ABSTRACT

INTRODUCTION
SAS/GRAPH software offers device-intelligent color graphics for producing charts, maps and plots in a variety of patterns. Users can customize graphs with the software, and present multiple graphs on a page. SAS/GRAPH software is a component of the SAS System, an applications system for data access, management, analysis, and presentation.

PLOTTING PROCEDURES

GCHART
The basic way to produce a vertical bar chart is as follows:

```
proc gchart data=sasuser.houses ;
vbar style ;
run ;
```

The previous chart produced a frequency chart (by default). We can change this to use a Y-axis variable by specifying SUMVAR:

```
vbar style / sumvar=price ;
run ;
```

If we want to use another variable to break each bar into sections, then we can use SUBGROUP:

```
vbar style / sumvar=price subgroup=baths ;
run ;
```

We can also divide bars into groups based on another variable by using GROUP:

```
vbar style / sumvar=price subgroup=baths group=bedrooms ;
run ;
```

To get a horizontal bar chart, rather than a vertical bar chart we can use the HBAR statement. This also produces some default statistics for each bar:

```
hbar style ;
run ;
```

To get a pie chart you use PIE statement:

```
pie style ;
run ;
```

To get a star chart use the STAR statement:

```
star style ;
run ;
```

GCONTOUR
Useful for viewing three dimensional data in two dimensions. Using a sample dataset, created with the following code:

```
data swirl;
do x=-5 to 5 by 0.25;
do y=-5 to 5 by 0.25;
if x+y=0 then
z=0;
else
z=(x*y*)((x*x-y*y)/(x*x+y*y));
output;
end;
end;
run;
```

We can produce a contour plot with this code:

```
proc gcontour data=test ;
plot y*x=z ;
run ;
```
We can use a pattern to make things more legible:

```latex
plot y*x=z / pattern;
run;
```

**G PLOT**

Make some sample data:

```latex
data sample;
do z=100 to 300 by 100;
do x=1 to 5;
y=ranuni(1)*10;
y2=ranuni(1)*10;
y3=ranuni(1)*10;
output;
end;
end;
run;
```

To plot some points use:

```latex
proc gplot data=sample;
where z=100;
plot y*x;
run;
```

To plot lines rather than points use:

```latex
symbol i=join;
plot y*x;
run;
```

To produce multiple plots on a single axis:

```latex
proc gplot data=sample;
plot y*x=z;
run;
```

To produce a graph for each combination of X & Y axis variables you can use the following, which in the example would produce 3 graphs (3 * 1):

```latex
where z=100;
plot (y y2 y3)*x;
run;
```

The 3 graphs from the previous example can be overlayed onto one graph by using the OVERLAY statement:

```latex
plot (y y2 y3)*x / overlay;
run;
```

To use independent right & left y-axes you can use the following:

```latex
proc gplot data=sample;
where z=100;
pplot y*x;
pplot2 y2*x;
run;
```

To do a simple bubble plot use this code:

```latex
proc gplot data=sample;
bubble y*x=z;
run;
```

To get a bubble plot with a right & left axis use this code:

```latex
bubble y*x=z;
bubble2 y2*x=z;
run;
```
G3D

Produces 3-dimensional graphs graphics two horizontal variables against one vertical variable. Using the following sample data:

```plaintext
data hat;
do x=-5 to 5 by 0.25;
do y=-5 to 5 by 0.25;
z=sin(sqrt(x*x+y*y));
output;
end;
end;
run;
```

We can produce a well known plot:

```plaintext
proc g3d data=hat;
plot y*x=z ;
run;
```

Or we can produce a scatter plot:

```plaintext
proc g3d data=hat;
scatter y*x=z ;
run;
```

G3GRID

Produces datasets for use with G3D or GCONTOUR. Can be used for interpolation and smoothing. If we create some “rough” data by taking half of the coordinates from our smooth data, we get the following:

```plaintext
data rough ;
set hat ;
if ranuni(1)<.5 then
    output ;
run ;
proc g3d data=rough ;
plot y*x=z ;
run;
```

Then we can use G3GRID to smooth the data we have:

```plaintext
proc g3grid data=rough out=smooth ;
grid y*x=z ;
run ;
proc g3d data=smooth ;
plot y*x=z ;
run;
```

GMAP

Produces two or three dimensional maps showing variations of a variable value with area. Supplied with a library of maps covering countries of the world and the U.S.A. in more detail.

To produce a choropleth map of Great Britain:

```plaintext
proc gmap map=maps.ukire
    data=maps.ukire2 ;
    id region ;
    choro region ;
run ;
```

To produce a block map:

```plaintext
block region ;
run ;
```

To produce a Prism map:

```plaintext
prism region ;
run ;
```

To produce a surface map:

```plaintext
surface country ;
run ;
```

GPROJECT

Converts spherical (latitude/longitude) coordinates into catesian (xy) coordinates.
GREduce
Reduces number of points needed to draw a map, and thus reduces detail within map too.

```sas
proc gmap map=maps.canada2 data=maps.canada2;
  id province;
  choro province;
run;
```

```sas
proc greduce data=maps.canada2
  out=can2(where=(density<3));
  id province;
run;
```

```sas
proc gmap map=can2 data=can2;
  id province;
  choro province;
run;
```

GREMOVE
Combines some unit areas in a map into larger area. See code in SAS sample library called - "Removing Internal Boundaries in a Map - GR35N01".

Before boundaries are removed.

![Map before boundaries removed](image1)

After boundaries have been removed.

![Map after boundaries removed](image2)

PRESENTATION PROCEDURES

GREPLAY
Replays graphics output using templates, allowing several graphics to be combined. The following code will display a screen from which templates may be chosen and graphics selected for use with them.

```sas
proc greplay igout=sashelp.eisgrph
gout=work.replays
tc=sashelp.templt
template=12r2
nofs;
  treplay 1:csf1 2:contract 3:exit 4:barchart;
run;
```

A collection of useful templates is provided in SASHELP.TEMPLT, however the user can create their own by using the procedure.

GANNO
Displays the graphics produced by the processing of annotate datasets. Many other procedures can also display annotate dataset output, along with their own output.

There are a range of macros available to make creating annotate datasets easier. To make them available use the following:

```sas
%annomac;
```

This displays the following output:

```sas
*** ANNOTATE macros are now available ***
For further information on ANNOTATE macros, enter, %HELPANO(macro), (for specific macros) %HELPANO(ALL), (for information on all macros) or %HELPANO (for a list of macro names)
```

To produce a circle and then display it:

```sas
data anno;
  %circle(10,20,5,*);
run;
proc ganno annotate=anno;
run;
```

GSLIDE
Can display graphics consisting of text and straight lines generated by TITLE, FOOTNOTE & NOTE statements. It can also generate data from ANNOTATE datasets.

```sas
proc gslide;
title1 h=6 This is a title on a slide;
note1 h=3 j=c Some text in the middle;
note2 h=4 j=c and some more;
footnote h=2 A footnote;
run;
```

GPRINT
GPRINT can convert text into graphics. To write some SAS procedure output to a catalog member you can do the following.

```sas
* Direct print to catalog member;
proc printto print=work.output.print;
run;
```
* Produce some output, which is written to catalog member:
  
  ```
  data=sasuser.houses;
  run;
  ```

* Close the output:
  ```
  proc printto;
  run;
  ```

To take the text produced and convert into one or more graphics, do the following:

* Point to the saved output:
  ```
  filename temp catalog 'work.output.print.output';
  ```

* Print output as a graphic:
  ```
  proc gprint fileref=temp;
  run;
  ```

Utility Procedures

**GDEVICE**

View, modify or create device drivers. Can use the interactive procedure as follows:

```
proc gdevice;
run;
```

Can list a device driver in batch as follows:

```
proc gdevice nofs;
list imggif;
run;
```

The output shows the device information:

```
GDEVICE procedure
Listing from SASHELP.DEVICES - Entry IMGGIF
Orig Driver: IMGGIF             Module:   SASGDIMG  Model:     3031
Description: Graphics Interchange Fmt--256 colors        Type: EXPORT
*** Institute-supplied ***
Lrows:  43  Xmax:   6.474 IN    Hsize:    0.000 IN  Xpixels:        615
Lcols:  76  Ymax:   3.631 IN    Vsize:    0.000 IN  Ypixels:    345
Prows:   0                      Horigin:  0.000 IN
Pcols:   0                      Vorigin:  0.000 IN
Aspect:   0.000                 Rotate:
Driver query:                   Queued messages: N
Paperfeed:   0.000 IN
OPTIONS
Erase:                   Autofeed:                Chartype:    0
Keep:                    Cell:                    Characters: 256
Autocopy:                Hardcopy:        0
Dash:                    hand:                   Autofeed:
Prompt - startup:        Hard:                     Paint:     0
end graph:        Fill:                    Fillinc:     1
mount pen:        Polyfill:                Maxpoly:     0
chg paper:        Symbol:                  Lfactor:     0
Pensort:       N
Devopts:      '1100100009200000'X
UCC:      '4749460000000000'X
Cback:      BLACK               Colortbl:
Color list:
  WHITE     RED       GREEN     BLUE      CYAN
  MAGENTA   YELLOW
CHARTYPE RECORD
Chartype Rows  Cols                  Font Name                 Scalable
FILE INFORMATION
Name:  testgraph.gsf
Options:  None                  Delimit:  PORT   Size:  604
Beginning Tutorials
Display, modify or create fonts.
To view a font use:

```
proc gfont name=swiss nobuild;
run;
```

Can create fonts by making a dataset which defines the coordinates and then using PROC GFONT:

```
data figures;
input char $ ptype $ x y segment lp $;
cards;
A W 0 64 0 P
A V 4 4 1 P
A V 60 32 1 P
A V 4 60 1 P
A V 4 4 1 P;
run;
```

```
libname gfont0 'c:\' ;
/* generate and display the font FIGURES */
proc gfont data=figures
name=figures;
run;
```

**GIMPORT**

Import graphics which are in CGM format, if you can get anything in that format! I couldn’t find any application that can save graphics in that format.

**GKEYMAP**

Generate characters not available on keyboard.

**GOPTIONS**

List all graphics options:

```
proc goptions;
run;
```

SAS/GRAPH software options and parameters

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOADMGDF</td>
<td>GDDM driver output an ADMGDF file</td>
</tr>
<tr>
<td>ASPECT=</td>
<td>Aspect ratio (width/height) for software characters</td>
</tr>
<tr>
<td>NOAUTOCOPY</td>
<td>Automatic hardcopy after display</td>
</tr>
<tr>
<td>NOAUTOFEED</td>
<td>Automatic paper feed after plot</td>
</tr>
</tbody>
</table>

```
NOTE: The PROCEDURE GOPTIONS used 0.55 seconds.
```

**GTESTIT**

A diagnostic tool to test configuration of devices by generating some test graphics.

```
proc gtestit;
run;
```
Video editing software, also allowing text compositing. Invoke the editor using:

```
Proc vedit ; run ;
```

**SUB-LANGUAGES**

**ANNOTATE**

Facility which allows the storage of graphics commands in data sets. These can then be used in conjunction with various procedures to produce or enhance graphics. The basic commands allow the drawing of: bars, lines, points, frames, pies, slices, polygons & text.

A range of macros are provided to ease the creation of ANNOTATE datasets. To make these macros available you must issue the following:

```
%annomac ;
```

The macro statements allow the drawing of various graphic objects with a single macro invocation for each object, for example:

```
data anno ;
   %bar(10,10,20,30,red,1,solid) ; run ;
proc ganno anno=anno ; run ;
```

**DSGI (DATA STEP GRAPHICS INTERFACE)**

Allows graphics output to be produced directly from a data step or SCL program. You can either create an entire graph with DSGI, or use it to enhance existing graphics. DSGI allows creation of the following graphics elements: arcs, bars, ellipses, elliptical arcs, lines, markers, pie slices, polygons, & text. The following code will draw a line on a blank graphic:

```
data dsgi ;
   rc=ginit() ;
   rc=graph('clear','text') ;
   rc=gdraw('line',2,30,50,50,50) ;
   rc=graph('update') ;
   rc=gterm() ;
   run ;
```

**FONTS**

There are a range of SAS/Graph fonts provided which can be found in SASHELP.FONT. A typical specification would be:

```
Title font=xswiss 'Expenditure Report' ;
```

**GRAPHICS STATEMENTS & WINDOWS**

There are a range of statements which affect many of the procedures described above. Here are some examples of what can be done.

**AXIS statement**

Up to 99 definitions of characteristics of an axes. Used by calling from procedure.

```
Axis1 order=(1 to 100 by 25) color=red
   label=('Percent Complete')
   major=(height=2 width=.5)
   minor=(number=1 height=2)
   offset=(0,0) width=2.5 ;
proc gplot data=test ;
   plot y*x / vaxis=axis1 ; run ;
```

**FOOTNOTE statement**

Defines all characteristics of footnotes. Stays in effect until overridden or cancelled.

```
Footnote justify=left 'Left text'
   j=center box=1 angle=45 'Cent.' ;
```

**GOPTIONS statement**

Allows setting of all graphics related options, so that default values can be overridden. Graphics options are reset to their defaults using:

```
Goptions reset=goptions ;
```

**LEGEND statement**

Allows customizing of Legends used with several graphics procedures. Legends are called from other procedures.

```
Legend1 across=1 down=3 cborder=red
   Position=(bottom inside center)
   Mode=share label=none ;
proc gchart data=test ;
   vbar y / legend=legend1 ; run ;
```

**NOTE statement**

Notes are similar to FOOTNOTE and TITLE statements, however they appear in the middle of a graphic, rather than the top (as for a TITLE) or bottom (as for a FOOTNOTE).

```
Note h=4 move=(15,70) 'My report'
   Move=(50,70) 'some data' ;
```

**PATTERN state m ent**

Defines characteristics of patterns used in graphs. Pattern definitions are used in order as patterns are required, starting at Pattern1. Various options are available including the ability to specify the angle and thickness of lines used in a pattern:

```
Pattern1 color=red value=m3x45 ;
```

**SYMBOL statement**

Specifies characteristics of symbols used to display data plotted in the GCHART procedure. This allows various types of plots to be produced including: scatter, high/low, regression, box, join, needle, spline & step:

```
Symbol1 c=red interpol=spline v=star ;
```

**TITLE statement**

Similar to FOOTNOTE & NOTE statements, except placing text at top of graphic:

```
Title j=c c=blue h=5 'My report' ;
```

**CONCLUSION**

SAS/Graph is a large and powerful product, encompassing a range of procedures which enable incredibly flexible graphics to be produced. It also contains the ANNOTATE facility for enhancing graphics and the DSGI language for drawing almost anything. This all adds up to a tool which can produce virtually any graphic image. It is particularly well suited over other graphic products where any of the following are required: automation, multi-platform support, large volume production.

**REFERENCES**

SAS/Graph Software: Volume 1, Reference, Version 6, First Edition
SAS/Graph Software: Volume 2, Reference, Version 6, First Edition

**CONTACT INFORMATION**

Your comments and questions are valued and encouraged. Contact the author at:

Philip Mason
Wood Street Consulting
16 Wood Street
Wallingford, Oxfordshire, OX10 0AY, ENGLAND
Work Phone: +44 1491 834615
Fax: +44 1491 834615
Email: Phil_mason@email.com