JMPer Cable

INSIDE THIS NEWSLETTER

Introduction to One-Click Bootstrapping

Phillip Ramsey, Consultant and Professor, University of New Hampshire

Bootstrapping is an important capability offered in JMP® Pro. The implementation of bootstrapping is simple and flexible because bootstrapped distributions can be generated within most JMP Pro statistical reports. This article provides a brief tutorial on basic bootstrapping and describes applications of bootstrapping within the contexts of standalone inferential methods, pedagogy, and as a component of other statistical methods.

Bootstrapping is a common English language metaphor for a process that is self-sustaining and continues without need for an external driving force. Bradley Efron (1979) adopted the metaphor bootstrap to describe a new resampling technique he had developed. His work was motivated by earlier work on jackknifing (Tukey, 1958).

Statistical bootstrapping as developed by Bradley Efron (1979) is a process whereby independent, random samples of size N are repeatedly drawn (possibly thousands of times) with replacement from an original data set of size N. Replacement means that any individual observation in the original data set may be selected multiple times or perhaps not all in any one resample. This assures that each bootstrap sample is very likely to differ in membership from the original data.

By computing sample statistics of interest on each resample, empirical distribution functions for these statistics are generated. The empirical distributions can in turn be used to estimate standard errors for these statistics and estimate confidence intervals for theoretical parameters the statistics estimate. The bootstrap confidence intervals (there are a number of versions of them) have the advantage that the computation does not necessarily require use of a parametric or a specific mathematical formula. Hall and Wilson (1991) show that bootstrapping is easily adapted to hypothesis testing.

The most straightforward way to compute a bootstrap $(1-\alpha)100$ percent confidence interval for a theoretical parameter is to use the appropriate percentiles of the bootstrapped distribution for that sample statistic, which estimates the theoretical parameter of interest. As an example, the 2.5th and 97.5th percentiles of the bootstrapped distribution of a sample average can be used to form a 95 percent confidence interval for the theoretical mean of the underlying distribution.

Bootstrap research found that in some cases, the bootstrap percentile confidence intervals could be substantially biased in terms of the actual coverage (Chernick, 2008). Bias correction algorithms exist for percentile confidence intervals. However they are not currently available in JMP Pro and further discussion is beyond the scope of this article.

Statistical inference based on the bootstrap

The following example shows how to use JMP Pro for bootstrapping to generate percentile confidence intervals for theoretical parameters of target populations. The example uses 20 temperature readings (Kelvin) originally from Cox and Snell (1981).

Suppose you want to find confidence intervals for the **interquartile range** (IQR) of the sample, which is the difference between the 25th and 75th percentiles of the sample distribution. Exact computation of confidence intervals for the IQR is generally not possible.

Begin by using the Distribution platform in JMP to generate a table of summary statistics for the original sample. The interquartile range does not show by default in the Summary Statistics table, so use the red triangle menu to request it, as shown in Figure 1. Also, because the bootstrap process operates on all statistics showing in the table to which it refers, uncheck all the default summary statistics that are not of interest in this example. The Distribution Summary table then displays only the interquartile range, as shown in Figure 1.

Next, place the cursor in the body of the report, right-click and select **Bootstrap** from the menu that appears (Figure 2). In the Bootstrap dialog, select the Fractional Weights option – this option provides smoother bootstrap samples Rubin (1981).

This example generates 200 samples. You can enter any number of samples you want, but keep in mind that each sample reruns the Distribution platform and computes the interquartile range for that sample. So the number of samples you choose depends on the computational resources you have. A large bootstrap request can be both resource and time intensive.

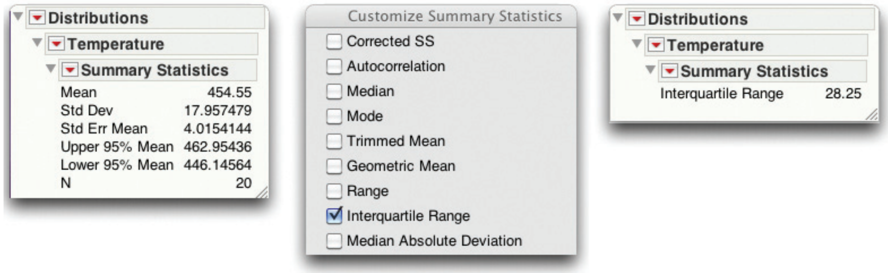


Figure 1 *Tailor Summary Statistics table to show only Interquartile Range*

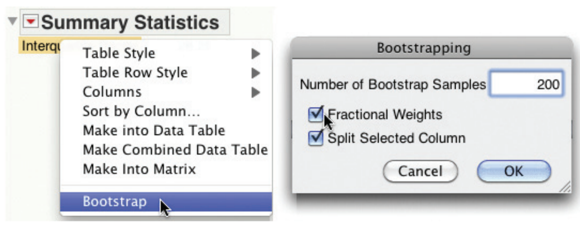


Figure 2 *Choose the Bootstrap with sample size 200*

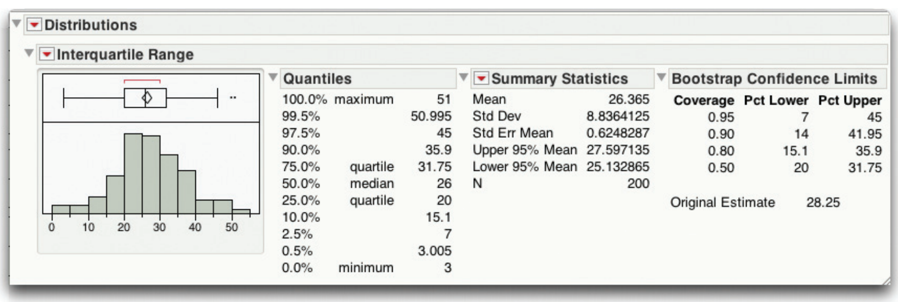


Figure 3 *Distribution of bootstrapped Interquartile Range*

When the bootstrap process is complete, you see a JMP table with the sampling results. These are columns called **BootID-** and **Interquartile Range** for the **Temperature** variable. The number of rows is the number of samples you requested plus one for the original sample computation. Now use the Distribution platform to look at summary statistics for the bootstrapped Interquartile Range.

JMP platforms that have the bootstrap option recognize the **BootID-** variable and display the Bootstrap Confidence Limits table, as shown in Figure 3.

The interquartile range computed in the original sample is 28.25 (Figure 1). Note that the interquartile range computed for the bootstrap sample of 200 is 25.4 and that the 95 percent bootstrap confidence limits, taken from the quantiles for the sample, are 7 and 45, which include the original estimate.

The bootstrap in education

Bootstrapping is starting to be adapted in statistics education (Lock, 2012) as a way to transition students from exploratory data analysis (EDA), where sample statistics are introduced, to confirmatory data analysis (CDA), where formal statistical inference is first introduced.

From the author's experience in teaching undergraduate, graduate, and Six Sigma courses in statistics, students have a difficult time grasping the concepts of a sampling distribution, standard error and confidence interval (or hypothesis test). That is, they struggle with the basics of statistical thinking.

Unfortunately, even hand computation of standard errors and parametric confidence intervals, when such computations are feasible, does little to motivate conceptual understanding on the part of the students. Often students become proficient in arithmetic computation without developing real conceptual understanding of the inferences to be drawn from such computation. This usually manifests itself by students correctly computing parametric confidence intervals (for example) and then subsequently stating incorrect or even nonsensical interpretations of those intervals. The students have apparently become proficient in arithmetic computation without the requisite ability to think statistically.

Beginning with the release of JMP® Pro 10, we introduced bootstrapping into the curricula for both academic and Six Sigma statistics courses. Rather than begin CDA with the traditional discussion of parametric sampling distributions, standard errors, and confidence intervals, first introducing bootstrapping motivates these concepts.

Anecdotally, the results so far have been quite encouraging in terms of students demonstrating deeper conceptual understanding of sampling distributions, confidence intervals, and hypothesis testing – the students seem more adept at proper statistical thinking.

Bootstrapped hypothesis tests are not directly available in JMP Pro, however quite a number of the more common hypothesis tests can be easily performed from the bootstrap results provided by JMP Pro. See Hall and Wilson (1991) for more detail on bootstrap hypothesis testing.

Bootstrapping in other statistical platforms

Besides being used as standalone methods for statistical inference, bootstrapping methods are increasingly incorporated into other statistical methods. A good example is partition modeling, where bootstrapping concepts are used to grow a random ensemble (or forest) of individual decision trees. JMP Pro has implemented this concept in the Partition platform with Bootstrapped Forest option.

Bootstrapping has also been incorporated into ANOVA, MANOVA, discriminant analysis, and regression methods. Many of the JMP Pro platforms for these analyses do have bootstrapping available. Unfortunately, bootstrap hypothesis tests in some cases require a bootstrapped standard error for each of the bootstrap samples; no exact computation exists for the standard error. The bootstrap standard error for each bootstrap sample is computed by the use of a double bootstrap (Chernick, 2008). Double bootstrapping is not currently supported in JMP Pro.

Overall, the use of bootstrapping and randomization tests in statistics education is growing and is only limited by the technology available to students, teachers and trainers. Its use is also consistent with the GAISE Report (ASA, 2012) recommendations for statistics education. JMP Pro provides a nice, easy-to-use platform for teachers to incorporate bootstrapping into the statistics course curricula.

Summary

In this article, we have discussed the uses of bootstrapping both as a form of statistical inference, especially where standard errors and confidence intervals cannot be easily calculated, and as an important pedagogical tool for statistics education. With the advent of ever more powerful computers, the use of bootstrapping and related methods will no doubt grow. JMP Pro provides a straightforward, easy-to-use bootstrapping capability such that the JMP user can incorporate bootstrapping into statistical analyses and also into the curricula for statistics education.

REFERENCES

Chernick, M. (2008). *Bootstrap Methods: A Guide for Practitioners and Researchers*, Second Edition. Wiley and Sons: NY, NY.

Cox, D. R. and Snell, E. J. (1981). *Applied Statistics: Principles and Examples*. London: Chapman and Hall.

Efron, Bradley (1979). *Bootstrap Methods: Another Look at the Jackknife*, The Annals of Statistics, 7, 1-26.

American Statistical Association (2012). *Guidelines for Assessment and Instruction in Statistics Education*.

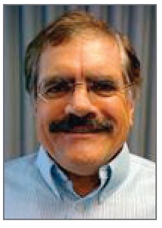
Hall, P. and Wilson S. (1991). *Two Guidelines for Bootstrap Hypothesis Testing*. Biometrics, 47, 757 – 762.

Lock, et. al. (2012). *UnLocking the Power of Data*. Wiley and Sons: NY, NY.

Rubin, D. B. (1981). *The Bayesian Bootstrap*, The Annals of Statistics, Vol. 9, No. 1, 130 – 134.

Tibshirani, et. al. (2008). *Elements of Statistical Learning*, Second Edition. Springer Verlag.

Tukey, J.W. (1958). *Bias and Confidence in Not-quite Large Samples*, The Annals of Mathematical Statistics. Vol. 29, 2, pp. 614.



Philip J. Ramsey, PhD, owns the North Haven Group, a quality and statistics consulting firm offering full service training and consulting in all levels and phases of Six Sigma as well as comprehensive training in design of experiments and predictive analytics. Ramsey is also a faculty member in the Department of Mathematics and Statistics at the University of New Hampshire (UNH).
He is co-author of *Visual Six Sigma* from SAS Press.

There's Always Something Slowing You Down and Add-Ins to Speed You Up Again

John Sall, Co-Founder and Executive Vice President, SAS

Are you frustrated that JMP doesn't have some feature that you need right now? I feel that way many times also. I find myself doing some task by hand over and over again. But now I have changed my attitude. If I am frustrated, then others are probably frustrated too at the same thing. I can invest an hour or two writing a script, and now it is worth doing, because I can submit it as an add-in for others to easily install. Here are three examples that I have found helpful.

Anonymizer

We often troubleshoot problems from users. Sometimes the problem is very data-specific, and we ask a user to send the data so that we can reproduce the problem here. But much of the time, users can't send us the data, because it is very proprietary or covered by other sharing restrictions. We often ask them if they can anonymize it somehow, removing the restricted parts or obscuring it in some effective way that still reproduces the problem. That can be a lot of work if done by hand but can be made easy by writing a script to do it.

It is a tall order because there are a lot of places that may need obscuring. It's not just the column names, but also the column values, and even value labels. If you change names, you can't just change them in the table; you have to change them in the scripts in the table.

The first thing was to make a copy of the table. The **Subset** command does that, though you do have to clear selections in the data table so that it will get the whole table. Here is the beginning of the script:

```
// Start the new table
dt<<Clear Column Selection;
// so Subset gets all columns
dt<<Clear Select; // so that
Subset gets all rows
newDt = dt << Subset( All rows,
All Columns );
newDt << SetName( "Anonomized" );
newDt << Begin Data Update;
```

Now you need to make new column names.

```
// Convert the column names
nameMap = Associative Array();
nc = N Col( dt );
For( i = 1, i <= nc, i++,
newName = "X__" || Char( i );
nameMap[Column( dt, i ) <<
GetName] = newName;
Column( newDt, i ) << SetName(
newName );
);.
```

You can get the script (called Anonymizer) from the JMP File Exchange and see the rest of the code, which does the character values, the value labels, and fixes the scripts. The resulting script has made a difference in our technical support and development troubleshooting. I am now seeing anonymized data tables that relate directly to specific user problems, rather than trying to simulate what the user has described.

Figure 1 shows an example of anonymization of the Big Class data table from the JMP Sample Data.

	name	age	sex	height	weight
1	KATIE	12	F	59	95
2	LOUISE	12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84
7	JAMES	12	M	61	128
8	ROBERT	12	M	51	79
9	BARBARA	13	F	60	112
10	ALICE	13	F	61	107

	X__1	X__2	X__3	X__4	X__5
1	V1_0	1000	V3_0	59	95
2	V1_1	1000	V3_0	61	123
3	V1_2	1000	V3_0	55	74
4	V1_3	1000	V3_0	66	145
5	V1_4	1000	V3_0	52	64
6	V1_5	1000	V3_1	60	84
7	V1_6	1000	V3_1	61	128
8	V1_7	1000	V3_1	51	79
9	V1_8	1001	V3_0	60	112
10	V1_9	1001	V3_0	61	107

Figure 1 *Example of anonymized data table*

Value labels viewer

The DDB Survey data table, shown below, is survey data and has more than 84,000 rows and 389 columns. Suppose you want to find groups of columns that have common categories (the same set of values) so that you can better organize the data table and more easily see the summaries in the Categorical platform, using the **Aligned Responses** option. The top table in Figure 2 shows a few question responses from the DDB. Most of the columns have value labels to display meaningful values in the table cells.

But how do you find groups of value label-aligned columns? Survey data is often messy – it is a lot of work to scroll through the data and select columns you think have the same set of value labels. What you want is some view of the data that shows all the value labels for the columns, grouped by the value labels. With this in mind, I wrote a script to be an add-in that does this.

The add-in is called Show All Value Labels and is available from the JMP File Exchange. This add-in produces the table shown at the bottom in Figure 2.

The new table lists all the column names in its source table that use value labels. The variables called Label1, Label2 and so on show the value labels assigned to the column. This table is linked to its source table.

Now it is easy to look through the value labels table and select a set of rows with the same value labels. These rows become selected columns in the original source table.

The data table is better organized, and like-labeled variables are more easily accessed when they are grouped. Use **Cols > Group Columns** to group the selected variables. Then highlight a group of columns (just click on the group name) and move the group (**Cols > Reorder Columns**) to a convenient location such as the beginning of the columns in the table, like the first table in Figure 3.

You can open a group of variables and select some or all of its members. The second table in Figure 3 shows a subset of the Activities group highlighted. These columns are then also highlighted in Categorical platform launch dialog. Click **Aligned Responses** on the dialog to see the results at the bottom in Figure 3. The Aligned Responses analysis gives interpretable results when the response variables have the same set of values.

Exclude Weakly Crossed Columns

When you have hundreds or thousands of survey questions over a multiyear survey, often not every question is asked every year. Questions go out of date, and new questions become relevant. Many questions may have only been

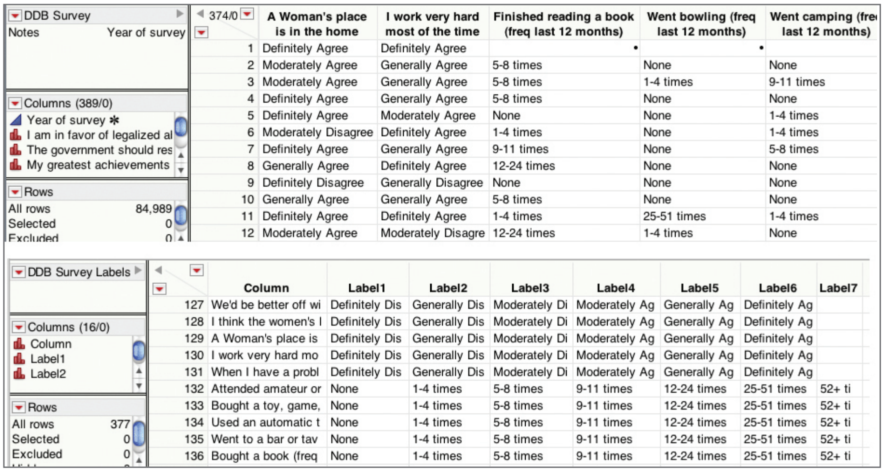


Figure 2 *DDB survey data table with table of value labels*

Source: Bowling Alone survey [http://bowlingalone.com]. You can download the data using the SPSS format, which has all the value labels assigned.

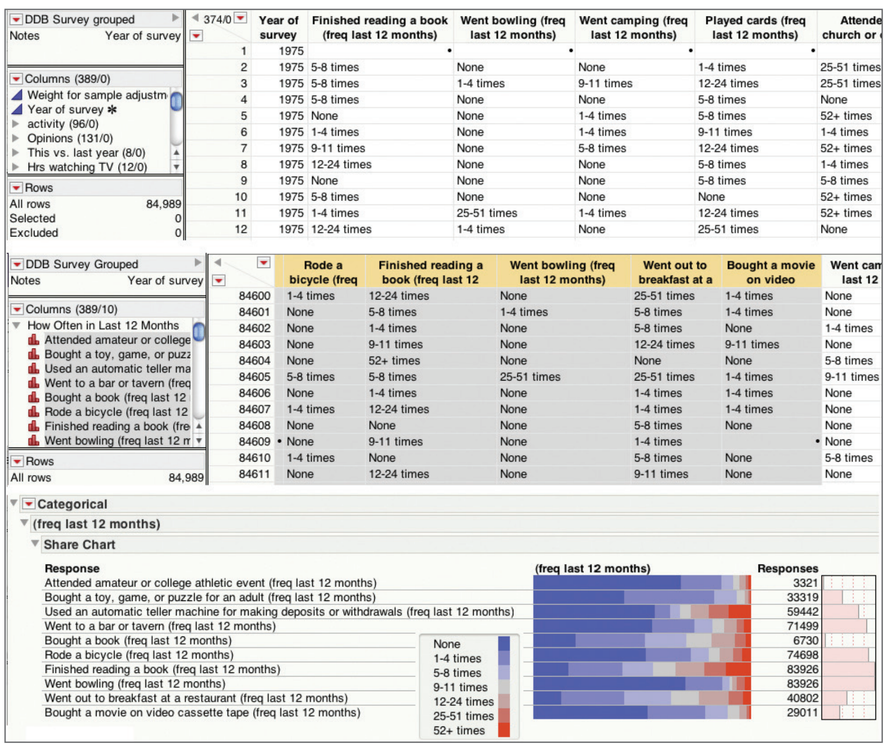


Figure 3 *Survey table with variables grouped by like value labels*

asked for one or two years in a 20-year survey. You probably don't want to see questions that were only asked once or twice. So you need a way to select the questions that were asked over many years, say at least 10 years over the 20-year Bowling Alone data. More specifically, you would like to exclude and hide columns (questions) that aren't asked on at least 10 years. The list of columns should look something like that in Figure 4.

There is an add-in script to do this called **Exclude Weakly Crossed Columns**. Execute the add-in script and fill out the dialog like the one shown in Figure 5 to see the result in Figure 4. This example uses all 104 of the grouped variables in the How Often in Last 12 Months category. This is another example of how add-in scripts can make your work load easier.

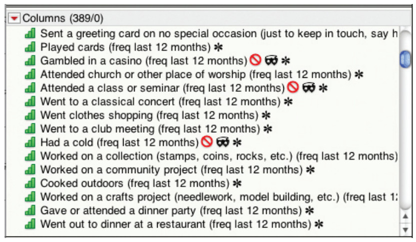


Figure 4 *Column list showing excluded and hidden*

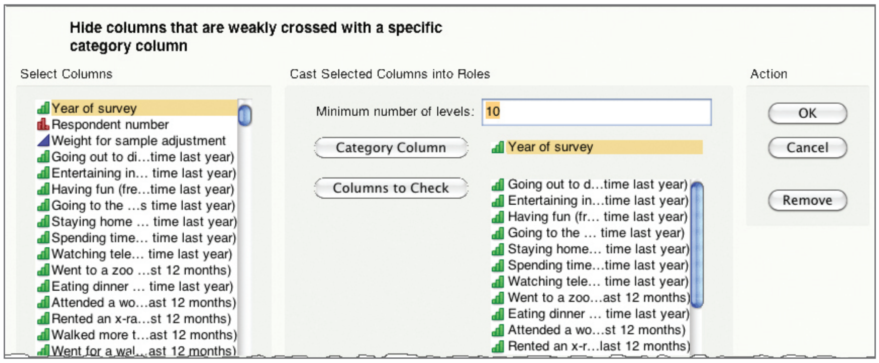


Figure 5 *Completed dialog for Exclude Weakly Crossed Columns dialog*

JMP Help at Your Fingertips

By Sheila Loring, JMP Technical Writer, SAS

When you need help generating that report or interpreting its results, time is of the essence. JMP provides you with a variety of documentation resources, whether you prefer reading online or in print. In addition to the traditional options, we're pleased to offer two new sources of help.

Online help

With JMP 10, help is available on the JMP website. As you can imagine, there are many benefits to online help. Searching the help with Google or another search engine can provide more robust results than searching the embedded JMP help or PDF files.

Find online help at **jmp.com/support/help**. On the help Web pages, you'll notice an orange search box. Enter search terms there to find topics only in the help (see Figure 1).

Embedded help

The help system inside JMP provides instant access to documentation. Many users rely on the question-mark tool in JMP to display information on a selected report graph or other graphical item. If you then find a topic and want to print out several pages, click the book title in the help window footer (see Figure 2) to open one of the PDF files.

When you search the help, search terms are highlighted on the page, which can be distracting. Turn off the highlighting by selecting **Options** and then select **Search Highlight Off**. The highlighting is removed from the next topics you view.

To display help topics that you recently viewed, look for the Home Window's Recent Help pane (on Windows only). Links to help topics, book PDF files and

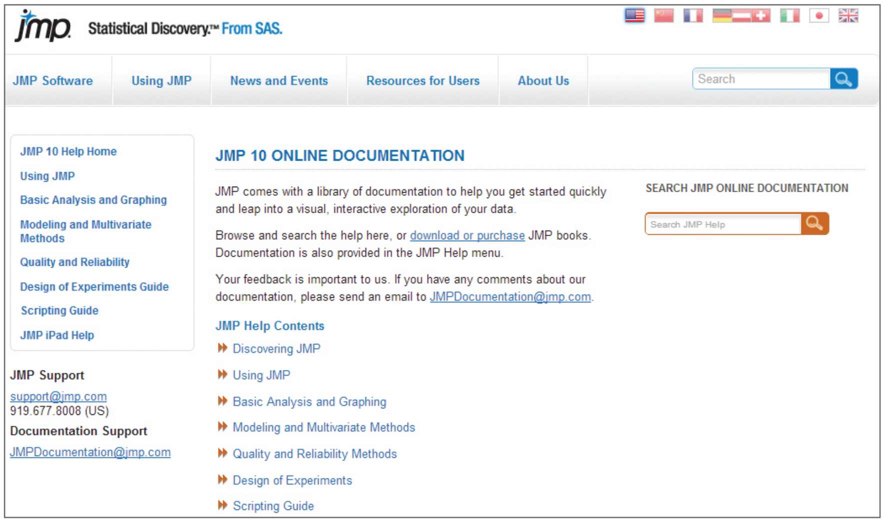


Figure 1 *Online Help documentation*

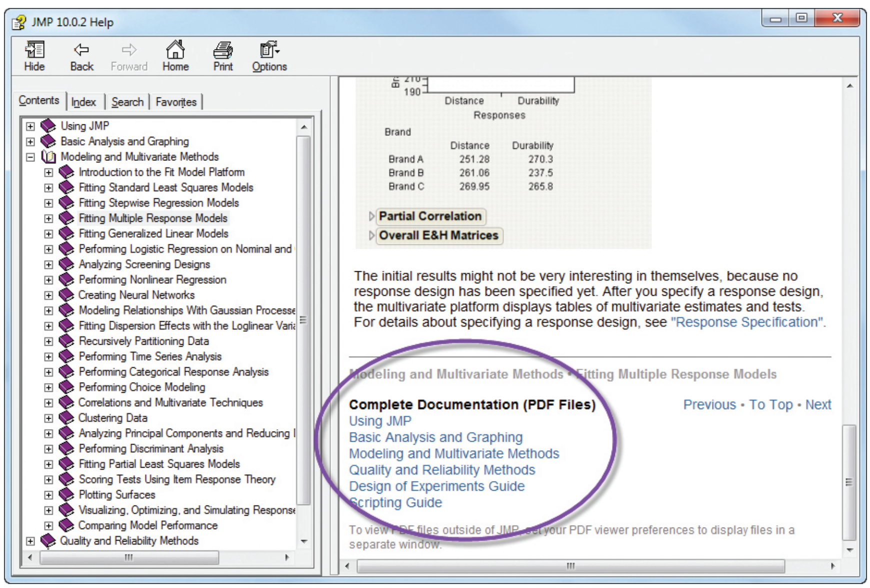


Figure 2 *JMP Help window on Windows*

other items from the Help menu appear in the pane.

Books in print and PDF

Readers often prefer searching or printing pages from the PDF books, which are available in the **Help > Books** menu. Can't figure out which book to look in? Acrobat provides an advanced search, which lets you search a directory of PDF files. **Select Edit > Advanced Search**, browse to select the JMP Documentation folder and enter the search terms. Results appear in a convenient outline view, grouped by book as shown in Figure 3.

If you prefer the printed word, purchase a JMP book from the SAS website or Amazon.com. Order printed books or download the PDF books at jmp.com/support/books.

E-books for the iPad®

We're also very excited to offer digital books (or e-books) for the first time with the release of JMP 10.0.1. Designed for viewing in iBooks on the iPad, e-books provide interactive options and let you personalize your reading experience. You can look up the definition of selected words and bookmark pages with a single tap. Carry JMP e-books everywhere, along with your other e-books. Purchase the e-books from the Apple iBookstore.

Which forms of documentation do you find most helpful? What features would you like to see? Let us know your thoughts so we can best meet your needs. Contact us at JMPDocumentation@jmp.com.

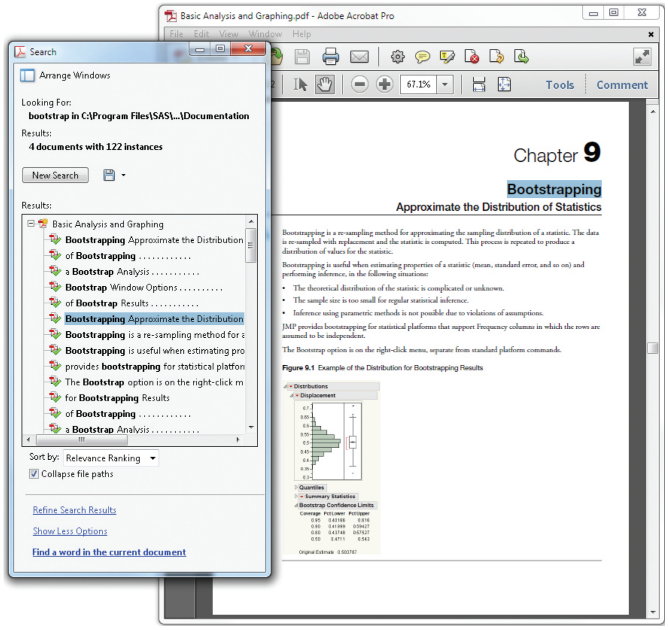


Figure 3 Searching JMP documentation

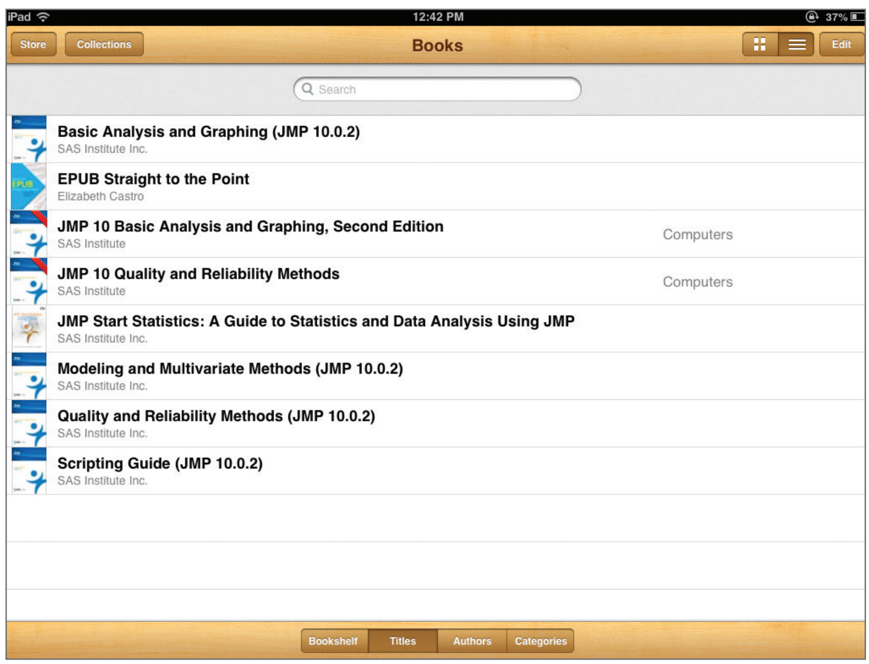


Figure 4 E-books on the iPad

Individual Run Replication

Bradley Jones, JMP Principal Research Fellow, SAS

Replication is one of the four basic principles of experimental design introduced by R. A. Fisher. The other three were the factorial principle, randomization and blocking.

The value of replication is that it provides an estimate of the run-to-run variability in the response that is unaffected if the model is incorrect.

In previous releases of JMP, replication meant to replicate a whole design. This is the traditional meaning of replication. To use replication you had to rerun every factor combination in the design a specified number of times. So, if a design had 12 unique factor combinations and you specified that you wanted one replicate, the resulting JMP table had 24 rows – every factor combination appeared twice. The interface for specifying replicates was available only after you created a custom design.

In JMP 10, instead of specifying how many times you want to repeat (replicate) the entire design, you decide how many individual runs to replicate. You make your request for replicates before rather than after design creation, and also specify which runs to replicate.

For example, suppose in the scenario above you want six replicated runs. Using the dialog in Figure 1, enter 6 in the edit box labeled Number of Replicate Runs. Then, click the Radio Button labeled User Specified and enter 18 total runs in the edit box. The resulting design has 18 runs with at least 6 replicated observations.

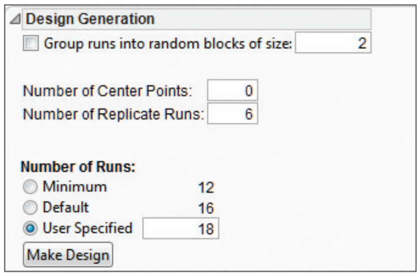


***Figure 1** Enter the number of individual run replicates*

How does this design compare to a design without replication?

To answer this question, consider an example that has two categorical factors each having three levels, and two continuous factors. The main effects model in all four factors only needs to consider the low and high ends of the range of the continuous factors. The full factorial design requires 36 runs, but suppose a budget constraint allows for only 18 runs. So generate a main effects model that has 12 default number of runs and then request six replicate runs for a total of 18.

Figure 2 shows the factor settings for a design with 12 unique factor combinations where seven of these combinations appear twice as illustrated by the red boxes.

Note that there is some symmetry in this design. The nine possible combinations of three level categorical factors each appear twice. Also, the factor X_4 is balanced with nine runs each at its low and high settings. However, the factor X_3 has only eight runs at its low setting but 10 runs at its high setting.

The default optimality criterion is D-optimality. Despite this slight lack of balance, the D-efficiency of this design

	X1	X2	X3	X4	Y
1	L1	L1	-1	1	83.65
2	L1	L1	1	-1	81.63
3	L1	L2	-1	-1	70.32
4	L1	L2	-1	-1	69.08
5	L1	L3	1	1	84.21
6	L1	L3	1	1	82.29
7	L2	L1	1	-1	82.81
8	L2	L1	1	-1	81.12
9	L2	L2	1	1	84.17
10	L2	L2	1	1	83.91
11	L2	L3	-1	-1	69.83
12	L2	L3	-1	1	81.87
13	L3	L1	-1	1	84.04
14	L3	L1	-1	1	84.72
15	L3	L2	-1	-1	68.68
16	L3	L2	1	1	84.17
17	L3	L3	1	-1	82.52
18	L3	L3	1	-1	84.37

***Figure 2** 18-run design with six replicated runs*

is 99.1%. In cases where the investigator specifies a number of replicates, the custom designer creates a design that is optimal given the constraint that it must supply the specified number of replicates.

For comparison, the design in Figure 3 shows the 18 run D-optimal design generated specifying no replicates. This table has the same pattern for the two three-level categorical factors. However, both of the continuous factors have nine runs each at their low and high settings. The D-efficiency of this design is 99.8 percent so the design with replication is

	X1	X2	X3	X4	Y
1	L1	L1	-1	1	84.67
2	L1	L1	1	1	84.29
3	L1	L2	-1	-1	70.63
4	L1	L2	1	-1	80.61
5	L1	L3	-1	1	83.34
6	L1	L3	1	-1	82.57
7	L2	L1	-1	-1	67.62
8	L2	L1	1	-1	81.96
9	L2	L2	-1	1	83.91
10	L2	L2	1	1	83.57
11	L2	L3	-1	-1	71.19
12	L2	L3	1	1	83.59
13	L3	L1	-1	-1	70.17
14	L3	L1	1	1	85.76
15	L3	L2	-1	1	83.65
16	L3	L2	1	-1	83.37
17	L3	L3	-1	-1	70.51
18	L3	L3	1	1	83.42

***Figure 3** 18-run design, no replicated runs*

less D-efficient by less than 1 percent than the optimal design.

What are the implications of individual run replication in analysis?

Both tables have simulated data. The a priori model for each design was the main effects model, but I added a term for the two-factor interaction involving X_3 and X_4 to the simulated data along with normal random error with a standard deviation of one. Here is the equation of the simulation:

$$Y = 80 + 3X_3 + 4X_4 - 3X_3X_4$$

Because there are multiple replicated runs in Figure 1, the JMP Fit Model platform automatically does a Lack-of-Fit test (Figure 4). Since the two-factor interaction is not in the model, the test is significant ($p > 0.0079$) indicating that the model is inadequate.

Armed with this information, the savvy investigator can add two-factor interactions to the fitted model in an effort to find the missing effect.

For the unreplicated design, there is no test for lack of fit because there are no pure error degrees of freedom to detect lack of fit. The RMSE of the fitted model is 3.59, which is three times larger than the true simulated error standard deviation of one. That is because the active two-factor interaction is not in the model, which results in an overestimate of the error standard deviation. This, in turn, means that tests of the model effects have lower power.

To be fair, it is not too difficult to detect the lack of fit using the data from the unreplicated design. Figure 5 shows a graph of the residuals from the fit of the unreplicated data plotted against X_3 and overlaying the two values of X_4 with different colors and associated lines.

Lack Of Fit

Source	DF	Sum of Squares	Mean Square	F Ratio
Lack Of Fit	5	48.027639	9.60553	9.5795
Pure Error	6	6.016300	1.00272	**Prob > F**
Total Error	11	54.043939		0.0079*
				Max RSq
				0.9901

Figure 4 *Lack-of-fit results for replicated runs design*

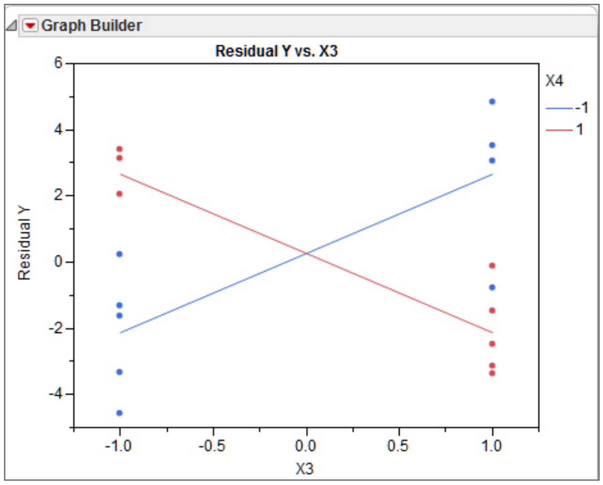


Figure 5 *Residual overlay plot indicating a strong two-factor interaction*

The interaction is clear because the overlaid lines cross. Producing this plot requires some additional effort and expertise on the part of the data analyst. By contrast, the lack-of-fit test is automatic.

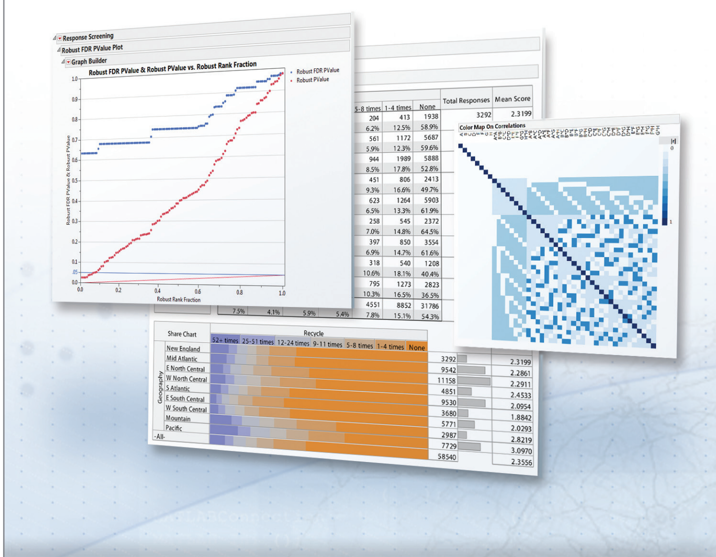
What is the bottom line?

There is safety in replication. That safety comes at a price in that for a given run budget, replicating runs means that you will not be able to estimate as many terms. Also individual point replication generally results in some increase in the variance of the coefficient estimates.

What individual run replication offers is more flexibility in the design specification – and flexibility is the whole point of Custom Design in JMP.

JMP® 11: Follow clues to new breakthroughs

Ready to make your data analysis more productive? In September, we'll proudly introduce new versions of JMP and JMP Pro, with customer-inspired advances that speed the pace of discovery from data import to analysis to presentation.



Working with Data: Making Hard Things Easy

TUESDAY, SEPT. 10, 10 – 11:30 A.M. ET

Join JMP creator John Sall and leaders from his software development team Tuesday, Sept. 10, as they introduce JMP 11 during a live webcast from Discovery Summit 2013 in San Antonio.

jmp.com/preview-jmp11

George Box: A Remembrance

Bradley Jones, JMP Principal Research Fellow, SAS

Excerpt from the JMP Blog

In this International Year of Statistics, we mourn the passing of George Box, truly a giant among 20th century statisticians. Indeed, he should not be relegated to the 20th century because he won the Brumbaugh Award in 2010 and 2007 for writing papers that made in their year of publication the largest single contribution to the development of industrial application of quality control.

Professor Box was the father of response surface methodology, which arose following his groundbreaking 1951 paper with K.B. Wilson. He also popularized the regular two-level fractional factorial designs, writing two great papers with Stu Hunter in 1961.

Another one of his great contributions was the wonderful book *Statistics for Experimenters*, which he wrote with William G. Hunter and Stu Hunter and published in 1978, the same year he served as president of the American Statistical Association. I remember the excitement I felt on reading the description of how the attainment of knowledge is an endless spiral proceeding alternately from deduction to induction and back. Even now, I recall with pleasure the discussion of the randomization distribution early in the book.

And let's not forget his contributions in Bayesian analysis (Box and Tiao 1973), time series (Box and Jenkins 1970) and control (Box and Luceño 1997).

Yes, the world of statistics has lost a great one.

To read the complete blog post and interesting comments from others, see:

blogs.sas.com/content/jmp/2013/03/29/george-box-a-remembrance/

You can become acquainted with this remarkable man through his autobiography – *An Accidental Statistician: The Life and Memories of George E. P. Box.*

Solving a Problem with the JSL Debugger

Melanie Drake, JMP Development Tester, SAS

Following is a case study that uses the JSL Debugger to solve a problem. This is based on a real script with the problem specified. You can download a simplified test script from the JMP File Exchange that shows the problem. The test script is listed as an Educational Demo (and called **Solving a Problem with the JSL Debugger**.)

The Problem

The original script ran without problems in English, but produced errors in Simplified Chinese. It appears that the way strings are handled in English is different than in Chinese: Specifically, escaped quotes (\!") are left out, which leads to errors when a string is parsed.

In English, here is a sample string that was constructed:

```
dt<<Select Where(dt:Numbers
== 51 & dt:Pick Strings ==
\!"North\!"
```

In Chinese, the string looked like this:

```
dt<<Select Where(dt:Numbers
== 51 & dt:Pick Strings ==
North
```

Notice the missing `\!"`...`\!"`, which caused errors when the string was parsed and that unquoted value couldn't be resolved.

First Look

The script was just short of 900 lines and produced an interactive report. The first 130 lines were setup, lines 131-886 defined expressions, and the rest of the script used those expressions.

The first thing I did was run the script in English and then in Chinese, which produced the same problem that the user had. Then I ran it in French, which confirmed that the problem was with any non-English language. I ran the rest of my tests in English and French, because French is easier for me to read than Chinese.

There were several lines that constructed the string shown above, and they all relied on complex coding that would be hard and time-consuming to execute line-by-line, entering and exiting multiple expressions.

Running just the appropriate string-building lines (after the script had been run once to initialize all variables) gave the same results in all languages, with the escape quotes used correctly.

However, the lines that formed the string used values from two data tables plus a third that was constructed from the first two. Thus, it was very hard to tell at a glance where all the variables being used were created and populated.

Enter the JSL Debugger

Running JMP in English, I opened the test script and clicked **Debug**. Since I had already run the script, all its variables were listed under **Globals**. There were too many to easily watch while stepping through the code. (I could also have learned this by running "`Show Globals()`" and watching the log grow very long.)

That was all I needed to know then, so I quit the Debugger and quit and restarted JMP so that my environment was completely clean. I opened the test script, and started the Debugger again.

I was interested only in the string that was being built, so I added that variable to my **Watch** list. To do this, I selected the variable in the script as shown in the Debugger (see Figure 1), then right-clicked, and selected **Add Watch**.

I added a break point on the first line after the setup portion was finished (Line 131 in the original. and Line 34 in my

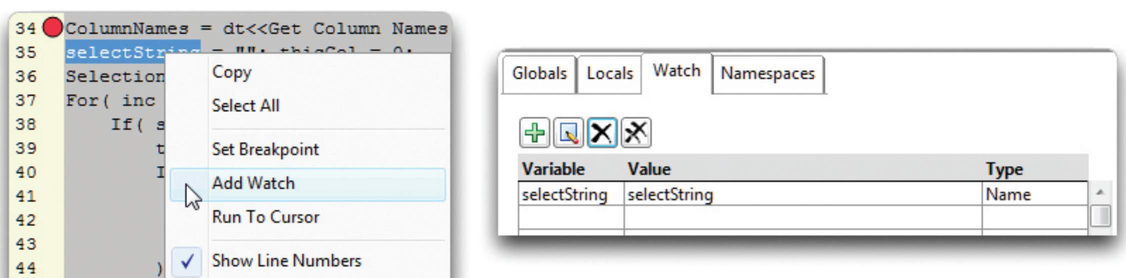
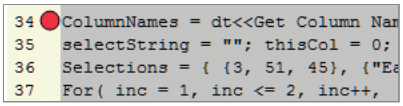


Figure 1 *Setting up the watch tab in the Debugger*

test script). Another break point is added at the first "real" instruction line of code after all the expressions are created (Line 887 in the original; this doesn't exist in the test script). To do this, I simply clicked next to the line number in the test script in the Debugger (as shown here).



First, I clicked **Run** to run the entire "setup" portion of the test script and saw that the string variable had not yet been created. Finding that out with one click was much faster than running the script manually line by line!

Next, I clicked **Run** to create all the expressions. The meat of the script was contained at the bottom of the original script, in only 10 lines. The first part was a loop that called expressions (which called expressions, which called expressions, etc.).

I used the **Step Into** button to step into each expression and find out what was happening. **Note: Step Over** would call an expression, run the whole thing plus any expressions it called within, and return a result, all without showing how the string was constructed. I needed to see the string as it was built, piece by piece.

Results

What I discovered was that each piece of the string was built within an `If()`. If the column with the information was character, then the value was quoted with the escaped quotes; otherwise, the value was not quoted.

This also meant that, out of almost 900 lines of code, I was really only interested in a small chunk of 19 lines (which nevertheless depended on the other 800+). These lines are re-created (and simplified) in Lines 38 – 54 in the test script.

In fact, the condition in other languages always returns 0. The condition piece of the `If()` is this (Lines 40 and 48 in the test sample):

```
Parse( "dt:" ||
ColumnNames[thisCol] )
   << Get Data Type() ==
   "Character"
```

In English, `col<<Get Data Type()` returns "Character" for a Character column. In French, for example, it returns "Caractère", which never evaluates as equal to "Character".

The result of `col<<Get Data Type()` always returns a localized answer.

Since the column in question was never determined to be equal to "Character" when run in Chinese (or French or any language other than English that JMP supports), the string was constructed as if the cell value was numeric. That was why the string in English had escaped quotes and the string in Chinese didn't.

Note: Although the Debugger doesn't show the results of the `Get Data Type()` message – only the result of the comparison – I could easily quit the Debugger, and select and run

```
Parse( "dt:" ||
ColumnNames[thisCol] ) <<
Get Data Type()
```

to see what was actually returned. The Debugger allowed me to find the problem quickly and painlessly.

A Simple Problem with a Simple Solution

In this case there was no bug in JMP: It is working as designed. The script needs only a small change to work in any language. Instead of discovering a column's data type by using the `<<Get Data Type()` message, get the first cell value and test whether it is a string:

```
Is String (Parse( "dt;" ||
    ColumnNames [thisCol] )
[1] );
```

I changed JMP to run in French, quit and restarted, and went through the debugger again.

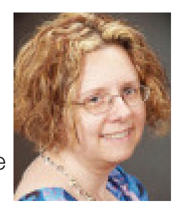
Suddenly, all was clear...

JMP Often Returns Localized Information

The problem was not with the assignments that built the string. The debugger displayed the problem – in non-English code, the condition within the `If()` that determined whether the column was character returned 0 for character columns. In English, the condition returned 1 for character columns.

```
Is String (Parse( "dt:" ||
     ColumnNames[thisCol])
     [1] );
```

As is the case with Perl, there is often more than one way to do something in JSL.



Melanie Drake is a Development Tester in the JMP Development group at SAS. She has worked at SAS for more than 11 years. Prior to working in her current position, she wrote and edited many of the JMP manuals and user guides. In particular, she updated, tested and maintained the JMP Scripting Language guide.

More Choices for Modeling Nonlinear Responses with JMP® 10

Mark Bailey, Principal Analytical Training Consultant, Education and Training, SAS
Clay Barker, JMP Senior Research Statistician Developer, SAS

Introduction

There are many instances where the response to a change in a variable is linear. In other cases, the response is not linear. A nonlinear response can sometimes be approximated by a model that is *linear in the parameters* (linear models). These models are a linear combination of terms in which the parameters appear singly in each term and only as a coefficient. A common linear model for a nonlinear response is a polynomial function, such as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 .$$

Finally, other cases of nonlinear responses require a model that is nonlinear in the parameters (nonlinear models). There are myriad such models, for example $y = \alpha + \beta e^{-\gamma x}$, that are used across a broad range of disciplines.

Let's watch as the Fit Curve platform handles two common situations:

- Determining *expiry* (shelf life) with an exponential decay
- Comparing a sample to a standard with a logistic calibration curves as used in bioassays or immunoassays.

Chlorine degradation example

The Chlorine degradation data table (shown here) contains measurements of

	Age	Concentration	Test
1	8	0.49	1
2	8	0.49	2
3	10	0.48	1
4	10	0.47	1
5	10	0.48	2
6	10	0.47	2
7	12	0.46	1
8	12	0.46	1
9	12	0.45	2
10	12	0.43	2
11	14	0.45	1
12	14	0.43	2

the chlorine level of 42 batches of a disinfectant at various ages (days since preparation). You need to determine when the level falls below the lower specification level of 0.4 (LSL).

To follow along:

1. Select **Analyze** > **Modeling** > **Nonlinear**.
2. Select Concentration and click **Y, Response**.
3. Select **Age**, click **X**, **Prediction Formula**, then click **OK**.

The decrease in the chlorine level over time is apparent from the scatterplot (see Figure 1).

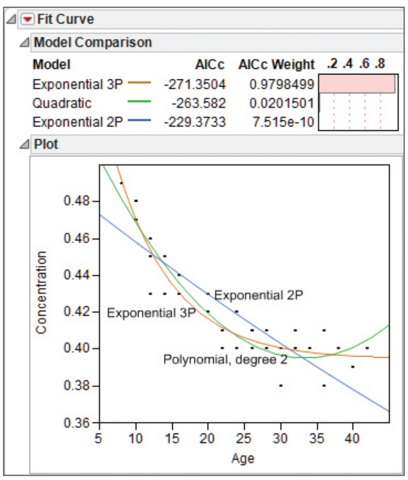


Figure 1 *Scatterplot with exploratory fits*

You can explore the shape of the data using commands in the red triangle menu on the Fit Curve title bar. The change is not linear, so start with a quadratic polynomial.

4. Select **Polynomials** > **Fit Quadratic**.

This model captures the curvature, but the parabolic shape must turn up as **Age** increases past the minimum point but the chlorine level does not rise in the future.

Next try a simple growth (decay) model.

5. Select **Exponential Growth and Decay** > **Fit Exponential 2P**.

This curve is forced to zero as **Age** goes to infinity – it performs worse than the polynomial.

Now use a growth curve with a non-zero asymptote.

6. Select **Exponential Growth and Decay** > **Fit Exponential 3P**.

This choice provides a reasonable model.

Compare models

The Model Comparison report shown in Figure 2 provides essential performance indicators. The model list is sorted by AICc. The best model has the smallest AICc. Note that the second best model has an AICc that is 7.8 higher, which indicates considerably less support from the data.

Fit Curve

Model Comparison

Model	AICc	AICc Weight	.2 .4 .6 .8	BIC	SSE	MSE	RMSE	R-Square
Exponential 3P	-271.3504	0.9798499		-265.4808	0.0030975	7.9424e-5	0.008912	0.9162612
Quadratic	-263.582	0.0201501		-257.7124	0.0037269	9.5561e-5	0.0097755	0.8992478
Exponential 2P	-229.3733	7.515e-10		-224.7919	0.0089208	0.000223	0.0149338	0.7588357

Figure 2 *Model Comparison reports*

The Prediction Model reports beneath the scatterplot (shown here) provide the form of the model and the interpretation of the parameters for each model fitted to the data.

Exponential 3P

Prediction Model

a + b * Exp(c * Age)

a = Asymptote

b = Scale

c = Growth Rate

Parameter Estimates

Parameter	Estimate	Std Error	Lower 95%	Upper 95%
Asymptote	0.394464	0.0035396	0.3875266	0.4014014
Scale	0.2653471	0.0391072	0.1886983	0.3419958
Growth Rate	-0.124905	0.0170651	-0.158352	-0.091458

The asymptote indicates that the degradation stops near the given LSL (lower specification level) of 0.40. The negative growth rate indicates that the decay is 0.125 units per day. **Note:** The sum of the scale and the asymptote indicates the starting level, 0.394464+0.2653471=0.6598111.

Inverse prediction

You can perform an inverse prediction to determine expiry. Select **Custom Inverse Prediction** from the red triangle on the Exponential 3P title bar. Use 0.95 for the confidence level. Select **Lower One Sided** from the drop-down menu, Enter 0.40 for Concentration and click **OK**.

Inverse Prediction

Predicted Values

Specified Concentration	Predicted Age	Std Error	Lower 95%
0.4000000	30.98178	2.829243	26.32809

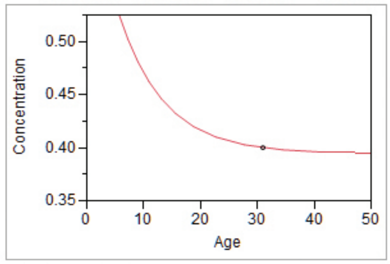


Figure 3 *Inversion Prediction Plot to estimate expiry*

		Sample	Conc (ug/mL)	Log Conc	Rep 1	Rep 2	Rep 3	Mean Assay
•	10	Standard	3	0.48	0.95	1.00	0.91	0.95
•	11	Standard	1.25	0.10	0.92	0.94	1.00	0.95
•	12	Sample	320	2.51	0.06	0.07	0.06	0.06
•	13	Sample	240	2.38	0.06	0.05	0.05	0.05

Figure 4 *Partial listing of the Dose-Response Analysis data table*

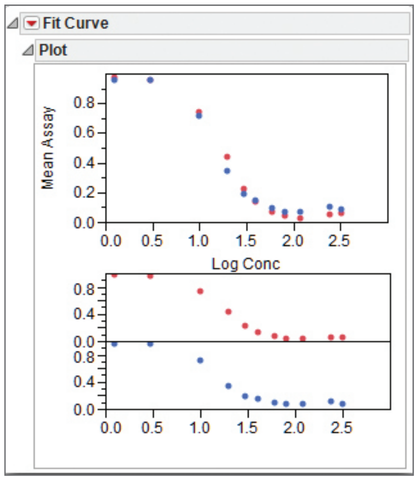


Figure 5 *Total and group plots for Mean Assay by Log Conc*

Figure 3 shows the fitted decay curve and the point estimate. The lower 95% confidence bound suggests that the disinfectant must be discarded and remade after 26 days.

Multiple samples

Now watch the Fit Curve platform use special features in a dose-response curve analysis. The **Standard & Sample Dose-Response Analysis** data table (Figure 4) includes an indicator (Sample) for the type of sample ("Standard" or Sample"), the concentration (**Conc (ug/mL)**) and log concentration (**Log Conc**), three standardized replicate assays, and the mean assay (**Mean Assay**).

To follow along:

1. Select **Analyze** > **Modeling** > **Nonlinear**.
2. Select **Mean Assay** and click **Y, Response**.
3. Select **Log Conc** and click **X, Predictor Formula**.
4. Select **Sample**, click **Group**, then click **OK**.

You see in Figure 5 that data for both samples are plotted together and individually (blue is "Standard," red is "Sample").

Now Fit the data using a logistic model:

5. Select **Sigmoid Curves** > **Logistic Curves** > **Fit Logistic 3P** from the red triangle on the Fit Curve title bar.

This model accounts for both asymptotes, which is not possible with a polynomial model.

Consider a more complex model:

6. Select **Sigmoid Curves** > **Logistic Curves** > **Fit Logistic 4P** from the red triangle on the Fit Curve title bar.

The additional flexibility afforded by this model fits the data better.

The addition of yet another parameter allows for asymmetry in the asymptotes:

7. Select **Sigmoid Curves** > **Logistic Curves** > **Fit Logistic 5P** from the red triangle on the Fit Curve title bar.

Look at the Model Comparison table in Figure 6 to see that the best model is Logistic 4P. The Logistic 3P model has essentially no support from the data (delta AICc is 92.80 – 60.38 = 32.42). The more flexible Logistic 5P model does not offer any real improvement.

The AICc Weight helps with model selection (Figure 6). You may interpret this case to say that, given one of the three fitted models is true, then there is a 83% chance that the Logistic 4P is the true model (AICcWeight = .8259).

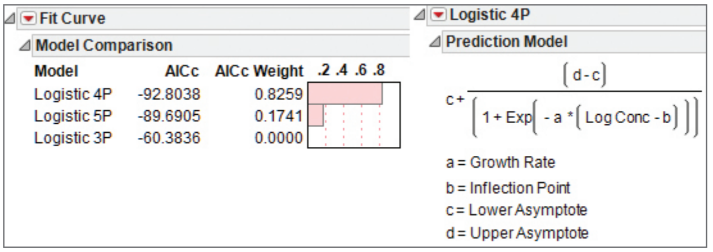
Since the Sample variable is a Group variable in the analysis, there are separate sets of parameter estimates for each group.

Figure 8 shows the curves based on the fitted model plotted together and separately for each group. It appears that the lower asymptote is different for the sample than it is for the standard.

The sample and standard curve appear to have a similar shape. If so, the relative potency can be estimated. To do this, select **Test Parallelism** from the red triangle menu on the Logistic 4P title bar.

Two hypothesis tests are provided. Both tests compare the full model to the reduced model. The full model estimates each parameter separately for the sample and the standard.

The reduced model estimates each parameter for the combined data, except the inflection point. In this example, the *F*-test is significant (0.0219) but the χ^2-test is not at α=0.05. The *F*-test result might be a type I error, or it might indicate that the lower asymptote is not the same for the sample and the standard.


Fit Curve
Model Comparison

Model	AICc	AICc Weight
Logistic 4P	-92.8038	0.8259
Logistic 5P	-89.6905	0.1741
Logistic 3P	-60.3836	0.0000

Logistic 4P
Prediction Model

$$c + \frac{(d-c)}{\left(1+\mathrm{Exp}\left(-a*\left(\mathrm{Log\ Conc}-b\right)\right)\right)}$$

a = Growth Rate
b = Inflection Point
c = Lower Asymptote
d = Upper Asymptote

Figure 6 *AICc results for Logistic 4P model*

Logistic 4P
Prediction Model
Summary of Fit
Parameter Estimates

Parameter	Group	Estimate	Std Error	Lower 95%	Upper 95%
Growth Rate	Sample	-5.386771	0.322846	-6.019537	-4.754004
Inflection Point	Sample	1.2300435	0.0132749	1.2040251	1.2560619
Lower Asymptote	Sample	0.034498	0.0093935	0.0160871	0.0529088
Upper Asymptote	Sample	0.9713194	0.0134447	0.9449683	0.9976706
Growth Rate	Standard	-6.087709	0.3901823	-6.852453	-5.322966
Inflection Point	Standard	1.1627806	0.0133358	1.136643	1.1889182
Lower Asymptote	Standard	0.0792695	0.0084472	0.0627134	0.0958256
Upper Asymptote	Standard	0.9603754	0.013253	0.9344	0.9863508

Figure 7 *Parameter estimates for each group*

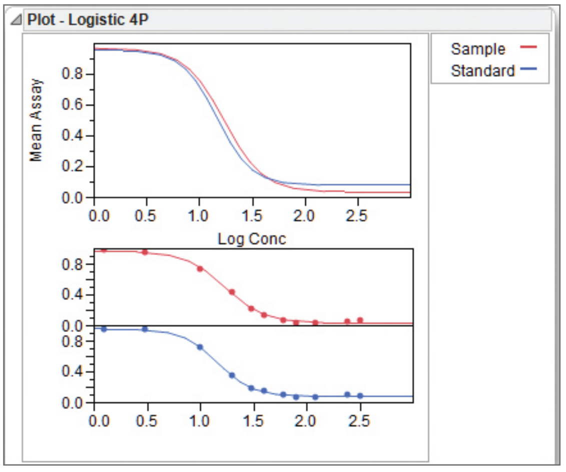


Figure 8 *Fitted curves for total Mean Assay and Sample Groups*

Parallelism Test - Logistic 4P
Test Results
Parallelism F Test

Parallel Fit SSE	Full SSE	NDF	DDF	F Ratio	Prob > F
0.0082438	0.0042315	3	14	4.425	0.0219*

Parallelism Chi-Square Test

ChiSquare	DF	Prob>ChiSq
0.004	3	0.9999

Relative Potencies
Relative Potency versus Sample

Group	Potency	Relative Potency	Std Error
Sample	3.3877074	1	0
Standard	3.2302119	0.9535097	0.0169253

The relative potency may be used when the curves are judged to be parallel. In this case, the potency of the sample is similar to that of the standard. The potency is the log concentration corresponding to the EC50 response.

JMP offers a newer approach to deciding about the relative shape in general, and parallelism in particular, that is based on an equivalence test for each parameter.

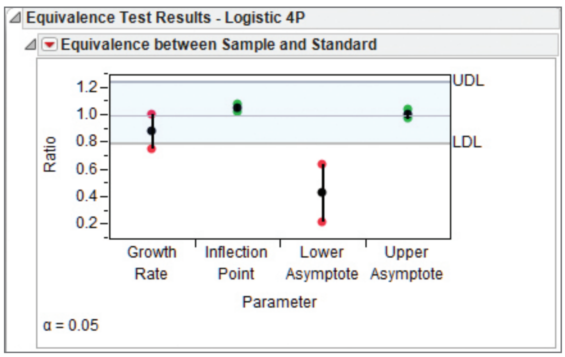
The newer approach is based on the two one-sided *t*=tests (TOST) approach, but it is presented graphically with confidence intervals for easier interpretation. The confidence interval for the ratio of the shape parameter estimates for the sample and the standard should be within a range deemed to be equivalent. The inflection point parameter may exceed this range, indicating a change in the relative potency.

To see this approach,

8. Select **Equivalence Test** from the red triangle menu on the Logistic 4P title bar.
9. Select "Standard" as the reference group and then click **OK**.

The two one-sided *t*=tests determine if the parameter estimates exceed a default 25 percent difference by ratio. It is apparent that the lower asymptote is not equivalent with the standard, so these curves are not parallel (Figure 9). This is confirmed by the Equivalence Summary Table.

See the following article, "Nonlinear Platform Summary: Nonlinear and Fit Curve Features," for a summary table of these features.



Equivalence Summary

Level	Lower Confidence Limit	Ratio	Upper Confidence Limit	Limit Exceeded
Growth Rate	0.757144	0.88486	1.012576	Lower
Inflection Point	1.030445	1.057847	1.085249	
Lower Asymptote	0.225888	0.435198	0.644509	Both
Upper Asymptote	0.97888	1.011396	1.043911	

Figure 9 *Graphical Results of TOST Equivalence Test*

References

Burnham, Kenneth P., and Anderson, David R. (2002) *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach* (Second Edition), Springer, New York, NY, page 70.

SAS Institute Inc. (2012), *Modeling and Multivariate Methods*, Cary, NC: SAS Institute Inc.

Nonlinear Platform Summary: Nonlinear and Fit Curve Features

Mark Bailey, Principal Analytical Training Consultant, Education and Training, SAS

JMP often provides more than one way to accomplish the same result to suit different needs and different styles of work, whether a small task or a large one. If you want to fit a model for a linear response to a single continuous variable, you could use either Fit Y by X or Fit Model. Both platforms give the same answers as far as whole model inference and parameter estimates inference. They differ, though, in details and exploitation capabilities. The Bivariate platform is optimized for the common special case of a single predictor. Fit Model is optimized for a broader range of models. This paradigm is also true of fitting nonlinear models.

Table 1 contrasts the two platforms for nonlinear modeling. Which platform is launched is determined by whether your column assigned the X role contains a Formula property. If it does, then Nonlinear launches; otherwise, Fit Curve launches.

The new Fit Curve platform provides polynomials through the fifth order and many commonly used nonlinear models. This platform provides a common framework for all of the nonlinear models that are included, but also allows for specialization to recognize unique applications, for example, testing equivalence or parallelism with logistic curve. Table 2 summarizes the specialized curves available in the Fit Curve platform.

Table 1 *Table of Nonlinear Features and Fit Curve Features*

Nonlinear Features	Fit Curve Features
Define any nonlinear model through a column formula, or select from editable model library.	Easy selection among popular models in platform menu.
Explicit choice of iteration options.	Iterations handled automatically.
Explicit control of convergence criteria.	Convergence criteria managed automatically.
Explicit specification of starting parameter values.	Heuristics for each curve type provide reasonable starting values.
Compare and test hierarchical models with Remembered Solutions.	Compare and select models using simultaneous plots of graphs of fitted model functions, and ranking by AICc, BIC and other criteria.
Full set of Profilers for response and model parameters.	Separate Prediction Profiler that shows the response, first derivative and second derivative for each model.
Grouped analysis through specific model terms.	Grouped analysis through casting column in Group role in launch dialog. Parameters compared across groups with Analysis of the Two One-Sided Tests (TOST) approach to find differences. Equivalence test across groups to find parallelism.
The loss function is least squared error by default, but the user may specify an alternate.	The loss function is the common least squared error.

Table 2 *Summary of specialized curves in the Fit Curve Platform*

Polynomials	Fit Linear through Fit Quintic
Sigmoid Curves	Logistic Curves (Fit Logistic 2P through Fit Logistic 5P) Fit Gompertz 3P Fit Gompertz 4P
Exponential Growth and Decay	Fit Exponential 2P and 3P Fit Biexponential 4P and 5P Fit Mechanistic Growth
Peak Models	Fit Gaussian Peak Fit Lorentzian Peak
Pharmacokinetic Models	Fit One Compartment Oral Dose Fit Two Compartment IV Bolus Dose Fit Biexponential 4P
Fit Michaelis Menten	Fit Michaelis Menten

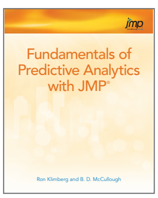
JMP® Books from SAS® Press

SAS Press, a very busy department within the Publications Division at SAS, continues to publish up-to-date books by outside authors. Topics by experts in a variety of fields are selected, carefully screened, edited and produced to offer SAS and JMP users the latest in theory, techniques, examples and case studies. In keeping with the latest in publishing technology, many books are now available as e-books.

Here is a selection of published books that show how to use JMP to help solve a variety of problems.

Fundamentals of Predictive Analytics with JMP®

(May 2013)
By Ron Klimberbg and B.D. McGullough



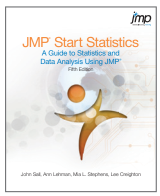
This new book bridges the gap between courses on basic statistics, which focus on univariate and bivariate analysis, and courses on data mining/predictive analytics. *Fundamentals of Predictive Analytics with JMP®* provides the technical knowledge and problem-solving skills needed to perform real data multivariate analysis. Using JMP® 10 and JMP® Pro, this book offers new and enhanced resources, including an add-in to Microsoft Excel, Graph Builder and data mining capabilities.

Written for students in undergraduate and graduate statistics courses, this book first teaches students to recognize when it is appropriate to use the tool, to understand what variables and data are required, and to know what the results might be. Second, it teaches them how to interpret the results, followed by step-by-step instructions on how and where to perform and evaluate the analysis in JMP.

With the new emphasis on business intelligence, business analytics and predictive analytics, this book is invaluable to everyone who needs to expand their knowledge of statistics and apply real problem-solving analysis.

JMP® Start Statistics: A Guide to Statistics and Data Analysis Using JMP®, Fifth Edition

(March 2012)
By John Sall, Ann Lehman, Mia Stephens, and Lee Creighton

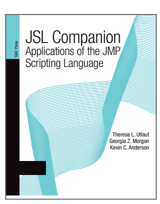


This new edition is a mix of software manual and statistics text that provides hands-on tutorials with just the right amount of conceptual and motivational material to illustrate how to use the intuitive JMP interface for data analysis. Each chapter features concept-specific tutorials, examples, review of concepts, step-by-step illustrations and exercises.

New features include the enhanced ability to manage a JMP session by easily tracking open and recently opened JMP tables; scripts, analyses, JMP projects, and other files; and vastly expanded tools for instructors to demonstrate statistical concepts.

JSL Companion: Applications of the JMP® Scripting Language

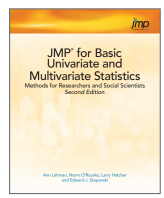
(December 2011)
By Theresa Utlaut, Georgia Morgan, and Kevin Anderson



This JSL companion provides novice scripters with a resource that helps them go beyond the basics of the JMP Scripting Language (JSL) and serves as a companion to writing applications. Taking a task-oriented approach rather than focusing on showing the syntax, the authors help users with much more than the basic task of capturing a script. The book starts with an introduction that is suitable for someone who is just beginning to learn JSL. It quickly moves to importing and saving data, working with variables, modifying data tables and working with JMP data structures (lists, matrices and associative arrays). Later chapters deal with JMP output, communicating with users, and customizing display.

JMP® for Basic Univariate and Multivariate Statistics: Methods for Researchers and Social Scientists

(March 2013)
By Ann Lehman, Norm O'Rourke, Larry Hatcher, and Edward Sepanski



Learn how to manage JMP data and perform the statistical analyses most commonly used in research in the social sciences and other fields. Updated for JMP® 10 and including new features on the statistical platforms, this book offers clearly written instructions to guide you through the basic concepts of research and data analysis. Step by step, you'll see how to obtain descriptive and inferential statistics, perform a wide range of JMP analyses and interpret the results, and summarize results clearly in a way suitable for publication.

This user-friendly book introduces researchers and students of the social sciences to JMP and to elementary statistical procedures, while the more advanced statistical procedures that are presented make it an invaluable reference guide for experienced researchers as well.

For more information about SAS Press and a complete list of available JMP and SAS books, go to **support.sas.com/publishing**.

About JMPer Cable

A Technical Publication for JMP® Users

Issue 28 Summer 2013

Editor: Ann Lehman

Production: Diana Witt

Contributors:
Mark Bailey
Melanie Drake
Bradley Jones
Sheila Loring
Phillip Ramsey
John Sall

Subscribe Today

If you enjoy receiving JMPer Cable in the mail, subscribe today! In the future, only subscribers will receive print copies of JMPer Cable. To avoid missing even one valuable issue, subscribe online at **jmp.com/jmpercable**.

You can also find JMPer Cable current and back issues, along with their data tables and scripts, online: **jmp.com/jmpercable**.

For additional user resources, go to **jmp.com/community**.

To order additional licenses of JMP, to learn about the most recent release, or to inquire about content of this JMPer Cable issue, call **1-877-594-6567**.

THE POWER TO KNOW.®

SAS Institute Inc. World Headquarters +1 919 677 8000

JMP is a software solution from SAS. To learn more about SAS, visit **sas.com**

For JMP sales in the US and Canada, call 877 594 6567 or go to **jmp.com**