SAS GRAPH BASICS

Many times when examining data or reviewing results of analyses, graphic plots are desired. SAS BASE provides the PROC PLOT and PROC CHART procedures, but these are character plots that often lack adequate resolution and can be cumbersome to read. A better option for quickly producing graphic output without resorting to other software is SAS GRAPH. This is a group of special procedures in SAS that can create high quality graphics which an be viewed directly on the screen or saved to graphic files.

While SAS GRAPH contains several procedures, only the following will be covered here:
PROC GPLOT, PROC GCHART, PROC G3D, and PROC GCONTOUR. These are the most commonly used procedures in data exploration and analysis.

PROC GPLOT - two dimensional line and scatter plots

Required Statements:

PLOT - The only required statement is PLOT. This statement tells GPLOT what variables you want plotted and how to display them. The form of the statement is:

```
PLOT <yvar>*<xvar> / options ;
```

where <yvar> and <xvar> may be single variable names or lists of variable names.

The <*yvar*> will be on the vertical axis and <*xvar*> on the horizontal.

Examples would be

```
PLOT y1*x1;
PLOT (y2 y2 y3 y4)*time;
PLOT (y1 y2 y3 y4)*(x1 x2 x3 x4);
```

Note that variables are separated by spaces and surrounded with parentheses. The last two examples plot all possible combinations of x and y variables.

Options:

These are given after the slash at the end of the PLOT statement. They control various aspects of the specified plot. Two popular options are:

OVERLAY - Used to plot multiple variables on one graph. Suppose you want to plot variables y and z versus variable x. The following is used:

```
PLOT y*x z*x / OVERLAY;
```

VREF and HREF - Used to draw vertical and horizontal reference lines at specified values. For example, To draw a vertical reference line at x=129.36, use:

```
PLOT y*x / VREF = 129.36;
```

PROC GCHART

Required Statements:

VBAR or HBAR - Produces either vertical or horizontal bar charts. The syntax for VBAR and HBAR is the same, namely:

```
VBAR <variables> / options;
HBAR <variables> / options;
```

Here <*variables*> is one or more variables of interest. As an example,

```
VBAR x1 x2;
```

produces a vertical bar chart for variables x1 and x2.

PIE - Produces a pie chart for specified variables. The form is:

```
PIE <variables> / options;
```

General Options:

FREQ - FREQ is used when one variable represents the levels to plot and another represents the actual counts or frequencies.

```
VBAR x1 / FREQ=num_alive;
```

VBAR/HBAR Options:

Several options exist for the VBAR and HBAR statements. Some of the more important ones are FREQ, PERCENT, CFREQ and CPERCENT. These define the bars to represent frequencies, percentages, or their respective cumulative counterparts, respectively and are used with categorical data. Examples:

```
VBAR x1 /FREQ;
HBAR location /CPERCENT;
```

For continuous data, e.g. yield, weights, etc., the options SUM and MEAN can be used in conjucntion with SUMVAR. An example in this case may be:

```
VBAR variety / MEAN SUMVAR=yield;
```

This makes a vertical bar chart for each variety. The bar heights represent the mean values for yield.

PIE Options:

Options here might be FILL=S to specify colored slices or FILL=X which gives cross-hatched slices. The FREQ=<*variable*> option may also be used here.

```
PIE mortality / FILL=S;
```

PROC G3D

Required Statements:

PLOT - Like GPLOT, the PLOT statement indicates what variables to put on the graph. PROC G3D however, plots three dimensional data. The form of the statement is:

PLOT ***<y variable>*<x variable> = <z variable> / options;***

The <*x variable*> and <*y variable*> represent the base of the plot while <*z variable*> provides the height of the plot at each x,y combination. You must have z values for every x,y combination. (If not, the procedure G3GRID can be used to interpolate between missing points). By default, this will draw a surface over the X-Y plane. Example:

```
PLOT dose*time = mortality;
```

SCATTER - Produces a 3 dimensional scatter plot of data points. The syntax is the same as PLOT.

General Options:

ROTATE and **TILT** - Determines the angle from which the 3D plot is viewed. Values are given in degrees.

```
PLOT x1*y1 = z1 / ROTATE=30 TILT = 80;
```

PLOT Options:

CTOP and **CBOTTOM** - Sets the color of surface top and bottom, respectively.

```
PLOT velocity*time = weight / CTOP=blue CBOTTOM=green;
```

XYTYPE - Surface is depicted with a grid mesh. By default this is shown for both x and y axes (XYTYPE=3). XYTYPE=1 and XYTYPE=2 result in grid lines for the x or y axes, respectively.

```
PLOT x*y=z / XYTYPE=1;
```

SCATTER Options:

NONEEDLE - Suppresses the vertical line connecting the data point and the X-Y plane. By default the line is drawn.

SHAPE - Specifies the shape to be drawn for each point. Some valid values are DIAMOND, CLUB, PILLAR, and HEART. Shape can also refer to a forth variable that contains the desired shape as its value.

GRID - Draws a grid on the X-Y plane.

```
SCATTER x1*y3=z7 / NONEEDLE SHAPE=HEART;
```

PROC GCONTOUR

Required Statements:

PLOT - The form of GCONTOUR statements is very similar to G3D. The PLOT statement again specifies what variables to plot. The output in this case is a contour plot.

PLOT <*y variable*>*<*x variable*> = <*z variable*> / *options*;

Options:

LEVELS - Can be used to specify the levels of the contour:

```
PLOT x*y = z / LEVELS=0 to 100 by 10;
PLOT x*y = z / LEVELS=10, 45, 60, 97;
```

The first statement will produce contour lines at 0, 10, 20, etc up to 100. The second creates contour lines at 10 45 60 and 97.

VREF and **HREF** - As with GPLOT, reference lines can be drawn on the plot using the VREF and HREF options, e.g.:

```
PLOT x*y = z / HREF=10, 20, 30, 40;
PLOT x*y = z / VREF = 120.9 to 140 by .1;
```

Other Statements:

SYMBOL*x* - The SYMBOL statement gives greater flexability in producing plots with GPLOT. It allows the data points to be plotted as various symbols, as well as joining of points with lines (solid and dashed). Colors of lines and points can also be set. Up to 10 SYMBOL statements can be defined as SYMBOL1 through SYMBOL10. The general syntax is:

```
SYMBOLx I=<interpolation> V=<value> C=<color> L=<line type>;
```

where the options are:

I= determines how or if data points are joined. Common values are NONE and JOIN for no interpolation and straight lines, respectively.

V= sets the plotting symbol. Try Diamond, Circle, Square, and Plus.

C= sets color. Common colors like blue, green, orange, etc can be used.

L= describes the line type and is optional. Default value is L=1 for a solid line. Higher values give various combinations of dotted and dashed lines.

Symbols can be defined anywhere in a SAS program, but must appear before they are used. An example of their use would be:

```
SYMBOL1 I=NONE V=DIAMOND C=BLACK;
SYMBOL2 I=NONE V=CIRCLE C=GREEN;

PROC GPLOT;
     PLOT y*x=1 y2*x=2/OVERLAY;
```

This produces a single plot with y values indicated by black diamonds, and y2 values indicated by green circles. A third variable can also be used with symbols. For example:

```
SYMBOL1 I=NONE V=DIAMOND C=BLUE;
SYMBOL2 I=NONE V=DIAMOND C=GREEN;

PROC GPLOT;
     PLOT y*x=year;
```

If the variable year has values of 1997 and 1998 then the plot will have diamonds representing y versus x with 1997 values blue and 1998 values green. These techniques can be very useful in data exploration.