Organizing Your SAS® Graphs

Steven E. Elkin, MBH Consulting, Inc. New York, NY

ABSTRACT

Writing programs to run SAS/GRAPH® software has long been a challenging task for many programmers. Understanding how to display, store and print the output are all difficult issues which must be faced even after appropriate SAS/GRAPH syntax has been developed. By using standard GOPTIONS[i], the GOUT option to store graphs in catalogs, and PROC GREPLAY[ii] to name the graphical entries in the catalog, a simplified method for organizing your SAS graphs is achieved. This method allows for both on-screen displays and printouts of your graph. Additionally, once stored in a SAS catalog, future access of your graph is possible without the need to re-submit the program which originally created the graph.

INTRODUCTION

This method for organizing SAS graphs is for an interactive display manager (DMS) environment making it ideal for work on a P.C. where the SAS DMS is typically used. There are three major components to this method. Accessing a standard set of GOPTIONS, storing the graphs in a permanent SAS catalog, and renaming the cataloged graphical entries using PROC GREPLAY. The greatest efficiency is achieved when incorporating this procedure as part of a macro so multiple graphical entries may be renamed and stored within the same permanent catalog.

STANDARD GOPTIONS

I have found that a single set of default GOPTIONS may be used in many different types of graphs. I have used the following set in survival curves, vertical bar charts and fully annotated time-plot graphs.

```
goptions
 reset=global
 device=win
 target=lj3sipsl
 gouttype=independent
 ftext=swiss
 i = j
 rotate = landscape
 horigin=1.0 in
 vorigin=0.75 in
 hsize  = 8.5 in
 vsize  = 6.95 in
 hpos = 80
 vpos = 50
 gunit=pct
 cback=white
 colors=(black)
;
```

Storing these default GOPTIONS in a single file is advantageous for two primary reasons. First of all, with a simple %INCLUDE statement a working set of GOPTIONS may be accessed easily without having to repeatedly cut and paste GOPTIONS from other programs. Secondly and most importantly, when re-displaying existing graphs stored in permanent catalogs, it is necessary to first submit the GOPTIONS originally used in the creation of these graphs. Keeping GOPTIONS in a single file using a standard file reference (e.g. %inc std_gopt;) to access the file makes this process easy.

STORING GRAPHS IN PERMANENT CATALOGS

Using the GOUT option in a PROC GPLOT[iii], GSLIDE, GCHART, etc., allows you to store the graphics output in a named catalog. By default, GOUT is set to WORK.GSEG, that is, the SAS work directory with a catalog named GSEG. This file, as are all working data set files, would then be deleted once the current DMS session ended. The syntax needed to obtain a permanent SAS catalog is simply GOUT = <lib reference>.<file reference>; where the LIB and FILE references are set to point to directories and file names which can be stored on your local or network drives. If the catalog referenced in the GOUT option already exists, then SAS will create the graph as a new entry in the existing catalog; otherwise, SAS will first create a new catalog (a file with the name as the filename reference and an extension of .SC2) before creating the graphics entry.

RENAMING ENTRIES WITH PROC GREPLAY

When a new graphics entry is produced in a catalog, the SAS system automatically names it according to the SAS/GRAPH procedure used. GPLOT for PROC GPLOT, GSLIDE for PROC GSLIDE, GCHART for PROC GCHART, etc. For subsequent entries in the same catalog SAS adds numbers to the names in ascending order. That is, if three graphs are produced using PROC GPLOT and stored in the same catalog, they will be named, GPLOT, GPLOT1, and GPLOT2; with the first graph being GPLOT and the last graph being GPLOT2.

One of the features of PROC GREPLAY is that it can name or rename cataloged graphical entries. In this method, the IGOUT statement is used to access the

catalog containing the graphs the DELETE statement is used to delete an older version of the same named graph if it exists, and finally, the modify statement is used to rename the graph as desired.

CONCLUSION

SAS/GRAPHS may be easily organized and named in permanent catalogs by accessing a standard set GOPTIONS, using the GOUT option in a SAS/GRAPH procedure, and the GREPLAY procedure. Both on-screen displays and printouts of your graphs are possible under the method presented here. A key feature of this method is that once stored in a SAS catalog, future access of your graph is possible without the need to re-submit the program which originally created the graph. A robust method for naming and storing graphics output is achieved when combining this method with a simple MACRO and WHERE clause (see sample code below).

Sample Code

In the following example, PROC GPLOT is used to plot the variables Y and X, subsetting by different parameters using a WHERE clause. The example produces four plots stored in a SAS catalog which is a file named X_YPLOTS.SC2 located in the directory named X:\PROJECT\OUTPUT. The four graphical entries are MALES, FEMALES, YOUNGER, and OLDER.

```
*** include standard GOPTIONS;
%include 'x:\std_code\gopt1.sas';

*** assign library where catalog will reside;
libname outlib 'x:\project\output';

*** a macro is used to produce plots
subsetting  by different parameters via a
WHERE clause;
%macro graph(wclause,label);

*** GOUT assigns a catalog named X_YPLOTS
stored in the directory OUTLIB;

*** the where clause subsets for different
parameters;

proc gplot gout=outlib.x_yplots;
    where &wclause;
    plot y*x;
    run; quit;

*** access the appropriate catalog assigned
by the GOUT statement in the PROC GPLOT with
GREPLAY using the IGOUT statement;

 *** If the catalog already has an entry
named for the resolved value of &LABEL, it
will be deleted, otherwise a NOTE in the SAS
log will appear that the entry is not on the
igout catalog.;

*** The modify statement renames the entry
from "GPLOT" to the resolved value of &LABEL,
the DELETE statement ensures that this can
take place;

proc greplay nofs;
    igout outlib.x_yplots;
    delete &label.;
    modify gplot / name=&label.;
    quit;

%graph(sex eq 1, males)
%graph(sex eq 2, females)
%graph(age lt 50, younger)
%graph(age ge 50, older)
```

REFERENCES

[i] SAS Institute Inc. (1990), SAS/GRAPH Software: Reference, Version 6, First Edition, Volume 1, Cary, NC: SAS Institute Inc., pp 291-302.

[ii] SAS Institute Inc. (1990), SAS/GRAPH Software: Reference, Version 6, First Edition, Volume 2, Cary, NC: SAS Institute Inc., pp 1191-1258.

[iii] SAS Institute Inc. (1990), SAS/GRAPH Software: Reference, Version 6, First Edition, Volume 2, Cary, NC: SAS Institute Inc., pp 1073-1130.

Steven E. Elkin
251 West 92nd St., Apt. 2F
New York, NY 10025
Daytime: (201) 503-6868, e-mail: steveeelk@aol.com