Paper 108-31

Data Driven Annotations: An Introduction to SAS/GRAPH's® Annotate Facility

Arthur L. Carpenter
California Occidental Consultants

ABSTRACT

When SAS/GRAPH was first introduced, it was the 'only game in town' for the generation of business graphics. In the current computing environment we have lots of graphics options, and it is not unusual for programmers new to SAS/GRAPH to ask why they should bother learning the command and syntax structure of SAS/GRAPH, let alone that of the Annotate Facility. The fact of the matter is that SAS/GRAPH is still the 'only game in town' with regards to the generation of data driven graphics.

When you want to create a large number of graphs (or even a single graph on a regular basis) and you do not want to place labels or other graphic enhancements manually, the Annotate Facility allows you to set up a system that places these items on the graph based on the data contained in the graph itself.

In this workshop we will learn how Annotate communicates to the graphics procedures through the annotate data table. We will see how the data itself can be used to place labels, draw lines, and otherwise dress-up a graph.

KEYWORDS

Annotate functions, Annotate Facility, Annotate table, Annotate variables, ANNO= option

INTRODUCTION TO THE ANNOTATE FACILITY

The Annotate Facility allows the user to create customized modifications to the graphic output. These modifications can be either predetermined or may be data driven. This means that, through the use of Annotate, you can greatly extend the already powerful capabilities of SAS/GRAPH.

Many users of SAS/GRAPH have avoided using Annotate because of what they perceive to be a rather steep and long learning curve. This is an unfortunate misconception, for using Annotate need not be difficult, and can be easily introduced by presenting the different fundamentals of the specialized Annotate data set. Using this data set Annotate looks for variables with specific names and attributes, and the values taken on by these variables in turn instruct the Annotate Facility as to the user's intentions.

What is the Annotate Facility?

The Annotate Facility is included within SAS/GRAPH® and acts as a bridge between the procedure selected by the user and the user's desire to customize the graphics output.

The power of the Annotate Facility is accessed through the use of a specialized data table. When using this data set, Annotate looks for variables with specific names and attributes, and the values taken on by these variables in turn instruct Annotate as to your intentions.

The Annotate Data Table

The Annotate data set is an ordinary SAS data set. It in no way differs from any other SAS data set. Unlike most SAS data sets, however, the Annotate data set is fairly rigidly defined in terms of the variables that it is to contain, and the attributes that these variables must have.

Although at first it may seem clumsy to pass specific information to a procedure through the use of dedicated data sets, procedures are actually designed to accept, interpret, and respond to SAS data sets. Therefore, an Annotate data set can contain the functional information that could not be included through options and procedure statements in the PROC step . The result is a stronger and more flexible approach.

The SAS/GRAPH procedures will sequentially read the observations from the Annotate data set and search for specific variables. The values taken on by these variables directs the Annotate Facility to perform the desired actions.

Each observation will request that Annotate perform a particular function. The requested function will cause Annotate to look for those variables that can be used to modify that particular function. Other variables that do not relate to that function for that observation will be ignored.

In this paper and workshop we will concentrate on the process of placing labels on graphs. Other Annotate functions are similar but will involve other variables.

THE ANNOTATE PROCESS

Since Annotate is used primarily to enhance a graph, the first step for the programmer faced with using the Annotate Facility is to determine what needs to be done. The answer will usually take the form of something like: 'add a label', 'include a legend in the upper right hand corner', or 'draw a triangle'. This information is passed to Annotate through specific variables in the Annotate data set. It is very important for you to remember that specific variables are used to answer the questions of:

- **What** is to be done?
- **How** is it to be done?
- **Where** is it to be done?

The variables that you use in the Annotate data set pass **ALL** the information to the graphics procedure. Just as in gourmet cooking where “the presentation is everything”, in Annotate, the selection of variables is everything.

Some of the variables that are used to answer these three questions include:

What	FUNCTION (this is the ONLY variable used to answer this question)
How	COLOR, SIZE, STYLE, POSITION
Where	X, Y, XSYS, YSYS

The variable FUNCTION is key to this process, and for the most part, dictates what other variables will be needed and used.

Once the FUNCTION has been determined (**WHAT**), its supporting variables are selected (**HOW**), and then finally the location variables (**WHERE**) are determined.

WHAT, WHERE, and HOW

For the new Annotate programmer it is very important to always remember to answer the WHAT question first. This gives you the reference point from which you can select the other variables.

What is to be done?

The character variable FUNCTION provides the information on WHAT is to be done. Virtually all Annotate data sets will have this variable defined for all observations. Since FUNCTION provides the user with the ability to express what is to be done, it is one of the best places for you to start when creating an Annotate data set.

Some of the common values of the variable FUNCTION include:

- BAR creates a fillable rectangle
- DRAW draws a line
- LABEL places text or symbols on the graphic
- MOVE allows movement to a specific point on the graphic
- PIE creates a fillable slice, arc, or circle
- POINT places a single point
- POLY starts the creation of a polygon
- POLYCONT continues the creation of a polygon
- SYMBOL places a symbol on the graphic

How is it to be done?

Once the value of FUNCTION has been determined, you can select from a list of attribute questions that address the issue of how the selected action is to be performed. Since the list of applicable supporting variables varies depending on the value of FUNCTION, the documentation is even organized by FUNCTION.

For FUNCTION='label', which will be used throughout this workshop, some of the attribute variables include:

- TEXT='*string*' add *string* to display
- COLOR='*color*' specify the color of text
- SIZE=*n* size of text
- STYLE='*font*' select font for text string

In these examples, since we are dealing with text, we will control the attributes in much the same way that options are used in TITLE and FOOTNOTE statements. Indeed there are corresponding Annotate variables for most text options including justification, character rotation, and angle of text.

Where is it to be done?

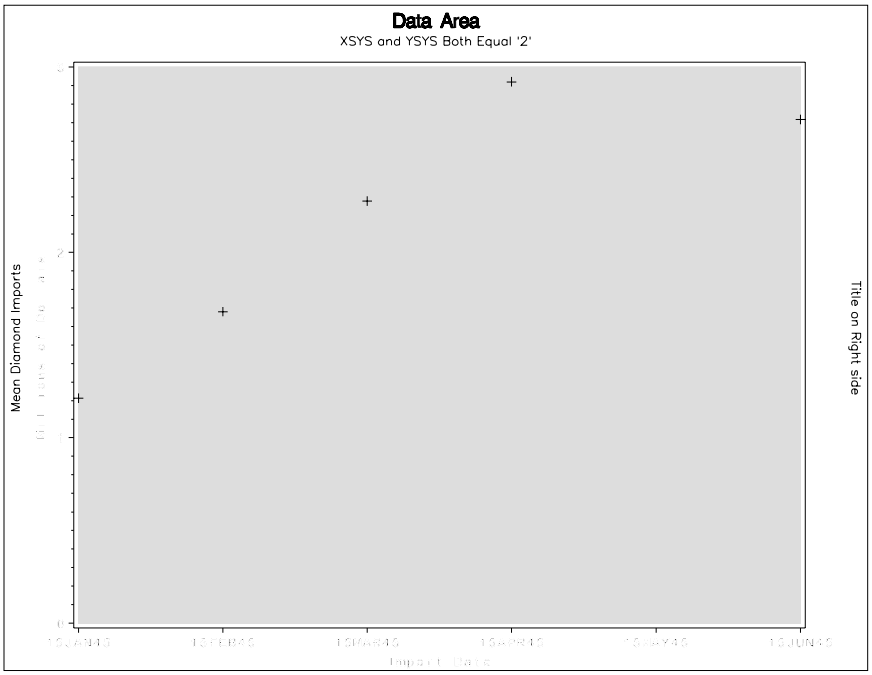
For nearly all of the values of FUNCTION, the location on the graph must be selected, i.e. WHERE on the graph should the annotation be placed. The coordinates are usually placed using the numeric variables X and Y. How these coordinates are interpreted depends on the coordinate system, which is specified by the XSYS and YSYS variables. These variables may be defined explicitly in a DATA step or their values may be data driven. In either case, X is used to define horizontal coordinates and Y the vertical.

The coordinate system itself can be selected by using the character variables XSYS and YSYS. Although these variables can take on one of twelve 'system' values, two of these values for XSYS and YSYS will satisfy most of your Annotate needs.

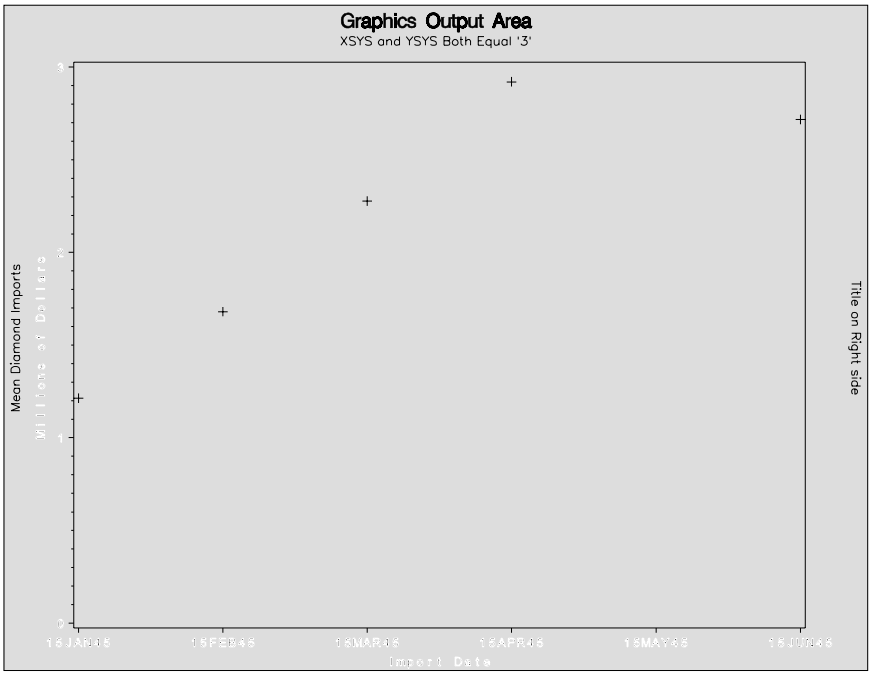
Where a particular value of X will be located depends on the value assigned to XSYS. When XSYS='3' (graphics output area percentage) a value of X=50 will be plotted in the horizontal middle of the page. However, when XSYS='2' (data value) the placement depends on the horizontal axis on the plot or graph. For instance if the axis ranges from 0 to 55, X=50 will be located on the far right of this axis area.

The values of XSYS and YSYS need not be constant in the Annotate data set. As a matter of fact, for a given observation, XSYS and YSYS do not even need to have the same values.

XSYS & YSYS='2' 'Absolute data value' places the point according to the values of the horizontal and vertical axes that are plotted on the graph.



XSYS & YSYS='3' 'Absolute Graphics Output Area percent' uses percentages of the entire graphics area, which are measured from the lower left corner.



A Simple Annotate Example

Labels are often placed on a scatter plot in such a way as to be associated with particular points. In the

following example a label (HIGH) will be placed at a point associated with the maximum imports on a scatter plot. This requires that you know the coordinates of the point that is to have the label, which in the following example is X=4 and Y=2.5 ❶. Because the values of XSYS and YSYS have been set to '2', the values of X and Y will correspond to the X and Y axes (MONTH and DIAMONDS respectively in this example).

In the data set ANNODAT.IMPORTS, the imports from three coded countries of various precious items is recorded in millions of dollars. The plot below shows the imports of diamonds by Warbucks Industries from the country coded as SFO.

```
* Place a label on the value at month 4;
goptions gfont=swiss htext=1.5;
DATA ANNPLT;
     LENGTH FUNCTION $8 TEXT $5;
     RETAIN XSYS YSYS '2' STYLE 'SWISS' ❷
         FUNCTION 'LABEL' ;
     SIZE = 2; ❸
     X=4;      Y=2.5; ❶
     TEXT='HIGH';
     OUTPUT;
     RUN;

PROC GPLOT DATA=annodat.imports ANNO=ANNPLT;
     where co_code='SFO';
     PLOT diamonds * MONTH / vaxis=.5 to 2.5 by .5;
     SYMBOL1 L=1 V=NONE I=JOIN;
     TITLE1 H=2 F=SIMPLEX '1945 Diamond Imports in Millions';
     footnote h=2 f=simplex j=l 'Figure 3';
     run;
     quit;
```

The X=4 and Y=2.5 ❶ determines the location of the label. The height of the characters and the font that is used, is controlled through the use of the STYLE and SIZE variables. STYLE ❷ is a character variable that contains the name of the font that is to be used, and SIZE ❸, a numeric variable, designates the height of the text. In Figure 3, the font is set to SWISS and the size of the text is set to 2.
Note the use of the LENGTH statement to set the length of the FUNCTION and STYLE variables. Variables that receive a default length based on an assignment statement run the risk of truncation if a longer value is assigned later in the step. Using the LENGTH statement for variables such as FUNCTION, STYLE, and COLOR insure that accidental truncation will not occur.

The Annotate data set, WORK.ANNPLT, contains:

OBS	FUNCTION	STYLE	TEXT	XSYS	YSYS	SIZE	X	Y
1	LABEL	SWISS	HIGH	2	2	2	4	2.5

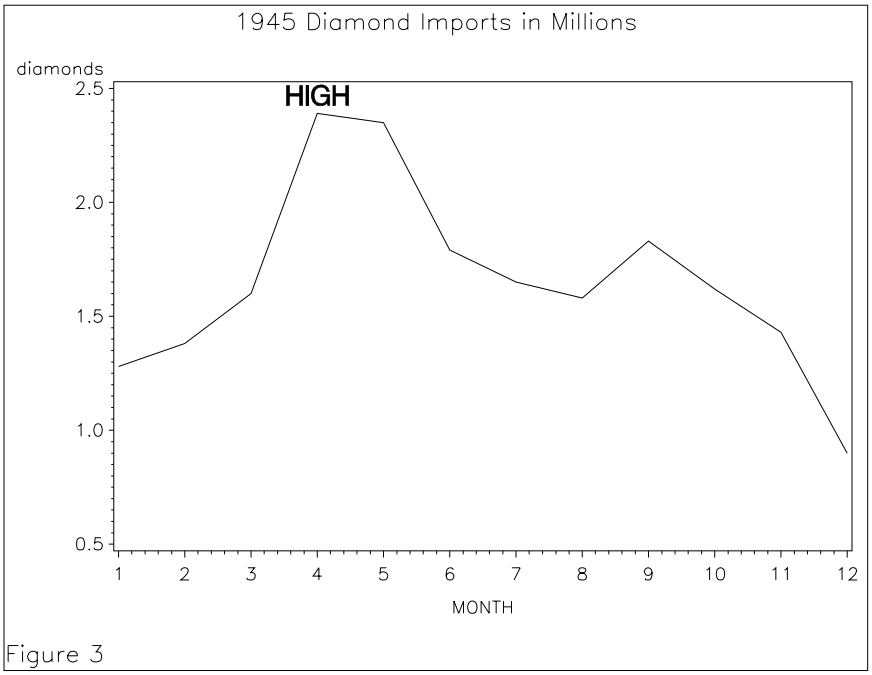


Figure 3

Unlike in this example, you will not generally know the location of the point of interest before the graph is drawn, so in order to automate the process of label placement, the location will need to be data driven. Fortunately the process of placing labels using values contained in the data is fairly straightforward and takes advantage of standard DATA step tools.

BUILDING THE ANNOTATE DATA TABLE

The Annotate data set is created using many of the same tools that are used to create other SAS data sets. Remember that an Annotate data set is just an ordinary SAS data set. The only thing that is special about an Annotate data set is that the Annotate Facility can only make use of specific variables, and these will often have prescribed attributes.

The Annotate data set can be built:

- by using assignment statements
- from an existing SAS data set (or other data source)

Using Assignment statements

The Annotate data set can be built from scratch by using assignment statements. This technique is best employed when the Annotate data set does not depend on incoming data and only a few observations are needed. Usually the assignment statements will be used in conjunction with a combination of the LENGTH and RETAIN statements. This approach was used in the DATA step that created the Annotate data for Figure 3 above.

```
DATA ANNPLT;
     LENGTH FUNCTION $8; ❶
     RETAIN XSYS YSYS '2'
            STYLE 'SWISS' ❷
            FUNCTION 'LABEL' ;
     SIZE = 2; ❸
     X=4;      Y=2.5;
```

```
        TEXT='HIGH';
        OUTPUT; ❹
        RUN;
```

❶ The LENGTH statement declares the number of bytes available to a variable. It is generally considered wise to specifically declare the length of the variables FUNCTION and COLOR as these are especially vulnerable to truncation.

❷ The RETAIN statement is a bit more efficient than the assignment statement when assigning a constant value to a variable. A single value can be assigned to a list of variables (both XSYS and YSYS both receive a value of '2'). Multiple variable assignments can be made in a single statement, and multiple RETAIN statements are also permitted.

❸ Various assignment statements can be used to create or modify values to variables.

❹ The completed observation is written to the Annotate data set through the use of the OUTPUT statement. When a data table is not named on the OUTPUT statement, the observation is written to all of the tables listed in the DATA statement (in this example ANNPLT).

To avoid truncation and to make sure that variables have compatible attributes when Annotate data sets are combined, it is a good idea to specify the length of character variables by using either the LENGTH or RETAIN statements. In the above example, the length of the variable TEXT is determined in an assignment statement to be $4. Including the variable TEXT in the LENGTH statement would have been more appropriate.

Building on an Existing Data Table

When the graphics display depends on an established SAS data set, that data set can often be used to build the Annotate data set as well. This technique is especially useful when you need to place labels or text strings at a location that is to be determined by the data itself.

Adding Annotate Variables to the Plot Data

The data set to be plotted and the Annotate data set do not necessarily need to be distinct. Annotate ignores any variables that are not in the Annotate dictionary. More specifically Annotate only looks for the specific variables that are used by the FUNCTION defined in the current observation.

This example adds a label to each point that has a Y value (diamond imports in millions of dollars) greater than $2,000,000. Two items of information are taken from the data which will be used by Annotate *i.e.* the value of the data point (TEXT), and the location that the point will be plotted on the graph (X and Y).

```
* label all months with imports exceeding $2 million;
data imports;
   set annodat.imports;
   length function $8 text $5; ❶
   retain xsys ysys '2' style 'swiss'
          function 'label' position '3' size 1.5;
   where co_code='SFO';
   if diamonds ge 2 then do; ❷
      x=month;
      y=diamonds;
      text=put(diamonds,5.2);
   end;
run;

* ANNOTATE and GPLOT use the same data set;
proc gplot data=imports anno=imports; ❸
   plot diamonds * month;
   symbol1 l=1 v=dot i=join;
```

```
   title1 h=2 'Diamond Imports From SFO in 1945';
   footnote h=2 f=simplex j=l 'Figure 4';
run;
quit;
```

❶ The Annotate variables are added to the PDV through the use of LENGTH, RETAIN, and assignment statements.

❷ The variables that hold the text for the label and the label's location receive values ONLY for the observations that are to receive annotation.

❸ The Annotate data set and the plot data are the same data set.

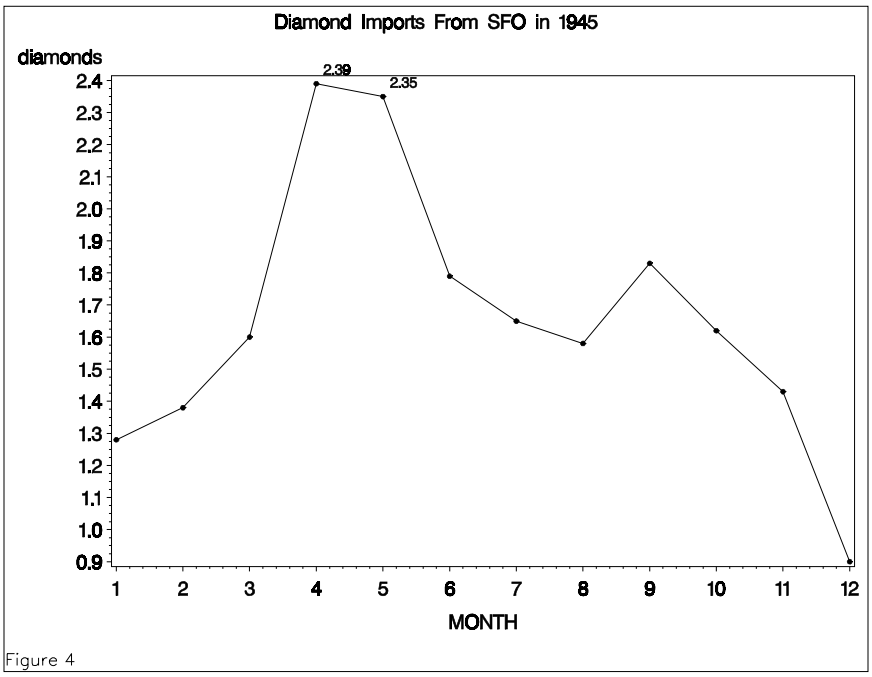


Figure 4

The data set IMPORTS contains both the plot information (DIAMONDS and MONTH) as well as the Annotate variables. Notice that only the two points with actual labels have values for X, Y, and TEXT. This means that, except for observations 4 and 5, all the other Annotate data will be ignored. This is not a very efficient use of our data table.

Variables in IMPORTS

Obs	diamonds	MONTH	function	text	style	xsys	ysys	x	y
1	1.28	1	label		swiss	2	2	.	.
2	1.38	2	label		swiss	2	2	.	.
3	1.60	3	label		swiss	2	2	.	.
4	2.39	4	label	2.39	swiss	2	2	4	2.39
5	2.35	5	label	2.35	swiss	2	2	5	2.35
6	1.79	6	label		swiss	2	2	.	.
7	1.65	7	label		swiss	2	2	.	.
8	1.58	8	label		swiss	2	2	.	.
9	1.83	9	label		swiss	2	2	.	.
10	1.62	10	label		swiss	2	2	.	.
11	1.43	11	label		swiss	2	2	.	.
12	0.90	12	label		swiss	2	2	.	.

If when we are creating the Annotate data we split this one table into two distinct data sets we can be

much more efficient.

Creating a Separate Annotate Data Set

The data set used to generate Figure 4 contained both the plot and Annotate data. It is usually more efficient to separate these two types of data. This example repeats the same plot as Figure 4, however, the data are broken up into two data sets.

```
* label all months with imports exceeding $2 million;
data imports(keep=diamonds month) ❶
     dianno (keep=xsys ysys style function position size
                  x y text);
   set annodat.imports(keep=co_code diamonds month);
   length function $8 text $5;
   retain xsys ysys '2' style 'simplex'
          function 'label' position '3' size 1.5;
   where co_code='SFO';
   output imports; ❷
   if diamonds ge 2 then do;
      * Conditional observation for ANNOTATE;
      x=month;
      y=diamonds;
      text=put(diamonds,5.2);
      output dianno; ❸
   end;
   run;

* ANNOTATE and GPLOT use separate data sets;
proc gplot data=imports anno=dianno;
   plot diamonds * month;
   symbol1 l=1 v=dot i=join;
   title1 h=2 'Diamond Imports From SFO in 1945';
   footnote h=2 f=simplex j=l 'Figure 5';
   run;
   quit;
```

❶ The DATA statement names both data sets and the KEEP= data set option is used to limit the tables to the variables of interest.

❷ All observations meeting the WHERE criteria are written to the data to be plotted.

❸ Only those observations that meet the criteria for an Annotate label are written to the Annotate table. Unlike in Figure 4 this data set will only have two observations for the Annotate Facility to process.

CONTROLLING TEXT LOCATION AND APPEARANCE

Usually when you are placing text on a graphic, the selected function will be ‘LABEL’. There is a great deal of control available for the appearance of text placed by Annotate. Most of the text control options that are available in the TITLE and SYMBOL statements, are also available in Annotate. The primary difference of course is that you will be specifying variables rather than options to achieve this control.

Using the POSITION Variable to Fine Tune Location

Usually the location specified by X and Y is not quite close enough to what you need. Should the label be above or below the point? To the left or right? The variable POSITION allows this fine tuning of the location.

POSITION takes on the values from ‘0' to ‘9' and ‘A’ to ‘F’. In the following example sixteen points are plotted each with a label of POS=x, where the x is the value of POSITION. The label itself uses its

POSITION so that you can get idea of the various values of POSITION and what they do for you.

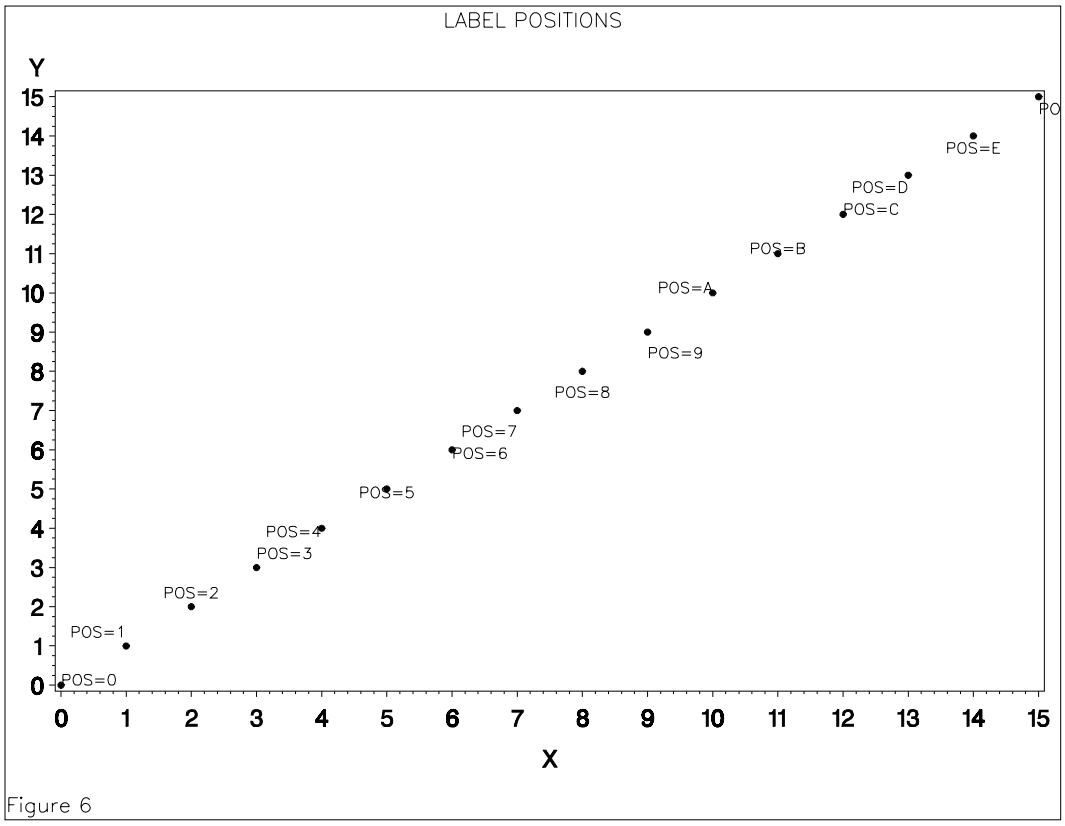


Figure 6

The code used in the generation of Figure 6 can be found in the appendix to this paper.

Text Appearance Variables

Several text appearance variables are similar to options in the TITLE and FOOTNOTE statements. These include:

TITLE / SYMBOL Option	Annotate Variable	Purpose
Font=	STYLE	Designate the graphic font.
Height=	SIZE	Determines the character size.
Rotate=	ROTATE	Rotate individual characters
Angle=	ANGLE	Changes the angle of the text line
Color=	COLOR	Assigns the color for the text

In TITLE and SYMBOL statements the font is specified by using the FONT= option, however in Annotate data sets, the variable STYLE is used to designate the font. Because STYLE can take on the value of the name of any font that is available to the graphics device, it is generally wise to give this variable a length of at least $8 to avoid truncation issues.

The size of the text is controlled by the SIZE variable. This numeric variable is used similarly to the HEIGHT= (H=) option in the TITLE and SYMBOL statements. The value of SIZE will be in the units currently specified in the GUNIT graphics option.

Normally the text is added along a horizontal baseline, however the ANGLE variable can be used to specify the number of degrees (0 to 360) to pivot the entire line. The angle is measured counter clockwise from the baseline.

It is also possible to rotate the individual characters within a text string. The ROTATE variable specifies the angle of rotation for each individual character. Very often the ROTATE and ANGLE options are used together on the same text string.

```
* label all months with imports exceeding $2 million;
data imports(keep=diamonds month)
     dianno (keep=xsys ysys style function position size
                  angle rotate x y text);
   set annodat.imports(keep=co_code diamonds month);
   length function $8 text $5;
   retain xsys ysys '2'
          ❶ style 'simplex' ❷ size 2
          ❸ angle 20 ❹ rotate -20
          function 'label' ❺ position '3';
   where co_code='SFO';
   output imports;
   if diamonds ge 2 then do;
      * Conditional observation for ANNOTATE;
      x=month;
      y=diamonds;
      text=put(diamonds,5.2);
      output dianno;
   end;
   run;
```

In this example the following text attributes are specified:

❶	STYLE	the font is set to SIMPLEX
❷	SIZE	character height is 2 units
❸	ANGLE	the entire label rises at an angle of 20 degrees
❹	ROTATE	each individual letter within the label is rotated back (minus degrees) to the horizontal
❺	POSITION	places the text above and to the right of the plotted point

The Annotate data set WORK.DIANNO is shown below.

OBS	FUNCTION	TEXT	XSYS	YSYS	STYLE	SIZE	ANGLE	ROTATE	POSITION	X	Y
1	label	2.39	2	2	simplex	2	20	-20	3	4	2.39
2	label	2.35	2	2	simplex	2	20	-20	3	5	2.35

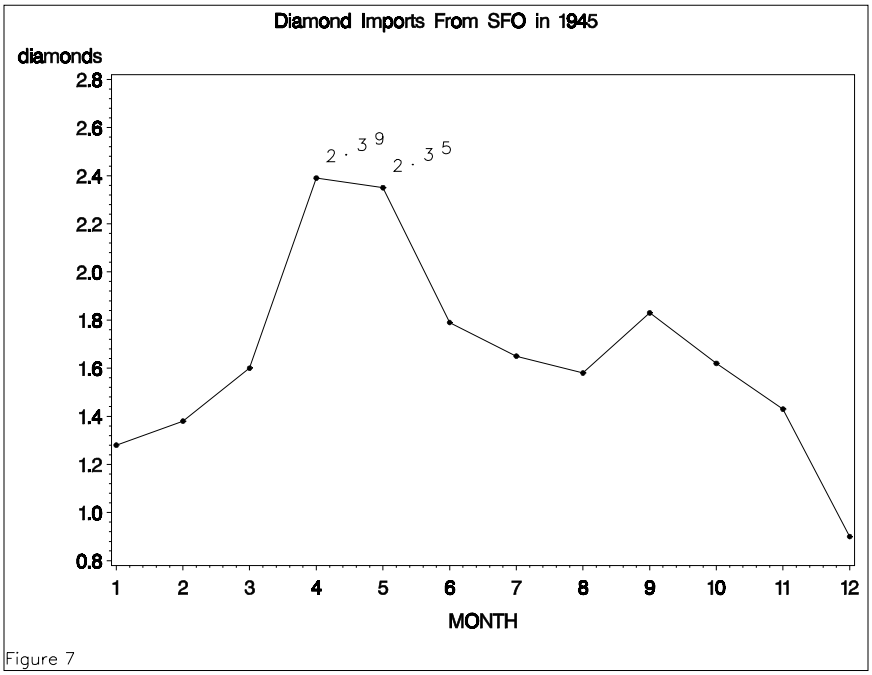


Figure 7

WORKING WITH HISTOGRAMS

When Annotate is used to modify graphs created by PROC GCHART, we need to look at the graph differently than when it is created by PROC GPLOT. Plots created by PROC GPLOT always have both an X and Y axis and thus use both the X and Y variables to determine the location for an Annotate action. Since histograms and other charts created with PROC GCHART do not have the same axis structure as do the plots created by PROC GPLOT, the variables X and Y do not necessarily have the same meaning for graphs created by PROC GCHART. Often this means that special Annotate variables are required to handle the placement of symbols and labels.

Histogram Location Variables

Some histograms may seem to have both vertical and horizontal axes, while pie charts seems to have neither. In fact these charts have MIDPOINT, GROUP, and RESPONSE axes. Annotate labels and symbols require orientation to these axes.

This association between the chart's axes and Annotate is accomplished through the use of special Annotate variables that are used only with PROC GCHART.

- MIDPOINT identifies the location on the midpoint axis
- GROUP identifies a location on the grouping axis
- X or Y used to associate the response axis value

On a vertical histogram the variable Y will be used to designate the vertical (response) axis, while on a horizontal histogram the variable Y will not be used and the horizontal axis variable X will be used to designate the response axis. In both cases the MIDPOINT or GROUP variables will be used to designate the location on the axis other than the response axis. For a vertical histogram without a GROUP= option for instance, the two variables MIDPOINT and Y would form the coordinate pair.

Placing Text Above Histogram Bars

For a vertical histogram the MIDPOINT variable is used instead of X to determine the location on the horizontal axis. The following example adds a label to the top of each vertical bar. The height of each bar is determined using a PROC SUMMARY step. This information is then used to build the Annotate data set.

```
proc sort data=annodat.imports
          out=imports;
   by month;
   run;

* Determine the average imports for each month;
proc summary data=imports;
   by month;
   var diamonds;
   output out=mean mean=mean;
   run;

* Create the axis  using ANNOTATE;
data anno (keep=xsys ysys function size text angle
                style position
                midpoint y); ❶
   set mean;
   length function $8;
   retain xsys ysys '2'  position '3' angle 55
          function 'label'
          size 2 style 'simplex';

   rename month=midpoint; ❶
   y = mean;
   text = left(put(mean,5.2));
   run;

axis1 order = (0 to 4 by 1)
      label = (a=90 '$ in Millions');
axis2 label = ('Month');

pattern1 color=red value=empty;

proc gchart data=mean;
    vbar month ❷ / type=mean sumvar=mean
                   anno=anno discrete
                   raxis=axis1 maxis=axis2; ❸
    TITLE1 'Average Diamond Imports for 1945';
    footnote h=2 f=simplex j=l 'Figure 8';
    run;
    quit;
```

❶ MIDPOINT is used instead of X to place the label horizontally.

❷ Since MONTH is the midpoint variable in the VBAR statement, it is used to provide the value of the Annotate variable MIDPOINT.

❸ The MAXIS (midpoint axis) statement is used to control the label for what is effectively the horizontal axis.

The first four observations of the data set ANNO includes the following:

OBS	MIDPOINT	FUNCTION	XSYS	YSYS	POSITION	ANGLE	SIZE	STYLE	Y	TEXT
1	1	label	2	2	3	55	2	simplex	1.21333	1.21
2	2	label	2	2	3	55	2	simplex	1.68000	1.68
3	3	label	2	2	3	55	2	simplex	2.27667	2.28
4	4	label	2	2	3	55	2	simplex	2.92000	2.92

The following chart is produced with the bars annotated with the monthly import values.

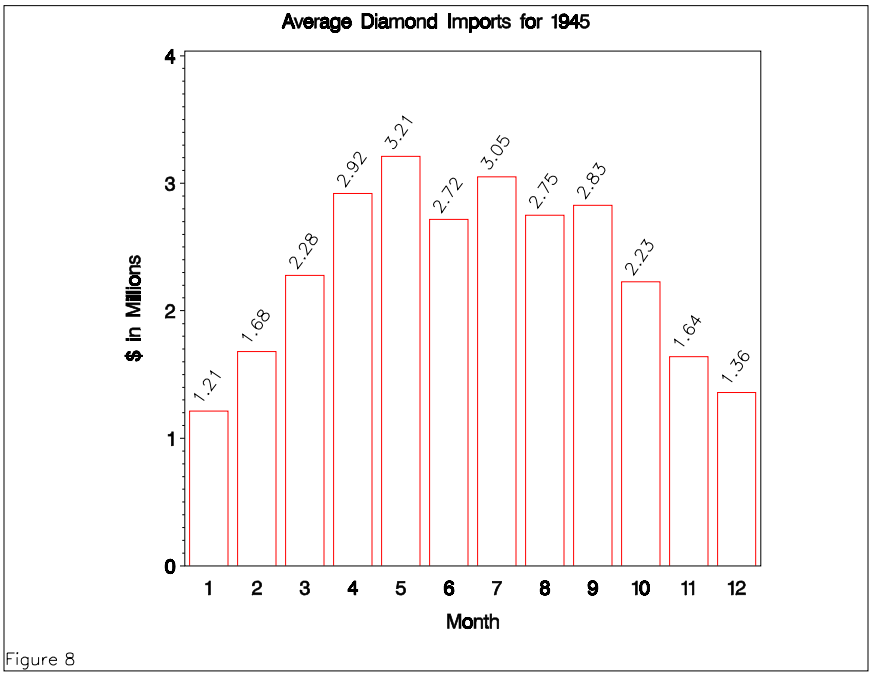


Figure 8

SUMMARY

Although this paper has merely scratched the surface of the capabilities of the Annotate Facility, hopefully it will be enough to get you started. Remember the whole process is all about the Annotate data set and its variables and their values. Use these variables to answer the WHAT, WHERE, and HOW questions. Primary of all of these variables is FUNCTION, and please always use FUNCTION to answer the WHAT question first. Everything else flows from the value of FUNCTION.

ABOUT THE AUTHOR

Art Carpenter's publications list includes three books, and numerous papers and posters presented at SUGI and other user group conferences. Art has been using SAS® since 1976 and has served in various leadership positions in local, regional, national, and international user groups. He is a SAS Certified Professional™ and through California Occidental Consultants he teaches SAS courses and provides contract SAS programming support nationwide.

AUTHOR CONTACT

Arthur L. Carpenter
California Occidental Consultants
P.O. Box 430

Vista, CA 92085-0430

(760) 945-0613
art@caloxy.com
www.caloxy.com

REFERENCES

Much of the text, data, and examples of this workshop and paper are drawn from the book *Annotate: Simply the Basics* by Art Carpenter (Cary, NC: SAS Institute Inc., 1999, 110pp.) and are used here with the author's permission. More detail and further examples can be found in this book (which you really ought to buy considering that you must be at least somewhat interested in the topic to have read this far in the paper).

TRADEMARK INFORMATION

APPENDIX

Code for Figures 1 & 2

```
*****************************************
* fig1_2.sas
*
* Show the areas controlled by XSYS and YSYS.
*
*****************************************;

filename filerefa "&pathcgm\fig1.cgm";
filename filerefb "&pathcgm\fig2.cgm";

goptions reset=all border;
 goptions device=cgmwp80l gsfmode=replace;
*goptions device=win;

* Calculate the average imports for each date;
proc sort data=annodat.imports out=dia1;
by date;
where date le '15jun45'd;
run;

proc means data=dia1 noprint;
by date;
var diamonds;
output out=dia2 mean=mean;
run;

************************************************;
* Create DATA AREA figure;
data figa;
retain xsys ysys '2';
function='move'; x='15jan45'd; y=0;
                 output;
function='bar '; x='15jun45'd; y=3;
                 style='solid';
```

```
                  color='graydd';
                  output;
run;

axis1 order=(0 to 3)
      label=(a=90);

goptions gsfname=filerefa;
proc gplot data=dia2 anno=figa;
plot mean*date / haxis = '15jan45'd to '15jun45'd by month
                  vaxis = axis1;
title1 h=2 f=swiss 'Data Area';
title2 h=1.5 f=simplex "XSYS and YSYS Both Equal '2'";
title3 h=1.5 f=simplex a=90 'Mean Diamond Imports';
title4 h=1.5 f=simplex a=-90 'Title on Right side';
footnote1;
label mean = 'Millions of Dollars'
      date = 'Import Date';
run;
************************************************;
* Create GRAPHICS OUTPUT AREA figure;
goptions gsfname=filerefb;

data figb;
retain xsys ysys '3';
function='move'; x=0; y=0;
                  output;
function='bar '; x=100; y=100;
                  style='solid';
                  color='graydd';
                  output;
run;

proc gplot data=dia2 anno=figb;
plot mean*date / haxis = '15jan45'd to '15jun45'd by month
                  vaxis = axis1;
title1 h=2 f=swiss 'Graphics Output Area';
title2 h=1.5 f=simplex "XSYS and YSYS Both Equal '3'";
title3 h=1.5 f=simplex a=90 'Mean Diamond Imports';
title4 h=1.5 f=simplex a=-90 'Title on Right side';
label mean = 'Millions of Dollars'
      date = 'Import Date';
run;
```

Code for Figure 6

```
*****************************************
* fig6.sas
*
* Demonstrate the use of the POSITION variable.
*
*****************************************;

filename fileref "&pathcgm\fig6.cgm";

goptions reset=all;
goptions border htext=2 ftext=swiss;
goptions device=cgmwp80l gsfmode=replace
         gsfname=fileref;
*goptions device=win;
```

```
* DEMONSTRATE THE POSITION VARIABLE WITH
* FUNCTION=LABEL;
DATA PLTDAT;
     LENGTH POS $1;
     INPUT POS $ X Y;
     CARDS;
     0  0  0
     1  1  1
     2  2  2
     3  3  3
     4  4  4
     5  5  5
     6  6  6
     7  7  7
     8  8  8
     9  9  9
     A  10 10
     B  11 11
     C  12 12
     D  13 13
     E  14 14
     F  15 15
     RUN;

DATA ANNDAT;
     SET PLTDAT;
     LENGTH FUNCTION $8 TEXT $15;
     RETAIN STYLE 'SIMPLEX' XSYS YSYS '2'
         FUNCTION 'LABEL' SIZE 1.5;
     TEXT = 'POS=' || POS;
     POSITION = POS;
     RUN;

PROC GPLOT DATA=PLTDAT ANNO=ANNDAT;
     PLOT Y * X;
     SYMBOL1 v=dot c=black;
     TITLE1 H=2 F=SIMPLEX 'LABEL POSITIONS';
     footnote h=2 f=simplex j=l 'Figure 6';
     run;
     quit;
```