The DO Loop
Statistical programming in SAS with an emphasis on SAS/IML programs

Log transformations: How to handle negative data values?

Rick Wicklin | APRIL 27, 2011

65635 50 Tweet +1 0

The log transformation is one of the most useful transformations in data analysis. It is used as a transformation to normality and as a variance stabilizing transformation. A log transformation is often used as part of exploratory data analysis in order to visualize (and later model) data that ranges over several orders of magnitude. Common examples include data on income, revenue, populations of cities, sizes of things, weights of things, and so forth.

In many cases, the variable of interest is positive and the log transformation is immediately applicable. However, some quantities (for example, profit) might contain a few negative values. How do you handle negative values if you want to log-transform the data?

Solution 1: Translate, then Transform

A common technique for handling negative values is to add a constant value to the data prior to applying the log transform. The transformation is therefore log(Y+a) where a is the constant. Some people like to choose a so that min(Y+a) is a very small positive number (like 0.001). Others choose a so that min(Y+a) = 1. For the latter choice, you can show that a = b – min(Y), where b is either a small number or is 1.

In the SAS/IML language, this transformation is easily programmed in a single statement. The following example uses *b=1* and calls the LOG10 function, but you can call LOG, the natural logarithm function, if you prefer.

```
proc iml;
Y = {-3,1,2,.,5,10,100}; /** negative datum **/
LY = log10(Y + 1 - min(Y)); /** translate, then transform **/
```

Solution 2: Use Missing Values

A criticism of the previous method is that some practicing statisticians don't like to add an arbitrary constant to the data. They argue that a better way to handle negative values is to use missing values for the logarithm of a nonpositive number.

This is the point at which some programmers decide to resort to loops and IF statements. For example, some programmers write the following inefficient SAS/IML code:

```
n = nrow(Y);
LogY = j(n,1); /** allocate result vector **/
do i = 1 to n; /** loop is inefficient **/
   if Y > 0 then LogY[i] = log(Y);
   else LogY[i] = .;
end;
```

The preceding approach is fine for the DATA step, but the DO loop is completely unnecessary in PROC IML. It is more efficient to use the LOC function to assign `LogY`, as shown in the following statements.

```
/** more efficient statements **/
LogY = j(nrow(Y),1,.); /** allocate missing **/
idx = loc(Y > 0); /** find indices where Y > 0 **/
if ncol(idx) > 0 then
   LogY[idx] = log10(Y[idx]);

print Y LY LogY;
```

Y	LY	LogY
-3	0	.
1	0.69897	0
2	0.7781513	0.30103
.	.	.
5	0.9542425	0.69897
10	1.146128	1
100	2.0170333	2

The preceding statements initially define `LogY` to be a vector of missing values. The LOC function finds the indices of `Y` for which `Y` is positive. If at least one such index is found, those positive values are transformed and overwrite the missing values. A missing value remains in `LogY` for any element for which `Y` is negative.

You can see why some practitioners prefer the second method over the first: the logarithms of the data are unchanged by the second method, which makes it easy to mentally convert the transformed data back to the original scale (see the transformed values for 1, 10, and 100). The translation method makes the mental conversion harder.

You can use the previous technique for other functions that have restricted domains. For example, the same technique applies to the SQRT function and to inverse trigonometric functions such as ARSIN and ARCOS.

tags: *Data Analysis*, *Efficiency*, *Statistical Programming*

- Bookmark on Delicious
- Digg this post
- Recommend on Facebook
- Share on FriendFeed
- Share on Linkedin
- share via Reddit
- Share with Stumblers
- Tweet about it
- Share on xing
- Print for later
- Tell a friend

46 Comments

Rick Wicklin
Posted June 2, 2011 at 2:02 pm | *Permalink*

Did you know that SAS now has a LOG1PX function that "returns the log of 1 plus the argument"? It's true!
http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a003121132.h

Reply

Saumya
Posted September 1, 2011 at 6:47 am | *Permalink*

Dear Rick
My data set includes stock return of around 1000 companies. In most cases sometimes the return data shows a -34.5 to -108 figures. How to make log transformation in this case. How much should be the constant value in this kind of data. Please help.

Reply

Rick Wicklin
Posted September 1, 2011 at 7:50 am | *Permalink*

It depends somewhat on what you're trying to do, but you might want to express the returns as a percentage, measured from the start of the time period (1 yr, 5 yrs, or whatever). Then the Negative returns are bounded by -100 percent, and you can safely compute log(101 + return).

Reply

Gjermund Grimsby