HOW NOT TO HATE ANNOTATE

Susan J. Kenny, Quintiles, Inc., Research Triangle Park, NC

INTRODUCTION

For new users, mastering the annotate facility within SAS/GRAPH® software can be a confusing, if not difficult, task. This goal of this paper is to dispel some of this confusion by presenting a overview of the key features of the annotate facility and detailing some examples that may help reduce the annotate learning curve.

ANNOTATE OVERVIEW

The most frequently used application of the annotate facility is to enhance SAS/GRAPH output. Generally, this means adding graphic elements to a graph produced from one of the SAS/GRAPH procedures, such as GPLOT or GCHART. Descriptions of the attributes of these graphic elements and instructions on how and where to add them to a graph are stored in an *annotate data set*. Each observation in the annotate data set represents a command to either perform an action or draw a graphic element. This data set is referenced from within the graph procedure (PROC) and the commands are invoked.

The Annotate Data Set

There are three types of variables that are important in the annotate data set. The *function* variable specifies what graphic element to draw or what action to take. The *positioning* variables specify the X,Y, and Z coordinates for where to draw the graphic element. The *attribute* variables specify the descriptions of the graphic elements, such as the color, font, or size. Each of these types of variables will be described briefly below.

Function Variables

The value of the function variable specifies the what type of graphic elements to draw or what action to take. The most frequently used values for this variables are: BAR, DRAW, FRAME, LABEL, MOVE, PIE, POINT, POLY, POLYCONT, AND SYMBOL. These values are fairly self explanatory for what type of graphic element or action is desired For example, FUNCTION='LABEL' specifies to place text, as defined in the attribute variable, at the location specified by the positioning variables.

There is another set of values for the function variable that specify programming actions. These are used more for advanced applications and will not be described here with the exception of the CNTL2TXT and TXT2CNTL functions, which will be described in an example below.

Positioning Variables

A through understanding of how the annotate facility determines where to place graphic elements is vital to creating a successful annotate data set. There are two important aspects relating to graphic placement: the coordinates and the coordinate system. The coordinates specify where to put the graphic elements and are simply defined as X=x; Y=y; and Z=z; for numeric coordinates. The Z variable is used only to annotate graphic output from the G3D procedure. When annotating output from GCHART, the annotate facility also has position variables of GROUP, MIDPOINT, and SUBGROUP, which reference the group, midpoint, or subgroup location for a horizontal or vertical bar chart, respectively.

The coordinate system determines how the X, Y, and Z coordinates are interpreted. The variables XSYS, YSYS, and ZSYS are

used to specify the coordinate system for the X, Y, and Z axes, respectively. Do not confuse these axes with the axes of the graphs produced by the SAS/GRAPH procedure. These are the axes for one of three possible page areas; the data area, the graphics output area, or the procedure output area. The data area is the portion of the graphic that is encompassed by the X- and Y- axes of the procedure graph. The graphics output area is essentially the entire page. The procedure output area is the area referenced by the SAS/GRAPH procedure and is generally the graphic output area minus the page margins. Each of these areas have units for interpretation. The data area can be interpreted in units of percent of the graph axis or the minimum to maximum values plotted by the procedure. Both the graphics output area and the procedure output area can be interpreted in units of percent area or number of cells. Remember that whenever percent or cells are used for units the (0,0) point is always at the lower left hand corner of the page area.

The specific values of the coordinate systems tell SAS to which area the X- and Y-coordinate values relate. These values are interpreted in absolute or relative terms. For example, XSYS='4' tells SAS to interpret the X-coordinate as the number of cells in the graphic output area while XSYS='A' indicates the X-coordinate should be interpreted as the number of cells relative to the last X-coordinate.

As a simple example of defining coordinate systems, using XSYS='1'; tells SAS that an assignment, such as X=50, should reference a location that is at 50 percent of the X axis. If YSYS='3' and Y=10 then this combination of coordinate system and coordinate values references a point that is located at 50% of the X axis and at 10% from the bottom of the entire graphic.

Creating an Annotate Data Set

SAS provides three methods for producing an annotate data set. The first is to use an input statement to read in desired values for each of the annotate variables. For example, the following code would produce an annotate data set that would place the text string 'Hello' at a location of (10%,10%) of the graphic output area and a star symbol using the special font at a location of (50%,50%).

```
data anno;
length function style $8 text $10;
retain xsys ysys '3';
input function $ x y size style $ text $;
cards;
label 10 10 2 triplex Hello!
symbol 50 50 5 special M
;
run;
```

The second method involves explicitly specifying values for each of the required annotate variables and then outputting each observation. For example, the following code will produce the exact annotate data set as above.

```
data anno;
length function style $8 text $10;
retain xsys ysys '3';
function='label'; x=10; y=10; size=2;
style='triplex'; text='Hello!';
output;
function='symbol'; x=50; y=50; size=5;
style='marker'; text='M';
output;
run;
```

The third method utilizes the annotate macros that are supplied by SAS. These macros provide a short hand way of specifying the values for each of the required annotate variables. Prior to using the annotate macros, you must call the macro library using the statement %ANNOMAC; . It is sufficient to issue this statement just once within a SAS session. The following code will produce an annotate data set similar to the other methods as above.

```
%annomac;
data anno;
%dclanno;
length text $10;
%label(10,10,Hello!,black,0,0,2,triplex,5);
%label(50,50,M,black,0,0,5,marker,5);
```

Note that there is no OUTPUT statement since the macros do an implicit output when executed. The %DCLANNO macro is used to automatically set the correct length and

data type for all annotate variables except the TEXT variable. Note also that to place the marker symbol, the %LABEL macro is used. There is no %symbol macro which corresponds to the FUNCTION='SYMBOL' statement. If you wish to use the special symbols that are available within the SYMBOL statement (see page 421 of the SAS/GRAPH manual), then you must use the explicit method above with the FUNCTION=SYMBOL. It is acceptable to combine the annotate macros and the explicit statement approach within the same DATA step.

Problem Solving

Complete familiarity with coordinate systems, page areas, and units is key to successfully using the annotate facility. If you are getting messages in the LOG window to the effect that an annotate observation is outside of the area, then you are probably using an incorrect combination of coordinate system values and actual coordinate values.

Remember that the annotate data set is a SAS data set like all others. Often, just printing out the annotate data set will highlight a problem. A common problem occurs when multiple annotate data sets are defined within one program and a LENGTH statement was not used to define the length of key variables and text. Upon combining multiple data sets with inconsistent variable attributes, SAS will use the attributes of the data set that is first specified on the SET statement. Therefore, if the first annotate data set used FUNCTION='MOVE' and the second data set used FUNCTION='LABEL', the combined data set would have FUNCTION values of 'MOVE' and 'LABE'. SAS will not produce any messages and unless you view a print of the data, the only hint you might have of this problem is that your annotated text will just not appear.

Using Annotate Versus Other Approaches

Often the NOTE, TITLE, FOOTNOTE statements with the MOVE= option, can be used successfully to place additional text on a graphic. Using these statements work well if you are certain that there will be no changes to the graph which will affect the placement of the graph axes. If there are changes, such as the addition of new titles or footnotes, the graph position will shift and the MOVE= option parameters will have to be amended to ensure that the text is positioned as desired. The ORIGIN= option within the AXIS statement can be used to secure the graph position yet this may result in unused space. Using the annotate facility to place text will prevent both of these situations.

Using an existing data set to provide desired locations of graphic elements is a very powerful way to create customized graphics. For example, using the output from the REG procedure can yield the necessary coordinates values needed to place the statistical equation right next to a regression line. Using an existing data set in conjunction with having the ability to associate graphic elements relative to the data area is a very powerful feature of the annotate facility.

EXAMPLES

Below are several examples that demonstrate the workings of the annotate facility. Nearly all of these examples relate to adding graphic elements relative to the data area defined by the procedure.

Annotating Special Characters In Axis Labels

In the current version of SAS/GRAPH, it is not possible to create an axis label that contains a super- or subscripted character with the AXIS statement. Instead, the user must add the special label with either a note statement or with annotate. The following is a simplified example.

Suppose you wish to have the X-axis label read:

$$AUC_{\infty} \text{ (ng*h/mL)}$$

We would want annotate to do the following tasks. First, using the X-Y axis lines as a reference, move to a location below the X-axis

and add the text ‘AUC’. Then relative to this text, move down and over slightly and add the infinity symbol. Next, relative to the symbol, move back up and add the remaining text, ‘(ng*h/mL)’. The following code accomplishes these tasks.

```
0001 %system(1,1);
0002 %label(35,-10,'AUC',black,
            0,0,1,zapf,6);
0003 %system(7,7);
0004 %label(0.5,-1,'o',black,0,0,1,math,6);
0005 %system(7,1);
0006 %label(1,-10,'(ng*h/mL)',black,
            0,0,1,zapf,6);
```

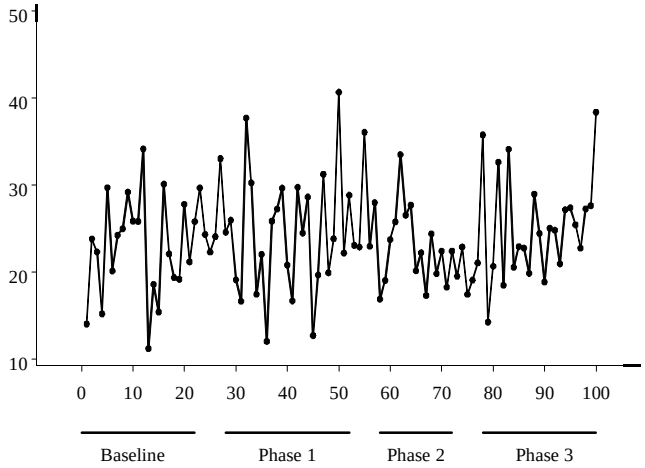
The program logic is as follows. The leading numbers correspond to the programming statement above.
1. use the % of x- and y-axis as coordinate system for both X- and Y- coordinates.
2. move to a point at 35% of the x-axis and -10% of the y-axis and write the text AUC .
3. change to relative coordinate system with units as % of x- and y-axis.
4. move to a point that is 0.5% lower than previous point and 0.2% to the right of the previous point and add the infinity symbol.
5. change y- coordinate system back to absolute units as % of y-axis. The x-coordinate system remains in as relative coordinates.
6. move to a point that is 1% to the right of previous point and -10% below y-axis. Note that -10% was used in first label statement.

Using Internal Coordinates

The annotate facility maintains two pairs of internal coordinates. XLAST and YLAST are the coordinates of the last graphics elements drawn or the coordinates from the last move. XLSTT and YLSTT are the coordinates of the last text drawn. These coordinates are useful when you wish to associate text with a recently drawn element, such as a line or a bar, or vice versa.

These coordinates are also useful when you wish to move to a defined point on a graph and then perform some annotate function relative to that location. This situation might occur if you wish to add additional graphic elements below an axis line. Figure 1 is an example where text and lines are drawn underneath the X-axis.

Figure 1.
Using Internal Coordinates



Using the internal coordinates allows us to find the graph origin and then move to a location relative to this point and create the text or line. Below is the code and the programming logic:

```
0001 %annomac;
0002 data anno;
0003 length text $15;
0004 %dclanno;
0005 %system(1,1);
0006 %move(0,0);
0007 %system(2,9);
0008 %line(0,-8,22,0,black,1,1);
0009 %line(28,0,52,0,black,1,1);
0010 %line(58,0,72,0,black,1,1);
0011 %line(78,0,100,0,black,1,1);

0012 array label (4) $ l1-l4 ('Baseline'
         'Phase 1' 'Phase 2' 'Phase 3');
0013 do x_label=10,40,65,90;
0014   n_label+1;
0015   %system(1,1);
0016   %move(0,0);
0017   %cntl2txt;
0018   %system(2,9);
0019   %label(x_label,-12,label{n_label},
          black,0,0,1.8,hwcgm005,5);
0020 end;
0021 run;
```

5. use percent of axis length for both x- and y-coordinate systems.
6. move to the origin of the graph, (0%,0%)
7. change to axis value for x- coordinate and relative % for y- coordinate. Since we first moved to the origin, then any future y- coordinates will be interpreted as a distance relative to this point.

8. draw line that goes from 0 to 22 along the x- axis and is 8% below the axis.
9-11. draw other lines using desired values of x. Note that y- coordinate is 0 since all coordinates are relative to the last coordinate specified.
12. use an array to store values of text for labeling.
13. at desired locations along the x- axis, do the following.
14. create a counter for the array index.
15-16. move to the origin of the graph.
17. copy the values of XLAST, YLAST (the internal coordinates for the location of the last move) into XLSTT, YLSTT (the internal coordinates which will be used for text)
19. change to axis value for x- coordinate and relative % for y- coordinate. Since we first moved to the origin, then any future y- coordinates will be interpreted as a distance relative to this point.
20. place desired labels using array at a location that is -12% below last y- position.

Labeling Plot Lines

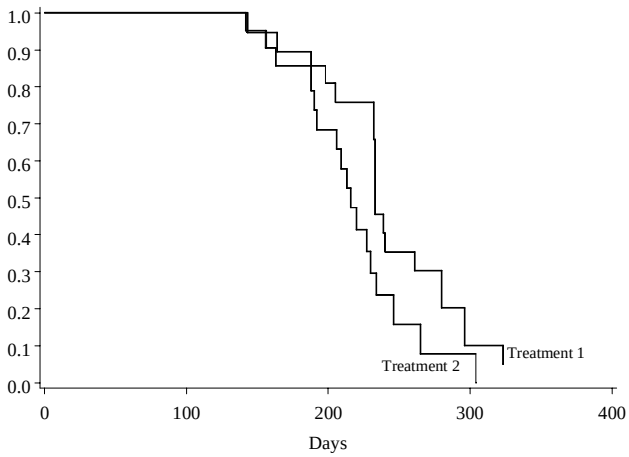
There may be occasions where you would like to place text relative to a statistical line, such as a regression line or survival curves, that is presented in a graph. Below is an example of how to place descriptive text next to multiple survival curves, as shown in Figure 2. This is an alternative to creating a legend to differentiate between the survival curves. The same programming logic could be used to annotate a regression equation next to a regression line.

The easiest way to annotate text near a statistical line (or point) is to use an output data set from the statistical procedure when creating the annotate data set. For annotating survival curves, data from the OUTSURV= output data set from within the LIFETEST procedure will provide the data values that define the survival curves. By using observations in this data set, we can determine where to place annotated text.

The following example was created using data from the SAS sample program LIFTEX0 within the STAT sample library. The following code was used to yield the output data set containing the survival values:

```
proc lifetest data = lifetxo plots = (s)
              outsurv=curves;
time days*censored(1);
strata group;
run;
```

Figure 2.
Annotating Survival Curves



Since we wish to place annotated text near the second to the last ‘step’ on the graph, we will want to retrieve the X- and Y-coordinates for these positions from the CURVES data set and use them as values for the annotate coordinates. The following code accomplishes the task of subsetting the CURVES data set and keeping the desired observation.

```
proc sort data=curves;
by group descending days; run;
data curves; set curves;
by group;
if first.group then count=0;
count+1;
if count=2 then output;
run;
```

The remaining code creates the annotation for each curve:

```
%annomac;
data anno;
set curves;
%system(2,2);
if group='pretrt2' then do;
  %label(days,survival,
         'Treatment 2',black,0,0,1,zapf,E);
end;
else do;
   %label(days,survival,
         'Treatment 1',black,0,0,1,zapf,3);
end;
run;
```

Note that in the %LABEL statements, we use the variable names ‘days’ and ‘survival’ as values for the x-, and y- annotate

coordinates, respectively. Using the variable names along with referencing an external data set, results in the values of these variables being used to provide the values for the annotate coordinates. Since the coordinate system for both the X- and Y- coordinates is defined in terms of data values, the annotated text will placed within the data area at the data locations.

There may be occasions where you wish to adjust the placement of the annotated text so that it is near the exact values for the X- and Y- variables. In this case, it is possible to adjust the data set values within the %LABEL statement. For example, the statement:

```
%label(days+10,survival-0.01,
       'Treatment 2',black,0,0,1,zapf,F);
```

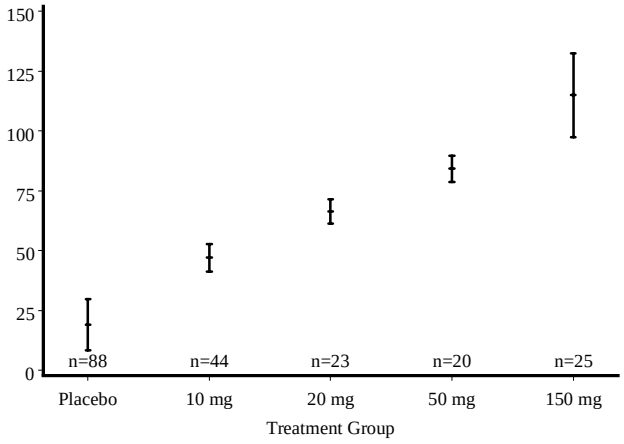
will place the text for treatment 2 at a location that is shifted 10 'days' to the right and slightly down from the survival value.

Annotating Within the Data Area

Adding descriptive text at data driven locations within a X-Y plot is an easy way to present additional data in a simple, meaningful way. Figure 3 is an example of adding the sample size for each discrete value of X.

This is very helpful when examining a plot of means and standard deviation for different groups. To add this text via annotation, the sample size for each treatment group is first obtained from an output data set from a procedure such as MEANS, UNIVARIATE, or FREQ. Below is the code used to produce the annotation for this graph.

Figure 3.
Annotating Within the Data Area




```
proc means n noprint;
class trt;
var resp;
output out=count n=n;
run;
%annomac;
data anno;
   set count(where=(trt ne .));
%system(2,1);
%label(trt,5,'N='||left(put(n,3.)),black,
       0,0,1.5,zapf,5);
run;
```

In this example, we are using a coordinate system that uses data values for the X- coordinates and percent of axis area for the Y- coordinates. The %LABEL statement will be executed for every observation in the data set COUNT. Note that we can create the concatenated text string within the %LABEL statement. Since we are adding text for each value of treatment, it will be necessary to create some extra space on each end of the X- axis to accommodate the text string. This can be accomplished by using the OFFSET= option within the AXIS statement.

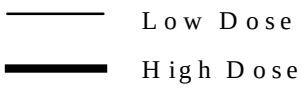
Creating Customized Legends

For many situations, the LEGEND statement is adequate for creating necessary text to describe a graph. However, there are some things the LEGEND statement cannot do. For example, all legend symbols are created using the same width and height, which is controlled by the SHAPE= option. It is not

possible to have some legend entries have larger symbols than others.

While these situations do not occur very often, using annotate to create the legend is often the only way to achieve the desired results. The annotate internal coordinates are used to ensure proper text positioning. Figure 4 is illustrates a legend with different sized entry symbols.

Figure 4.
Creating a Legend



Below is the code and program logic used to produce this legend.

```
0001 data anno;
0002 length text $ 10;
0003 %dclanno;
0004 %system(3,3);
0005 %line(80,15,85,15,black,1,2);
0006 %cntl2txt;
0007 %system(9,3);
0008 %label(+5,16,'Low Dose',black,
           0,0,2,zapf,6);
0009 %system(3,3);
0010 %line(80,9,85,9,black,1,5);
0011 %cntl2txt;
0012 %system(9,3);
0013 %label(+5,10,'High Dose',black,
           0,0,2,zapf,6);
```

4. use the percent of graphic area to position the legend. This is generally determined by trial and error.
5. draw the thin line for the Low Dose group.
6. copy the internal values of XLAST, YLAST into XLSTT, YLSTT.
7. change the coordinate system to (9,3) so that future X- coordinates are interpreted as relative coordinates in units of %. The next text command (%LABEL) will use these internal coordinates as the starting point for taking the text action.
8. using a +5 in the %LABEL statement for the X-value indicates that annotate will move 5% in the X direction as the starting location for the beginning of the text.

9-13. Repeat logic for the High Dose Group.

Note in the above example that the line and the text are not positioned on the same Y % coordinate (e.g. 15 versus 16 in first entry). This is because both %LINE and %LABEL begin place their elements ‘top’ justified. The text must be shifted up slightly so that the line appears to the left of the middle of the text, not at the top left.

CONCLUSION

In summary, the annotate facility provides nearly endless capability for customizing graphic output. A complete understanding of coordinate systems and units for coordinate values is essential for successfully creating an annotate data set. Many simple text additions to graph output can be accomplished by using text statements such as TITLE but the annotate facility is preferred when additions are added relative to the data area encompassed by the procedure graph. Several examples were presented that may help the beginning user to understand the inner workings of the annotate facility.

REFERENCES

SAS Institute Inc. (1990), *SAS/GRAPH Software, Version 6, First Edition*, Cary, NC:SAS Institute, Inc.

AUTHOR CONTACT

Susan J. Kenny, Ph.D.
Quintiles, Inc.
PO Box 13979
Research Triangle Park, NC 27709-3979
919-941-1463
skenny@quintiles.com