Graphing Them Together: Overlaying Plots with Macros and SAS/GRAPH®

Lawrence Altmayer
U.S. Census Bureau

ABSTRACT

SAS/GRAPH is a powerful tool for creating plots of series of data. When we want to plot more than one series of data on a set of axes, using multiple symbol statements allows us to do this. This paper demonstrates a way of using macros to concatenate several data sets into one and prepare it for the overlay, since SAS/GRAPH can only read one per plot. We then use macros to specify the symbols to use, and macro looping to generate the actual statements. The paper also gives details on titling. Finally, we discuss how to use the symbol statement to specify the size and *type* of symbol, which may not be as straightforward as it seems...

INTRODUCTION

The graphing procedure described in this paper is used for plotting series of data processed in the U.S. Census Bureau's Time Series Analytic Repository (TSAR). There are a few levels of programs, required for specifying various setup variables and the data set for graphing using SAS/GRAPH. This paper gives the essential lines of macro code a user would have to run to get the final overlapping graphs. Sample data and code for generating the graph is available from the author on request.

The programs required to generate the graph include:

- a batch control file (bcf), called by a one line UNIX® script
- a "driver" program, which sets up some necessary input variables, and the data set for input to SAS/GRAPH
- graph macro, containing SAS/GRAPH code for generating the overlay plot, using the above data set

GRAPH DRIVER PROGRAM

The graph driver program performs several tasks. One DATA step creates macro variables used in defining the horizontal axes and reference lines for the graph. Another DATA step inputs the graph processing control input file. In this DATA step, macro variables for each series (name, lower and upper bounds, and increment) are created. A macro variable for the total number of series is also defined. The code is as follows:

```
data temp;
   infile &grphpcif end=lastone missover;
input series $ gfrom gto gby;
trimn=trim(left(put(_n_,4.)));
call symput('series'||trimn, series);
call symput('gfrom'||trimn,
trim(left(gfrom)));
call symput('gto'||trimn, trim(left(gto)));
call symput('gby'||trimn, trim(left(gby)));
if lastone then call
   symput('numsers',trimn);
      run;
```

Finally, since SAS/GRAPH requires one data set at a time as input, we must concatenate the data sets for the series into one. This is done with the following set of code:

```
❶     data series;
❷       set %do i=1 %to &numsers;
           &survey..&&series&i %end;;
❸       %do i=1 %to &numsers;
❹         if &&series&i^=. then do;
❺           grp="&&series&i";
❻           series=&&series&i;
❼           end; %end;;
❽       drop %do i=1 %to &numsers;
❾          &&series&i %end;;
❿     run;
```

In this code, we use a SET statement with macro code to generate the names of the individual data sets we are concatenating, in line 2. Next, we use example 8 (Plotting Three Variables) from the SAS version 8 online documentation help for SAS/GRAPH PROC GPLOT, which describes how to use a categorical variable, corresponding an individual series name in this case, to produce one data set necessary for the SAS/GRAPH input. This code is shown in lines 4-7. Finally, since we no longer need the variables indicating the original individual series names, we drop these, as in lines 8-9.

GRAPH MACRO

The graph macro uses SAS/GRAPH (specifically, PROC GPLOT) to generate the overlay plots. We first need to generate the individual title associated with each plot line. For this, we have to obtain the title from a separate 'descriptor' file which contains this. For the various plots, different symbols are also needed to distinguish the various plot lines. The different symbols are generated using a list of symbols assigned to a macro variable with a %LET statement. After this, SYMBOL statements required for the individual plot lines are generated by a macro loop.

TITLES

Titles are obtained from a separate 'descriptor' file as mentioned above. We set up a data set with only the unique titles for the overlay plots of interest. We talk here only about title2 (there are 3 titles in the final set of plots).

In the code below, titl&i (i=2 for this iteration) contains one observation for each category associated with the 5 plots. The macro variable title&i is created with a CALL SYMPUT near the end of the DATA step. TRIMN is assigned values consecutively from 1, to the number of categories. On the last iteration of the DATA step, numtitl&i (i=2, here) gets the value of the total number of categories.

```
data titl&i (drop=trimn);
   set titl&i end=last&i;
   by title&i;
   if first.title&i;
   trimn=trim(left(put(_n_,4.)));
   call symput("title&i"||trimn,
     trim(left(title&i)));
   if last&i then
     call symput("numtitl&i",trimn);
run;
```

Finally, the following code (for i=2):

```
title2 h=1.5 f=zapfi
   %do j = 1 %to &numtitl2; &&title1&j
      %if &j ^= &numtitl2 %then ,;
   %end;;
```

generates the following TITLE statement:

```
title2 h=1.5 f=zapfi Total Building
Materials Group, Total Durable Goods
Stores, Total Durable Goods Stores
Excluding Automotive Dealers(5008-
5500), Total Retail, Total Retail
Excluding Automotive (5005-5500);
```

for this example.

SYMBOLS

We first associate the list of symbols we want to use, to a macro variable with a %LET statement, as follows:

```
%let valulist=square diamond triangle
   plus x;
```

If the number of series has been assigned previously to the graph macro call, we can generate the SYMBOL statements with the following macro loop:

```
%do i = 1 %to &numsers;
   symbol&i i=joinv=%scan(&valulist,
   &i,%str( )) h=0.8 c=black;
%end;
```

For &numsers = 5, this generates the following series of statements:

```
symbol1 i=join v=squareh=0.8 c=black;
symbol2 i=join v=diamond h=0.8 c=black;
symbol3 i=join v=triangle h=0.8 c=black;
symbol4 i=join v=plus h=0.8 c=black;
symbol5 i=join v=x h=0.8 c=black;
```

We need the c=black (for color) even if we are using a black and white printer, because otherwise, SAS assumes it should go through a predefined *series* of colors for each given symbol. For a black and white printer, this shows up as several plots with the same black and white symbol. By specifying c=black, SAS 'skips' its color list, and gives us just one color for the desired symbol (black, here) and the 5 different symbols we specify.

PROC GPLOT

The final set of overlay plots is generated with a PROC GPLOT with the following PLOT statement:

```
PROC GPLOT
data=series(where=(data>="01dec&gyy10"d));
 PLOT series*date = grp / href
  ="01dec&gyy9"d "01dec&gyy8"d ...
  "01dec&gyy1"d haxis=axis1 vaxis=axis2
  autovref lvref=35 frame;
run;
```

where grp is a categorical variable referring to the series being plotted here (of 5 series, in this case). The use of categorical variables is described in the SAS v8 online documentation as described above. Here, href refers to the horizontal axis specifications, and contains a series of ten date variables generated with previously defined macro variables. The PLOT statement also contains various other options required for the final plots.

REFERENCE

SAS Online Doc ® Version 8. Copyright (c) 1999 SAS Institute Inc., Cary, NC, USA

ACKNOWLEDGMENT

The author would like to thank Diane Roebuck for her helpful comments in the development of this paper.

AUTHOR

Lawrence Altmayer
U.S. Census Bureau
ESMPD, Room 1223-4, MS 6200
Washington, DC 20233-6200
(301)457-2581
Lawrence.W.Altmayer@census.gov