Introduction

LazStats, among others, are ongoing projects that I have created for use by students, teachers, researchers, practitioners and others. There is no charge for use of these programs if downloaded directly from a World Wide Web site. The software is a result of an “over-active” hobby of a retired professor (Iowa State University.) I make no claim or warranty as to the accuracy, completeness, reliability or other characteristics desirable in commercial packages (as if they can meet these requirements.) They are designed to provide a means for analysis by individuals with very limited financial resources. The typical user is a student in a required social science or education course in beginning or intermediate statistics, measurement, psychology, etc. Some users may be individuals in developing nations that have very limited resources for purchase of commercial products.

The series of statistics packages began years ago while I was teaching educational psychology and industrial technology at Iowa State University. As previously mentioned, packages have been written in Basic for the Altair computer, Amiga computer, Radio Shack, Commodore 64, and the PC. The first package was called “FreeStat”. With the advent of Windows on the PC, a new version of the package was created using Borland’s Turbo Pascal and Microsoft’s Visual Basic. These packages were named “Statistics and Measurement Program Learning Environment (SAMPLE.) Upon my retirement I began the “OpenStat” series. The first OpenStat was written with the use of Borland’s C++ Builder. OpenStat2, OS2 and OS3 were written with Borland’s Delphi (Pascal) compiler. Another version, LinuxOStat, was written with Borland’s Kylix compiler for the Linux operating system(s). LazStats was written using the free Lazurus – Free Pascal package. Each of these versions contains slight variations from each other. Because there are many students out there who are learning computer programming as well as statistics, the source code for each of these versions is also available through the Internet downloading. It should be clear that programming is my hobby and I enjoy trying various languages and compilers. Other languages such as Lisp and Forth could be and are used to write statistical routines. Years ago I wrote programs in Fortran and Cobol (ugh!) C++ seems to have become the standard for many commercial and industrial uses and Pascal is such a nice language for the teaching of programming that I have devoted more time to these languages. The advantage of the Pascal compilers from the Borland Corporation is the *very* fast compile times and excellent execution speeds they provided. Lazurus seems to be similar and the code can be compiled for different operating systems. There are several languages which are interpretive languages that have been developing over the past decade that seem to be popular for statistical programming. R and S are such “languages” popular among “serious” statisticians. Unfortunately, the learning curve is rather high for the occasional user and student who would rather spend time on the statistics rather than the programming challenges.

While I reserve the copyright protection of these packages, I make no restriction on their distribution or use. It is common courtesy, of course, to give me credit if you use these resources. Because I do not warrant them in any manner, you should insure yourself that the routines you use are adequate for your purposes. I strongly suggest analyses of textbook examples and comparisons to other statistical packages where available. You should also be aware that I am constantly revising, correcting and updating OpenStat. For that reason, some of the images and descriptions in this book may not be exactly as you see when you execute the program. I update this book from time to time to try and keep the program and text coordinated.

Installing LazStats

LazStats should be successfully installed on Windows 95, 98, ME, XT, NT and Vista systems.. A free setup package (INNO) has been used to package LazStats and associated files for installation on your computer. I include in the setup file the executable file, HTML Help files, and sample data files.

To install LazStats for Windows, follow these steps:

1. Connect to the internet address: http://statpages.com/miller/LazStats/

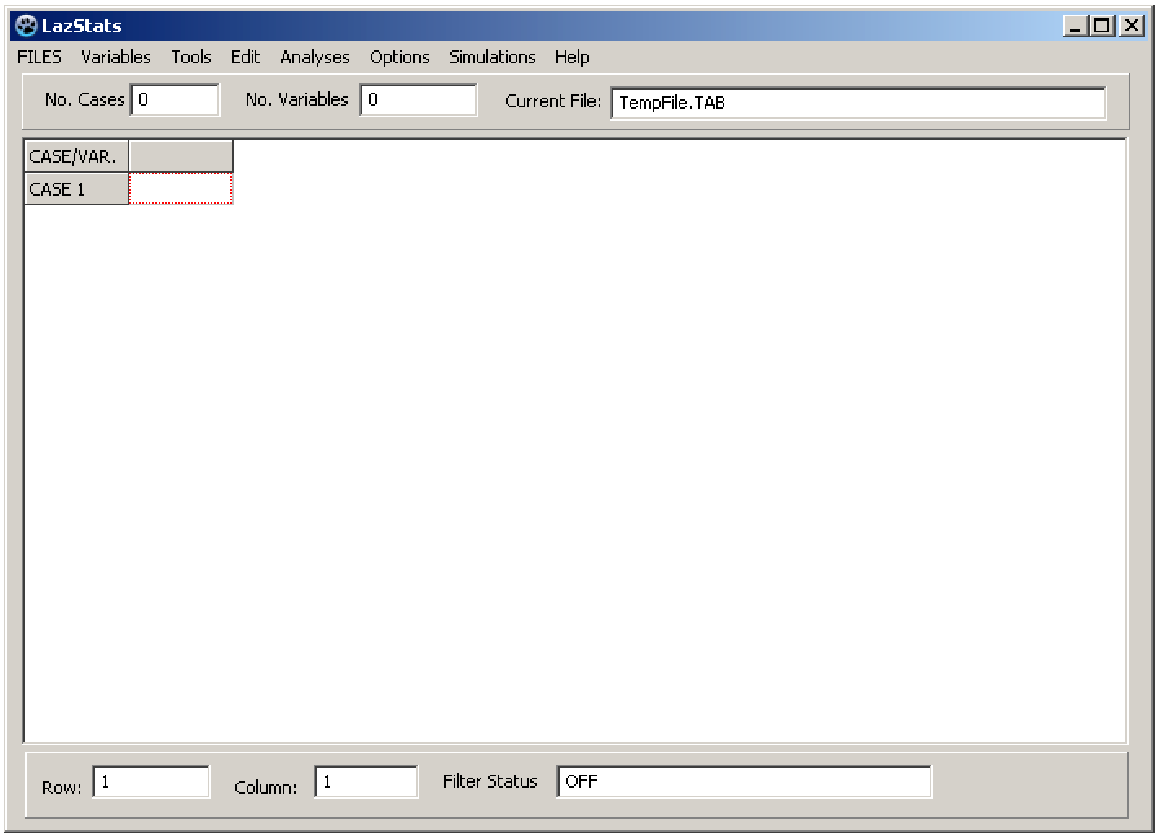
2. Click on the link to the LazStats program file. Your Internet browser should automatically begin the download process to a directory of your choice on your computer.

3. Once you have downloaded the program file, simply double click the name of the file and the program will begin.

4. I recommend you also download the sample files into the same directory in which you placed the program.

Starting LazStats

To begin using a Windows version of LazStats use the Windows Explorer to change to the directory where you stored LazStats files and double click the LazStats.exe file (or its icon.) The initial screen you see will be a form that displays a small window to accept use of the program. An "Options" file will be created. The OPTIONS.FIL contains the default values for how your data will be stored and what the default missing value is for a variable (more on this later.) Next, the following form should appear:



No. Cases 0

No. Variables 0

Current File: TempFile.TAB

Row: 1

Column: 1

Filter Status OFF

Figure 1. The LazStats Main Form

The above form contains several important areas. The "grid" is where data values are entered. Each column represents a "variable" and each row represents an "observation" or case. Each case of data you enter will have a case number. At the top of this "main" form there is a series of "drop-down" menu items. When you click on one of these, a series of options (and sometimes sub-options) that you can click to select. Before you begin to enter case values, you need to "define" each variable to be entered in the data grid. Select the "VARIABLES" menu item and click the "Define" option. More will be said about this in the following pages.

Files

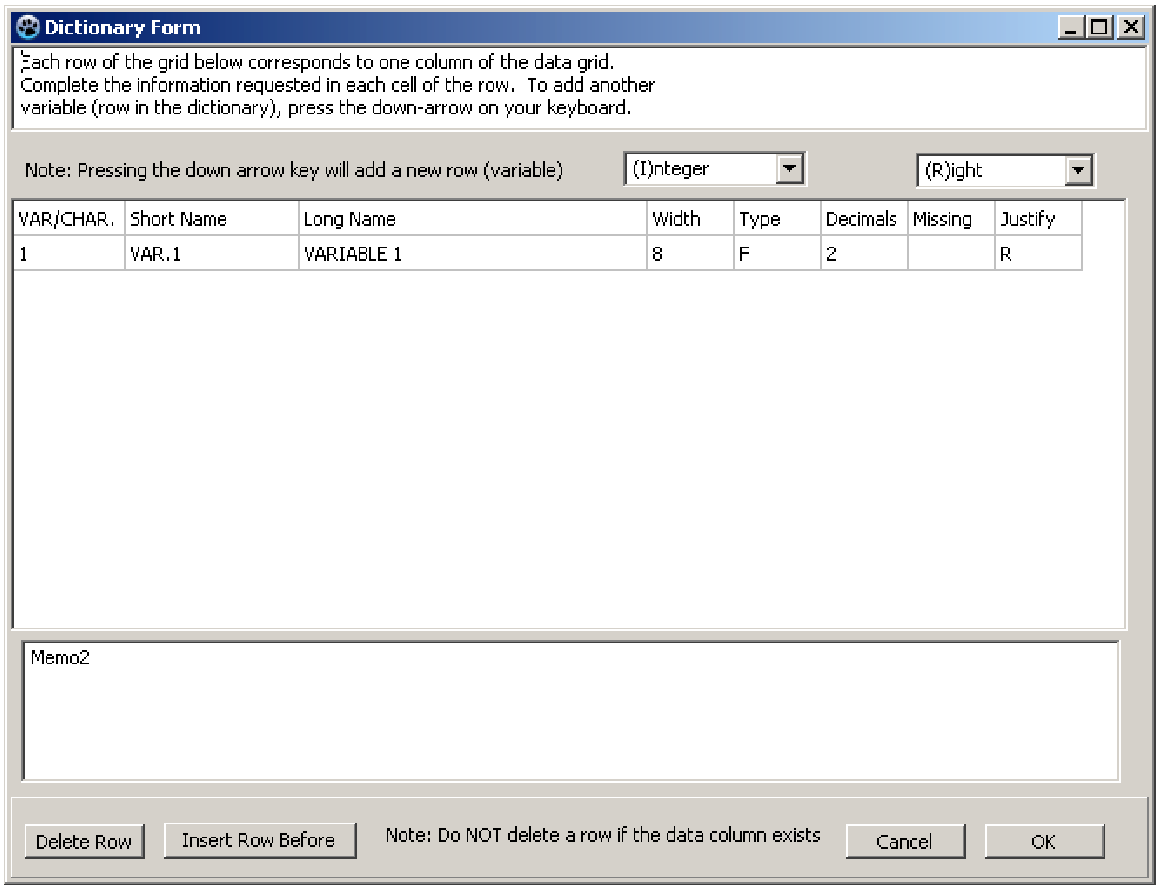
The “heart” of LazStats or any other statistics package is the data file to be created, saved, retrieved and analyzed. Unfortunately, there is no one “best” way to store data and each data analysis package has its own method for storing data. Many packages do, however, provide options for importing and exporting files in a variety of formats. For example, with Microsoft’s Excel package, you can save a file as a file of “tab” separated fields. Other program packages such as SPSS can import “tab” files. Here are the types of file formats supported by LazStats:

1.Text type of files (with the extension .LAZ) NOTE: the file format in this text file is unique to LazStats!
2.Tab separated field files (with the file extension of .TAB.)
3.Comma separated field files (with the file extension of .CSV.)
4.Space separated field files (with the file extension of .SSV.)

My preference is to save files as .LAZ and .TAB files. This gives me the opportunity to analyze the same data using a variety of packages. For relatively small files (say, for example, a file with 20 variables and 1000 cases), the speed of loading the different formats is similar and quite adequate.

Creating a File

When LazStats begins, you will see a “grid” of two rows and two columns. The left-most column will automatically contain the word “Case” followed by a number (1 for the first case.) The top rows will contain the names of the variables you have defined. You can change the name of the variables and define additional variables by clicking on the menu item labeled “VARIABLES” and then clicking on the “Define” option. A “form” will appear that looks like the figure below:



Each row of the grid below corresponds to one column of the data grid.
Complete the information requested in each cell of the row. To add another variable (row in the dictionary), press the down-arrow on your keyboard.

Note: Pressing the down arrow key will add a new row (variable)

(I)nteger

(R)ight

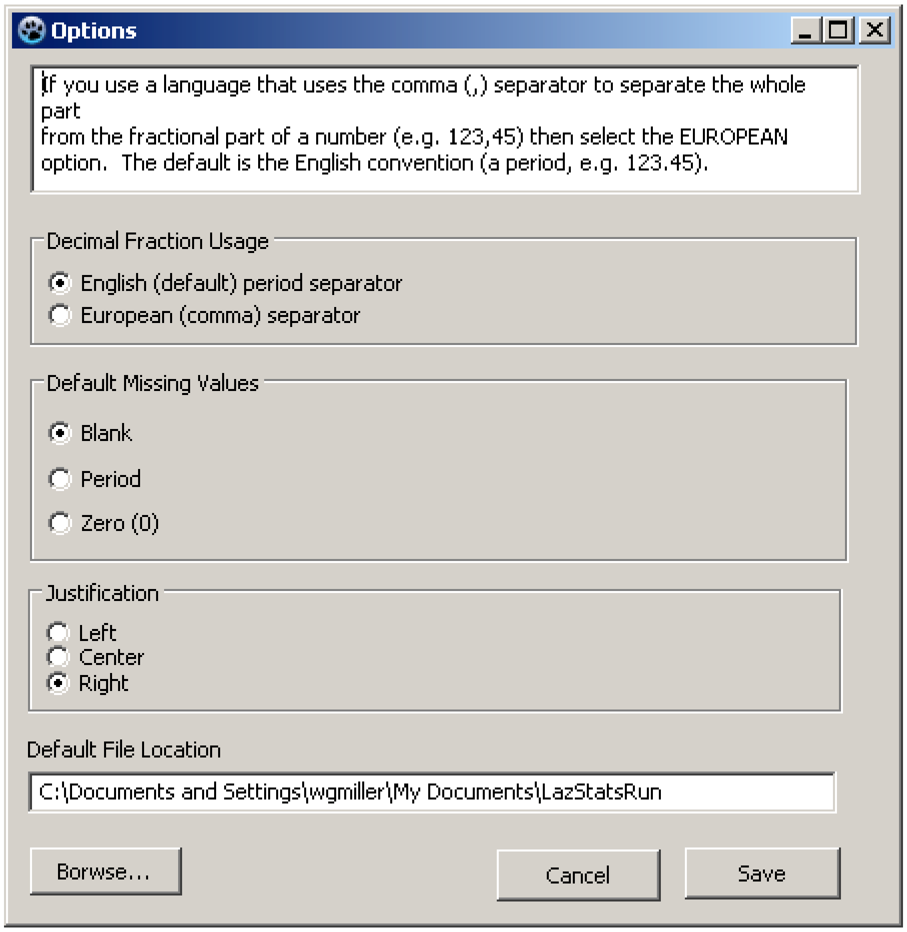
VAR/CHAR.	Short Name	Long Name	Width	Type	Decimals	Missing	Justify
1	VAR.1	VARIABLE 1	8	F	2		R

Memo2

Note: Do NOT delete a row if the data column exists

Figure 2. The Variable Definition Form

In the above figure you will notice that a variable name was automatically generated for the first variable. To change the default name, double click the box with the default name and enter the variable name that you desire. It is suggested that you keep the length of the name to eight characters or less. You may also enter a longer label for the variable. If you save your file as a .LAZ file, this long name (as well as other descriptive information) will be saved in the file (the use of the long label has not yet been implemented for printing output but may be in future versions.) To proceed, simply click the OK button in the lower right of this form. The default type of variable is a “floating point” value, that is, a number that may contain a decimal fraction. If a data field (grid cell) is left blank, the program will usually assume a missing value for the data. The default format of a data value is eight positions with three positions allocated to fractional decimal values (format 8.3.) By clicking on any of the specification fields you can modify these defaults to your own preferences. You can change the number of decimal places (0 for integers.) You will find that some analyses require that a variable be defined as an integer and others as floating point values. The drop-down box labeled “(I)nteger” lets you click on the type of variable you are defining and automatically record the character value that defines that type. If you press the “down-arrow” on your keyboard, another variable with default values will be added. You can also click a large button at the bottom to add delete or insert a variable. Another way to specify the default format and missing values is by modifying the "Options" file. When you click on the Options menu the following form appears:


Options

If you use a language that uses the comma (,) separator to separate the whole part
from the fractional part of a number (e.g. 123,45) then select the EUROPEAN option. The default is the English convention (a period, e.g. 123.45).

Decimal Fraction Usage
[✓ English (default) period separator]
European (comma) separator

Default Missing Values
[✓ Blank]
Period
Zero (0)

Justification
Left
Center
[✓ Right]

Default File Location: C:\Documents and Settings\wgmiller\My Documents\LazStatsRun

Figure 3. The Options Menu

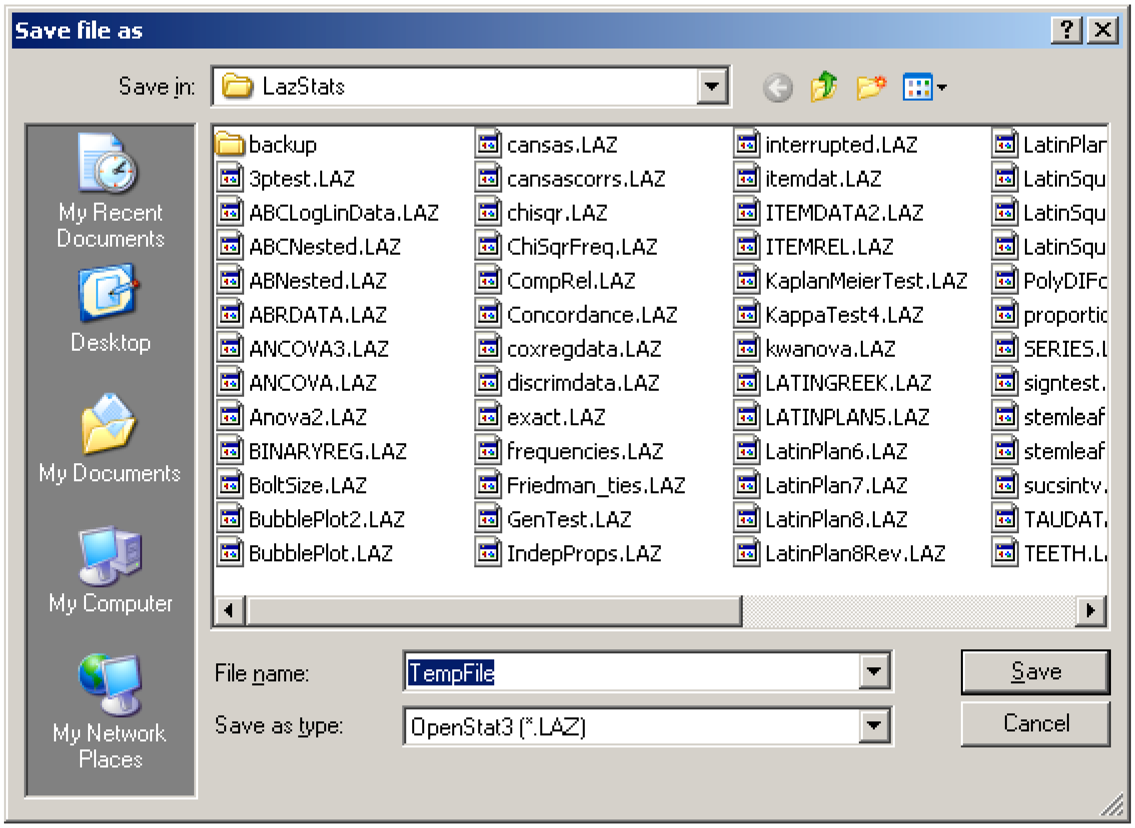
In the options form you can specify the Data Entry Defaults as well as whether you will be using American or European formatting of your data (American's use a period (.) and Europeans use a comma (,) to separate the integer portion of a number from its fractional part.) You can double click the Default File Location edit box to change the path to your files. In many countries, the separation of the whole number from the fractional part of a floating point number is a comma (,) and not a period (.) as in the United States. A user that uses the comma separator is designated a “European” user. The default is the American usage. It is possible to convert one type to another. The example files all use the American standard. If you use the European standard, you will need to examine the “default” confidence intervals shown on many of the statistics dialog forms – they may have a period (e.g. 0.05) instead of a comma (0,05) as needed in the European format. One can click on the value and change it to an appropriate format.

Entering Data

When you enter data in the grid of the main form there are several ways to navigate from cell to cell. You can, of course, simply click on the cell where you wish to enter data and type the data values. If you press the “enter” key following the typing of a value, the program will automatically move you to the next cell to the right of the current one or down to the next cell if you are at the last variable. You may also press the keyboard “down” arrow to move to the cell below the current one. If it is a new row for the grid, a new row will automatically be added and the “Case” label added to the first column. You may use the arrow keys to navigate left, right, up and down. You may also press the “Page Up” button to move up a screen at a time, the "Home" button to move to the beginning of a row, etc. Try the various keys to learn how they behave. You may click on the main form's Edit menu and use the delete column or delete row options. Be sure the cursor is sitting in a cell of the row or column you wish to delete when you use this method. A common problem for the beginner is pressing the "enter" key when in the last column of their variables. If you do accidentally add a case or variable you do not wish to have in your file, use the edit menu and delete the unused row or variable. If you have made a mistake in the entry of a cell value, you can change it. In the grid cell you can use the delete key, backspace key, enter characters, etc. to make the corrections for a cell value Notice that as you make grid entries and move to another cell, the previous value is automatically formatted according to the definition for that variable. If you try to enter an alphabetic character in an integer or floating point variable, you will get an error message when you move from that cell. To correct the error, click on the cell that is incorrect and make the changes needed.

Saving a File

Once you have entered a number of values in the grid, it is a good idea to save your work (power outages do occur!) Go to the main form's File menu and click it. You will see there are several ways to save your data. A "dialog box" will then appear as shown below for a .LAZ type of file:



Save in: LazStats
File name: TempFile
Save as type: OpenStat3 (*.LAZ)

Figure 4. The Save Dialog Form

Simply type the name of the file you wish to create in the File name box and click the Save button. After this initial save operation, you may continue to enter data and save again. Before you exit the program, be sure to save your file if you have made additions to it. Notice that .LAZ file type is labeled as an OpenStat3 type of file. This is because LazStats is a conversion of a previously developed Borland Delphi Pascal program called "OS3".

If you do not need to save specifications other than the short name of each variable, you may prefer to "export" the file in a format compatible to other programs. The Export Tab File option under the File menu will save your data in a text file in which the cell values in each row are separated by a tab key character. A file with the extension .TAB will be created. The list of variables from the first row of the grid are saved first, then the first row of the data, etc. until all grid rows have been saved. If there are blanks in any value cells, the default missing value will be written for that cell. Alternatively, you may export your data with a comma or a space separating the cell values. Basic language programs frequently read files in which values are separated by commas or spaces. If you are using the European format of fractional numbers, DO NOT USE the comma separated files format since commas will appear both for the fractions and the separation of values - clearly a design for disaster!

Some Common Errors!

Empty Cells

The beginning user will often see a message something like "" is not a valid floating point value. The most common cause of this error occurs when a procedure attempts to read a blank cell, that is, a cell that has been left empty by the user. The new user will typically use the down-arrow to move to the next row in the data grid in preparation to enter the next row of values. If you do this after entering the values for the last case, you will create a row of empty cells. You should put the cursor on one of these empty cells and use the Edit->Delete Row menu to remove this blank row.

The user should define the "Missing Value" for each variable when they define the variable. One should also click on the Options menu and place a missing value in that form. Not all LazStats procedures allow missing values so you may have to delete cases with missing values for those procedures.

Incorrect Format for Floating Point Values

A second reason you might receive a "not valid" error is because you are using the European standard for the format of values with decimal fractions. Most of the statistical procedures contain a small "edit" window that contains a confidence level or a rejection area such as 95.0 or 0.05. These will NOT be valid floating point values in the European standard and the user will need to click on the value and replace it with the correct form such as 95,0 or 0,05.

String labels for Groups

Users of other statistics packages such as SPSS or Excel may have used strings of characters to identify different groups of cases (subjects or observations.) LazStats uses sequential integer values only in statistical analyses such as analyses of variance or discriminant function analysis An attempt to use a string (alphanumeric) value will cause an "not valid" type of error. It is best to do the conversion of string labels to integers and use the integer values as your group variable.

Floating Point Errors

Sometimes a procedure will report an error of the type "Floating Point Division Error". This is often the outcome of a procedure attempting to divide a quantity by zero (0.) As an example, assume you have entered data for several variables obtained on a group of subjects. Also assume that the value observed for one of those variables is the same (a constant value) for all cases. In this situation there is no variability among the cases and the variance and standard deviation will be zero! Now an attempt to use that zero variance or standard deviation in the calculation of z scores, a correlation with another variable or other usage will cause an error (division by zero is not defined.)

Values too Large (or small)

In some fields of study such as astronomy the values observed may be very, very large. Computers use binary numbers to represent quantities. LazStats procedures use "double precision" storage for floating point values. The double precision value is stored in 64 binary "bits" in the computer memory. In most computers this is a combination of 8 binary "bytes" or words. The values are stored with a characteristic and mantissa similar to a scientific notation. Of course bits are also used to represent the sign of these parts. The maximum value for the characteristic is typically something like 2 raised to the power of 55 and the mantissa is 2 to the 7th power. Now consider a situation where you are summing the product of several of very large values such as is done in obtaining a variance or correlation. You may very well exceed the 64 bit storage of this large sum of products! This causes an "overflow" condition and a subsequent error message. The same thing can be said of values too small. This can cause an "underflow" error and associated error message.

The solution for these situations of values too large or too small is to "scale" your initial values. This is can be done by subtracting (or adding for very small values) a constant from the original values to decrease (or increase) the values. This does not affect the covariability or variance of the variables but does affect the means. The results will have to be "re-scaled" to reflect the original measurement scale, typically by simply adding the constant back to the means.