

## When Graphing, Anything is Possible: Once you know how to ANNOTATE (Tips and techniques to make using Annotate more effective and easier to use)

Lori D. Griffin, Hoechst Marion Roussel, Inc., Kansas City, MO

### ABSTRACT

Have you ever created a graph, especially one involving annotate, tested it to your screen and then printed it but didn't get on paper what you saw on your screen? The following paper should help you understand as well as fix problems like this.

The intent of this paper is to help clear up some of those foggy areas surrounding, mainly, the ANNOTATE facility with consideration of other related graphing problems. Annotating can be FUN once you have a fundamental understanding of the basics needed to annotate. This paper will help to develop a good foundation from which to work. Some of the topics covered will be HPOS and VPOS, XSYS and YSYS, understanding the device you use, using the POSITION variable for text placement, annotate macros, and much more. All the programming code used to create the examples will be provided during the poster presentation.

### INTRODUCTION

This presentation is intended for all types of programmers, from beginners to the more advanced. I believe everyone will find at least one tip or technique they can take home and try out. Most of the presentation is geared toward the basics for annotating. A few other graphing areas are also covered.

### Small TIPS TO REMEMBER

#### Tip #1:

Before using the graphic annotate macros you must first submit the annomac macro.

```
%ANNOMAC;
```

The ANNOMAC macro compiles the annotate macros and makes them available for use.

#### Tip #2:

For those that prefer to develop their annotate data set using a CARDS statement, just keep in mind that a CARDS statement cannot be used within a macro.

#### Tip #3:

Beware of multiple run statements. Have you ever had the same graph displayed more than once?

```
proc gslide;
title 'graph number 1';
run;
run;
```

The above program displays the same graph twice due to the multiple RUN statements.

### SIZE AND STYLE

There are two important key words to remember, which are Size and Font (**SIZE** versus **HEIGHT** and **STYLE** versus **FONT**). This may seem very basic to someone familiar to annotate but for those that are just beginning this is a difference of which to take note. Height is what you use to set the height of the text for such items as a title, footnote, axis .... When using annotate, however, you specify the height using the **SIZE** option. Font is what you use to set the text description for titles, footnotes, axis and other items on your graph. When using annotate, however you specify the text description using the **STYLE** option.

```
function='label'; x=10; y=10;
text='figure1'; STYLE='swissb';
SIZE=3; output;
```

## The importance of LENGTH

When using the LABEL function or the %LABEL macro for annotating multiple labels and the results are truncated. What do you do? See example below.

```
data anno1;
%annomac;
%label(20,10,'Figure1',,,,,,,,,);
%label(20,8,'This is Figure2',,,,,,,,,);

proc gslide anno=anno1;
run;
```

The above code produces the following results:

```
Figure1
This is
```

Notice how the second label is truncated, it should be 'This is Figure2'. The text of the first label that contained 'Figure1', which was seven characters long, set the length for any additional text references. So as a result, the second LABEL only printed the first seven characters which was 'This is '.

To fix this problem just add an initial length statement.

```
data anno1;
%annomac;
length text $15.;
%label(20,10,'figure1',,,,,,,,,);
%label(20,8,'This is Figure2',,,,,,,,,);

proc gslide anno=anno1;
run;
```

The following statements use the annotate functions instead of the annotate macros but perform the same as the %LABEL macro.

```
Function='LABEL'; x=20; y=10;
text='Figure1'; output;
Function='LABEL'; x=20; y=10;
text='This is figure2'; output;
```

When using the annotate functions it is a good idea to set an initial length statement for the variable function.

Length function \$8.;

If you attempted to submit the following code without the Length statement, you will receive the message that follows in your log and nothing will be graphed.

```
data anno1;
xsys='3'; ysys='3';
function='POLY'; x=15; y=10;
style='solid'; color='green'; output;
function='POLYCONT'; x=85; y=10;
style='solid'; color='green'; output;
function='POLYCONT'; x=50; y=40;
style='solid'; color='green'; output;
function='POLYCONT'; x=15; y=10;
style='solid'; color='green'; output;
proc print; run;

proc gslide anno=anno1;
run;
quit;
```

```
NOTE: ERROR DETECTED IN ANNOTATE= DATASET WORK.ANNO.
NOTE: PROBLEM IN OBSERVATION 4 -
POLYGON DEFINITION CONTAINS TOO FEW VERTICES. ACTION TAKEN
POINT
```

What has happened is that the first function, function='POLY', has set the length for any additional function variables to a length of four. So the next function, function='POLYCONT', is only recognized as function='POLY' and the CONT is truncated. Probably the best way to debug your annotate data set is to add a proc print statement at the end of your data set and this will all become very clear.

## HPOS AND VPOS:

One might ask: How do people determine their HPOS and VPOS settings? What I have found

is that some people like to set the HPOS and VPOS to the default of their screen display. And if you do a large amount of testing to your screen, you probably get used to the screen display. If you like it on the screen then you probably want your printer output to resemble that. There are others who prefer to set their HPOS and VPOS to 100, this way you can use either cells or percents when annotating and the results should be the same. There are still others that prefer to set their HPOS and VPOS to the default of their printers. What I found to be easiest is to choose a setting early on in your graphing career and stick with it, because eventually you become accustomed to the height(size) and fonts(style) needed for the best presentation of your data.

The HPOS= graphics option specifies the number of **columns** in your graphics output area. The VPOS= graphic option specifies the number of **rows** in your graphics output area. The HPOS= and VPOS= graphic options, both divide the graphics output area into cells. If the VPOS is used, it overrides the value of the LROWS and PROWS device parameters. And if the HPOS is used, it overrides the value of the LCOLS and PCOLS device parameters.

If you specify HPOS=80 and VPOS=32, the graphics output area will be divided into a grid with 80 columns and 32 rows. Specifying HPOS=0 and VPOS=0 causes the device driver to use the default hardware character cell size for the device. The default is determined by the LCOLS and LROWS if orientation is landscape or by the PCOLS and PROWS if the orientation is portrait.

Now you know that LCOLS, LROWS, PCOLS, and PROWS are device parameters. You might wonder, how do I know what my device parameters are set at? First you must know what your device is. For example, if your device is PS for Post Script use the following code to display the PS device parameters.

```
proc gdevice catalog=sashelp.devices nofs;
list PS;
run;
```

```
GDEVICE procedure
Listing from SASHELP.DEVICES - Entry PS

Orig Driver: PS          Module: SASGDPSL  Model: 1201
Description: PostScript devices          Type: PRINTER
*** Institute-supplied ***
Lrows: 0 Xmax: 8.500 IN Hsize: 7.500 IN Xpixels: 2550
Lcols: 0 Ymax: 11.000 IN Vsize: 10.000 IN Ypixels: 3300
Prows: 81 Horigin: 0.500 IN
Pcols: 75 Vorigin: 0.500 IN
Aspect: 0.000 Rotate:
Driver query: Queued messages: N
Paperfeed: 0.000 IN
```

The above is only a very small sampling of the information concerning the PS device driver. I have displayed less than half the information this procedure produces. But as you can see the LROWS, LCOLS, PROWS and PCOLS is displayed in this sampling. I would advise the beginner to the most advance graphics programmer to take the time and run this procedure. This procedure alone may help explain why your graph looks different when you run it to different devices.

#### XSYS and YSYS:

XSYS, YSYS and ZSYS variables specify a coordinate system for each axis (X, Y, and Z, respectively). When dealing with the coordinate systems you will encounter three different areas: data area, procedure output area, and graphics output area. When dealing with these areas, also referred to as drawing areas, you need to have a good understanding of the effect that titles, footnotes, and legends have on these areas. Below is the annotate code that created the shaded areas on the examples of the data areas. The only difference in the code is the XSYS and YSYS specifications.

```
/* DATA AREA */
data anno1; set x;
function='move'; x=0; y=0; output;
function='bar'; x=100; y=100; style='x1';
line=1; color='yellow'; output;
```

```

/* GRAPHICS OUTPUT AREA */
data anno2; set x;
xsys='3'; ysys='3';
function='move'; x=0; y=0; output;
function='bar'; x=100; y=100; style='x1';
line=1; color='yellow'; output;

/* PROCEDURE OUTPUT AREA */
data anno3; set x;
xsys='5'; ysys='5';
function='move'; x=0; y=0; output;
function='bar'; x=100; y=100; style='x1';
line=1; color='yellow'; output;

```

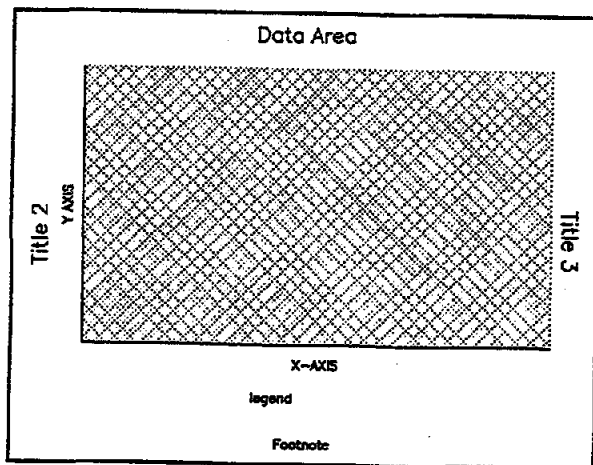


Figure 1 Data Area

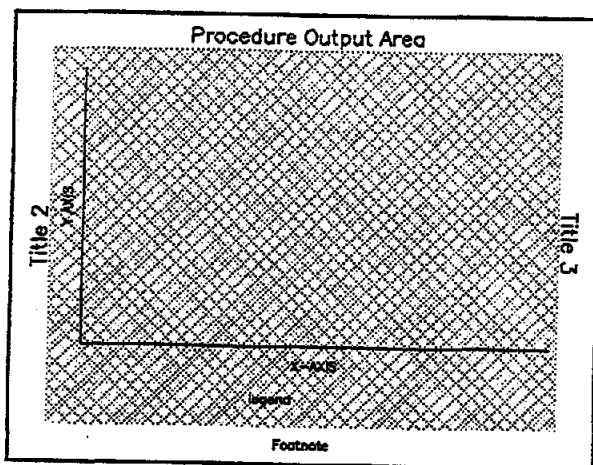


Figure 2 Procedure Output Area

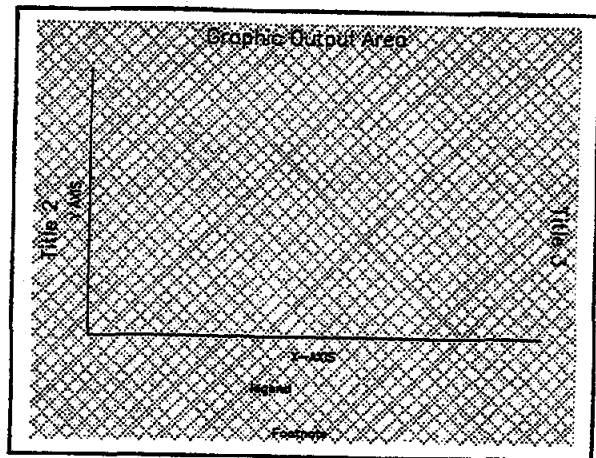


Figure 3 Graphic Output Area

Please pay close attention to the title statements that have been angled on the left and right hand side of the graphs. You will notice that the data and procedure output areas are both affected by this title but not the graphic output area. It is a good idea to become familiar with these drawing areas, because they are a fundamental part of annotating.

## POSITION

Position is an important variable and is only available with the function LABEL. Position controls the placement and alignment of your text string. Often times position can save you from adjusting the X and Y coordinates to change the placement of your text.

Figure 4 was created using the following code:

```

data x;
x=1; y=1;
x=2; y=1;
run;

data anno1; set x;
length function $8. text $12. style $8.
color $8.; xsys='3'; ysys='3';

function='move'; x=50; y=0; output;
function='draw'; x=50; y=100;

```

```
color='steel'; output;
function='move'; x=0; y=50; output;
function='draw'; x=100; y=50;
color='steel'; output;
```

```
hsys='3';
function='pie'; x=50; y=50; angle=0;
rotate=360; color='yellow';
style='solid'; size=5; output;
```

```
function='label'; x=50; y=50;
color='blue'; position='1'; size=10;
text='position="1"'; output;
```

```
proc gslide anno=anno1;
run;
quit;
```

Figures 5 through 18 were created by modifying the above code. The only modifications, include both the position variable and the text string. These examples along with the code may help you to better understand the variable Position.

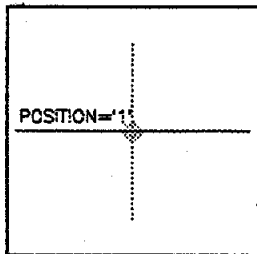


Figure 4 One cell above, Right aligned

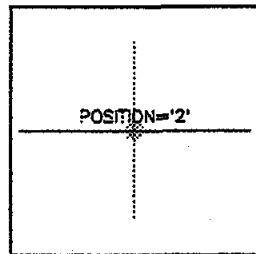


Figure 5 One cell above, Centered

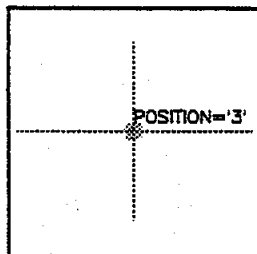


Figure 6 One cell above, Left aligned

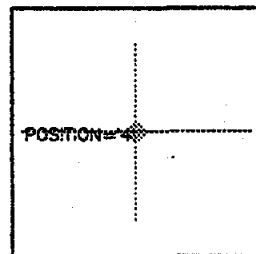


Figure 7 Centered, Right aligned

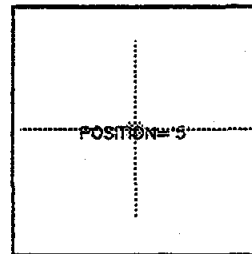


Figure 8 Centered, Centered

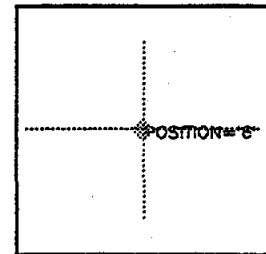


Figure 9 Centered, Left aligned

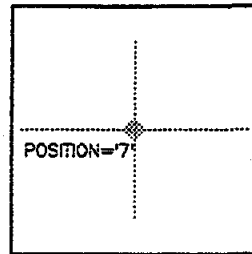


Figure 10 One cell below, Right aligned

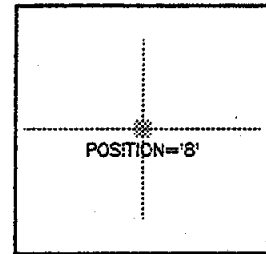


Figure 11 One cell below, Centered

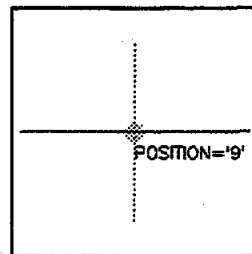


Figure 12 One cell below, Left aligned

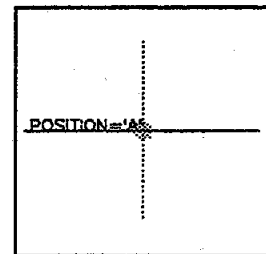


Figure 13 Half cell above, Right aligned

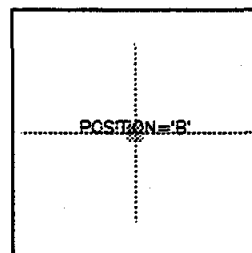


Figure 14 Half cell above, Centered

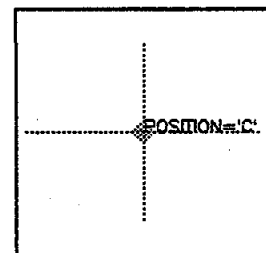


Figure 15 Half cell above, Left aligned

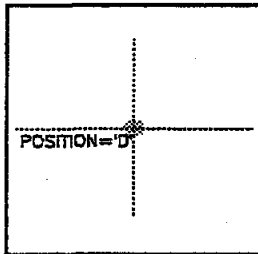


Figure 16 Half cell below, Right aligned

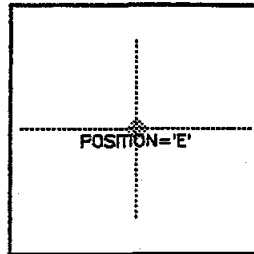


Figure 17 Half cell below, Centered

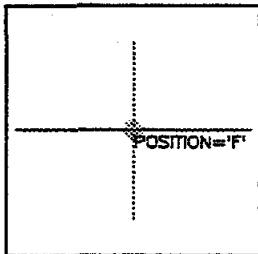


Figure 18 Half cell below, Left aligned

Position is very helpful, if you want to change your color or font in the middle of a your text string. Position='0', is a special text value that allows you to pause and then continue your text string. For further information on the position variable, please refer to your SAS® Graph manuals.

## CONCLUSION

Hopefully, the information presented in this paper will prove beneficial in any endeavors with annotate. This paper has in no way tried to answer all those annotate problems and questions. Those who have had experience with annotate, know one paper would not nearly be enough. What I have attempted to do is give a few helpful tips and to get you thinking about some of the fundamentals you need to know when attempting to annotate.

## AUTHOR INFORMATION

Any questions, comments, or suggestions contact:

Lori Griffin  
Statistical Analyst  
Applications Programming  
Marion Merrell Dow, Inc.  
10400 Hickman Mills Dr.  
Kansas City, Missouri 64137  
(816) 966-7044

SAS, SAS/GRAPH are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.