ABSTRACT

The production of report-quality graphics in an efficient manner is very important to most analysts. SAS/GRAPH® and especially PROC GPLOT have a number of options and added features that allow the user to construct high-quality plots quickly and easily.

Occasionally, however, the user may need to create a plot with requirements that do not match the built-in features and options. When this happens, the programmer must reach into the proverbial bag of tricks. There are several techniques or tricks that can be used to enhance plots produced by PROC GPLOT. Many of these can be easily and quickly applied even by a new or less experienced programmer. One of the more useful of these techniques (even for an experienced user) involves the addition of points to the plot. When properly coordinated with the SYMBOL statements, these additional points can produce axes, confidence bounds, histograms and special plots.

This paper discusses programming techniques for the inclusion of points into the plotted data set. Methods are shown which allow the user to maintain data dependent control over the graph without using macros or the ANNOTATE facility. The coordination of created variables and data points with the SYMBOL statement is also discussed.

INTRODUCTION

The GPLOT procedure in SAS/GRAPH is one of the most flexible plotting procedures in the SAS/GRAPH package of programs. In addition to a large number of PROC and PLOT options, the procedure also supports AXIS statements for both axes and a LEGEND statement. For the plot itself, the SYMBOL statements allow the user a great deal of control over the plotted symbols and lines. Even with all this control there are times when a particular plot cannot be generated with the available options.

Most programmers develop a set of tools or tricks to customize their plots. One of these tools or tricks is graph control through the use of plotted points which have been added to the original data. When done properly, these points can be used to generate error bars, confidence bounds, histograms and special purpose plots. The primary alternative to this approach is the ANNOTATE facility which allows even more flexibility, but is more difficult to master. The points addition technique is quickly and easily applied and allows the user to gain a large amount of control without the need to master either ANNOTATE or MACRO.

APPROACH

Primary control of the graph portion of a GPLOT plot is through the use of SYMBOL statements. These statements allow the user to connect points and specify the character to use when plotting points. All of the figures used in this paper make use of various aspects of the SYMBOL statement. When the user adds points to the plotted data set and controls how they are plotted using SYMBOL statements, even more flexibility is achieved.

The data used throughout this paper is taken from a study of a type of sea star known as patiria. Five transects, each about fifty meters in length, were swum and the location and abundance of patiria were noted. The first twenty observations in the data set are shown in Table 1. The means and two standard errors for abundances grouped into five meter segments are shown in Figure 1. In this figure the symbol statement:

```
SYMBOL1 V=NONE I=STD2MJ C=RED L=1;
```

was used. The I=STD2MJ calculates the mean and standard error for each segment.

<table>
<thead>
<tr>
<th>DATE</th>
<th>SEGMENT</th>
<th>TRANSCODE</th>
<th>PATIRIA</th>
<th>DISTANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2JUN81</td>
<td>1</td>
<td>A</td>
<td>0.19999</td>
<td>1.0406</td>
</tr>
<tr>
<td>2JUN81</td>
<td>2</td>
<td>A</td>
<td>0.59999</td>
<td>1.053</td>
</tr>
<tr>
<td>2JUN81</td>
<td>3</td>
<td>A</td>
<td>0.59999</td>
<td>1.057</td>
</tr>
<tr>
<td>2JUN81</td>
<td>4</td>
<td>A</td>
<td>0.59999</td>
<td>1.057</td>
</tr>
<tr>
<td>2JUN81</td>
<td>5</td>
<td>A</td>
<td>0.59999</td>
<td>1.057</td>
</tr>
<tr>
<td>2JUN81</td>
<td>6</td>
<td>A</td>
<td>0.59999</td>
<td>1.057</td>
</tr>
<tr>
<td>2JUN81</td>
<td>7</td>
<td>A</td>
<td>0.59999</td>
<td>1.057</td>
</tr>
<tr>
<td>2JUN81</td>
<td>8</td>
<td>A</td>
<td>0.59999</td>
<td>1.057</td>
</tr>
<tr>
<td>2JUN81</td>
<td>9</td>
<td>A</td>
<td>0.59999</td>
<td>1.057</td>
</tr>
<tr>
<td>2JUN81</td>
<td>10</td>
<td>A</td>
<td>0.59999</td>
<td>1.057</td>
</tr>
</tbody>
</table>

TABLE 1
and joins the points for the plot. This plot is quick and easy but has some limitations. Symbols such as squares cannot be added on the mean and upper and lower bounds (V=SQUARE plots all the data thus obscuring the error bars) and the plotted data points influence the scaling of the vertical axis. In addition, this technique cannot be used if the means have already been calculated.

![Figure 1: Error Bars Using 1=STD2MJD](image)

These inconveniences can be avoided by controlling the plot directly. PROC MEANS is used to calculate the means and standard errors prior to the GPLOT step. Each observation in the data set created by PROC MEANS contains the mean and the standard error for that segment. The desired plot (Figure 2) has three points for each segment (actually there are four as the mean is plotted twice as we shall see shortly). The one point is expanded to four by the use of the following statements (all the code used to generate the figures is attached at the end of the paper):

```plaintext
KEEP SEGMENT MEAN;
  * OUTPUT THE MEAN;
  OUTPUT;
  * OUTPUT THE MEAN PLUS 2 STD ERRORS;
  MEAN = MEAN + 2*SE;
  OUTPUT;
  * OUTPUT THE MEAN LESS 2 STD ERRORS;
  MEAN = MEAN - 2*SE;
  OUTPUT;
  OUTPUT THE MEAN A SECOND TIME;
  MEAN = MEAN + 2*SE;
  OUTPUT;
```

The mean is plotted twice (first and last) so that the line which joins the segments will join them at the segment mean. The symbol statement used is:

```
SYMBOL1 V=SQUARE 1=JOIN C=RED L=1;
```

![Figure 2: Error Bars Using Generated Points](image)

Multiple lines may be placed on a single plot by adding a third variable to the PLOT statement. Figure 3 notes the location of each aggregation of patiria on each transect. This was done simply by creating a transect counter (TRANCODE, see Table 1, is not numeric and cannot be used as a vertical variable). PROC FORMAT converts this "artificial" variable back to values that are easily understood by the user of the plot. The plot and symbol statements used were:

```plaintext
PROC FORMAT;
  VALUE TRAN 1='A' 2='B' 3='C' 4='D' 5='E';

DATA Pl; SET PATIRIA;
  BY TRANCODE;
  IF FIRST.TRANCODE THEN TRANSECT + 1;
  FORMAT TRANSECT TRAN.;
  PROC GPLOT DATA=Pl GOUT=DBPLOT.PATIRIA;
  PLOT TRANSECT * DISTANCE = TRANSECT;
  SYMBOL1 V=SQUARE 1=JOIN C=RED L=1 R=5;
```

The R=5 in the SYMBOL statement allows one symbol statement to be repeated or reused five times.

![Figure 3: Sampled Transects](image)

Although Figure 3 shows the location of the aggregations, it does not show their relative magnitude. Figure 4 depicts the plot of the location and magnitude of each aggregation for each transect. The resulting mess contains all the information but in an unusable format. At best this plot separates the transects by using trancode in the plot.
Each value of tranocode is automatically tied to successive SYMBOL statements which have the transect code as the plot character.

```
PLOT PATIRIA * DISTANCE = TRANCODE;
SYMBOL1 V='A' F=SIMPLEX I=JOIN
   C=RED L=1;
SYMBOL2 V='B' F=SIMPLEX I=JOIN
   C=RED L=1;
SYMBOL3 V='C' F=SIMPLEX I=JOIN
   C=RED L=1;
SYMBOL4 V='D' F=SIMPLEX I=JOIN
   C=RED L=1;
SYMBOL5 V='E' F=SIMPLEX I=JOIN
   C=RED L=1;
```

The transects can easily be separated by adding a constant (determined by the transect number) to each abundance value.

```
DATA PI; SET PATIRIA;
BY TRANCODE;
IF FIRST.TRANCODE THEN TRANSECT = 1;
   PATIRIA = PATIRIA + (TRANSECT-1)*10;
PROC GPLOT DATA=PI GOUT=DBPLOT.PATIRIA;
   PLOT PATIRIA * DISTANCE = TRANCODE;
```

The values of PATIRIA are not adjusted for transect 'A', however the second transect, 'B', has ten added to every value of PATIRIA, 20 is used for the third transect etc. In this way the transect lines no longer overlap. The selection of the constant (10 in this case) was based on inspection of Figure 4, however in most applications the data itself would be used to determine the constant (using the MAX= option in PROC MEANS is one way).

```
DATA PI; SET PATIRIA;
BY TRANCODE;
IF FIRST.TRANCODE THEN TR = 1;
   PATIRIA = PATIRIA + (TR-1)*10;
OUTPUT;
IF LAST.TRANCODE THEN DO;
   DISTANCE=10;
   PATIRIA=(TR-1)*10; OUTPUT;
   DISTANCE=0;   OUTPUT;
   PATIRIA=(TR )*10; OUTPUT;
END;
```

The axes can be artificially constructed by adding points (three for each transect) to the data set to be plotted. These three additional points are added after the last data point in each transect. The vertical value of each of the secondary horizontal axes is determined by the transect number and the constant displacement (in this case 10 was used again as in Figure 2).

```
DATA PI; SET PATIRIA;
BY TRANCODE;
IF FIRST.TRANCODE THEN TR = 1;
   PATIRIA = PATIRIA + (TR-1)*10;
OUTPUT;
IF LAST.TRANCODE THEN DO;
   DISTANCE=10;
   PATIRIA=(TR-1)*10; OUTPUT;
   DISTANCE=0;   OUTPUT;
   PATIRIA=(TR )*10; OUTPUT;
END;
```

The first point of this axis will be at the right side of the plot and will have a value that corresponds to the value that represents a zero abundance for patiria on that transect. The second point is the corner at the left and the third is the point on the the upper left. Figure 6 is an improvement, but

```
DATA PI; SET PATIRIA;
BY TRANCODE;
IF FIRST.TRANCODE THEN TR = 1;
   PATIRIA = PATIRIA + (TR-1)*10;
OUTPUT;
IF LAST.TRANCODE THEN DO;
   DISTANCE=10;
   PATIRIA=(TR-1)*10; OUTPUT;
   DISTANCE=0;   OUTPUT;
   PATIRIA=(TR )*10; OUTPUT;
END;
```

Figure 5 is usable, however it is no longer easy for the user of the plot to determine the abundance of a particular value in the third transect; the vertical scale no longer applies. Each of the transects should have its own set of axes and its own vertical scale.

```
```

FIGURE 5 - TRANSECT ABUNDANCES ON SEPARATE LINES

FIGURE 6 - TRANSECT ABUNDANCES ON SEPARATE LINES
it does not solve all the problems. Certainly the choice of the vertical separation constant allows some overlap and the axes run together. Since the axes use the same SYMBOL statements as the transect data, the last data point is joined to the axis and each axis also has plot symbols.

Figure 7 was generated to solve these two problems. The transect axes were given separate transect codes so that they could be tied to their own SYMBOL statements and the vertical separation constant was increased. These changes are shown below.

```
DATA Pl; SET PATIRIA;
BY TRANCODE;
IF FIRST.TRANCODE THEN TR + 1;
TRANSECT = TR;
PATIRIA = PATIRIA + (TR-1)*11;
OUTPUT;
IF LAST.TRANCODE THEN DO;
TRANSECT = TR + 5;
DISTANCE= 10;
PATIRIA=(TR-1)*11; OUTPUT;
DISTANCE= 0;
PATIRIA=(TR-1)*11*10; OUTPUT;
END;
PROC GPLOT DATA=Pl GOUT=DBPLOT.PATIRIA;
PLOT PATIRIA * DISTANCE - TRANSECT;
SYMBOL1 V='A' F=SIMPLEX I=JOIN
C=RED L=1;
SYMBOL2 V='B' F=SIMPLEX I=JOIN
C=RED L=1;
SYMBOL3 V='C' F=SIMPLEX I=JOIN
C=RED L=1;
SYMBOL4 V='D' F=SIMPLEX I=JOIN
C=RED L=1;
SYMBOL5 V='E' F=SIMPLEX I=JOIN
C=RED L=1;
SYMBOL6 V=' ' F=SIMPLEX I=JOIN
C=BLACK L=1 R=5;
```

Notice the addition of the SYMBOL6 for the axes. The constant is now 11, however the height of each individual axis is still 10. This prevents them from running together.

Figure 8 polishes the plot by removing the legend and cleaning up the primary vertical axis by using the AXIS statement. More complicated plots can be generated using the combined techniques of additional data points and SYMBOL statement control.

```
FIGURE 8 - TRANSECT ABUNDANCES ON SCALED SEPARATE LINES

FIGURE 9 demonstrates one such plot. In this plot the same abundances as were shown in Figure 8 are reflected about the center line creating a plot that gives a better visual feel for the location and magnitude of aggregations of the organisms under study.

```
HOLDIT = PATIRIA;
* UPPER BOUND;
TRANSECT = TR*3-2;
PATIRIA = (TR-1)*11 + .5*HOLDIT; OUTPUT;
* HILO VALUE;
TRANSECT = TR*3-1;
PATIRIA = (TR-1)*11; OUTPUT;
PATIRIA = (TR-1)*11 + .5*HOLDIT; OUTPUT;
PATIRIA = (TR-1)*11; OUTPUT;
* LOWER BOUND;
TRANSECT = TR*3;
PATIRIA = (TR-1)*11 - .5*HOLDIT; OUTPUT;
```

This code generates three separate lines for each transect: the upper bound, the lower bound and the vertical and center lines. All the lines are plotted using a common SYMBOL statement.
SUMMARY

There are many ways to control the plots produced by PROC GLOT. The options available within the proc are many and varied. Although macros and the ANNOTATE facility can be used to produce an even larger variety of plots, an additional method discussed in this paper coordinates the addition of data points into the plot data set with the use of the SYMBOL statement. This approach is often easiest for a new user to master and is useful as a quick tool for even the advanced programmer.
Arthur L. Carpenter
CALOXY
4239 Serena Avenue
Oceanside, CA 92056
(619) 724-8579

SAS/GRAPH is a registered trademark of
SAS Institute Inc., Cary, NC, USA.