

Take the Struggle Out of Data Wrangling With JMP[®]

Robert Carver

WHITE PAPER

Table of Contents

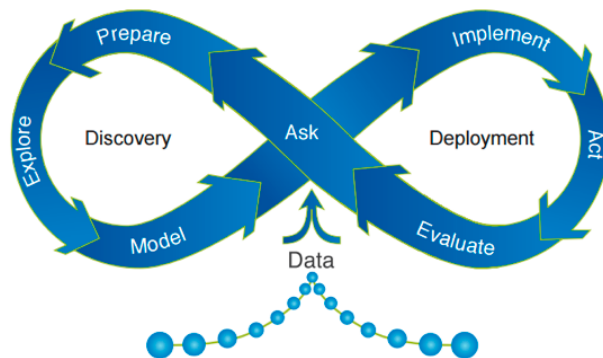
Introduction.....	1
A continuous process in an organizational context	1
Essential database operations.....	2
Detecting and addressing common data hygiene problems	4
On the hunt for dirty data: Finding and fixing common problems	5
Messy data	5
Outliers	8
Missing data.....	16
Strategies for dealing with missing data	20
Imputation concepts and tools.....	21
Special Considerations for Time Series.....	23
Patterns	24
Data requirements driven by analytic approach.....	25
What shape is a data table?	26
Stacking wide data	26
Feature engineering: Common issues and appropriate strategies ..	28
Distribution of observations	30
High dimensionality: Abundance of columns	31
High dimensionality: Abundance of rows.....	32
Date and time-related issues.....	33
Some date functions: Extracting parts	34
Concluding thoughts.....	34
References.....	35

Introduction

Although reliable estimates are hard to come by, there seems to be consensus that data preparation consumes roughly 80% of the time required for a statistical project (Press, 2016). This paper¹ provides an introduction to the unglamorous, time-consuming, laborious, and sometimes dreaded “dirty work” of statistical investigations – variously known as data wrangling, data cleaning, janitor work (Lohr, 2014) or simply data management. JMP users can perform wrangling operations within one software environment, without having to hop off and onto different platforms.

A continuous process in an organizational context

Successful analyses require a disciplined process like the one shown here (SAS, 2016) wherein a question gives rise to fact-gathering, analysis and implementation of a solution, which in turn is monitored and gives rise to further questions. Though process is often portrayed as circular, this schematic has the added appeal of connoting an infinite loop where data plays a central role. In this image, though, the step identified as “Prepare” is not drawn remotely to scale. Also, in any phase we may need to step back to an earlier phase. This paper will deal mainly with the Prepare and Explore phases.



Though this paper focuses on data preparation, there are four other themes that we'll keep in mind:

- **Data projects occur within a work context.** The questions that drive any study arise from that context and in turn affect the scope and nature of the data preparation required. Successful analysis requires domain knowledge and an understanding of the nature and consequences of the issues under study.
- **Workflow efficiency is valuable.** Because data preparation is so time-consuming and varies from project to project, it is vital to establish and follow procedures that are efficient. JMP supports every stage of the process.

¹ Portions of this paper are adapted from my book *Preparing Data for Analysis With JMP*, SAS Press, 2017.

- **Reproducibility contributes to workflow efficiency.** Some projects are one-time, unique tasks, but many occur repeatedly. We need to preserve a complete record of precisely how analysts transform raw data into the data used.
- **Your mind is your most valuable analytical software.** As powerful and efficient as JMP is, it cannot determine if you and your team have collected data about the right variables to answer your questions or whether your models make sense within the work context. JMP will certainly make data wrangling more efficient, but you must bring domain knowledge and critical judgment to the process.

Throughout the paper, we'll assume that readers are acquainted with the JMP menu system and know how to access or create a JMP data table. For coverage of these essential topics, see the "Discovering JMP" chapter in the *JMP Documentation Library* (2019). This paper covers these topics:

- Essential database operations.
- Detecting and addressing common data hygiene problems.
 - o Messy data: Incorrect modeling types, inaccurate or obsolete data.
 - o Outliers and other dubious observations.
 - o Missing observations.
 - o Suspicious patterns.
- Issues driven by the analytic approach.
 - o Data requirements for different platforms.
 - o Reshaping a data table.
 - o Feature engineering: Transformations and data reduction.
 - o Special consideration for time and date data.
- Concluding thoughts.

Essential database operations

Many statistical and analytics projects call for combining data from multiple sources, selecting columns, filtering rows and other tasks ordinarily requiring database software. JMP performs such operations via dialogs, without the need to write SQL code.

In addition to locating all the variables that might be relevant, analysts often confront the need to reconcile disparities across data sources. Nonstandard abbreviations or coding schemes, differing representations of times and dates, varying units of measurement abound. Before all the data can be assembled into a single well-organized table suitable for analysis, the differences need to be ironed out.

JMP can import data from a large variety of software formats, as well as directly from the internet or databases. In version 15, beyond reading data from Excel, SPSS Minitab and other statistical packages, it is easy to import directly from a URL or from a table within a PDF. In this brief section, we'll pick up at the point where you may have a few JMP data tables and want to build a data set specifically for your current project.

If the columns share at least one primary key column, there are three fundamental approaches available within JMP:

- 1) Use **Tables > Join** to perform a standard join operation, selecting the desired columns and placing them into a new data table or adding them to the main table.
- 2) Use **Query Builder** to do the same. Query Builder supports multiple table joins and greater flexibility in specifying conditions as you build an SQL query right from JMP.
- 3) Designate columns with **Link ID** or **Link Reference** column attributes to enable virtual joins that make columns from a table available to an analysis platform called upon from a main table. This saves memory and storage by not creating a third large combined table out of two tables.

Because **Query Builder** has such extensive functionality, here is a short example drawn from my 2016 JMP Discovery Summit presentation. The presentation described a project to understand the variation in the number of Olympic medals awarded to different countries, using available demographic data from public sources. The data wrangling required assembling data from four sources.

Query Name: Olympics Query Data Source: JMP

Available Columns:

- t2.GAUL
- t2.IOC
- t2.ISO4217-cur
- t2.ISO4217-cur
- t2.ISO4217-cur
- t2.ISO4217-cur
- t2.is_independe
- t2.Capital
- t2.Continent
- t2.TLD
- t2.Languages
- t2.Geoname ID
- t2.EDGAR
- t3.NOC
- t3.Edition
- t3.TotMedals
- t3.N(Medal, Me
- t3.N(Medal, Wc
- t1.Code
- t1.Flag

Included Columns:

Variable Name	JMP Name	Format	Aggregation
t4.Country Name	Country Name		None
t4.Country Code	Country Code		None
t4.Year	Year	Best	None
t1.Code	Code		None
t1.Flag	Flag		None
t3.TotMedals	TotMedals	Fixed Dec	None
t3.N(Medal, Men)	N(Medal, Men)	Best	None
t3.N(Medal, Women)	N(Medal, Women)	Best	None

Query Preview:

	Country Name	Country Code	Year	Code	Flag	TotM
1	Afghanistan	AFG	1960	AFG		
2	Afghanistan	AFG	1964	AFG		
3	Afghanistan	AFG	1968	AFG		
4	Afghanistan	AFG	1972	AFG		
5	Afghanistan	AFG	1976	AFG		
6	Afghanistan	AFG	1980	AFG		
7	Afghanistan	AFG	1984	AFG		
8	Afghanistan	AFG	1988	AFG		
9	Afghanistan	AFG	1992	AFG		

Order By:

Run Query Save Save As... Close Help

Within **Query Builder**, one chooses the source tables, the columns that link the various tables, and selects columns for the final data table. After identifying the tables and specifying the join conditions, one sees a list of open tables, available columns – each with prefix like t1, t2, etc., to indicate the source table – and basic information about the columns selected thus far.

Note also that one can specify formats, aggregation criteria, filters and order by (sort) sequences. JMP provides a preview of the resulting query table. This dialog composes an SQL query, viewable in the SQL tab, and allows the user to save the SQL code for future replication and documentation.

Detecting and addressing common data hygiene problems

Every project is different, and each data set will have its own hygiene challenges. That said, we can categorize typical issues broadly, and we can use those categories to anticipate and diagnose problems. Kandel et al. (2011) suggest that the goal of data preparation is to make data usable, credible and useful. Usable refers to getting them into a form that is suitable for the intended analysis tools. JMP analysis platforms require data to be in columns and rows, with proper data types and modeling types. Credible data are representative of the target process or population for the purposes of the study. Finally, useful data are necessarily usable and credible and must also be “responsive to one’s inquiry” (272).

These criteria direct attention to questions about whether a data table contains acceptable columns and rows. Do the variables (columns) adequately capture the model constructs to serve the current purpose? Do the observations (rows) suitably represent the subjects being investigated or modeled? If a study is about organisms, do the rows represent organisms? Was there any systematic bias – intended or otherwise – in sampling the data? These substantive questions speak to the analyst’s domain knowledge and understanding of the nature of the inquiry.

While Kandel’s goals set the stage, data preparation deals in specifics. Kim et al. (2003) have offered a comprehensive taxonomy of dirty data, including these common obstacles:

- **Messy data or data errors:** Are there individual values that are simply incorrect? Typographical errors are common when data were collected or digitized manually. Sensors can malfunction and record impossible or implausible values. Free-form survey data are plagued with inconsistent spelling and abbreviations. Some such errors may be difficult or impossible to detect at scale, but the initial investigation should seek them out. Some columns may contain correct values but carry the wrong data or modeling types.

- **Outliers:** Are there extreme observations that might be erroneous or might signal an observation that does not come from the target population? Some continuous distributions are severely skewed and may include extreme values that are accurate and hence should be retained, but that can potentially distort a visualization or the results of the planned data analysis.
- **Missing data:** After assembling a data table, one finds empty cells. How prevalent are they? Do they occur randomly or with meaningful patterns? If a model requires complete cases for all variables in the model, we can lose entire rows if there are any gaps across the variables. Might we infer meaning from a blank cell? Might we sometimes be able to impute or estimate a suitable replacement value for a blank and thereby retain the observation?
- **Suspicious patterns:** Many studies seek out meaningful patterns across columns of data. In time series data, we typically investigate patterns across rows. But sometimes, we find patterns that are suspicious – perhaps too good to be true. Such patterns are worrisome because they can indicate fraud or human error. Orel (1968) chronicles the controversial history of Gregor Mendel's pea data; more recently, there has been considerable attention given to misconduct and mistakes in medical studies (Adam, 2019).

Imagine downloading price data for a publicly traded stock and finding that there is no variation at all. Or exploring several columns and finding near-perfect correlations. Are the columns alternative measures of a single dimension? Was one column computed from others by the creator of the data table? Some strong patterns are legitimate, but the analyst may want to flag and investigate suspicious patterns.

On the hunt for dirty data: Finding and fixing common problems

Before one can clean a dirty data table, one needs to find the dirt. Veteran JMP users are probably familiar with several approaches to data exploration to find errors and unusual observations. In this section, we demonstrate several sensible ways to start. These examples are intended to be illustrative as opposed to comprehensive, because the quirks of any individual data table are so varied. This section illustrates several classic problems and demonstrates JMP platforms that speed the work of data wrangling.

Messy data

As a first example of messy data, consider the **Color Preference Survey** data from the **Sample Data Library** in JMP. The data table contains responses from 80 individuals, simply inquiring about favorite colors. Responses to the survey questions are character data, and the first three might be considered numeric, depending on the planned analysis.

Response ID	Time Started	Date Submitted	What is your gender?	How old are you?	What is your favorite color? (select one)	Red: W do yo
1	2018/11/16 11:17:08 AM	Nov 16, 2018 11:17:08 am	Male	50 - 59	Blue	Red
2	2018/11/16 11:17:19 AM	Nov 16, 2018 11:17:20 am	Female	40 - 49	Purple	Red
3	2018/11/16 11:18:44 AM	Nov 16, 2018 11:18:45 am	Female	10 - 19	Green	Red
4	2018/11/16 11:33:32 AM	Nov 16, 2018 11:33:33 am	Female	30 - 39	None of the above	Red
5	2018/11/16 6:27:37 PM	Nov 16, 2018 6:27:37 pm	Male	2 others	3 others	Red
6	2018/11/18 6:57:19 PM	Nov 18, 2018 6:57:20 pm	Male	50 - 59	Blue	Red
7	2018/11/18 6:58:05 PM	Nov 18, 2018 6:58:05 pm	Male	30 - 39	Blue	Red
8	2018/11/18 7:04:21 PM	Nov 18, 2018 7:04:21 pm	Female	20 - 29	Red	Red
9	2018/11/18 7:13:34 PM	Nov 18, 2018 7:13:34 pm	Female	40 - 49	Blue	Red
10	2018/11/18 8:49:59 PM	Nov 18, 2018 8:49:59 pm	Female	50 - 59	Yellow	Red
11	2018/11/18 10:18:05 PM	Nov 18, 2018 10:18:05 pm	Male	10 - 19	Green	Red
12	2018/11/18 10:20:50 PM	Nov 18, 2018 10:20:50 pm	Male	50 - 59	Green	Red
13	2018/11/18 10:32:40 PM	Nov 18, 2018 10:32:40 pm	Male	40 - 49	Blue	Red
14	2018/11/18 12:48:59 AM	Nov 19, 2018 12:48:59 am	Male	40 - 49	Blue	Red
15	2018/11/19 12:53:25 AM	Nov 19, 2018 12:53:25 am	Female	10 - 19	Yellow	Red
16	2018/11/19 3:18:04 AM	Nov 19, 2018 3:18:04 am	Male	50 - 59	Blue	Red
17	2018/11/19 4:13:07 AM	Nov 19, 2018 4:13:07 am	Male	60+	Blue	Red
18	2018/11/19 4:32:54 AM	Nov 19, 2018 4:32:54 am	Male	50 - 59	Blue	Red
19	2018/11/19 7:15:01 AM	Nov 19, 2018 7:15:01 am	Female	<10	None of the above	Red
20	2018/11/19 7:29:28 AM	Nov 19, 2018 7:29:28 am	Male	50 - 59	Blue	Red
21	2018/11/19 7:56:07 AM	Nov 19, 2018 7:56:07 am	Female	40 - 49	None of the above	Red

The modeling type icons in the **Columns** pane are a good place to begin. Most of the columns are nominal, which is appropriate, and a few are multiple response columns. We see that **Response ID** is coded as continuous. Since it is just a sequential identifier, it might be better changed to the Ordinal modeling type. One can do this by right-clicking the icon and making the change right in place.

The next two columns are both time stamps presumably collected by the survey software, but they have different types and formats. This is also evident in the header graphs (new in JMP 15) at the top of each column. When we open the **Column Info** for both, we see what is going on.

Time Started in table 'Color Preference Survey'

Column Name

Time Started

☐ Lock

Data Type

Numeric

Modeling Type

Continuous

Format

y/m/d h:m:s

Width

22

Dec

0

Input Format

y/m/d h:m:s

Column Properties

Date Submitted in table 'Color Preference Survey'

Column Name

Date Submitted

☐ Lock

Data Type

Character

Modeling Type

Nominal

Column Properties

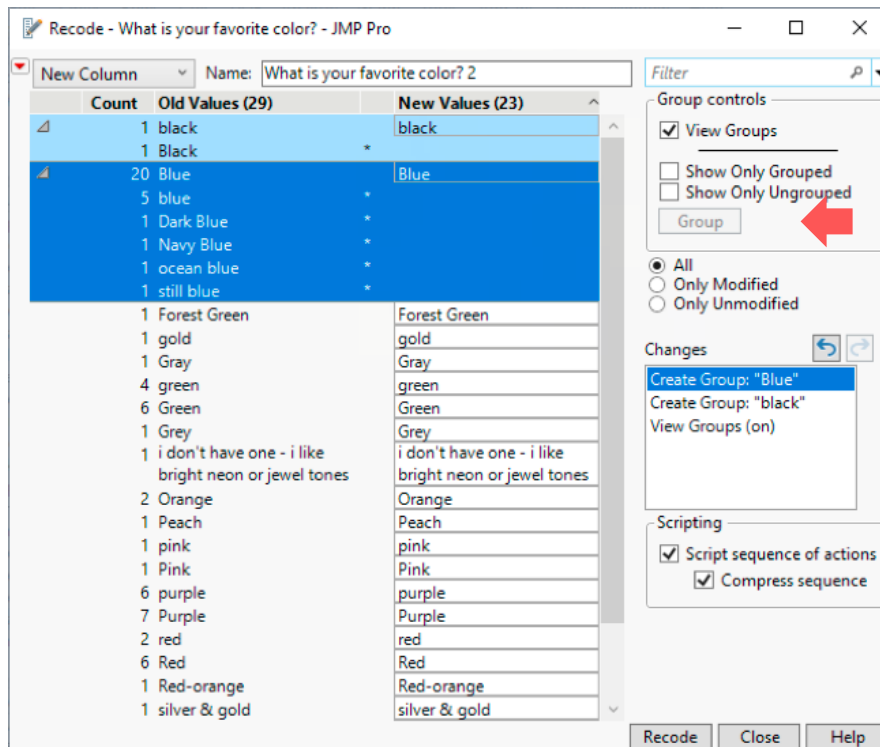
Time Started is numeric, its modeling type is continuous, and it uses one of the date and time formats. In contrast, **Date Submitted** is a nominal character variable. If these columns have no role in any anticipated analysis, these differences are irrelevant. For any analytic purpose, though, the differences have implications. Further, if one wanted to compute the amount of time each respondent spent completing the survey, both columns would need to be numeric to perform the calculation.

JMP provides several character and date extraction functions that allow for straightforward conversion of a character string to a data and time column, should that serve the analyst's purpose.

Another example of messy data in this table is the next-to-last column called **What is your favorite color?** This is a free-response question on the survey, and respondents typed in their favorite colors.

What is your favorite color?	
Blue	
Purple	
Green	
purple	
Red	
24 others	
Purple	
Red	
ocean blue	
black	
Blue	
Navy Blue	
Blue	
Orange	
still blue	
Yellow	
Green	
Green	
blue	
Blue	

The header graph shows bars representing the frequency of five colors, and notes that there are 24 more colors named in the column. Closer inspection shows that two of the colors are Purple and purple. Glancing down the first 14 rows of the column, we find seven varieties and spellings of blue. A bar chart of the column would reveal all 29 colors. Again, depending on the goals of an analysis, we might want to leave these as the users entered them, or convert them all to "Blue." For the latter purpose, the **Recode** platform in the **Cols** menu is the way to go. Here we see a **Recode** dialog in progress, using the **Group** function to recode specified old values into new values. **Recode** provides the option of changing the data in place or to a new column. Best practices in data wrangling strongly lean in the direction of conserving the data in its original form and placing revised values in a new column. In this way, one can always go back.



Note also the **Scripting** region of this dialog. JMP will preserve the sequence of changes made during the recoding, and you can save the script right to the data table by using a red triangle option. In this way, we have a record of changes made to the data. This is our first encounter with reproducibility.

Notice also that there are a few entries here that will require the judgment of the project team or the client. Is “red-orange” a useful color category? Or does it belong with Red? Or Orange? The software gives us a way to implement the choice, but the choice is on us.

Outliers

To illustrate the next few common problems, we’ll use some data from the World Bank’s collection of World Development Indicators (WDI). The WDI DataBank (<https://databank.worldbank.org/reports.aspx?source=world-development-indicators>) is a public use database of demographic and economic time series for the nations of the world. Our data begins in 1990 and includes 27 measures in addition to country and year identifiers. The table has 32 columns and 6,235 rows (215 countries x 29 years). The World Bank collates data provided by the nations themselves to various international agencies. To begin, we’ll examine the first several continuous columns. In a full study, we’d want to investigate all columns individually or in meaningful groups, but the goal here is to demonstrate the approach and methodology.

WDI - JMP Pro

File Edit Tables Rows Cols DOE Analyze Graph Tools Add-Ins View Window Help

Source: World Bank, World Dev
 Source
 Subset Script
 Subset Script 2
 Distribution
 Distribution ...co2_kgperPPP

Columns (36/0)
 Country Code
 Country Name
 Region
 Income Group *
 Year
 air_pass
 battle_deaths
 broadband
 cell
 co2_kgperPPP
 co2_mt_pc
 food
 gdp_pc
 gdp_pc_lcu
 gni
 gni_pc
 gni_pc_lcu
 gni_ppp
 h2o_safe
 h1h1_exp_pcap

Country Code	Country Name	Region	Income Group	Year	air_pass	battle_deaths	broadband	cell	co2_kgperPPP	co2
25	ABW	Aruba	Latin America & ...	High income	2014	*	18.49800087	134.5922251	0.2200241355	8.40
26	ABW	Aruba	Latin America & ...	High income	2015	*	18.20932454	135.13384	*	*
27	ABW	Aruba	Latin America & ...	High income	2016	*	*	*	*	*
28	ABW	Aruba	Latin America & ...	High income	2017	*	*	*	*	*
29	ABW	Aruba	Latin America & ...	High income	2018	*	*	*	*	*
30	AFG	Afghanistan	South Asia	Low income	1990	241400	1478	*	0	0.21
31	AFG	Afghanistan	South Asia	Low income	1991	212300	3302	*	0	0.18
32	AFG	Afghanistan	South Asia	Low income	1992	212300	4276	*	0	0.09
33	AFG	Afghanistan	South Asia	Low income	1993	197000	4071	*	0	0.0
34	AFG	Afghanistan	South Asia	Low income	1994	238400	8937	*	0	0.08
35	AFG	Afghanistan	South Asia	Low income	1995	250400	5499	*	0	0.07
36	AFG	Afghanistan	South Asia	Low income	1996	255600	3177	*	0	0.06
37	AFG	Afghanistan	South Asia	Low income	1997	89600	6396	*	0	0.05
38	AFG	Afghanistan	South Asia	Low income	1998	52700	6256	*	0	0.1
39	AFG	Afghanistan	South Asia	Low income	1999	140200	4629	*	0	0.04
40	AFG	Afghanistan	South Asia	Low income	2000	149705	5235	*	0	0.1
41	AFG	Afghanistan	South Asia	Low income	2001	*	5055	*	0	0.03
42	AFG	Afghanistan	South Asia	Low income	2002	*	890	0.113740162	0.0566504324	0.04
43	AFG	Afghanistan	South Asia	Low income	2003	*	687	0.067120278	0.0569777631	0.05
44	AFG	Afghanistan	South Asia	Low income	2004	*	715	0.000829232	2.487667492	0.0434410878
45	AFG	Afghanistan	South Asia	Low income	2005	*	1595	0.000877515	4.7864517	0.0528662233
46	AFG	Afghanistan	South Asia	Low income	2006	*	4750	0.00193099	9.733604444	0.0605396092
47	AFG	Afghanistan	South Asia	Low income	2007	*	6906	0.001878513	17.53816162	0.07137879
48	AFG	Afghanistan	South Asia	Low income	2008	*	5552	0.001831902	28.94006019	0.1246189965
49	AFG	Afghanistan	South Asia	Low income	2009	*	6341	0.003570876	37.49420045	0.1639766479
50	AFG	Afghanistan	South Asia	Low income	2010	1999127	6864	0.005207761	35.46776644	0.1771048975
51	AFG	Afghanistan	South Asia	Low income	2011	2279341.1795	7405	46.44405817	0.2498358821	0.41
52	AFG	Afghanistan	South Asia	Low income	2012	1737962.1264	7719	0.004886478	49.972753	0.1910314926
53	AFG	Afghanistan	South Asia	Low income	2013	2044188	8056	0.004727136	52.96647314	0.1655358908
54	AFG	Afghanistan	South Asia	Low income	2014	2309428	12285	0.004579031	56.19133269	0.1549103548
55	AFG	Afghanistan	South Asia	Low income	2015	1929907	17273	0.020947642	58.42052823	*

Rows
 All rows 6,235
 Selected 0
 Excluded 0
 Hidden 0
 Labelled 0

Before delving into outlier detection, let's briefly deal with another common messy data issue: reconciling value sequences for an ordinal character variable. With a curated database like the WDI, data errors are uncommon. However, when importing the WDI data into JMP, **Income Group** was assigned a data type of Character and modeling type of Nominal. While the data type is correct, for most purposes it is preferable to treat **Income Group** as ordinal, since the World Bank classifies countries in four groups from low income to high income. JMP software's default ordering is alphabetical, but one can specify a **Value Order** (signified by a bold asterisk * next to the column name), which is adjusted with the up and down arrows in the **Column Info** dialog.

'Income Group' in table 'WDI'

Column Name: Income Group

☐ Lock

Data Type: Character

Modeling Type: Ordinal

Column Properties

Value Order

Specify the order of data items and/or pick rules that determine the order.

☒ Custom Order

Custom Order

Low income
 Lower middle income
 Upper middle income
 High income

Sorted Order

Sorted Order (0)

Rules for sorted items

☐ Row Order Levels
☒ Common Order
☒ Numerical Order

Enter value

☒ Use Locale Comparisons for Characters

OK
 Cancel
 Apply
 Help

The shape of this data table is stacked (sometimes called tall or narrow), meaning that we have repeated observations of each variable stacked sequentially and **Year** occupies a column. An alternative arrangement would be split or wide: There would be a separate column for each variable for each year. In a wide table, for example, we might have the annual number of airline passengers in 29 distinct columns such as **air_pass_1990**, **air_pass_1991**, and so on.

Above, we see that the header graphs provide a quick view of the shape and range of continuous columns as well as the levels of categorical columns. The first five continuous columns are:

- **air_pass**: number of passengers carried by air transport, both domestic and international.
- **battle_deaths**: number of battle-related deaths, combatant and civilian.
- **broadband**: fixed broadband subscriptions per 100 people.
- **cell**: cellular subscriptions per 100 people.
- **co2_kgperPPP**: CO2 emissions (kg per purchasing power parity \$ of GDP).

The **Columns Viewer** platform in the **Cols** menu gives an overview of several columns at once, and is an efficient way to screen for missing data, outliers and data type irregularities.

WDI (6235 rows, 36 columns)

Columns View Selector

Select Columns

36 Columns

Enter column name

Country Code

Country Name

Region

Income Group

Year

air_pass

battle_deaths

broadband

cell

co2_kgperPPP

co2_mt_pc

food

gdp_pc

gdp_pc_lcu

gini

gini_pc

gini_pc_lcu

Clear Select

Subset

Show Summary

Show Quantiles

Find Columns with Properties

Summary Statistics

5 Columns

Clear Select

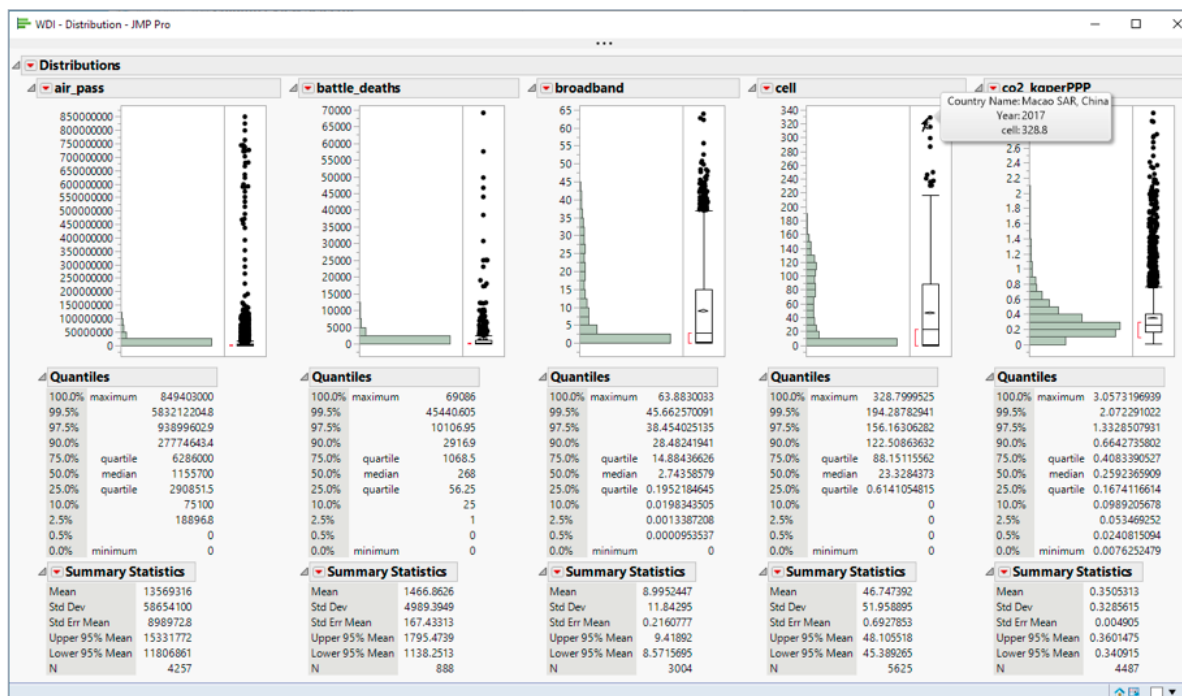
Distribution

Columns	N	N Missing	Min	Max	Mean	Std Dev
air_pass	4257	1978	0	849403000	1356931637	58654100204
battle_deaths	888	5347	0	69086	1466.8626126	4989.3949143
broadband	3004	3231	0	63.8830033	8.9952447259	11.842950299
cell	5625	610	0	328.7999525	46.74739169	51.958895127
co2_kgperPPP	4487	1748	0.0076252479	3.0573196939	0.3505312536	0.3285615247

It is clear at once that there are many missing observations, and for **battle_deaths** and **broadband**, more cells are empty than are present. Knowing that **cell** is the number of mobile subscriptions per 100 people, we might question the maximum value of approximately 328.8.

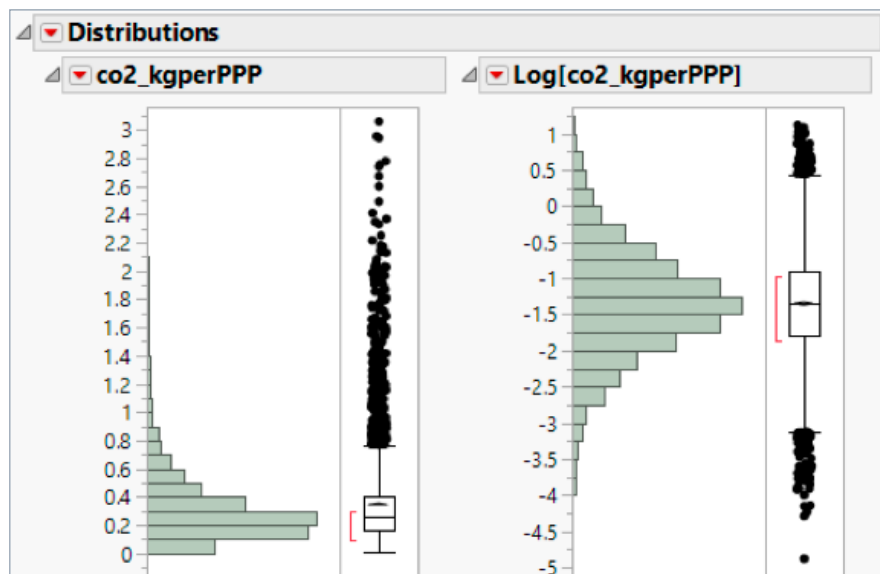
The high degree of missingness for **battle_deaths** is good news: During the 29 years of data, most countries had no battle deaths to report. On the other hand, in a study of climate change, missing data for **co2_kgperPPP**, a measure of carbon dioxide emissions, could impede analysis; one might want to ask when the time series began and whether some regions of the world have been systematically under-reporting. We'll come back to such missing data questions in the next section.

By clicking on the **Distribution** button, we open the distribution reports for the five columns.



All five of the distributions are strongly right-skewed and all have numerous outliers. This suggests that outliers are to be expected and probably are not, say, data entry errors. In the fourth histogram, **cell**, the hover label shows that the oddly high number of subscriptions per 100 persons was for Macao in 2017. By brushing the few outliers in the boxplot, we would find that all of these extreme observations come from Macao and Hong Kong, where cell phone usage is particularly popular.

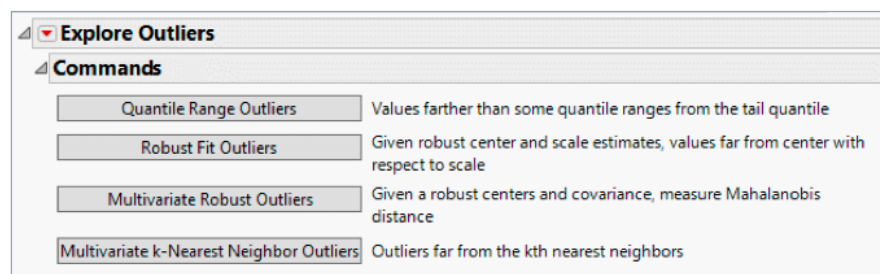
For many inferential modeling techniques, one might prefer to avoid working with such strongly skewed data. There are data transformations that can help ameliorate the issue. For example, taking the natural log of a variable like **co2_kgperPPP** creates a quite symmetric variable with far fewer outlying values than the original column. In a later section, we'll discuss several ways to accomplish data transformations.



Beyond visual inspection in the **Columns Viewer** and **Distribution** reports, the **Analyze > Screening > Explore Outliers** platform can be very helpful in deciding how to address outlier issues for continuous columns. We'll continue to illustrate using the same five columns.

Since there are multiple working definitions of "outlier," JMP offers four basic approaches to identifying extreme and unusual values. These are fully documented in *Fitting Linear Models* within the *JMP Documentation Library* (2019), found on the **Help** menu in JMP. We briefly summarize the distinctions among the four approaches here, and then illustrate use of the options.

Note that each approach offers several user-controlled options. Here the goal is to describe the four methods, drawing analogies to familiar contexts. The first two options use univariate measures on selected columns, and the second two are intended for multiple columns.



- **Quantile Range Outliers:** This method defines outliers as being points lying beyond a multiple of an interquantile range specified by the user. The default values are 0.1 and 3, meaning that points more than three times the difference between the 90th and 10th percentiles are considered outliers.
- **Robust Fit Outliers:** This method computes robust measures of center and dispersion for the variable. It then defines outliers as those lying more than K times the robust dispersion measure from the robust center.
- **Multivariate Robust Outliers:** This method and the next method refer to outliers from multi-column relationships for the columns selected. It calculates the Mahalanobis distances between each point to the robust center of a multivariate normal distribution, relative to the estimated correlations. In a bivariate context, this is akin to looking for points lying “far” from a linear pattern.
- **Multivariate k-Nearest Neighbor Outliers:** Finally, this method takes a fundamentally different approach. It computes the Euclidean distance of each individual point from its Kth nearest neighbor, where K is supplied by the user. Hence, rather than expressing distance from a center, distance is relative to the nearest of a subset of points.

Of course, once outlying values have been identified, there is the question of what should be done, and those judgments require domain knowledge. Depending on the context and the process by which the data were generated and captured, the most appropriate approach will vary. In some instances, outliers are caused by measurement error, instrument failure, or human error in recording or transcribing the data. In others, outliers are best handled by a variable transformation such as taking a logarithm.

Here is a typical report after selecting **Quantile Range Outliers**. The user may change the tail quantile and multiplier specifications and decide to color outliers and/or exclude them from further analysis. For this discussion, let's focus on the bottom of the report.

Explore Outliers

Commands

Quantile Range Outliers

Outliers are values Q times the interquantile range past the lower and upper quantiles.

Tail Quantile

0.1

Q

3

☐ Restrict search to integers

☐ Show only columns with outliers

Rescan

Close

Select Rows

Color Cells

Exclude Rows

Color Rows

Add to Missing Value Codes

Change to Missing

Select columns and choose an action.

Column	10% Quantile	90% Quantile	Low Threshold	High Threshold	Number of Outliers	Outliers (Count)
air_pass	75100	2.78e+7	-8.3e+7	1.11e+8	62	111598546 112353099 114128000 115419921 115928738 116713587 116847033 1
battle_deaths	25	29169	-86507	115926	19	12054 12149 12227 12285 17203 17273 17981 18950 23038 24950 25000 25050 307
broadband	0.01983	28.4824	-85.368	113.87	0	
cell	0	122.509	-367.53	490.035	0	
co2_kgperPPP	0.09892	0.66427	-1.5971	2.36033	12	2.368216 2.3685549 2.4093136 2.4910621 2.5988959 2.6707496 2.7387848 2.753254

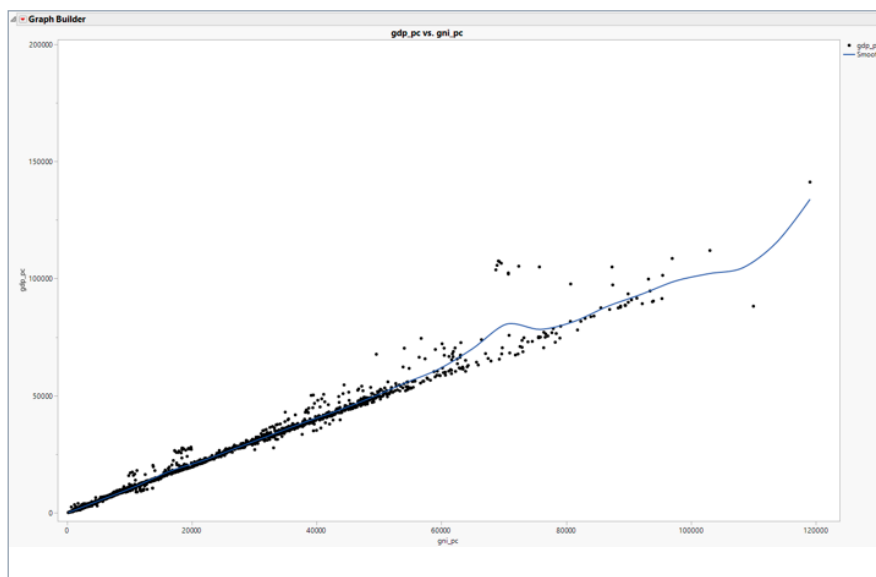
Recall that there are more than 6,200 observations in the entire table. **air_pass** is the column with the largest number of extreme values, with 62 cells lying beyond the threshold limits, defined here as less than three times the 10th percentile or greater than three times the 90th percentile. The first several outlying values are listed in the last column of the report.

Earlier we investigated the maximum value of the cell subscribers column. In the boxplot, there were numerous upper tail points more than 1.5 times the interquartile range. However, by the quantile range criteria used here, there are no outliers at all. The boxplot uses one widely accepted criterion for identifying outliers. The **Explore Outliers** platform affords us several alternatives to meet the needs of a project or organization.

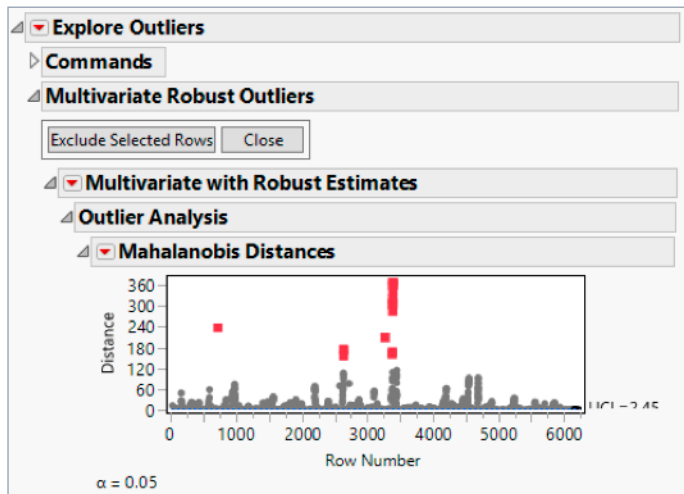
To illustrate the multivariate outlier tools, let's consider two additional columns from the WDI data table:

- **gdp_pc**: Gross Domestic Product, GDP, per capita (constant 2010 US\$, i.e., adjusted for inflation)
- **gni_pc**: Gross National Income, GNI, per capita (constant 2010 US\$)

Both columns measure per capita income, but in different ways. Before exploring outliers, let's look at a scatterplot of the two columns.

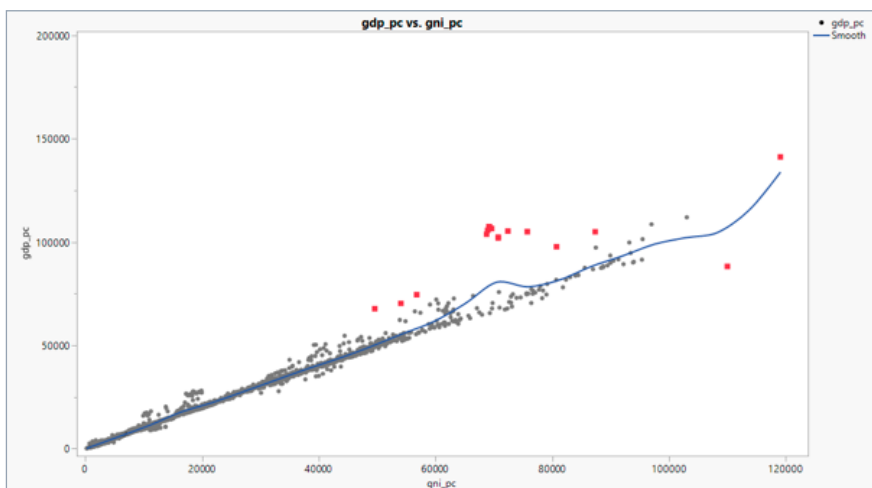


Clearly, there is a strong linear pattern. This makes sense, because both variables measure different aspects of a single thing. However, it is also clear that some points depart from the pattern. Let's see how the **Multivariate Robust Outliers** command identifies outliers.



This platform computes a Mahalanobis distance for each point, essentially measuring how far a point lies from a multivariate distribution. Think of it as analogous to a residual in this two-dimensional case, except that the distance is orthogonal to the linear pattern rather than vertical. In the report, we see the distance for each row in the table. I have changed markers and shapes for those points with distances that appear to be considerably larger than the rest.

When we now look back at the scatterplot, we see which points stand out.



Most of the highlighted points are near the center of the **gdp_pc** distribution, though high in the **gni_pc** distribution. Though we can inspect the scatterplot and find such values visually, this platform not only finds them in a systematic way but offers the option to exclude outliers from further analysis.

Missing data

With real data, missing observations are very common and can profoundly affect the analysis in a project. In this section we first consider some useful concepts about reasons for missing data, then see some more tools to assess the scale of the missing data problem. Finally, we illustrate four basic approaches to ameliorating the problem. At the outset, we should note that the specialized missing data platforms focus on numeric data, but missing categorical data is also potentially problematic.

What do we mean by missing data, and why is it so important? In the most general sense, it simply refers to empty cells within a data table. Sometimes we know or can surmise the reasons that led to empty cells, and sometimes we cannot. Sometimes, the very fact that an individual case has no value for a given column is informative in its own right.

As an introductory example, consider the “Youth Risk Behavior Surveillance System” that generated the data found in the JMP Sample Data table titled **Health Risk Survey**. This iteration of the survey was taken in 2005, and researchers at the US Centers for Disease Control and Prevention interviewed nearly 14,000 9th to 12th grade students asking a large number of questions about health and risky teen behaviors. The most current **User’s Guide** (CDC, 2018) for this sample data describes its methodology as follows:

The National Youth Risk Behavior Survey (YRBS) uses a three-stage cluster sample design to produce a representative sample of 9th through 12th grade students. The target population consisted of all public, Catholic, and other private school students in grades 9 through 12. A weighting factor was applied to each student record to adjust for nonresponse and the oversampling of black and Hispanic students in the sample.

This description points to two kinds of missingness from the outset: Some target populations are over- or under-represented due to the sampling method, due to fact that some people decline to be interviewed, or perhaps due to the fact that they have stopped attending school. Therefore, there are individuals – entire rows of the table – absent from the sample whom researchers would have wanted to include. Additionally, sampled individuals may have elected to skip some questions while answering others. Why skip a question? Perhaps a student is not sure of an answer or has forgotten. Perhaps the student simply didn’t realize he had skipped past a question. Or perhaps answering the question would be risky in some way.

Consider this question on the survey: “Have you carried a weapon to school one or more times in the past 30 days?” This is a Yes or No closed-choice question. Out of 13,760 respondents, 157 left the question blank. We do not know why an individual omitted the question, but one might reasonably speculate that a 17-year-old who had never brought a gun to school would answer “No.”

These kinds of challenges can occur in any type of data collection. Huge data sets with many columns are infamous for their sparsity – an enormous number of columns with mostly blank cells. Why is sparsity a challenge? Sometimes it is about computational inefficiency and storage demands, but for the purposes of this discussion, there are two big statistical issues when many numeric values are missing: bias and unknown variability. Missingness affects both the estimator and standard errors, meaning that inference is jeopardized.

The severity of the problems depends on (1) the amount of missingness relative to the sample size and (2) the mechanisms giving rise to missingness. Similarly, the choice of practical responses to missing-data issues also depend on how and why those pesky blank cells occur in the first place. Even in cases where the reasons for missingness are well understood, it may not be trivial to select the most appropriate approach to data preparation and analysis. When the underlying mechanism is not well understood, it is even more challenging but ultimately up to the analyst to decide how to proceed. There is no purely statistical rationale that is valid in all studies, and we have various options depending on the situation.

Little and Rubin differentiate among three types of missing data (2002; Gelman and Hill, 2006; Penn, 2007; Sterne et al., 2009), which can be understood in the context of the YRBS example.

- **MCAR: Missing Completely at Random.** Blank cells occur essentially without any pattern, and the likelihood of missingness is the same for every case. Observed values provide no information about the missing values. Missingness does not depend on any observed data. A data value is missing because of a transcription or data entry error that might occur for any subject in the study. When it is justifiable to think that data are MCAR, such as when students are unaware of skipping past a question, then complete case analysis is likely to be unbiased.
- **MAR: Missing at Random.** The probability that an observation of Y is missing is a function of observed variables that are completely reported. When data are MAR then analytic techniques that require complete cases may yield biased estimates. For example, if it is plausible to assume that students' propensity to skip a question about drinking behavior depends on other factors such as age, we might consider NAs for the drinking question to be MAR.
- **MNAR: Missing Not at Random.** If the pattern of missingness (not the values themselves, but the fact that they are missing) depends on unobserved data, then empty cells are missing not at random. In other words, "[e]ven after the observed data are taken into account, systematic differences remain between the missing values and the observed values." (Sterne, et al., 2009, p. 157). If, for example, the fact that a student engages in a non-reported illegal activity increases the probability that the student will skip a question about something else, the NAs would be missing not at random. MNAR data sets generally lead to biased estimates. Another version of MNAR would be a student who feels so unsafe at school that he or she doesn't complete the survey.

In some studies, the very fact that some cells are blank carries meaning. As Sall (2013) notes, “when you are studying the data to predict if a loan is bad and the loan applicant leaves his income or his current debts missing, we are probably better to assume that the applicant is being evasive here for a reason.” This would be a case of MNAR – missing not at random. Rather than omit the case or attempt to estimate a plausible income value, we might create a code to indicate that a value is missing and include the case within the analysis.

There are several ways to work around missing data, but the mechanisms that give rise to the blanks in a data set must be understood before undertaking to “fill in the blanks” as it were. Before discussing workarounds, let’s examine ways to detect the severity of the problem.

We already know that the **Columns Viewer** platform counts the number of missing cells in a column, and we have other platforms that go much further. Just below **Explore Outliers** on the **Analyze > Screening** menu is the **Explore Missing** platform. To begin illustrating this feature, let’s return to the WDI data and continue to look at four WDIs related to income. Earlier we examined inflation-adjusted, constant-dollar GDP and GNI per capita. We now add two more columns:

- **gdp_pc_lcu**: GDP per capita (constant local currency)
- **gni_pc_lcu**: GNI per capita (constant local currency)

The initial **Missing Value Report** just shows the count of empty cells in the four columns.

Explore Missing Values

Commands

- Missing Value Report: Number of missing values for each column
- Missing Value Clustering: Hierarchical clustering of rows and columns missingness
- Missing Value Snapshot: Patterns of missing values with graphical map
- Multivariate Normal Imputation: Least squares prediction from the nonmissing variables in each row
- Multivariate SVD Imputation: Imputation for wide problems using a singular value decomposition with the power-method adapted for missing values
- Automated Data Imputation: Automatically selects best dimension for low-rank approximation based on the data and has streaming imputation capabilities

Automated Data Imputation Controls

Missing Columns

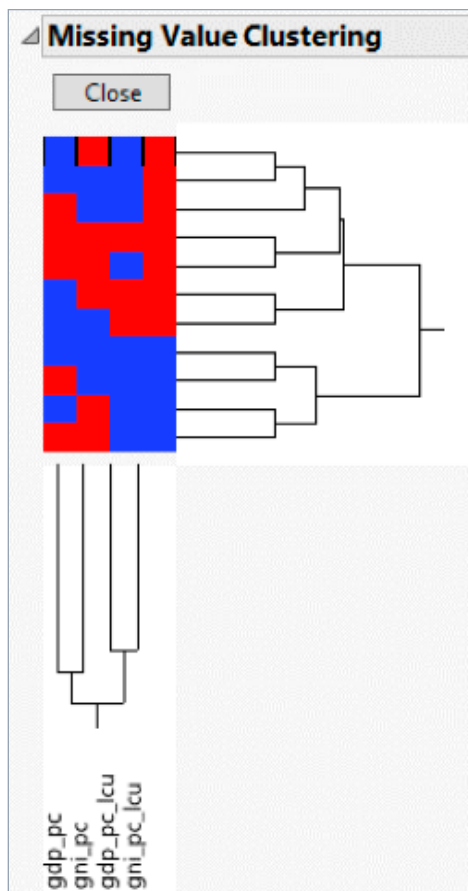
☐ Show only columns with missing

Select columns and choose an action.

Column	Number Missing
gdp_pc	905
gdp_pc_lcu	559
gni_pc	2445
gni_pc_lcu	2351

The platform includes five more commands, two of which visualize the multivariate patterns of missingness, and three impute (infer) plausible values to fill empty cells. We'll come back to imputation after considering the **Missing Values Clustering** command.

Clustering generates a dendrogram, which is a hierarchical graph of missing observations by column and row, and a cell plot. Blue areas show where observations are present and red indicates missing observations. The idea in this plot is to detect groups of columns with similar patterns of blank cells. When you hover over one of the cells in the cell plot, a hover label lists the rows represented. It is particularly valuable if we decide to use one of the automated imputation listed among the commands. These are discussed later.



The **Tables > Missing Data Pattern** platform generates a data table related to the dendrogram, with the additional advantage of including the frequency of each pattern. If we think of the cell plot as a stack of 11 rows of four colored squares, we see that the fourth row from the bottom is entirely blue. In other words, there are some number of rows in the WDI data table that have a complete set of rows for the four income variables. How many? We cannot tell from the cell plot, but the **Missing Data Pattern** table provides the detail.

	Count	Number of columns missing	Patterns	gdp_pc	gdp_pc_lcu	gni_pc	gni_pc_lcu
1	3628	0	0000	0	0	0	0
2	31	1	0001	0	0	0	1
3	96	1	0010	0	0	1	0
4	1567	2	0011	0	0	1	1
5	2	2	0101	0	1	0	1
6	6	3	0111	0	1	1	1
7	128	1	1000	1	0	0	0
8	1	2	1001	1	0	0	1
9	32	2	1010	1	0	1	0
10	193	3	1011	1	0	1	1
11	551	4	1111	1	1	1	1

We see that 3,628 rows are complete, and only 551 are missing all four variables. The **Patterns** column indicates how many columns are missing, and which of the four has blank cells. This can be very useful, for example, in choosing columns to include in a multivariate model.

Strategies for dealing with missing data

There are four basic approaches to the problem, and there is no single strategy that dominates all others. The best approach depends on the goals of the project, the data types involved and the domain knowledge of the analyst.

Some statistical methods require complete cases (rows) of data to produce estimates. For example, linear regression models need values of the response variable and all factors for every row in a data table. Suppose we want to build a model to estimate Y from three X variables. If X3 is missing for one subject and X2 is missing for another, then both subjects are dropped from the analysis, which is likely to bias the results except under certain restrictive assumptions.

In a study where data are being combined from several sources, the consequences of missing data may be different for the ultimate user of the data than for the original constructor of the database (Rubin, 1996). Depending on the study goals, the analyst may be able to sidestep some missing values or use an analytic technique that does not require complete cases. She may be able to impute values to fill in blank cells or use a technique that treats missingness as informative.

Little and Rubin (2002, pp. 19-20) propose four categories of missing-data strategies, and JMP implements all four.

1. Procedures based on completely recorded units. If there are relatively few missing observations, use case-wise deletion and forge ahead. Similarly, if a variable has mostly missing values, omit it from the analysis. This is an acceptable strategy if the missing cases are rare and if one can reasonably assume that cases are missing completely at random. Many analysis platforms drop cases with missing data automatically.

2. Weighting procedures. This is common with survey data where population subgroups appear in the sample disproportionately to their occurrence in the population. Many platforms in JMP anticipate the use of sampling weights, which are often supplied with public-use survey data.
3. Imputation-based procedures. There are several methods described below that fill in blank cells with plausible estimates, and then the analyst applies methods that require complete cases. JMP offers several of the available methods for imputation.
4. Model-based procedures. These approaches involve modeling using the observed data, applying maximum likelihood methods for estimation of parameters and variability.

Because the focus of this paper is wrangling the available data to prepare it for traditional analysis, we'll deal only with imputation-based procedures.

Imputation concepts and tools

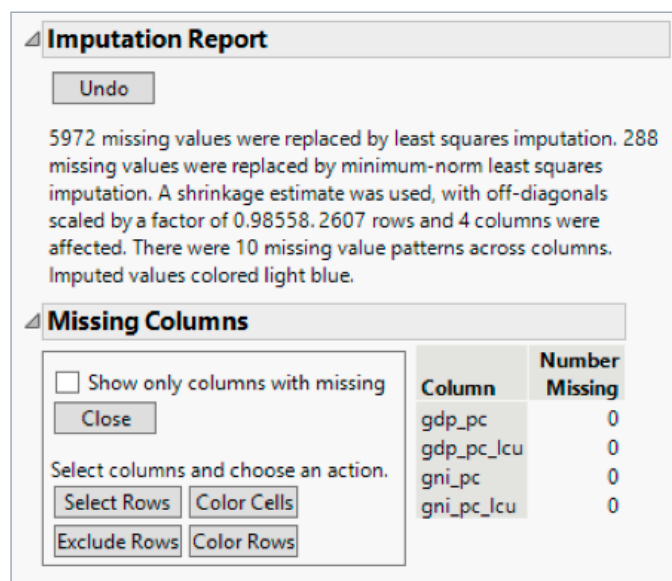
When there are too many missing cells to ignore, it may be possible to use educated estimates of those values so that we have complete cases to work with. The methods used to construct plausible replacement values are known collectively as imputation methods. In this section we look at examples of these approaches to "filling in the blanks."

A word of caution is important here: JMP will make the requisite calculations and replacements but will not make judgments about whether a particular method is logically, theoretically or practically justified in a given study. That responsibility belongs to the analyst, who must have sufficient domain knowledge to make the choice. Imputation has substantial consequences for model-building, and analysis and should be approached with eyes wide open.

Earlier we mentioned the **Health Risk Survey** example, when we might logically infer that a missing answer to a delicate question could be "Yes." As another example, we turn to the **Aircraft Incidents** sample data table. The data provide information about 1,906 US domestic aircraft accidents or other safety-related incidents (e.g., smoke in the cockpit) in 2001. By far, most rows represent accidents. Two of the columns are **Injury Severity** and **Total Fatal Injuries**. There are 1,551 missing cells in the **Total Fatal Injuries** column and none in the **Injury Severity** column. Inspection shows that only flights with fatalities have a value in the second column, and the rest are missing. Clearly it makes sense to replace the blanks with 0's in this column.

The **Explore Missing Values** platform provides three imputation methods, very briefly summarized here. They all require continuous variables. All three methods presume that any missing values are missing at random, and each uses data from other specified columns and rows to estimate the missing values. We illustrate here using the four income columns from the **WDI** data.

- **Multivariate Normal Imputation:** User selects a group of columns, and missing cells are populated with least squares estimates using data from the same row. This method changes the current table in memory, and reports on the results. After inspecting the imputed values for reasonableness, one can undo the operation or save the data table under a new name.



Imputation Report

Undo

5972 missing values were replaced by least squares imputation. 288 missing values were replaced by minimum-norm least squares imputation. A shrinkage estimate was used, with off-diagonals scaled by a factor of 0.98558. 2607 rows and 4 columns were affected. There were 10 missing value patterns across columns. Imputed values colored light blue.

Missing Columns

☐ Show only columns with missing

Close

Select columns and choose an action.

Select Rows Color Cells

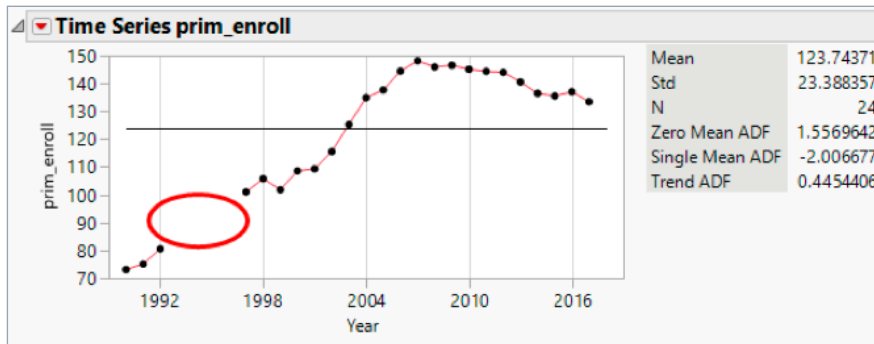
Exclude Rows Color Rows

Column	Number Missing
gdp_pc	0
gdp_pc_lcu	0
gni_pc	0
gni_pc_lcu	0

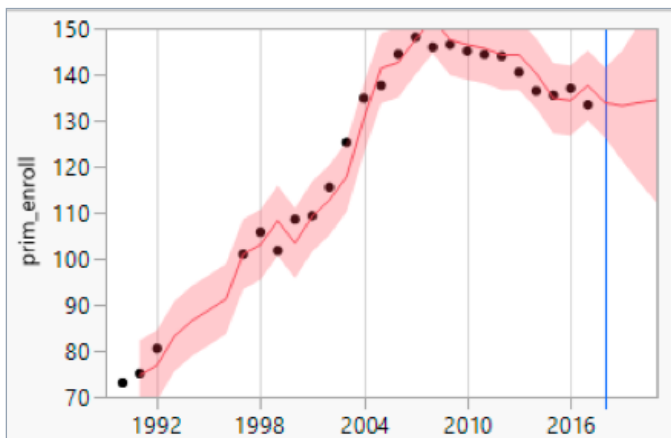
- **Multivariate SVD Imputation:** When a data table has more columns than rows, it may be advisable to use an imputation method that relies on singular value decomposition (SVD). Initially, missing values are replaced with column means, and then SVD transforms the data matrix as an iterative series of rotations and rescaling.
- **Automated Data Imputation (JMP Pro Only):** This method is an extension of multivariate SVD, and uses an approximation method known as matrix completion. The user can set some parameters of the algorithm and can choose among options for saving the new imputed data values.

Special Considerations for Time Series

These imputation methods may not be appropriate for time series data. If a time series is stationary, exhibiting only random variation, then it might make sense to use a column average or to model the data based on other columns. However, if the series exhibits cyclical movement or a strong trend, then one might turn to the **Time Series** and **Time Series Forecasts** platforms. As a simple example, look at primary school enrollments in Rwanda between 1990 and 2017. The series was interrupted, presumably due to the civil war and genocide in the early 1990s, and we are missing data for four years.



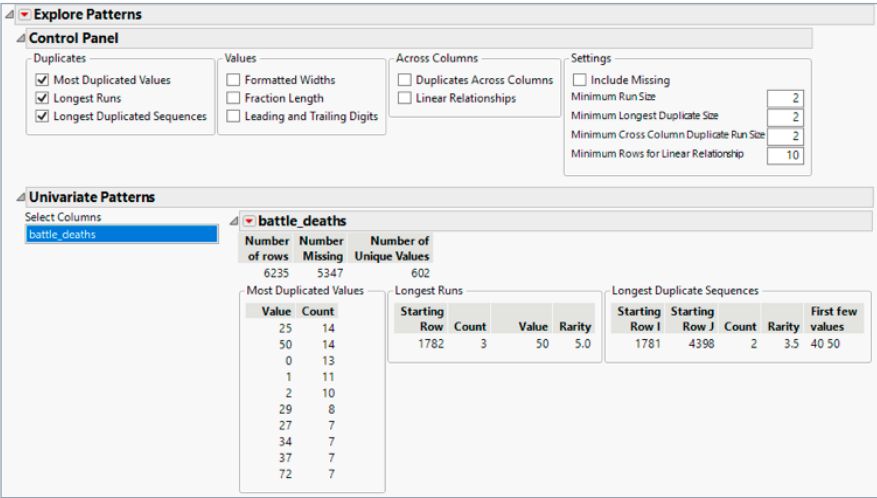
The black horizontal line is the mean value, which would be a poor estimate for the missing years. Here are the results of an autoregressive model (AR(2,1)). We see the fitted model and confidence region, and one may save the estimated values to the data table, including interpolated values for the four missing years.



Patterns

Consider a process that is continuously monitored by automatic sensors in which a sensor fails and records a consecutive series of identical values. Or a data set entered by hand, in which columns are transposed or some rows repeated. The **Explore Patterns** utility performs several different quality checks looking for suspicious patterns like duplicated values within a column, a long run of identical values, or patterns across columns (e.g., replicated values or linear relationships).

To illustrate, consider the variable **battle_deaths** in the **WDI** data. The default **Explore Patterns** report for that single column summarizes repeated values and where they occur.



One might expect that the chaos of war would result in randomly varying numbers of fatalities in a country. If a government were able to accurately account for deaths, it would be surprising to find precisely the same number of battle-related fatalities in several years, let alone consecutive years. And yet, this report shows that the numbers 25 and 50 appear quite often in the data, and that there is a run of three years in which a country reported 50 deaths. Using a red triangle option, we can select that run, reported by Ethiopia for 2002 through 2004. The human cost of conflict in that country since 1990 has been horrifying and highly variable. Scanning the data table, it is clear that further investigation is warranted to determine what happened in 2001.

	Country Code	Country Name	Region	Income Group	Year	air_pass	battle_deaths
1767	EST	Estonia	Europe & Central...	High income	2016	372274	•
1768	EST	Estonia	Europe & Central...	High income	2017	13103	•
1769	EST	Estonia	Europe & Central...	High income	2018	•	•
1770	ETH	Ethiopia	Sub-Saharan Africa	Low income	1990	620300	49698
1771	ETH	Ethiopia	Sub-Saharan Africa	Low income	1991	635600	12227
1772	ETH	Ethiopia	Sub-Saharan Africa	Low income	1992	756400	165
1773	ETH	Ethiopia	Sub-Saharan Africa	Low income	1993	752000	36
1774	ETH	Ethiopia	Sub-Saharan Africa	Low income	1994	716400	75
1775	ETH	Ethiopia	Sub-Saharan Africa	Low income	1995	749900	29
1776	ETH	Ethiopia	Sub-Saharan Africa	Low income	1996	743000	52
1777	ETH	Ethiopia	Sub-Saharan Africa	Low income	1997	772300	•
1778	ETH	Ethiopia	Sub-Saharan Africa	Low income	1998	789900	1040
1779	ETH	Ethiopia	Sub-Saharan Africa	Low income	1999	861400	30704
1780	ETH	Ethiopia	Sub-Saharan Africa	Low income	2000	944595	25050
1781	ETH	Ethiopia	Sub-Saharan Africa	Low income	2001	1027528	40
1782	ETH	Ethiopia	Sub-Saharan Africa	Low income	2002	1103184	50
1783	ETH	Ethiopia	Sub-Saharan Africa	Low income	2003	1117329	50
1784	ETH	Ethiopia	Sub-Saharan Africa	Low income	2004	1403293	50
1785	ETH	Ethiopia	Sub-Saharan Africa	Low income	2005	1667316	83
1786	ETH	Ethiopia	Sub-Saharan Africa	Low income	2006	1720306	50
1787	ETH	Ethiopia	Sub-Saharan Africa	Low income	2007	2290179	177
1788	ETH	Ethiopia	Sub-Saharan Africa	Low income	2008	2715017	50
1789	ETH	Ethiopia	Sub-Saharan Africa	Low income	2009	2914056	515
1790	ETH	Ethiopia	Sub-Saharan Africa	Low income	2010	3347022	151
1791	ETH	Ethiopia	Sub-Saharan Africa	Low income	2011	4440917.6908	51
1792	ETH	Ethiopia	Sub-Saharan Africa	Low income	2012	5001121.8652	50
1793	ETH	Ethiopia	Sub-Saharan Africa	Low income	2013	5671501	77
1794	ETH	Ethiopia	Sub-Saharan Africa	Low income	2014	6274582	25
1795	ETH	Ethiopia	Sub-Saharan Africa	Low income	2015	7074779	49
1796	ETH	Ethiopia	Sub-Saharan Africa	Low income	2016	8242114	50
1797	ETH	Ethiopia	Sub-Saharan Africa	Low income	2017	9566378	•
1798	ETH	Ethiopia	Sub-Saharan Africa	Low income	2018	•	•
1799	FIN	Finland	Europe & Central...	High income	1990	4450200	•

Data requirements driven by analytic approach

The analysis plan for a project should influence the data management and preparation activities. Modeling methodologies have their own requirements for the organization of a data table, for units of analysis and for data types. As a simple and familiar example, models built on paired observations will expect to find data pairs in separate columns.

Hence, one needs to be constantly mindful of the stages to follow when preparing a set of data for analysis. Data preparation happens within the context of the full investigative cycle for a reason, and that goes beyond variable selection. The shape of the table must match the analytic goals, and data types and modeling types must be suitable. For explanatory and predictive models, we may need to engineer features from raw data and may want to manage the dimensionality of a table.

What shape is a data table?

Earlier, when first examining the WDI data table, we briefly discussed the shape of a data table as being either wide or long/narrow in format. Perhaps the most common reason for arranging data into a wide format is that we are working with longitudinal or repeated measures data. Some analysis platforms may require that each repeated measurement be treated as a variable (see, e.g., Grace-Martin, 2015, or SAS, 2019). For some procedures, JMP can analyze data from a repeated measures design in either wide or long format, as discussed in SAS 2016b. Because repeated measured of the same subject or observational unit are likely to be correlated with one another, they are better treated as multiple variables rather than a single variable.

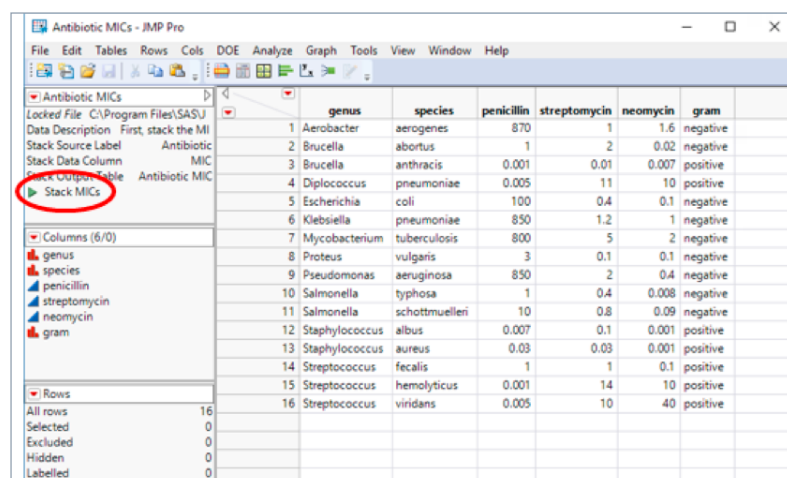
When the analytical goal involves explicitly viewing time as a factor or predictor, as in time-series modeling or visualization, then a long format might be more suitable. For example, a data table should be long to use the Time dimension in the JMP **Bubble Plot** platform.

The **Tables** menu provides two platforms – **Stack** and **Split** – to rearrange data from wide to long and vice versa. Without getting lost in the details of performing these operations, let’s look at the implications of shape and also take the opportunity to see how JMP facilitates reproducibility of complex wrangling operations.

Stacking wide data

Wainer and Lysen (2009) report on a noteworthy historical example dating to 1951 when “famous graphic designer Will Burtin (1908–1972) published a graphic display that was admired for the clarity and economy with which it showed the efficacy of three antibiotics on 16 different kinds of bacteria. The dependent variable was the minimum concentration of the drug required to prevent the growth of the bacteria in vitro – the minimum inhibitory concentration (MIC).”

Burtin’s intention was to clearly communicate the comparative efficacy of the three antibiotics – penicillin, streptomycin and neomycin – in inhibiting the growth of each specific bacterium. Burtin’s data table is included in the JMP Sample Data Library as **Antibiotic MICs**.



	genus	species	penicillin	streptomycin	neomycin	gram
1	Aerobacter	aerogenes	870	1	1.6	negative
2	Brucella	abortus	1	2	0.02	negative
3	Brucella	anthracis	0.001	0.01	0.007	positive
4	Diplococcus	pneumoniae	0.005	11	10	positive
5	Escherichia	coli	100	0.4	0.1	negative
6	Klebsiella	pneumoniae	850	1.2	1	negative
7	Mycobacterium	tuberculosis	800	5	2	negative
8	Proteus	vulgaris	3	0.1	0.1	negative
9	Pseudomonas	aeruginosa	850	2	0.4	negative
10	Salmonella	typhosa	1	0.4	0.008	negative
11	Salmonella	schottmuelleri	10	0.8	0.09	negative
12	Staphylococcus	albus	0.007	0.1	0.001	positive
13	Staphylococcus	aureus	0.03	0.03	0.001	positive
14	Streptococcus	fecalis	1	1	0.1	positive
15	Streptococcus	hemolyticus	0.001	14	10	positive
16	Streptococcus	viridans	0.005	10	40	positive

The 16 bacteria are in the rows, and the columns contain the genus and species for each, as well as three variables representing the MICs for each of the three antibiotics. Finally, the sixth column indicates whether “the bacteria in question take up Gram stain or not (The stain is named after its inventor, Hans Christian Gram [1853–1938]).” An inspection of the data makes clear that the MICs span several orders of magnitude. The widely differing scales pose a challenge for graphing, and we’ll take that up shortly.

The wide layout served Burtin’s purpose of comparing the quantity of each treatment to manage each species. However, for other types of analysis or visualization, a long (stacked) format is preferable. In fact, this data table includes a saved script (see the circled green arrow item in the **Tables** panel) as well as a usage note in the **Data Description Table Variable**. The description reads as follows:

First, stack the MIC columns (run the script). Then, create a new column and take the Log(MIC).

Background: In the fall of 1951 Burtin published a graph showing the performance of the three most popular antibiotics on 16 different bacteria.

The response, the minimum inhibitory concentration (MIC), represents the concentration of antibiotic required to prevent growth in vitro. The covariate “gram” describes the reaction of the bacteria to Gram staining.

This data table has 16 rows and six columns (a 16x6 matrix). The creator of this data table used the **Stack** platform to create a long data table, which turns out to be 48x5, and hence larger. Rather than saving two data tables, they used a red triangle option to save the script for the stacking operation within the original data table. Every JMP dialog generates a JMP Scripting Language (JSL) script consisting of code that, when executed, will perform the desired operation. When you run a JMP platform, the resulting report, graph or data table has a red triangle menu that includes an option to save the script, and one can easily save the script as a table variable in the originating table. To regenerate the long table, one clicks the green arrow next to the script title, in this case **Stack MICs**, to produce the data table.

	genus	species	gram	Antibiotic	MIC
1	Aerobacter	aerogenes	negative	penicillin	870
2	Aerobacter	aerogenes	negative	streptomycin	1
3	Aerobacter	aerogenes	negative	neomycin	1.6
4	Brucella	abortus	negative	penicillin	1
5	Brucella	abortus	negative	streptomycin	2
6	Brucella	abortus	negative	neomycin	0.02
7	Brucella	anthracis	positive	penicillin	0.001
8	Brucella	anthracis	positive	streptomycin	0.01
9	Brucella	anthracis	positive	neomycin	0.007
10	Diplococcus	pneumoniae	positive	penicillin	0.005
11	Diplococcus	pneumoniae	positive	streptomycin	11
12	Diplococcus	pneumoniae	positive	neomycin	10
13	Escherichia	coli	negative	penicillin	100
14	Escherichia	coli	negative	streptomycin	0.4
15	Escherichia	coli	negative	neomycin	0.1
16	Klebsiella	pneumoniae	negative	penicillin	850
17	Klebsiella	pneumoniae	negative	streptomycin	1.2
18	Klebsiella	pneumoniae	negative	neomycin	1
19	Mycobacterium	tuberculosis	negative	penicillin	800

This table creator took the additional step of documenting the wrangling steps for a reader by providing a data description note. The future reader could be another team member, or it could be the initial wrangler returning to the data in the future. This is a good illustration of the practice of “being kind to your future self.”

Feature engineering: Common issues and appropriate strategies

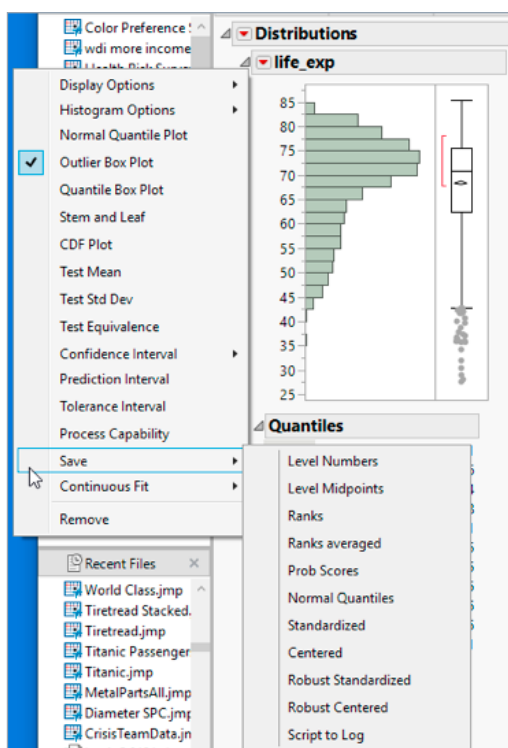
In machine learning, practitioners speak about featurization or feature engineering, the process of transforming available data into more useful factors to train a model. The key drivers in deciding how to transform the data in a table are characteristics of the data (outliers, skewness, missingness, high-dimensionality and so on) and the plan for analyzing the data. For example, skewness is problematic for traditional inference methods based on assumptions of normality, but it is not necessarily troublesome with other techniques. For other modeling methods, the comparative scale of model variables is an important consideration. Distance-based methods like k-means clustering or k-nearest neighbors aim to gather observations that are proximate in multidimensional space. When applying such techniques, it is often advisable to express variables in a standard scale like z-scores.

We can define four general types of data characteristics that lead us to transform a variable; these are summarized in the table below. JMP provides all the solutions listed in the table, sometimes in multiple ways and within multiple platforms. The list is long, and it is beyond the scope of this paper to cover all the issues, but in this section, we highlight a few that might be unfamiliar to many readers.

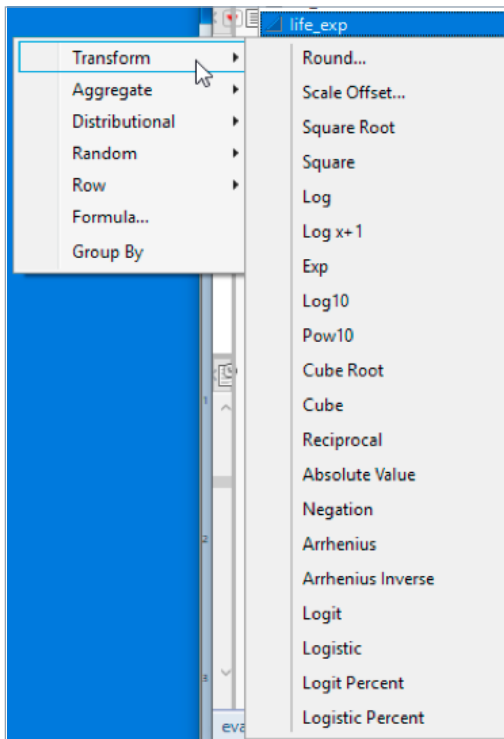
Category	Common Problems	Common Solutions
Distribution of observations		
	Noisy data: high degree of variation in continuous data	Binning, nominal discretization, smoothing (for time series)
	Skewness or outliers	Functional transformation (e.g., logarithmic)
	Scale differences among variables to be used	Standardization
	Too many levels in a categorical variable	Recoding, clustering
High dimensionality: Abundance of columns		
	Correlated or redundant variables	Drop columns, principal components analysis (PCA), singular value decomposition (SVD)
	Sparsity (many 0s) or missing observations across columns	PCA, SVD
Abundance of rows		
	Table partitioning for training, validation and testing	Create validation column, subset
	Variety of observational units	Aggregation by strata, cluster or time period
Date and time-related issues		
	Inconsistent date or time data	Date functions

Distribution of observations

In some situations, we want to tame either the shape or the variability of one or more variables. We have already touched on functional transformations to address outliers or highly skewed data. With continuous data, the red triangle **Save** options include several alternatives to discretize, rescale or center a variable.



We've seen a logarithmic transformation applied to a right-skewed distribution. One can create a new transformed variable as a new column in a data table, or temporarily right within a platform so that transformations can be explored within one's workflow with minimal disruption. There are many functions available as temporary local transformations, visible by right-clicking a continuous column name in **Graph Builder**, for example. These local transformations allow for rapid exploration during analysis, without having to step back and create a new column before knowing whether the transformation will be useful.



High dimensionality: Abundance of columns

Traditionally, statistical analysts sought more data rather than less. In the current era, we are often confronted with more variables and more observations than needed. In this section, we consider a few strategies to overcome the embarrassment of riches that can arise from the plethora of data now available. Recall the WDI data table with four different measures of per capita income. If one wants to build a model that includes an income feature, some of the columns are probably superfluous. One strategy is just to select the single column that is most informative for the modeling task.

On the other hand, we know that all four columns have missing observations. Principal components analysis (PCA) is a technique to extract information content from multiple columns, distilling the information into perhaps one or two columns with complete observations. PCA creates a relatively small number of separate linear combinations of several numeric variables in a way that captures as much of the variation in the original variables as possible. An appealing characteristic of PCA is that it can create usable components even in the presence of missing data or columns with sparse data.

A full treatment of PCA is well beyond the scope of this white paper, but here is a highly condensed introduction. Interested readers should consult the *Multivariate Methods* portion of the *JMP Documentation Library*.

Conceptually, given a set of m variables whose combined total variation is V , PCA creates a set of m weighted linear combinations of the variables. The first principal component is that linear combination whose variance is the maximum of all possible combinations, capturing as much of the joint variability as possible. From there, the algorithm finds a second component that is orthogonal to the first and therefore uncorrelated with the first and that captures the maximum amount of remaining variation. This process continues through all m components.

In the simplest case, if there were just two correlated columns under consideration, the goal of PCA is to combine them in such a way that the first principal component captures most of their combined variability and therefore we can build a model using either the first component rather than two variables, or that we can use two uncorrelated principal components and (in the context of linear regression) bypass the issue of multicollinearity.

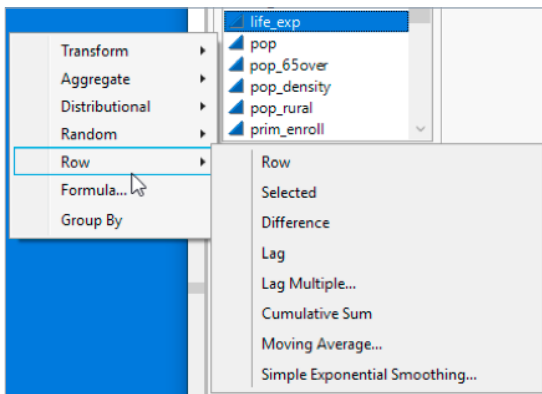
High dimensionality: Abundance of rows

It may seem counterintuitive to suggest that there are times when we might want to reduce our sample size. Particularly with data mining or machine learning projects or with studies that use data sets with large numbers of observations, there are at least three scenarios in which it may be advantageous or desirable to build models with fewer rather than more rows:

- In general, we don't want to develop hypotheses and then test them with the same data. When we have the luxury of an enormous number of observations, we can productively use part of the data for model-building, and then validate the models with a holdout sample. A standard practice in data mining is to divide a body of data into training and test subsamples. In JMP, we can easily partition a large sample for exploration, modeling and validation by creating a new two- or three-level dummy validation column. Many analysis platforms support the use of a validation column.
- If the observational unit is at a finer level than the modeling task demands, or if we have data from multiple sources that use different observational units, we may want to aggregate rows in some fashion. For example, we might find some city-level data when we want county- or state-level data, or we have repeated measures that we want to combine for each individual. The **Tables** menu and **Analyze > Tabulate** provide the necessary functionality.
- When building models to predict rare events, we may want to over-sample the rare event and under-sample others. Hence, we might randomly suppress rows corresponding to the unusual target values using row selection options in the **Rows** menu.

Date and time-related issues

In JMP, as in other software, times and dates are stored internally as numbers, but may be displayed in a variety of more recognizable forms. JMP stores date-based data internally as the number of seconds elapsed since Jan. 1, 1904, and provides numerous formatting options as well as date and time functions to carry out the special operations demanded by date data. Here again, JMP anticipates these needs, and **Column Info** and **Formula** provide much of the functionality. In addition, all analysis launch windows include temporary row functions.



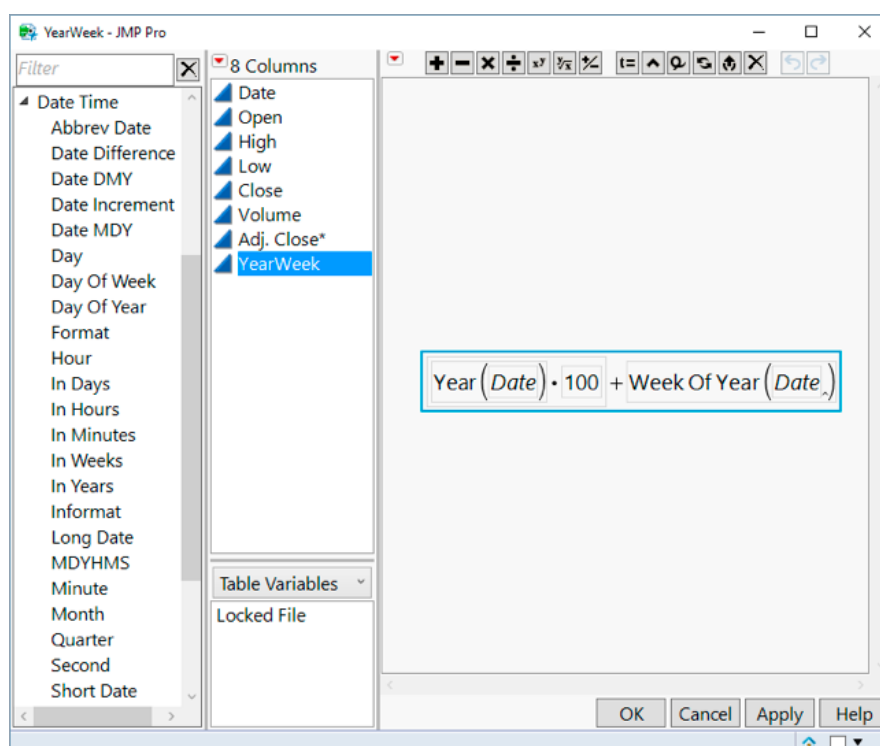
Among the special issues for time and date data are these:

- Although time is continuous, ordinary arithmetic operations don't apply.
- JMP uses the 1904 bases, but not all programs do the same. For example, some versions of Excel use 1904, while others use 1900 as the starting point. IBM SPSS uses Oct. 14, 1582, and UNIX systems use Jan. 1, 1970. For projects drawing data from various originating sources, it could be necessary to take these differences into account.
- Sometimes, we may want to extract a part of a timestamp, for example the day of the week from a date, or just the hour from a time.
- Similarly, we might want to aggregate rows across time, which could require extraction of, say, the year from dates expressed as mm/dd/yyyy.
- For observations that have a time associated with them, sometimes we might be less interested in a specific observation than in its change from a prior value or in the prior value itself. This is easily accomplished with **Differences** and **Lags**.

Some date functions: Extracting parts

JMP has numerous functions for working with time-related columns, particularly with respect to isolating portions of a time stamp. When analyzing stock price data, one might want to focus on the month or quarter of a date. In another setting, one might be interested in the hour of the day, or day of the week.

As a simple illustration, consider a stock price data table of daily prices that we want to aggregate into weekly averages. Part of that process is converting a date into a year plus week number format. I've done just that by creating a new column called **YearWeek**, which is computed from the **Date** column, and illustrates one of the many ways that date functions can be used.



Concluding thoughts

Data preparation is an essential and critical part of any project. It requires an understanding of an organization's needs as well as statistical methodologies. With the proper understanding and tools, one can wrangle data efficiently, at scale, and within the iterative flow of a project. JMP anticipates a wide range of common data wrangling needs and situations, and it permits users to clean and regularize dirty data with ease. Rather than needing multiple software services to perform extract-transform-load (ETL) operations, one can accomplish all of the entire ETL process seamlessly within the analytic workflow. Moreover, the JMP environment can record and document every aspect of data wrangling so that there are reproducible audit trails. Users stay within the single software environment throughout.

References

- Adam, D. (2019). "The data detective". *Nature*, Vol. 571, 25 July, 2019.
- Carver, Robert H. (2017). *Preparing Data for Analysis With JMP*. Cary, NC: SAS Institute Inc.
- Carver, Robert H. (2016). "Speeding up the dirty work of analytics". JMP Discovery Summit 2016, Cary NC. Available at <https://community.jmp.com/t5/Discovery-Summit-2016/Speeding-Up-the-Dirty-Work-of-Analytics/ta-p/24106>.
- Centers for Disease Control and Prevention (2018). 2017 YRBS Data User's Guide. (June), Atlanta: CDC.
- Gelman, A. and Hill, J. (2006). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Missing-data imputation, Chapter 25. New York: Cambridge Univ. Press. Available at <http://www.stat.columbia.edu/~gelman/arm/missing.pdf>.
- Grace-Martin, K. (2015). "The Wide and Long Data Format for Repeated Measures Data", available at <http://www.theanalysisfactor.com/wide-and-long-data/>.
- Han, J., Kamber, M. and Pei, J. (2011) *Data Mining: Concepts and Techniques*, 3rd Edition. Waltham MA: Elsevier, Chapter 3.
- Kandel, S., Heer, J., Plaisant, C., Kennedy, J., van Ham, F., Riche, N.H., Weaver, C., Lee, B., Brodbeck, D. and Buono, P. (2011). "Research directions in data wrangling: Visualizations and transformations for usable and credible data". *Information Visualization*. Vol. 10(4), pp. 271-288.
- Kim, W., Choi, B.-J., Hong, E.-K., Kim, S.-K., and Lee, D. (2003). "A taxonomy of dirty data," *Data Mining and Knowledge Discovery*, 7, pp. 84--91.
- Little, R. and Rubin, D. (2002). *Statistical Analysis With Missing Data*, 2nd ed. Hoboken, NJ: Wiley.
- Lohr, Steve (2014). "For Big-Data Scientists, 'Janitor Work' Is Key Hurdle to Insights," *New York Times*. Available at <https://www.nytimes.com/2014/08/18/technology/for-big-data-scientists-hurdle-to-insights-is-janitor-work.html>.
- Orel, V. (1968). "Will the Story on 'Too Good' Results of Mendel's Data Continue?." *Bioscience*, Vol. 18, No. 8 (August 1968), pp. 776-778. Available at <https://www.jstor.org/stable/1294326>.
- Penn, D.A. (2007). "Estimating missing values from the General Social Survey: An application of multiple imputation," *Social Science Quarterly*, Vol. 88, No. 2 (June), pp. 573-584.

Press, G. (2016) "Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says." Forbes, available online at <http://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#4fe4cf1e7f75>.

Rubin, D.B. (1996) "Multiple imputation after 18+ years." Journal of the American Statistical Association, Vol. 91, No. 434, (June) pp. 473-489.

Sall, J. (2013) "It's not just what you say, but what you don't say: Informative missing values." JMP Blog, Oct. 29. Available at <http://blogs.sas.com/content/jmp/2013/10/29/its-not-just-what-you-say-but-what-you-dont-say-informative-missing/>.

SAS (2016) "Managing the Analytical Life Cycle for Decisions at Scale: How to Go From Data to Decisions as Quickly as Possible." Cary, NC: SAS Institute Inc.

SAS (2019) JMP 15 Documentation Library. Cary, NC: SAS Institute Inc.

Sterne, J.A.C., White, I.R., Carlin, J. B., Spratt, M., Royston, P., Kenward, M.G., Wood A.M. and Carpenter, J.R. (2009). "Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls." BMJ: British Medical Journal. Vol. 339, No. 7713 (18 July), pp. 157-160.

Wainer, H. and Lysen, S. (2009) "That's Funny...A window on data can be a window on discovery." American Scientist (July/August).



About SAS and JMP

SAS, the world's leader in analytics, created JMP (pronounced "jump") in 1989 to empower scientists, engineers and other data analysts to explore and analyze data visually and interactively. Since then, JMP has grown from a single product into a family of statistical discovery tools, each one tailored to meet specific needs. John Sall, SAS co-founder and Executive Vice President, leads the JMP business unit.



SAS Institute Inc. World Headquarters

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

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies. 111281_G125681.0520