CHAPTER
11

Annotate Dictionary

Overview

This chapter describes the functions, variables, and macros that are used with the Annotate facility.

Refer to Chapter 10, “The Annotate Data Set,” on page 403 for these topics:

- an introduction to Annotate concepts
- a discussion of how to build Annotate data sets and generate Annotate graphics
- examples of graphs that use the Annotate facility.

For an explanation of the error messages that are produced by the Annotate facility, see the Help system.

Annotate Functions

In an Annotate data set, the value of the FUNCTION variable specifies what action the observation performs. Annotate functions act in conjunction with the other Annotate variables that determine where and how to perform the action. Many of these variables are function dependent, that is, what they do depends on the function they are used with. For example, with the LABEL function the STYLE variable specifies a font; with the BAR function, STYLE specifies a pattern.

This section describes all of the values of the FUNCTION variable. For each function it

- describes the function's action.
- notes whether the function updates the internal coordinate variables XLAST, YLAST and XLSTT, YLSTT.
- describes how other Annotate variables behave with the function. The variables that are used with each function are listed in Figure 11.1 on page 430. For a complete description of each variable, see "Annotate Variables" on page 455.

For a summary of drawing and programming tasks performed by the FUNCTION variable, see Table 10.2 on page 409.

Figure 11.1 Annotate Variables used with Annotate Functions

Annotate Graphics Functions ▶ ▼ Annotate Variables		BAR	CNTL2TXT	COMMENT	DEBUG	DRAW	DRAW2TXT	FRAME	LABEL	MOVE	PIE	PIECNTR	PIEXY	POINT	POLY	POLYCONT	POP	PUSH	SWAP	SYMBOL	TXT2CNTL
Data Set Variables	ANGLE								■		■		■								
	CBORDER								■											■	
	CBOX								■											■	
	COLOR	■				■	■	■	■		■			■	■	■				■	
	FUNCTION	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
	GROUP	■				■			■	■	■	■		■	■	■				■	
	HSYS					■	■	■	■		■	■								■	
	HTML	■							■			■			■					■	
	LINE	■				■	■	■			■				■						
	MIDPOINT	■				■			■	■	■	■		■	■	■				■	
	POSITION								■												
	ROTATE								■		■										
	SIZE	■				■	■	■	■		■	■	■		■					■	
	STYLE	■						■	■		■				■					■	
	SUBGROUP	■				■			■	■	■	■		■	■	■				■	
	TEXT			■					■											■	
	WHEN	■				■	■	■	■	■	■	■	■	■	■	■				■	
	X	■				■			■	■	■	■		■	■	■				■	
	XC	■				■			■	■	■	■		■	■	■				■	
	XSYS	■				■		■	■	■	■	■		■	■	■				■	
	Y	■				■			■	■	■	■		■	■	■				■	
	YC	■				■			■	■	■	■		■	■	■				■	
	YSYS	■				■		■	■	■	■	■		■	■	■				■	
	Z	■				■			■	■	■	■		■	■	■				■	
	ZSZS	■				■			■	■	■	■		■	■	■				■	
Internal Variables*	XLAST YLAST	■	■			■				■	■	■	■	■			■	■	■		■
	XLSTT YLSTT		■						■								■	■	■	■	■

*The internal variables are automatically updated by the graphics functions indicated.

BAR Function

Draws a fillable rectangle whose lower-left corner is defined by the internal variables (XLAST, YLAST) and whose upper-right corner is defined by the X, Y variable pair.

Updates: XLAST, YLAST

Syntax

FUNCTION='BAR';

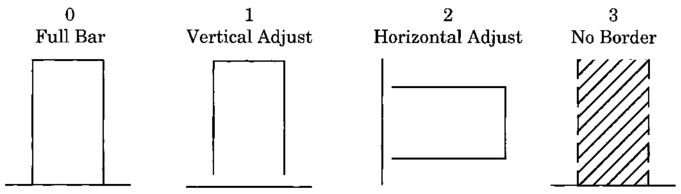
Associated Variables

COLOR='*color*'
: specifies the color of either the interior of the bar or the outline of the bar. *Color* can be any SAS/GRAPH color name. The part of the bar affected depends on the value of the STYLE variable. If STYLE specifies a pattern or fill, the COLOR variable determines the color of the interior. If STYLE specifies an empty pattern, the COLOR variable determines the color of the outline of the bar.

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
: specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

LINE=0...3
: specifies the direction in which to adjust the outline of the bar. Use LINE values 1 and 2 to offset a particular bar from an axis or adjoining area. Figure 11.2 on page 431 illustrates LINE values.

Figure 11.2 LINE Values for Bars



SIZE=*thickness*
: specifies a line thickness for the rectangle

STYLE='*fill-pattern*'
: specifies the pattern that fills the bar. *Fill-pattern* can be the following bar and block patterns:

SOLID S	a solid fill.

EMPTY E	an empty fill.
style<density>	a shaded pattern: *style* can be R | X | L *density* can be 1...5

WHEN='B' | 'A'
: specifies when to draw the bar in relation to other procedure output. See "WHEN Variable" on page 474.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate*
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
: define the upper-right corner of a bar (rectangle) whose lower-left corner is (XLAST,YLAST). Use the Z variable only when you are annotating output from the G3D procedure. Figure 11.3 on page 432 illustrates the use of these coordinates.

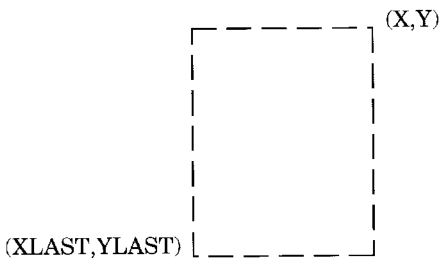
XSYS='*coordinate-system*'
: specifies the coordinate system for the X or XC variable. The XC variable can be used only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
: specifies the coordinate system for Y or YC variable. The YC variable can only be used with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
: specifies the coordinate system for the Z variable. See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

Details Figure 11.3 on page 432 shows how the XLAST, YLAST, and X, Y variables define the bar. The values of the XLAST and YLAST variables are usually initialized with a MOVE function or another function that updates the XLAST and YLAST pair.

Figure 11.3 Points Used to Construct a Bar



CNTL2TXT Function

Copies the values of the internal coordinates stored in the variable pairs (XLAST, YLAST) to (XLSTT, YLSTT).

Updates: XLSTT, YLSTT

Syntax

FUNCTION='CNTL2TXT';

Details

You can use CNTL2TXT to calculate the position of labels on a graph. For example, the following DATA step uses CNTL2TXT to position a pie slice label in the center of the arc and just beyond the arc itself, as shown in Figure 11.6 on page 434.

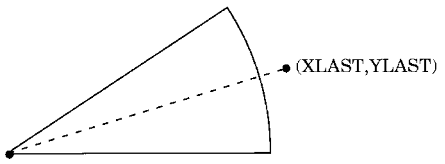
First, use the PIE function to draw the pie slice:

```
data pielabel;
   retain xsys ysys '3';
   length function style $ 8;
   function='pie'; size=20; x=30; y=30;
      style='empty'; rotate=45; output;
```

Then use the PIEXY function to calculate a point outside of the arc as shown in Figure 11.4 on page 433.

```
/* find a point that is half of the arc (rotate*.5) */
      /* and is 4 units beyond the radius (size=1.1) */
   function='piexy'; angle=rotate*.5; size=1.1; output;
```

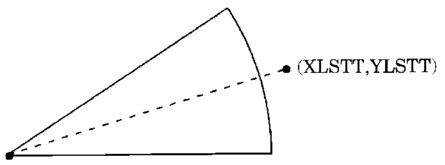
Figure 11.4 Position Calculated with the PIEXY Function



At this point, the XLAST and YLAST variables contain the coordinates of the point that is calculated by PIEXY. However, (XLAST, YLAST) cannot be used directly by text functions. Use CNTL2TXT to copy the coordinates in (XLAST, YLAST) to the XLSTT and YLSTT variables, which text functions can use. Figure 11.5 on page 433 shows the results.

```
function='cntl2txt'; output;
```

Figure 11.5 Coordinates after Using the CNTL2TXT Function

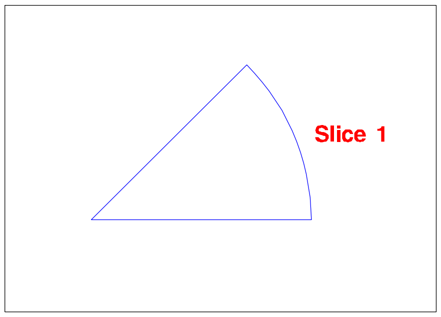


Now you can use the LABEL function to write the label as shown in Figure 11.6 on page 434. Specify missing values for the X and Y variables to force LABEL to use the XLSTT and YLSTT variables instead of the X and Y variables.

```
/* write the label 'Slice 1' and position it to     */
      /* the right of the point stored in XLSTT and YLSTT */
   function='label'; text='Slice 1'; angle=0; rotate=0;
      position='6'; style='swissb'; size=4; x=.; y=.;
      output;
run;

   /* draw the Annotate graphics */
proc ganno anno=pielabel;
run;
quit;
```

Figure 11.6 Labeled Pie Slice



COMMENT Function

Inserts comments within the Annotate data set. The observations generated by the COMMENT function are ignored when the data set is processed.

Syntax

FUNCTION='COMMENT';

Associated Variables

TEXT='*text-string*'
specifies the comment to write to the data set.

DEBUG Function

Writes the values of internal coordinates and Annotate variables to the SAS log before and after processing the next command (unless it is DEBUG) in the Annotate DATA step.

Syntax

FUNCTION='DEBUG';

DRAW Function

Draws a line in the graphics output from the (XLAST, YLAST) coordinates to the (X, Y) coordinates specified in the function.

Updates: XLAST, YLAST

Syntax

FUNCTION='DRAW';

Associated Variables

COLOR='*color*'
: specifies the color of the line that is being drawn. *Color* can be any SAS/GRAPH color name.

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
: specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

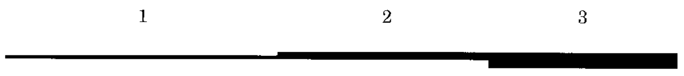
HSYS='*coordinate-system*'
: specifies the coordinate system for the SIZE variable. See "HSYS Variable" on page 461 for an explanation of *coordinate-system*.

LINE=1...46
: specifies the line type of the line that is being drawn. See "Specifying Line Types" on page 248 for an illustration of the line types.

SIZE=*line-thickness*
: specifies the thickness of the line that is being drawn. The units depend on the value of the HSYS variable. For example, if HSYS='3', the SIZE variable is in units of percent of the graphics output area. If HSYS='4', the SIZE variable is in units of cells of the graphics output area.

As the thickness of the line increases, it may be impossible to center around a given coordinate. For example, if you specify a thickness of value 2 and HSYS='4', the first line is drawn at the (X, Y) coordinates. The second is drawn slightly above the first. The exact amount varies by device, but it is always one pixel in width. A thickness of value 3 produces one line above, one line at, and one line below the (X, Y) coordinate position. See Figure 11.7 on page 436 for examples of line thicknesses.

Figure 11.7 Sample Line Thicknesses Used with the SIZE Variable



WHEN='B' | 'A'
: specifies when to draw the line in relation to other procedure output. See "WHEN Variable" on page 474.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
: specify the endpoint of a line drawn from (XLAST, YLAST) to (X,Y).

XSYS='*coordinate-system*'
: specifies the coordinate system for the X or XC variable. The XC variable can be used only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
: specifies the coordinate system for the Y or YC variable. The YC variable can be used only with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
: specifies the coordinate system for the Z variable (PROC G3D only). See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

DRAW2TXT Function

Draws a line from (XLAST, YLAST) to (XLSTT, YLSTT) without updating any of those variables.

Syntax

FUNCTION='DRAW2TXT';

Associated Variables

COLOR=’*color*’
: specifies the line color. *Color* can be any SAS/GRAPH color name.

HSYS=’*coordinate-system*’
: specifies the coordinate system for the SIZE variable. See “HSYS Variable” on page 461 for an explanation of *coordinate-system*.

LINE=1...46
: specifies the line type of the line that is being drawn. See “Specifying Line Types” on page 248 for an illustration of the line types.

SIZE=*line-thickness*
: specifies the thickness of the line that is being drawn. See “DRAW Function” on page 435 for details.

WHEN=’B’ | ’A’
: specifies when to draw the line in relation to generation of the procedure output. See “WHEN Variable” on page 474.

Details DRAW2TXT is useful for underlining text.

DRAW2TXT does not update the (XLAST, YLAST) or (XLSTT, YLSTT) coordinates; neither can it interrupt a POLYCONT sequence.

FRAME Function

Draws a border around the portion of the display area defined by the XSYS and YSYS variables. Optionally specifies a background color for the framed area.

Syntax

FUNCTION=’FRAME’;

Associated Variables

COLOR=’*color*’
: specifies the frame color and, if the STYLE variable is specified, fills the interior of the frame. *Color* can be any SAS/GRAPH color name.

HSYS=’*coordinate-system*’
: specifies the coordinate system for the SIZE variable. See “HSYS Variable” on page 461 for an explanation of *coordinate-system*.

LINE=1...46
: specifies the line type with which to draw the frame. See “Specifying Line Types” on page 248for an illustration of the line types.

SIZE=*line-thickness*
: specifies the thickness of the line with which to draw the frame. See “DRAW Function” on page 435 for details.

STYLE='*fill-pattern*'

specifies the pattern that fills the area that is bounded by the frame. *Fill-pattern* can be the following bar and block patterns:

SOLID S	a solid fill.
EMPTY E	an empty fill.
style<density>	a shaded pattern: *style* can be R | X | L *density* can be 1...5

See also the discussion of fill patterns for bars and blocks in VALUE= on page 214.

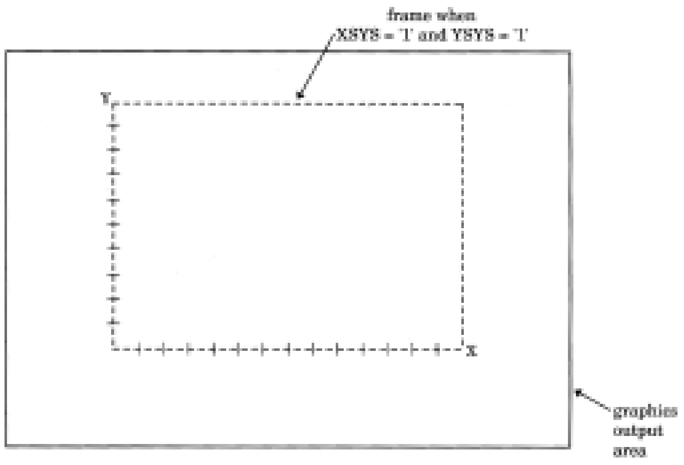
WHEN='B' | 'A'

specifies when to draw the frame in relation to other procedure output.

XSYS='*coordinate-system*'
YSYS='*coordinate-system*'

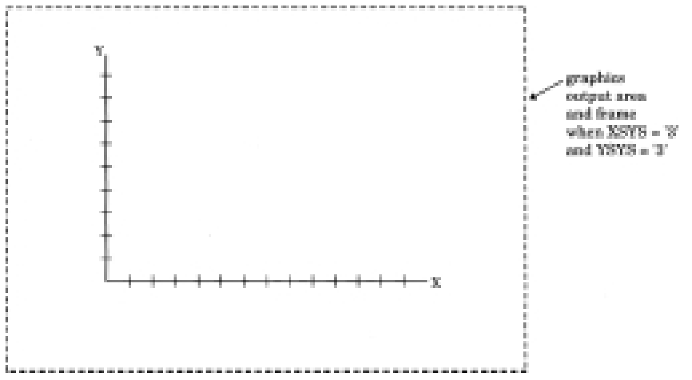
define the area to be enclosed by the frame. For example, if XSYS='1' and YSYS='1', the frame encloses the axis area as shown in Figure 11.8 on page 438. See "XSYS Variable" on page 477 and the YSYS variable on "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

Figure 11.8 Frame Created When XSYS='1' and YSYS='1'



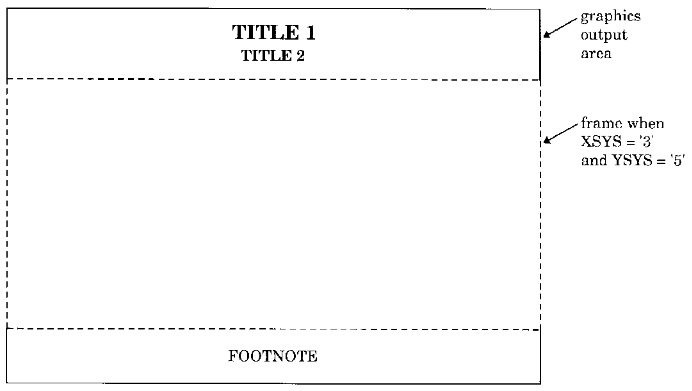
If XSYS='3' and YSYS='3', the frame encloses the entire graphics output area, as shown in Figure 11.9 on page 439.

Figure 11.9 Frame Created When XSYS='3' and YSYS='3'


Y
X
graphics output area and frame when XSYS = '3' and YSYS = '3'

The values for XSYS and YSYS do not have to be the same. If XSYS='3' and YSYS='5', the frame encloses the entire width of the graphics output area; however, vertically, the frame only encloses the procedure output area as shown in Figure 11.10 on page 439.

Figure 11.10 Frame Created When XSYS='3' and YSYS='5'



See "XSYS Variable" on page 477 and "YSYS Variable" on page 480 for an explanation of these variables and the areas that they affect.

Details Use FRAME to simulate the CBACK= graphics option on devices (such as plotters) that do not support that option. For devices that do support the CBACK= graphics option, FRAME works in addition to that option. FRAME does not alter the (XLAST, YLAST) coordinates. See "CBACK" on page 309 for more information on CBACK=.

LABEL Function

Places text in the graphics output. Associated variables can control the color, size, font, base angle, and rotation of the characters displayed.

Updates: XLSTT, YLSTT

Syntax

FUNCTION='LABEL';

Associated Variables

ANGLE=0...360
: specifies the baseline angle of the character string with respect to the horizontal. The pivot point is at (X, Y), and the rotation is in a counterclockwise direction.

CBORDER='*color*' | 'CTEXT'
: draws a colored border around the text. *Color* can be any SAS/GRAPH color name.

CBOX='*color*' | 'CBACK
: draws a solid, colored box behind the text. *Color* can be any SAS/GRAPH color name.

COLOR='*color*'
: specifies the color of the text. *Color* can be any SAS/GRAPH color name.

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
: specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

HSYS='*coordinate-system*'
: specifies the coordinate system for the SIZE variable. See "HSYS Variable" on page 461 for an explanation of *coordinate-system*.

POSITION='*text-position*' | '0'
: controls the text string placement and alignment. *Text-position* can be one of the characters 1 through 9, A through F, <, +, or >. Invalid or missing values default to POSITION='5'. POSITION should always be a character variable of length 1. For details, see "POSITION Variable" on page 466.

ROTATE=*rotation-angle*
: specifies the rotation angle of each character in the string. It is equivalent to the ROTATE= option in the FOOTNOTE, NOTE, and TITLE statements.

SIZE=*height*
: specifies the height of the text string. The SIZE variable units are based on the value of the HSYS variable.

STYLE='*font*' | "'*hardware-font-name*'" | 'NONE'
: specifies the font with which to draw the text that is specified by the TEXT variable. See "STYLE Variable (Fonts)" on page 471 for a description of the various font specifications.

TEXT='*text-string*'
: specifies the text to be written. *Text-string* can be up to 200 characters. Define the TEXT variable with sufficient length to contain all of the characters in your text string. If you need longer strings, use separate observations and POSITION='0' to continue the text.

WHEN='B' | 'A'
specifies when to draw the text strings in relation to other procedure output.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
specify the start point of the text string. The Z variable can be used only with the G3D procedure. Optionally, you can modify the placement of the text string with the POSITION variable.

XSYS='*coordinate-system*'
specifies the coordinate system for the X or XC variable. Use the XC variable only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
specifies the coordinate system for the Y or YC variable. Use the YC variable only with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
specifies the coordinate system for the Z variable. See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

MOVE Function

Moves the drawing pointer to a specific location without drawing a line.

Updates: XLAST, YLAST

Syntax

FUNCTION='MOVE';

Associated Variables

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

WHEN='B' | 'A'
specifies when to perform the move in relation to other procedure output. See also "WHEN Variable" on page 474.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
specify the coordinates to which the pen is to be moved. The Z variable can only be used with the G3D procedure.

XSYS='*coordinate-system*'
specifies the coordinate system for the X or XC variable. Use the XC variable only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
specifies the coordinate system for the Y or YC variable. Use the YC variable only with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
specifies the coordinate system for the Z variable. See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

Details Use MOVE to prepare for a DRAW command, a BAR command, or programming functions.

PIE Function

Draws pie slices in the graphics output.

Updates: XLAST, YLAST to coordinates for center of the slice.

Syntax

FUNCTION='PIE';

Associated Variables

ANGLE=*starting-angle*
specifies the starting angle of the slice arc. The default is 0.00 (horizontal) if the ANGLE variable is not specified for the first slice. After the first slice, the default is the ending angle of the slice arc just drawn if ANGLE=. (missing). Therefore, you can specify consecutive pie slices more easily by omitting the start and end calculations that are otherwise required. If you want the next slice to start at an angle that is different from the ending angle of the previous slice, you must specify a value for the ANGLE variable.

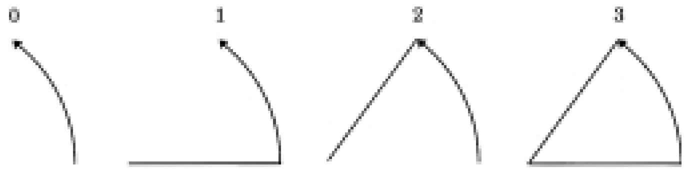
COLOR='*color*'
specifies the color of the pie slice, if a pattern is specified in the STYLE variable. If you specify STYLE='EMPTY', the COLOR variable also specifies the outline color of the pie slices. *Color* can be any SAS/GRAPH color name.

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

HSYS='*coordinate-system*'
specifies the coordinate system for the SIZE variable. See "HSYS Variable" on page 461 for an explanation of *coordinate-system*.

LINE=0...3
specifies which slice line (or lines) to draw. See Figure 11.11 on page 443 for line values and their actions. LINE=0 draws only the outside of the arc and enables you to draw a circle.

Figure 11.11 LINE Values Used with the PIE Function



ROTATE=*rotation-angle*
specifies the angle of rotation or the delta angle of the slice arc. The default is 0.00.
For example, if you specify these statements, the slice arc that is drawn begins at 90 degrees (vertical) and ends at 135 degrees (90+45):

```
function='pie'; angle=90; rotate=45; output;
```

The ANGLE variable is internally updated to the end value, 135 degrees. The value is modified only internally. If a second PIE is used and the ANGLE variable contains a missing value, the start angle is assumed to be the previous end, or 135 degrees. The arc continues from that point.
If you specify the previous statements and then specify these statements, the slice begins at 135 degrees (the end angle from the previous slice) and extends another 45 degrees to the end point, 180 degrees.

```
function='pie'; angle=.; rotate=45; output;
```

This action repeats for every missing angle in the sequence.

SIZE=*radius*
specifies the radius of the circle being drawn. The SIZE variable uses units that are determined by the HSYS variable.

STYLE='*fill-pattern*'
specifies the value of the pattern that fills the pie slices. *Fill-pattern* can be the following pie patterns:

PSOLID PS	a solid fill.
PEMPTY PE	an empty fill.

P*density*<*style*<*angle*>>	a shaded pattern:
	density can be 1...5
	style can be X | N
	angle can be 0...360

For example, if STYLE='P5N15', a pie slice with a fill of parallel lines is produced. The fill uses the heaviest density to draw the lines, and the parallel lines are drawn at a 15-degree angle from perpendicular to the radius of the pie slice. See also the discussion of fill patterns for pie and star charts in VALUE= on page 217.

WHEN='B' | 'A'
specifies when to draw the pie slice in relation to other procedure output. See also "WHEN Variable" on page 474.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
define the center of the slice. The pivot point for all slices is the point referenced by X, Y, and Z (with PROC G3D only). The first PIE command that is issued sets the center at the (X,Y) value. If subsequent values for X and Y are missing, the coordinates of the center point are used.

XSYS='*coordinate-system*'
specifies the coordinate system for the X or XC variable. Use the XC variable only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
specifies the coordinate system for the Y or YC variable. Use the YC variable only with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
specifies the coordinate system for the Z variable. See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

See Also

"CNTL2TXT Function" on page 432

PIECNTR Function

Sets new center and radius values for later use by the PIEXY function but does not draw an arc.

Updates: XLAST, YLAST

Syntax

FUNCTION='PIECNTR';

Associated Variables

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

HSYS='*coordinate-system*'
specifies the coordinate system for the SIZE variable. See "HSYS Variable" on page 461 for an explanation of *coordinate-system*.

SIZE=*radius*
specifies the new radius of the pie slice. The new radius is used by a subsequent PIEXY function. The HSYS variable determines the SIZE variable units.

WHEN='B' | 'A'
specifies when to draw the pie slice in relation to other procedure output.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
define the center and radius of the slice. All slices are referenced from that center. Use the Z variable only with the G3D procedure.

XSYS='*coordinate-system*'
specifies the coordinate system for the X or XC variable. Use the XC variable only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
specifies the coordinate system for the Y or YC variable. Use the YC variable only with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
specifies the coordinate system for the Z variable. See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

PIEXY Function

Calculates a point on the outline of the slice arc.

Updates: XLAST, YLAST

Syntax

FUNCTION='PIEXY';

Associated Variables

ANGLE=*rotation-angle*

specifies the angle of rotation when moving around the perimeter of a pie. The ANGLE variable determines the angle at which the point is located relative to 0 (the three o'clock position). The default is 0.00.

SIZE=*radius-multiplier*

determines the distance from the center of the slice to the point that is being calculated. The point's distance is the current value of the SIZE variable multiplied by the radius (that is, the SIZE variable) of the previously drawn slice. To position a graphics element inside the pie slice, set the SIZE variable to less than 1; to position it outside of the pie slice, set the SIZE variable to greater than 1. For example, if you specify these statements, the point calculated is 1.1 times the radius (where the radius is taken from the SIZE variable that is used with the previous FUNCTION='PIE' or FUNCTION='PIECNTR' observation).

```
function='piexy'; size=1.1; output;
```

WHEN='B' | 'A'

specifies when to update the internal coordinate pair (XLAST, YLAST) in relation to other procedure output. See also "WHEN Variable" on page 474.

Details PIEXY does not draw anything but places the calculated coordinates of the point in the internal coordinate pair (XLAST, YLAST). Then you can use XLAST and YLAST with other functions to perform other graphics actions, such as labeling pie slices. If you need to use the calculated position for a text function, use the SWAP or CNTL2TXT to put (XLAST, YLAST) into (XLSTT, YLSTT).

PIEXY assumes that a pie slice has been drawn or that FUNCTION='PIECNTR' has been used. Erroneous results can occur if a slice has not been drawn and PIEXY is invoked.

Figure 11.12 on page 446 shows a pie slice that is drawn with the PIE function. Figure 11.13 on page 446 shows a point beyond the arc that was calculated using the PIEXY function.

Figure 11.12 Pie Slice Drawn with the PIE Function

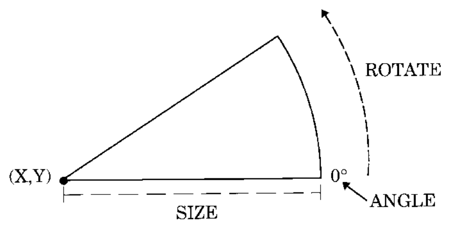
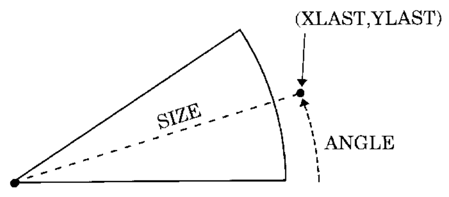


Figure 11.13 Point Calculated with the PIEXY Function



See Also

"CNTL2TXT Function" on page 432

POINT Function

Places a single point at the (X, Y) coordinates in the color you specify. The point is one visible pixel in size.

Updates: XLAST, YLAST

Syntax

FUNCTION='POINT';

Associated Variables

COLOR='*color*'
: specifies the color of the point to be drawn. *Color* can be any SAS/GRAPH color name.

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
: specify coordinates when used with HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

WHEN='B' | 'A'
: specifies when to draw the point in relation to other procedure output.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
: specify the coordinates of the point that is to be drawn. Use the Z variable only with the G3D procedure.

XSYS='*coordinate-system*'
: specifies the coordinate system for the X or XC variable. Use the XC variable only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
: specifies the coordinate system for the Y or YC variable. Use the YC variable only with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
: specifies the coordinate system for the Z variable. See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

POLY Function

Specifies the beginning point of a polygon. Associated variables can define the fill pattern and color, as well as the line type that outlines the polygon.

Syntax

FUNCTION='POLY';

Associated Variables

COLOR='*color*'
: specifies the color of the interior of the polygon, if a pattern is specified for the STYLE variable. The outline color is specified with the POLYCONT function. *Color* can be any SAS/GRAPH color name.

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
: specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with data coordinate systems 1, 2, 7, and 8.

LINE=1...46
: specifies the line type that outlines the polygon. See "Specifying Line Types" on page 248 for an illustration of the line types.

SIZE=*thickness*
: specifies a line thickness for the polygon

STYLE='*fill-pattern*'
: specifies the value of the pattern that fills the polygon. *Fill-pattern* can be the following map patterns:

MSOLID MS	a solid pattern
MEMPTY ME	an empty pattern
M*density*<*style*<*angle*>>	a shaded pattern: *density* can be 1...5 *style* can be X | N *angle* can be 0...360.

For example, if STYLE='MSOLID' for the POLY function, the fill area that is drawn by the POLYCONT sequence uses a solid fill. If STYLE='M5N15', the fill area uses a shaded fill of parallel lines. The *fill-pattern* value M5N15 specifies that the lines use the heaviest density, are parallel, and are drawn at a 15-degree angle from the horizontal. See also the discussion of fill patterns for maps in VALUE= on page 215.

WHEN='B' | 'A'
specifies when to begin the polygon in relation to other procedure output.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
specify the initial point of the polygon that is being created. Use the Z variable only with the G3D procedure.

XSYS='*coordinate-system*'
specifies the coordinate system for the X or XC variable. Use the XC variable only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
specifies the coordinate system for the Y or YC variable. Use the YC variable only with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
specifies the coordinate system for the Z variable. See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

Details Use POLY with POLYCONT to define and fill areas in the graphics output. POLY and POLYCONT do not update the (XLAST, YLAST) coordinates.

See Also

"POLYCONT Function" on page 449

POLYCONT Function

Continues drawing a polygon begun with the POLY function. POLYCONT specifies each successive point in the polygon definition.

Syntax

FUNCTION='POLYCONT';

Associated Variables

COLOR='*color*'
specifies the polygon outline color. *Color* can be any SAS/GRAPH color name. You can specify an outline color only with the first POLYCONT command in the sequence; all subsequent POLYCONT commands ignore the COLOR variable. If you do not specify a color, the POLYCONT function uses the interior color that was specified with the POLY function.

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

WHEN='B' | 'A'
specifies when to draw the polygon in relation to other procedure output.

X=*horizontal-coordinate*
Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC='*character-type-horizontal-coordinate*'
YC='*character-type-vertical-coordinate*'
specify a point on the outline of the polygon that is being created. Use the Z variable only with the G3D procedure.

XSYS='*coordinate-system*'
specifies the coordinate system for the X and XC variable. Use the XC variable only with XSYS='2'. See "XSYS Variable" on page 477 for an explanation of *coordinate-system*.

YSYS='*coordinate-system*'
specifies the coordinate system for the Y and YC variable. Use the YC variable only with YSYS='2'. See "YSYS Variable" on page 480 for an explanation of *coordinate-system*.

ZSYS='*coordinate-system*'
specifies the coordinate system for the Z variable. See "ZSYS Variable" on page 482 for an explanation of *coordinate-system*.

Details The polygon definition is terminated by a new POLY command or by any of these functions:

BAR

DRAW

DRAW2TXT

FRAME

LABEL

MOVE

PIE

PIECNTR

PIEXY

POINT

SYMBOL

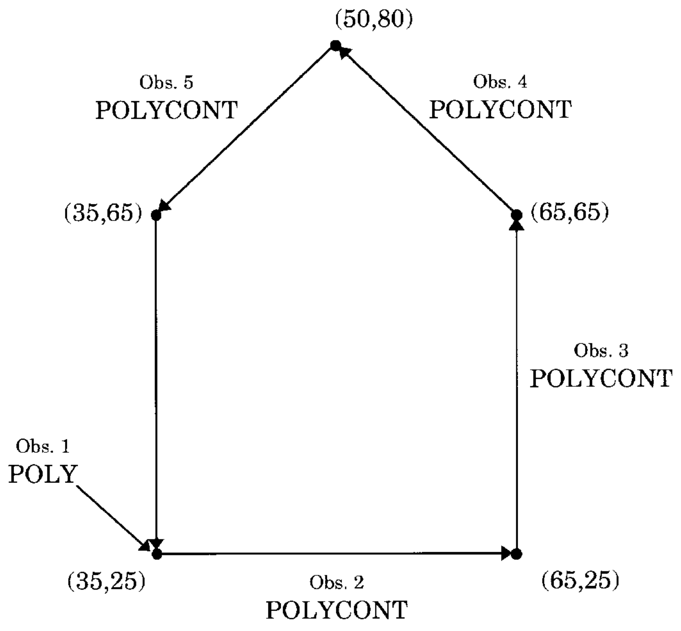
Use POLY and POLYCONT together to draw a polygon. The (X, Y) observation from the POLY function and the last (X, Y) observation from POLYCONT are assumed to connect. Thus, you are not required to respecify the first point. For example, these statements draw a pentagon like the one in Figure 11.14 on page 451:

```
data house;
   retain xsys ysys '3';
   length function $ 8;
```

```
      /* start at the lower left corner */
   function='poly'; x=35; y=25; output;
      /* move to the lower right corner */
   function='polycont'; x=65; y=25; output;
      /* move to the upper right corner */
   function='polycont'; x=65; y=65; output;
      /* move to the center top*/
   function='polycont'; x=50; y=80; output;
      /* move to the upper left corner and complete the figure */
   function='polycont'; x=35; y=65; output;
run;

proc ganno anno=house;
run;
quit;
```

Figure 11.14 Pentagon Produced with the POLY and POLYCONT Functions



Missing values for the X and Y variables that are specified with POLYCONT are interpreted differently from the way that they are interpreted with the other functions. Other functions use the missing values to request a default value. POLYCONT interprets a missing value as a discontinuity (that is, a hole) in the polygon. If you are not using the data coordinate system and you specify an X or Y value of –999 in a POLYCONT observation, the default of (XLAST, YLAST) is used. Missing values indicate holes and are handled identically in the Annotate facility and the GMAP procedure. See “Displaying Map Areas and Response Data” on page 738 for more information on handling missing values.

POP Function

Removes the (XLAST, YLAST) and (XLSTT, YLSTT) values from the LIFO stack and updates the internal coordinate pairs with the retrieved values.

Updates: (XLAST, YLAST) and (XLSTT, YLSTT)

Syntax

FUNCTION='POP';

Details Use POP when you want to access the values of (XLAST, YLAST) and (XLSTT, YLSTT) that you most recently stored with the PUSH function. See the PUSH function for a description of the LIFO stack.

PUSH Function

Adds current (XLAST, YLAST) and (XLSTT, YLSTT) values to the LIFO stack.

Syntax

FUNCTION='PUSH';

Details The LIFO (last-in-first-out) stack is a storage area where you can keep internal coordinate values for later use by utility functions without recalculating those values. LIFO stacks manage the stored data so that the last data stored in the stack is the first data removed from the stack.

Use the stack to save the current values of (XLAST, YLAST) and (XLSTT, YLSTT) and use them with functions later in the DATA step. You store and retrieve these values from the stack with the PUSH and POP functions. The PUSH function copies the current values of XLAST, YLAST, XLSTT, and YLSTT onto the stack. The POP function copies values from the stack into XLAST, YLAST. XLSTT, and YLSTT.

SWAP Function

Exchanges values of (XLAST, YLAST) with (XLSTT, YLSTT) and vice versa.

Updates: (XLAST, YLAST) and (XLSTT, YLSTT)

Syntax

FUNCTION='SWAP';

Details Use SWAP when you want to use both the (XLAST, YLAST) and (XLSTT, YLSTT) coordinates for text and nontext functions, respectively.

SYMBOL Function

Places symbols in the graphics output. Associated variables can specify the color, font, and height of the symbols displayed.

Updates: XLSTT, YLSTT

Syntax

FUNCTION='SYMBOL';

Associated Variables

CBORDER='*color*' | 'CTEXT'
: draws a colored border around the text. *Color* can be any SAS/GRAPH color name.

CBOX='*color*' | 'CBACK'
: draws a solid, colored box behind the text. *Color* can be any SAS/GRAPH color name.

COLOR='*color*'
: specifies the symbol color. *Color* can be any SAS/GRAPH color name. The COLOR variable behaves in the same way as the COLOR= option in the SYMBOL statement. See COLOR= on page 228 for details

GROUP=*group-value*
MIDPOINT=*midpoint-value*
SUBGROUP=*subgroup-value*
: specify coordinates for HBAR and VBAR charts from the GCHART procedure. Use these variables only with the data coordinate systems 1, 2, 7, and 8.

HSYS='*coordinate-system*'
: specifies the coordinate system for the SIZE variable. See "HSYS Variable" on page 461 for an explanation of *coordinate-system*.

SIZE=*height*
: specifies the height of the symbol that is being drawn, using units determined by the HSYS variable. The SIZE variable is equivalent to the HEIGHT= option in the SYMBOL statement. See HEIGHT= on page 229 for details.

STYLE='*font*' | "'*hardware-font-name*'" | 'NONE';
: specifies the font that is used to draw the symbol that is specified by the TEXT variable. See "STYLE Variable (Fonts)" on page 471 for a description of the various font specifications.

  When the STYLE variable is used with the SYMBOL function, it behaves the same as the FONT= option in the SYMBOL statement. By default, no font is specified and the symbol that is specified by the TEXT variable is taken from the special symbol table. If you use STYLE to specify a symbol font, such as Marker, the string that is assigned by the TEXT variable is the character code for a symbol. If you use STYLE to specify a text font, such as Swiss, the string assigned by the TEXT variable is displayed as text. See FONT= on page 229 for details.

TEXT=’*special-symbol*’ | ’*text-string*’;

specifies the symbol to be displayed. *Special-symbol* can be up to eight characters long. Values for *special-symbol* are those described in the VALUE= option of the SYMBOL statement and are illustrated in VALUE= on page 241.

If you also specify a text font with the STYLE variable, you can specify a text string that is displayed as the symbol. The maximum length for *text-string* is 200 characters.

When the TEXT variable is used with the SYMBOL function, it behaves the same as the VALUE= option in the SYMBOL statement. See VALUE= on page 241 for details.

WHEN=’B’ | ’A’

specifies when to draw the symbols in relation to other procedure output.

Y=*vertical-coordinate*
Z=*depth-coordinate* (PROC G3D only)
XC=’*character-type-horizontal-coordinate*’
YC=’*character-type-vertical-coordinate*’

specify the point at which the symbol is placed. Use the Z variable only with the G3D procedure.

XSYS=’*coordinate-system*’

specifies the coordinate system for the X or XC variable. Use the XC variable only with XSYS=’2’. See “XSYS Variable” on page 477 for an explanation of *coordinate-system*.

YSYS=’*coordinate-system*’

specifies the coordinate system for the Y or YC variable. Use the YC variable only with YSYS=’2’. See “YSYS Variable” on page 480 for an explanation of *coordinate-system*.

ZSYS=’*coordinate-system*’

specifies the coordinate system for the Z variable. See “ZSYS Variable” on page 482 for an explanation of *coordinate-system*.

Details SYMBOL is similar to the LABEL function with these exceptions:

- SYMBOL draws symbols. If you do not specify a font, SYMBOL can use the symbols found in Figure 8.21 on page 243.
- The text cannot be rotated or angled.
- The text string cannot be longer than eight characters.
- The text string is always centered with respect to *x* and *y*.

TXT2CNTL Function

Copies the values (XLSTT, YLSTT) to (XLAST, YLAST), replacing previous values of (XLAST, YLAST).

Syntax

FUNCTION='TXT2CNTL';

Details TXT2CNTL allows nontext functions to use the ending position of a text string as a starting or ending point.

Annotate Variables

When an Annotate data set is processed, the Annotate facility looks at the values of specific variables in order to draw graphics. This section describes all of the Annotate variables in alphabetical order. Not all variables are used with all functions. Refer to the description of the individual functions in "Annotate Functions" on page 429 for more information about how each variable is used with each function. For a summary of Annotate variables and their uses, see Table 10.1 on page 407.

ANGLE Variable

Specifies the angle at which the graphics output is drawn.

Type: numeric

Default: function dependent

Syntax

ANGLE=0...360;

Functions

The ANGLE variable is function dependent.

If function is...	then the variable specifies...
LABEL	the baseline angle of the character string with respect to the horizontal. With the LABEL function, the default is ANGLE=0.
PIE	the starting angle of the slice arc. The default for the first PIE function is ANGLE=0; thereafter, the default is the last value that is calculated by ANGLE and ROTATE.
PIEXY	the angle of rotation when moving counterclockwise around the perimeter of a pie. With the PIEXY function, the default is ANGLE=0.

CBORDER Variable

Draws a colored border around text or symbols.

Type: character

Length: 8

See also: CBOX

Syntax

CBORDER='*color*' | 'CTEXT';

color

specifies the color of the border that surrounds the text or symbol. *Color* is any SAS/GRAPH color name. See Chapter 7, "SAS/GRAPH Colors," on page 139 for more information about specifying colors.

Specifying a null value for *color* (CBORDER=' ')cancels the CBORDER variable.

CTEXT

draws the border in the same coloras the text or symbol. The text color is determined by (1) the COLOR variable or (2) the CTEXT=graphics option or (3) the first color in the colors list.

Functions

You can use the CBORDER variable with these functions:

LABEL

SYMBOL

Details

Once you have specified CBORDER, it remains in effect for all subsequent observations that use the LABEL or SYMBOL function and draws a border around all text or symbols. To turn off the border for subsequent text or symbols, specify CBORDER=' '.

To fill the area defined by CBORDER, use the CBOX variable in conjunction with CBORDER.

CBOX Variable

Draws a solid box behind the text or symbol and fills the box with the specified color.

Type: character

Length: 8

See also: CBORDER

Syntax

CBOX='*color*' | 'CBACK';

color
specifies the color that fills the box. *Color* is any SAS/GRAPH color name. See Chapter 7, "SAS/GRAPH Colors," on page 139 for more information about specifying colors.

Specifying a null value for *color* (CBOX=' ')cancels the CBOX variable.

CBACK
fills the box with the same color as the background color of the graph. The background color is either (1) the color specified by the CBACK= graphics option or (2) the default background color for the device.

Functions

You can use the CBORDER variable with these functions:

LABEL

SYMBOL

Details

Once you have specified CBOX, it remains in effect for all subsequent observations that use the LABEL or SYMBOL function and adds a solid, colored box to all text or symbols. To turn off boxing for subsequent text or symbols, specify CBOX=' '.

The color of the text or symbol within the box is controlled by the COLOR variable.

By default, the solid box has no border. To add a colored border to the box, use the CBORDER variable in conjunction with CBOX.

COLOR Variable

Specifies the color used by the function.

Type: character

Length: 8

Default:

1 first color in colors list of the COLORS= graphics option

2 first color in device's default colors list.

Syntax

COLOR='*color*';

color
specifies any SAS/GRAPH color name. See Chapter 7, "SAS/GRAPH Colors," on page 139 for more information about specifying colors.

Functions

The COLOR variable is function dependent.

If function is...	then the variable specifies...
BAR	the color that outlines and, optionally, fills the bar if a pattern is specified in the STYLE variable. If no pattern is specified, *color* determines only the color of the outline.
DRAW or DRAW2TXT	the color that draws the line.
FRAME	the color for the frame. If a fill pattern is specified, *color* also determines the color of the interior or the area bounded by the frame.
LABEL	the color that draws the text.
PIE	the color for the pie slice if a pattern is specified with the STYLE variable. If no pattern is specified, *color* determines the color of the outline of the pie slice.
POINT	the color for the point.
POLY	the fill color for the interior of the polygon if a pattern is specified with the STYLE variable. If the STYLE variable is missing or EMPTY, *color* is ignored.
POLYCONT	the color that outlines the polygon when used with the first POLYCONT function. COLOR is ignored for subsequent POLYCONT functions in the POLYCONT sequence.
SYMBOL	the color that draws the symbol.

FUNCTION Variable

Specifies a graphics command or programming function for the Annotate facility to perform.

Type: character

Length: 8

Default: LABEL

Syntax

FUNCTION='*function-name*';

function-name

specifies the name of an Annotate function. Values for *function-name* are

BAR	draws and, optionally, fills a rectangle.
CNTL2TXT	copies (XLAST, YLAST) to (XLSTT, YLSTT), overwriting the previous values of (XLSTT, YLSTT).
COMMENT	places comments in your data set. The observation is ignored when the data set is processed.
DEBUG	writes the values of all Annotate variables to the SAS log before and after the next observation.
DRAW	draws a line in the graphics output.
DRAW2TXT	draws a line from (XLAST, YLAST) to (XLSTT, YLSTT).
FRAME	draws a border around the area defined by XSYS and YSYS. It also to specifies a background color for the framed area .
LABEL	draws text and is the default for the FUNCTION variable.
MOVE	moves to the specified point (does not draw a line).
PIE	draws a pie slice, arc, or circle that can be filled.
PIECNTR	sets new center and radius values. The PIEXY function can use this information in a later observation. PIECNTR does not draw a pie.
PIEXY	returns the coordinates of a point on a pie slice. Other functions can use this information in a later observation.
POINT	draws a point.
POLY	begins drawing a polygon (first vertex). Use the POLYCONT function in successive observations to supply the remaining vertices.
POLYCONT	continues drawing a polygon.
POP	gets values from the LIFO stack and changes the current value of (XLAST, YLAST) and (XLSTT, YLSTT) to those values.
PUSH	puts the current values for (XLAST, YLAST) and (XLSTT, YLSTT) in the LIFO stack.
SWAP	exchanges the values of (XLAST, YLAST) and (XLSTT, YLSTT).
SYMBOL	draws a symbol. See Figure 8.21 on page 243 for a list of the symbols.
TXT2CNTL	copies the values (XLSTT, YLSTT) to (XLAST, YLAST), overwriting the previous values of (XLAST, YLAST).

All other variables in the observation that contain the function act as parameters for the action. For a detailed description of each function and the Annotate variables that can be used in conjunction with it, see “Annotate Functions” on page 429.

GROUP Variable

Positions graphics elements on the bars of a vertical or horizontal bar chart drawn using the GROUP= option in the GCHART procedure.

Type: Numeric or character; must match the type of the GROUP= variable used in the GCHART procedure.

Length: Should match the length of GROUP= variable in the GCHART procedure.

Default: none

Restriction: Used only with vertical or horizontal bar charts produced by the GCHART procedure.

Syntax

GROUP=*group-value*;

group-value

references value(s) of the variable that is identified by the GROUP= option in the GCHART procedure either as a variable name or as an explicit data value. *Group-value* can be one of the following:

group-variable	the name of a group variable.
group-data-value	a specific numeric data value.
'*group-data-value*'	a specific character data value.

To annotate all the bars in a horizontal or vertical bar chart, specify a variable name. To annotate a bar chart for a specific value of the GROUP variable, specify a specific value.

Functions

You can use the GROUP variable only with the data coordinate systems 1, 2, 7, and 8, and with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

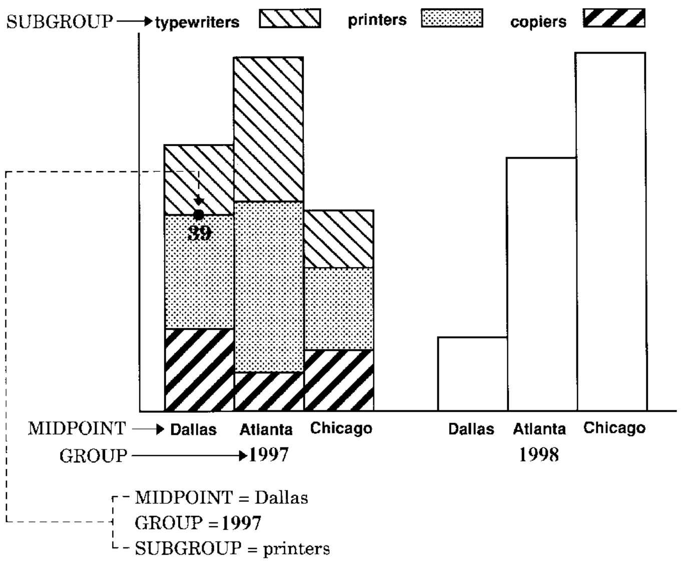
POLY

POLYCONT

SYMBOL

Details Using the GROUP variable is similar to using the X and Y variables with data system coordinates to position graphics elements in a vertical or horizontal bar chart. Figure 11.15 on page 461 shows how the GROUP variable works with the SUBGROUP and MIDPOINT variables to label the bars of a vertical bar chart.

Figure 11.15 Using the GROUP Variable to Position a Label in a Bar Chart



The label showing the number of units that were sold in Dallas in the year 1997 is positioned by the values that are assigned to these Annotate variables:

- GROUP=YEAR (where YEAR is a variable in the GCHART data set)
- MIDPOINT=CITY (where CITY is a variable in the GCHART data set)
- SUBGROUP=ITEM (where ITEM is a variable in the GCHART data set).

HSYS Variable

Defines the coordinate system and area of the output used by the SIZE variable to display the Annotate graphics.

Type: character
Length: 1
Default: 4

Syntax

HSYS='*coordinate-system*';

coordinate-system
specifies a value that represents a coordinate system. Values can be 1 through 9 and A through C as shown in the following table:

Absolute Systems	Relative Systems	Coordinate System Units
1	7	percentage of data area
2	8	data values
3	9	percentage of graphics output area

Absolute Systems	Relative Systems	Coordinate System Units
4	A	cell in graphics output area
5	B	percentage of procedure output area
6	C	cell in procedure output area

These values are also used by the XSYS and YSYS variables. See “Coordinate Systems” on page 411 for a description of the areas and coordinate systems.

Functions

You can use HSYS with these functions, all of which also use the SIZE variable:

DRAW

DRAW2TXT

FRAME

LABEL

PIE

PIECNTR

SYMBOL

Details The coordinate system that you specify with the HSYS variable affects how the function interprets the value of the SIZE variable. For example, if you use HSYS='3' and SIZE=10 with the DRAW function, the thickness of the line is 10 percent of the graphics output area. If you use HSYS='1' and SIZE=10 with DRAW, the thickness of the line is 10 percent of the data area.

HTML Variable

Defines a link in the HTML file created for a drill-down graph. This link is associated with an area of the graph and contains valid HTML syntax that can point to a report or another graph that you want to display when the user drills down on the area.

Type: character

Length: up to 1024

Default: none

Syntax

HTML='*link-string*';

link-string

specifies the text that defines the link for drill-down. For more information about drill-down graphs and how to specify the link string, see “About Drill-down Graphs” on page 90

Functions

You can use the HTML variable with these functions:

BAR

LABEL

PIE

POLY

SYMBOL

Details

Use a LENGTH statement to set the length of the HTML variable to the longest string you need for the link string. Be sure to set the HTML value to a null if you continue writing observations to the annotate data set after you are done assigning links. For example, the following code defines link information for two squares, but then sets the HTML variable to null when drawing a frame; otherwise the backgound area within the frame will use the link information from the last defined HTML value and become a hot zone in the graph.

```
data squares;
     length function style color $ 8
            html text $ 15;
     xsys='3'; ysys='3';

        /* draw a green square */
     color='green';
     function='move'; x=10; y=65; output;
     function='bar';  x=30; y=95; style='solid';
        html='href=green.gif'; output;

        /* draw a red square */
     color='red';
     function='move'; x=60; y=65; output;
     function='bar';  x=80; y=95;
        html='href=red.gif'; output;

        /* draw a blue frame */
     function='frame'; color='blue'; style='empty';
        /* set null link for background area in frame */
        html=''; output;
run;
```

LINE Variable

Controls the drawing of a line by determining either the type of line to draw or the relative position of the line.

Type: numeric

Default for all functions: 1

Syntax

LINE=*line-type*;

Functions

The LINE variable is function dependent.

If function is...	then the variable specifies...
BAR	the direction in which to adjust the outline of the bar. When used with the BAR function, values are 0 through 3. See Figure 11.2 on page 431 for an illustration of these values.
DRAW, DRAW2TXT, FRAME, or POLY	the type of line to draw. When used with these functions, values are 1 through 46. See Figure 8.22 on page 249 for an illustration of the line types.
PIE	the lines to include in the outline of the slice. When used with the PIE function, values are 0 through 3. See Figure 11.11 on page 443 for an illustration of these values.

MIDPOINT Variable

Positions graphics elements on the bars of a vertical or horizontal bar chart drawn by the GCHART procedure.

Type: Numeric or character; must match the type of the MIDPOINT= variable in the GCHART procedure.

Length: Should match the length of the MIDPOINT= variable in the GCHART procedure.

Default: none

Restriction: Used only with vertical or horizontal bar charts produced by the GCHART procedure.

Syntax

MIDPOINT=*midpoint-value*;

midpoint-value
references midpoint data value(s) in the GCHART procedure either as a variable name or as an explicit data value. *Midpoint-value* can have one of the following forms:

midpoint-variable	the name of a midpoint variable.
midpoint-data-value	a specific numeric data value.

'midpoint-data-value'	a specific character data value.

Generally, specify a variable name if you want to annotate all of the bars in a horizontal or vertical bar chart. To annotate a bar chart for a specific value of the MIDPOINT variable, specify a specific value.

Functions

You can use the MIDPOINT variable only with the data coordinate systems 1, 2, 7, and 8, and with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

POLYCONT

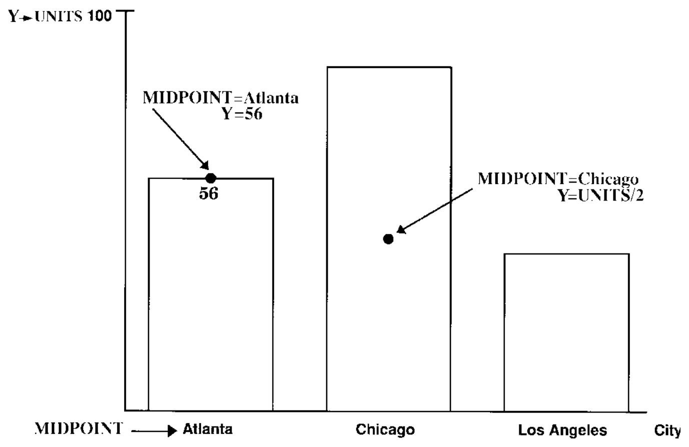
SYMBOL

Details Using the MIDPOINT variable is similar to using the X and Y variables to position graphics elements in a vertical or horizontal bar chart when using data system coordinates. For example, suppose you produce a vertical bar chart in which the chart variable CITY produces a bar for each city in a data set. The height of each bar is determined by the value of the SUMVAR= variable, UNITS.

You can label these bars by assigning the chart variable CITY to the Annotate MIDPOINT variable. The MIDPOINT variable provides the *x* coordinate for the label. By default, Annotate assigns the statistic variable, in this case the SUMVAR= variable, UNITS, to the Annotate Y variable, which provides the *y* coordinate for the label.

Figure 11.16 on page 465 shows how the values of the MIDPOINT and Y variables position the label that shows the number of units sold in Atlanta. The value, which is calculated and printed by the LABEL function, is 56.

Figure 11.16 Using the MIDPOINT Variable to Position a Label in a Bar Chart



The labels in this figure are positioned by the values that are assigned to these Annotate variables:

- □ MIDPOINT=CITY (where CITY is the chart variable); the MIDPOINT variable provides the horizontal coordinate in the vertical bar chart.
- □ Y=UNITS (where UNITS is the SUMVAR= variable); the Y variable provides the vertical coordinate. By specifying Y=units/2, you can vertically center the label in the bar.

Note: In a horizontal bar chart, the MIDPOINT variable controls the *y* coordinate and the statistic variable controls the *x* coordinate. △

CAUTION:

Be careful when using MIDPOINT and X and Y variables in the same data set. Using the MIDPOINT and X variables in an Annotate data set that is used to annotate a VBAR chart or the MIDPOINT and Y variables in the same data set used to annotate an HBAR chart can cause unexpected results. When annotating a VBAR chart, the Annotate facility uses the MIDPOINT variable as the horizontal coordinate if it exists in the Annotate data set and ignores the X variable. Consequently, you should use the MIDPOINT variable as the horizontal coordinate for all observations in an Annotate data set if you use it for one.

A similar behavior occurs if you use both the MIDPOINT and Y variables in an Annotate data set that is used to annotate HBAR charts. The MIDPOINT variable is always used, regardless of whether it has a missing value, and the Annotate facility ignores the Y variable. In this case, as well, use the MIDPOINT variable for the vertical coordinate for all observations in an Annotate data set if you use it for one. △

POSITION Variable

Controls placement and alignment of a text string specified by the LABEL function.

Type: character

Length: 1

Default: 5

Syntax

POSITION=’*text-position*’ | ’0’;

text-position

specifies the placement of the text string in relation to the position that is defined by the X and Y variables. *Text-position* can be one of the characters 1 through 9, A through F, <, +, or >. These characters represent the positions that are described in the following table:

Horizontal Position > Vertical Position V	Right Aligned	Centered	Left Aligned
One cell above	1	2	3
Centered	4 | <	5 | +	6 | >
One cell below	7	8	9
Half cell above	A	B	C
Half cell below	D	E	F

These positions are illustrated in Figure 11.18 on page 468.

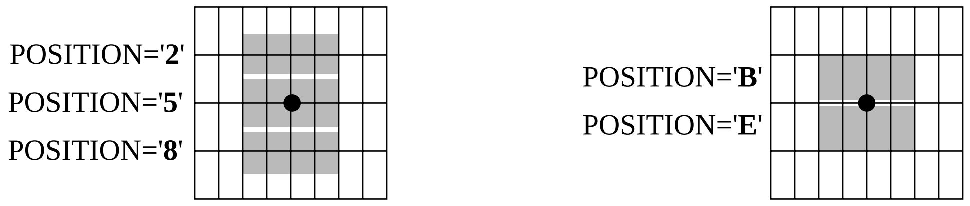
'0'

specifies a pause in the string in order to change an attribute, such as the color of the text.

Details

Stacking text strings To stack text strings, specify a different position value of each string. Figure 11.17 on page 467 shows two ways to stack text.

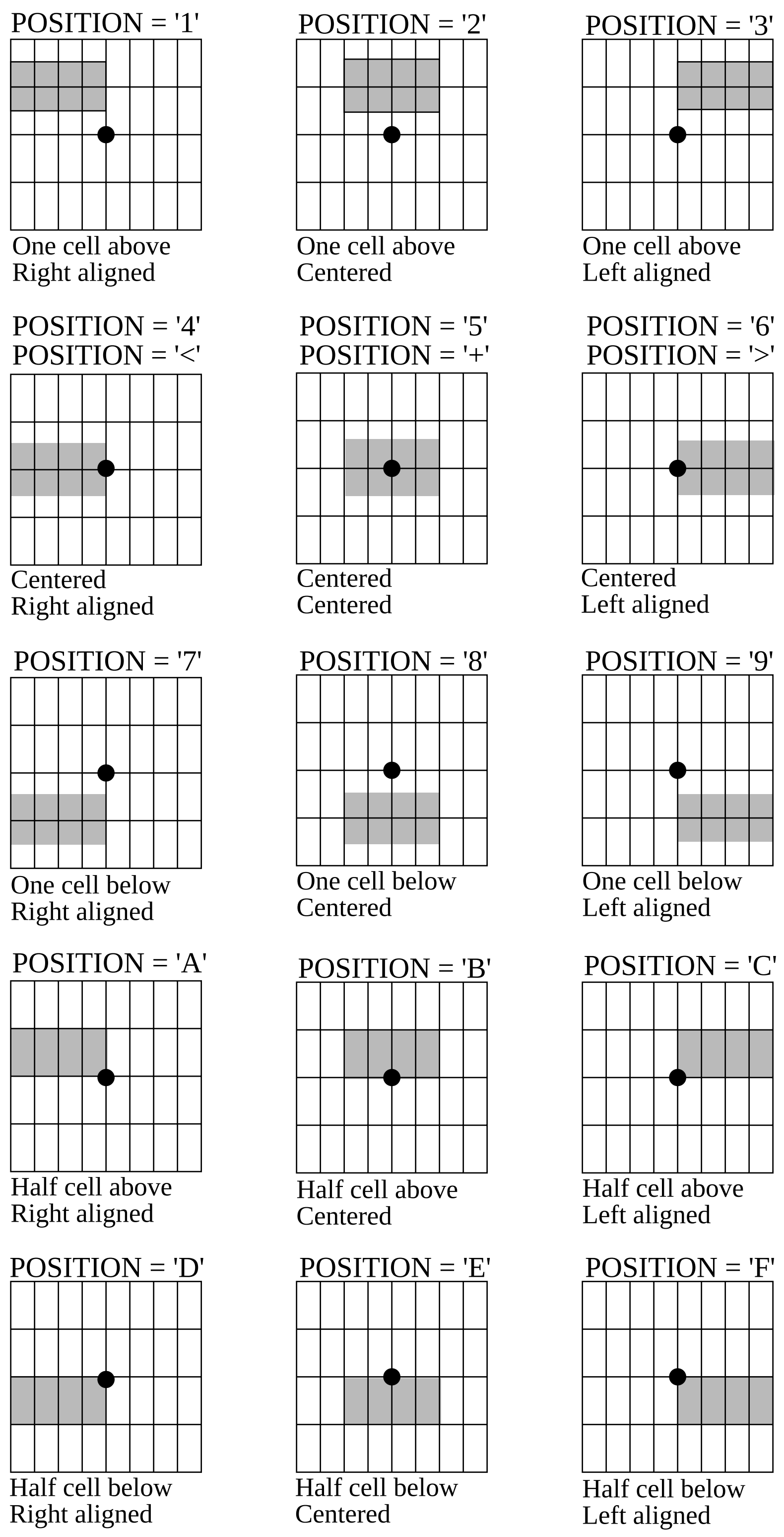
Figure 11.17 Combining POSITION Values to Stack Text



Positioning numeric labels The <, +, and > positions perform the same function as 4, 5, and 6, respectively, but are recommended only for labels that are numbers. The <, +, and > positions are especially useful when you are labeling a horizontal bar chart. You can use <, +, or > if the numbers in your font are significantly smaller than the text and you are having trouble centering labels. If the numbers in your font are the same height or close to the same height as the text, you can use positions 4, 5, and 6 to center the labels.

Note: You cannot stack <, +, and > positions as you can 4, 5, and 6 positions. △

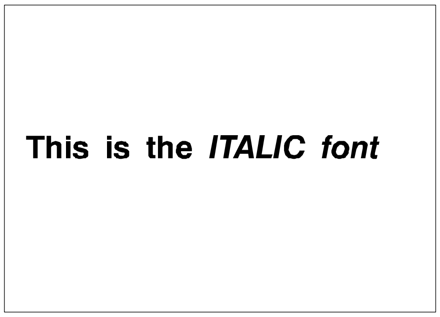
Figure 11.18 Effect of POSITION Values on Text Strings



Changing attributes in the middle of a text string 0 is a special value to use when you want to pause and then continue a text string. With this value you can change colors, fonts, and so on in the middle of a line, while retaining the exact position of the text at the pause. When POSITION='0', the combined text string is left-justified beginning at the point that is defined by the X and Y variables. However, you must define missing values for X for the continuation string. The following Annotate data set changes the font in the middle of the string. The result is shown in Figure 11.19 on page 469.

```
data anno;
   length style $ 8 text $ 12;
   xsys='3'; ysys='3'; hsys='3'; x=5; y=50;
      style='swissb'; size=10; text='This is the';
      position='0'; output;
   x=.; style='swissbi'; text=' ITALIC font';
       output;
run;
```

Figure 11.19 Using POSITION='0' to Change the Attributes of a Text String



ROTATE Variable

Specifies the angle at which to rotate the graphics element.

Type: numeric

Default: 0.00

Syntax

ROTATE=*rotation-angle*;

Functions

The ROTATE variable is function dependent.

If function is...	then the variable...
PIE	specifies the sweep of the generated arc that begins at the angle that is specified by the ANGLE variable that is used with the PIE function.
LABEL	rotates the individual text characters with respect to the baseline.

SIZE Variable

Determines the size of the graphics element with which it is used.

Type: numeric

Length: 8

Default: 1.00

Syntax

SIZE=*size-factor*;

Functions

The SIZE variable is function dependent.

If function is...	then the variable...
DRAW, DRAW2TXT, FRAME, POLY, or POLYCONT	determines the thickness of the line being drawn.
LABEL	specifies the height of the text.
PIE or PIECNTR	determines the radius of the pie.
PIEXY	sets the radius multiplier.
SYMBOL	selects the height of the symbol.

Details The SIZE variable uses the coordinate system that is specified by the HSYS variable. See “HSYS Variable” on page 461 for details.

STYLE Variable (Patterns)

Specifies a pattern for bars, pies, frames, and rectangles

Type: character

Length: 8

Default: EMPTY | PEMPTY | MEMPTY

Syntax

STYLE='*fill-pattern*';

fill-pattern

specifies a pattern to use with the graphics element. The value for *fill-pattern* is function dependent:

If function is...	then fill-pattern is...
BAR or FRAME	SOLID | EMPTY | *style<density>*
PIE	PSOLID | PEMPTY | P*density<style<angle>>*
POLY	MSOLID | MEMPTY | M*density<style<angle>>*

See the appropriate function for a complete description of *fill-pattern* values. See "PATTERN Statement" on page 211 for more information about patterns.

STYLE Variable (Fonts)

Specifies a font for text or symbols produced by the LABEL or SYMBOL functions.

Type: character

Length: Depends on specification.

Default: default hardware font

Syntax

STYLE='*font*' | "'*hardware-font-name*'" | 'NONE';

font

specifies a font. *Font* can be either the name of a software font that is stored in a catalog or a hardware font specification of the form HW*xxxnnn*. For example, STYLE='CENTB' specifies a software font that is stored in the catalog SASHELP.FONTS. The maximum length for *font* is 8 characters.

hardware-font-name

specifies the name of a hardware font as shown in the Chartype window of the device entry. The maximum length for *hardware-font-name* is 256 characters. *Hardware-font-name* must be enclosed in both double and single quotation marks, for example, STYLE="'Palatino-Italic'".

NONE

specifies the default hardware font.

See Chapter 6, "SAS/GRAPH Fonts," on page 125 for more information about specifying fonts.

If the value of the STYLE variable is missing, SAS/GRAPH software searches for a font specification in this order:

1. the font specified by the FTEXT= graphics option
2. the hardware font, if the device supports one
3. the SIMULATE font.

SUBGROUP Variable

Positions graphics elements within subgrouped bars of a vertical or horizontal bar chart produced by the GCHART procedure.

Type: Numeric or character; must match the type of the SUBGROUP variable used in the GCHART procedure.

Length: Should match the length of the SUBGROUP= variable in the GCHART procedure.

Default: none

Restriction: The bar charts must have been produced using the SUBGROUP= option.

Syntax

SUBGROUP=*subgroup-value*;

subgroup-value

references value(s) of the SUBGROUP= variable in the GCHART procedure either as a variable name or as an explicit data value. *Subgroup-value* can have one of the following forms:

subgroup-variable	the name of a subgroup variable.
subgroup-data-value	a specific numeric data value.
subgroup-data-value	a specific character data value.

Generally, specify a variable name if you want to annotate all of the bars in a horizontal or vertical bar chart. To annotate a bar chart for a specific value of the SUBGROUP variable, specify a specific value.

Functions

You can use the SUBGROUP variable only with the data coordinate system 1, 2, 7, or 8, and with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

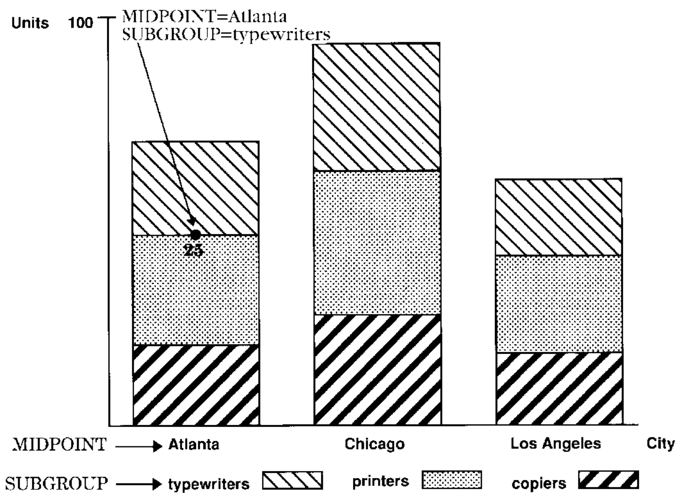
POLYCONT

SYMBOL

Details Using the SUBGROUP variable is similar to using the X and Y variables with data system coordinates to position the graphics elements in subgroup segments in vertical and horizontal bar charts.For example, in a vertical bar chart that produces a bar for each city in a data set, you can easily label the subgroups in each bar by setting *subgroup-variable* to the GCHART variable by which the bar is being subgrouped. This variable provides the *y* coordinate of the label.

The MIDPOINT variable works well with the SUBGROUP variable to provide the *x* coordinate. In this example, if you set the MIDPOINT variable to the GCHART variable that contains the names of the cities, the MIDPOINT variable provides your *x* coordinate. Rather than providing the X and Y variables, you would use the SUBGROUP and MIDPOINT variables. Figure 11.20 on page 473 shows how the SUBGROUP variable works with the MIDPOINT variable to label the bars of a vertical bar chart.

Figure 11.20 Using the SUBGROUP Variable to Position a Label in a Bar Chart



The label showing the number of printers sold in Atlanta is positioned by the values that are assigned to these Annotate variables:

- □ MIDPOINT=CITY (where CITY is a variable in the GCHART data set)
- □ SUBGROUP=ITEM (where ITEM is a variable in the GCHART data set).

TEXT Variable

Specifies the text or symbol to be placed on the graphics output.

Type: character

Length: up to 200

Default: blank string

Syntax

TEXT='*text-string*' | '*special-symbol*';

text-string
specifies the text that is used as a label (LABEL or COMMENT function) or symbol (SYMBOL function). The maximum length for *text-string* is 200 characters.

special-symbol
specifies the name of a symbol from the special symbol table that is illustrated in Figure 8.21 on page 243. The maximum length for *special-symbol* is eight characters.

Functions

You can use the TEXT variable with these functions:

COMMENT

LABEL

SYMBOL

Details Use a LENGTH statement to set the length of the TEXT variable if the length of a text string is longer than one character.

WHEN Variable

Specifies when the function is performed in relation to generating other graphics output for the procedure or in relation to generating other Annotate graphics.

Type: character

Length: 1

Default: B

Syntax

WHEN='B' | 'A';

B | A
specifies whether to draw the annotation before (B) or after (A) the graph. These values are not case sensitive. A missing value is equivalent to specifying B.

Functions

You can use the WHEN variable with these functions:

BAR

DRAW

DRAW2TXT

FRAME

LABEL

MOVE

PIE

PIECNTR

PIEXY

POINT

POLY

POLYCONT

SYMBOL

Details Normally, observations in an Annotate data set are processed sequentially. If you use the WHEN variable, all those observations with a WHEN value of B are processed first, the procedure output is then processed (if one is to be produced), and finally the observations with a WHEN value of A are processed.

X Variable

Identifies the *x* coordinate of where a graphics element is to be drawn.

Type: numeric

Default: value of XLAST or XLSTT

Syntax

X=*horizontal-coordinate*;

Functions

You can use the X variable with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

POLYCONT

SYMBOL

Note: The X or XC variable is required unless either the MIDPOINT, GROUP, or SUBGROUP variable provides the horizontal coordinate. △

Details Specify a corresponding vertical coordinate when using the X variable. This vertical coordinate can be specified with the Y, YC, MIDPOINT, or SUBGROUP variables, depending on the type of graph that you are annotating.

The X variable uses the units that are specified in the XSYS variable. If you use XSYS='2' and the data axis is typed as character, use the XC variable instead of the X variable.

If the value of the X variable is missing for a function that requires it, the value of the XLAST variable is used with nontext functions and the value of the XLSTT variable is used with text functions.

XC Variable

Identifies the *x* coordinate of a graphics element when the coordinate value is character.

Type: character

Length: Should match that of the plot variable in the procedure.

Default: the value of XLAST or XLSTT

Restrictions: Used only with output from the GCHART and GPLOT procedures. Ignored if the axes are numeric.

Syntax

XC='*character-type-horizontal-coordinate*';

Functions

You can use the XC variable with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

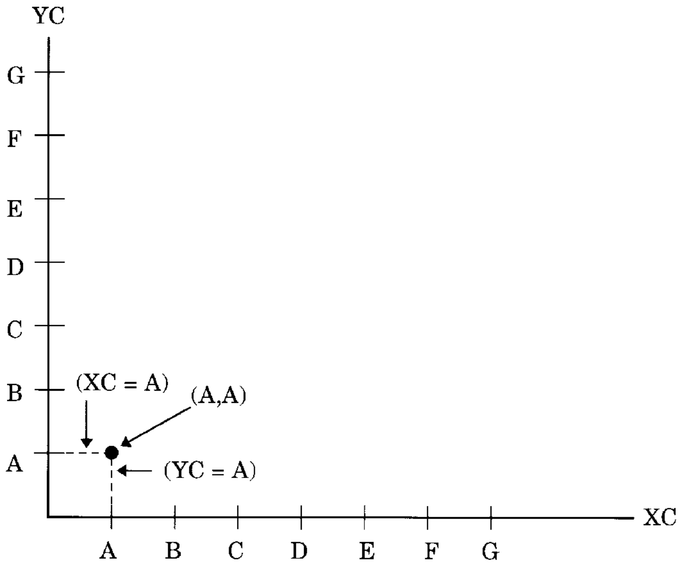
POLYCONT

SYMBOL

Details The XC variable is the character equivalent of the X variable. Use XC when the axis values are character. You must also specify a value of 2 (absolute data values) for the XSYS variable. (See also “XSYS Variable” on page 477.) If you use a value other than 2 for the XSYS variable, the graphics output is not displayed properly.

Figure 11.21 on page 477 illustrates the XC variable.

Figure 11.21 Using the XC and YC Variables with Character Data



Note: The X or XC variable is required unless either the MIDPOINT, GROUP, or SUBGROUP variable provides the horizontal coordinate. △

CAUTION:

Do not use the X and XC variables in the same data set. Using both X and XC variables in the same data set can cause unpredictable results. △

XSYS Variable

Defines the coordinate system and area of the output used by the X and XC variables to display the Annotate graphics.

Type: character

Length: 1

Default: 4

Syntax

XSYS=’*coordinate-system*’;

coordinate-system
specifies a value that represents a coordinate system. Values can be 1 through 9 and A through C as shown in the following table:

Absolute Systems	Relative Systems	Coordinate System Units
1	7	percentage of data area
2	8	data values
3	9	percentage of graphics output area
4	A	cell in graphics output area
5	B	percentage of procedure output area
6	C	cell in procedure output area

These values are also used by the HSYS and YSYS variables. See “Coordinate Systems” on page 411 for a description of the areas and coordinate systems.

Functions

You can use the XSYS variable with these functions:

BAR

DRAW

FRAME

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

POLYCONT

SYMBOL

Details The coordinate system that you specify with the XSYS variable affects how the function interprets the value of the X or XC variable.

Note: Not all coordinate systems can be used with all Annotate variables. For any restrictions, see the individual variables in this section. △

Y Variable

Identifies the *y* coordinate of where a graphics element is to be drawn.

Type: numeric

Default: value of YLAST or YLSTT

Syntax

Y=*vertical-coordinate*;

Functions

You can use the Y variable with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

POLYCONT

SYMBOL

Note: The Y or YC variable is required unless either the MIDPOINT, GROUP, or SUBGROUP variable provides the vertical coordinate. △

Details Specify a corresponding horizontal coordinate when using the Y variable. You can specify the horizontal coordinate with the X, XC, MIDPOINT, or SUBGROUP variable, depending on the type of graph you are annotating.

The Y variable uses the units specified in the YSYS variable. If you use YSYS='2' and the axis data is type character, use the YC variable instead of the Y variable.

If the value of the Y variable is missing for a function that requires it, the value YLAST is used for nontext functions and the value of YLSTT is used for text functions.

YC Variable

Identifies the *y* coordinate of a graphics element when the coordinate value is character.

Type: character

Length: Should match that of the plot variable in the procedure.

Default: YLAST | YLSTT

Restrictions: Used only with output from the GCHART and GPLOT procedures. Ignored if the axes are numeric.

Syntax

YC='*character-type-vertical-coordinate*';

Functions

You can use the YC variable with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

POLYCONT

SYMBOL

Details The YC variable is the character equivalent of the Y variable. Use YC when the axis values are character. You must also specify a value of 2 (absolute data values) for the YSYS variable. (See “YSYS Variable” on page 480.) If you use a value other than 2 for the YSYS variable, the graphics output is not displayed properly.

See Figure 11.21 on page 477 for an illustration of the YC variable.

Note: The X or XC variable is required unless either the MIDPOINT, GROUP, or SUBGROUP variable provides the horizontal coordinate. △

CAUTION:

Do not use Y and YC variables in the same data set. Using both Y and YC variables in the same data set can cause unpredictable results. △

YSYS Variable

Defines the coordinate system and area of the output used by Y and YC to display the Annotate graphics.

Type: character

Length: 1

Default: 4

Syntax

YSYS=’*coordinate-system*’;

coordinate-system

specifies a value that represents a coordinate system. Values can be 1 through 9 and A through C, as shown in the following table:

Absolute Systems	Relative Systems	Coordinate System Units
1	7	percentage of data area
2	8	data values
3	9	percentage of graphics output area
4	A	cell in graphics output area
5	B	percentage of procedure output area
6	C	cell in procedure output area

These values are also used by the HSYS and XSYS variables. See "Coordinate Systems" on page 411 for a description of the areas and coordinate systems.

Functions

You can use the YSYS variable with these functions:

BAR

DRAW

FRAME

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

POLYCONT

SYMBOL

Details The coordinate system that you specify with the YSYS variable affects how the function interprets the value of the Y or YC variable.

Note: Not all coordinate systems can be used with all Annotate variables. For any restrictions, see the individual variables in this section. △

Z Variable

Identifies the *z* coordinate of where a graphics element is to be drawn.

Type: numeric

Length: 8

Default: none

Restrictions: Used only with output from the G3D procedure.

Syntax

Z=*depth-coordinate*;

Functions

You can use the Z variable with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

POLYCONT

SYMBOL

Details The Z variable uses the units that are specified in the ZSYS variable.

ZSYS Variable

Defines the coordinate system and area of the output used by Z variable to display the Annotate graphics.

Type: character

Length: 1

Default: 2

Syntax

ZSYS='*coordinate-system*';

coordinate-system
specifies a value that represents a coordinate system. Values can be 1, 2, 7, or 8 as shown in the following table:

Absolute Systems	Relative Systems	Coordinate System Units
1	7	percentage of data area
2	8	data values

See “Coordinate Systems” on page 411 for a description of the areas and coordinate systems.

Functions

You can use the ZSYS variable with these functions:

BAR

DRAW

LABEL

MOVE

PIE

PIECNTR

POINT

POLY

POLYCONT

SYMBOL

Details The coordinate system that you specify with the ZSYS variable affects how the function interprets the value of the Z variable.

Note: Not all coordinate systems can be used with all Annotate variables. For any restrictions, see the individual variables in this section. △

Annotate Internal Coordinates

The Annotate facility maintains two sets of internal coordinates that are stored in the variable pairs (XLAST, YLAST) and (XLSTT, YLSTT). One set of variables (XLAST, YLAST) stores coordinate values that are generated by nontext functions and the other set (XLSTT, YLSTT) stores coordinates generated by text functions. These two variable pairs supply default values when the X or Y variable contains a missing value.

Both pairs are initially set to 0 and remain 0 until a function updates the values. You cannot assign explicit values to these variables, but you can manipulate their values with some of the Annotate functions.

XLAST, YLAST Variables

Track the last values specified for the X and Y variables when X and Y are used with nontext functions.

Details

The coordinate values that are stored in the (XLAST, YLAST) variables are automatically updated by these nontext functions: BAR, DRAW, MOVE, PIE, and POINT. These values are then available for use by other nontext functions that follow in the DATA step. (The DRAW2TXT graphics function uses XLAST and YLAST but does not update them.)

Because (XLAST, YLAST) are updated internally, you cannot specify values for them. However, their values can be manipulated by these programming functions:

CNTL2TXT

PIECNTR

PIEXY

POP

PUSH

SWAP

TXT2CNTL

XLSTT, YLSTT Variables

Track the last values specified for the X and Y variables when X and Y are used with text-handling functions.

Details

The coordinate values stored in the (XLSTT, YLSTT) variables are automatically updated by the LABEL and SYMBOL text functions. These values are then available for use by other text functions that follow in the DATA step.

Because (XLSTT, YLSTT) are updated internally, you cannot specify values for them. However, their values can be manipulated by these programming functions:

CNTL2TXT

DRAW2TXT

POP

PUSH

SWAP

TXT2CNTL

Annotate Macros

You can use Annotate macros within a SAS DATA step to simplify the process of creating Annotate observations. With a macro, you specify a function and assign variable values in one step without having to write explicit variable assignment statements. You can mix assignment statements and macro calls in the same DATA step.

This section describes all of the Annotate macros including the complete syntax and a description of the parameters. For more information on accessing and using macros, and for a summary of operations performed by the Annotate macros, see “Using Annotate Macros” on page 500.

%ANNOMAC

Compiles Annotate macros and makes them available for use.

Variables affected: none directly

Syntax

%ANNOMAC;

Details In a SAS session, you must submit the ANNOMAC macro before you can use the Annotate macros.

%BAR, %BAR2

Draws a fillable rectangle with opposing corners at (*x*1,*y*1) and (*x*2,*y*2).

Variables affected: COLOR, FUNCTION, LINE, STYLE, X, Y

Updates: XLAST, YLAST

Syntax

%BAR (*x*1, *y*1, *x*2, *y*2, *color, line, style*);

%BAR2(*x*1, *y*1, *x*2, *y*2, *color, line, style, width*);

Parameter	*Description*
*x*1, *y*1	numeric constants or numeric variable names, the *x* and *y* coordinates of the lower-left corner of the bar.
*x*2, *y*2	numeric constants or numeric variable names, the *x* and *y* coordinates of the upper-right corner of the bar.
color	literal; the color of the bar. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. Use an asterisk (*) to indicate the previously defined value for *color*.

line	numeric constant or numeric variable name; the direction in which to adjust the outline of the bar. Use line values 1 and 2 to offset a particular bar from an axis or adjoining area. Values for *line* are

0	draw all edges.
1	vertical adjust.
2	horizontal adjust.
3	draw no edges.

See Figure 11.2 on page 431 for representations of LINE values.

style	literal; the pattern that fills the bar. Values for *style* are the following bar or block patterns:

SOLID S	a solid fill.
EMPTY E	an empty fill.
style<density>	a shaded pattern: *style* can be R | X | L *density* can be 1...5.

See also the discussion of fill patterns for bars and blocks in VALUE= on page 214. *Note:* All parameters must be expressed as values, not variable names.

width	sets the size variable and controls the line thickness of the outline around the bar or polygon.

%CIRCLE

Draws an empty circle with the center at (*x*, *y*).

Variables affected: ANGLE, FUNCTION, ROTATE, SIZE, STYLE, X, Y

Updates: XLAST, YLAST

Syntax

%CIRCLE (*x, y, size, color*);

Parameter	*Description*
x,y	numeric constant or numeric variable name; the *x* and *y* coordinates of the center of the circle.
size	numeric constant or numeric variable name; the radius of the circle, using the units that are specified by the HSYS variable.
color	literal; the color of the outline of the circle. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation

marks. Use an asterisk (*) to indicate that the previously defined value for *color* should be used.

See Also

"%SLICE" on page 497 for drawing a filled circle.

%CNTL2TXT

Copies the values of the internal coordinates (XLAST, YLAST) to the text coordinate (XLSTT, YLSTT).

Variable affected: FUNCTION

Updates: XLSTT, YLSTT

Syntax

%CNTL2TXT;

Details %CNTL2TXT is useful when you are calculating the position of labels on a graph. See "CNTL2TXT Function" on page 432 for an example.

See Also

"%SLICE" on page 497 for drawing a filled circle.

%COMMENT

Inserts a comment into an Annotate data set.

Variables affected: FUNCTION, TEXT

Syntax

%COMMENT (*text-string*);

Parameter	*Description*
text-string	specifies the text that you want inserted into the Annotate data set; *text-string* can be a character string enclosed in quotation marks or character variable name.

%DCLANNO

Automatically sets the correct length and data type for all Annotate variables except the TEXT variable.

Variables affected: All except TEXT

Syntax

%DCLANNO;

%DRAW

Draws a line to the (*x*, *y*) coordinate.

Variables affected: COLOR, FUNCTION, LINE, SIZE, X, Y

Updates: XLAST, YLAST

Syntax

%DRAW (x, y, *color*, *line*, *size*);

Parameter	*Description*
x,y	numeric constant or numeric variable name; the *x* and *y* coordinates of the ending point of the line that you want to draw.
color	literal; the color of the line. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. Use an asterisk (*) to indicate the previously defined value for *color*.
line	numeric constant or numeric variable name; the type of line to draw. Values for *line* are integers from 1 to 46. See "Specifying Line Types" on page 248 for representations of line types.
size	numeric constant or numeric variable name; the width of the line to draw, using the units specified by the HSYS variable.

Details The point from which the line is drawn is usually set with the MOVE macro.

%DRAW2TXT

Draws a line from the coordinate (XLAST, YLAST) to the text coordinate (XLSTT, YLSTT).

Variables affected: COLOR, FUNCTION, LINE, SIZE

Syntax

%DRAW2TXT (*color*, *line*, *size*);

Parameter	*Description*
color	literal; the color of the line. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. Use an asterisk (*) to indicate the previously defined value for *color*.
line	numeric constant or numeric variable name; the type of line to draw. Values for *line* are integers from 1 to 46. See “Specifying Line Types” on page 248 for representations of line types.
size	numeric constant or numeric variable name; the width of the line to draw.

%FRAME

Draws a border around the portion of the display area defined by the reference system. Optionally fills the area.

Variables affected: COLOR, FUNCTION, LINE, SIZE, STYLE

Syntax

%FRAME (*color*, *line*, *size*, *style*);

Parameter	*Description*
color	literal; the color of the line. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. Use an asterisk (*) to indicate the previously defined value for *color*.
line	numeric constant or numeric variable name; the type of line to draw. Values for *line* are integers from 1 to 46. See “Specifying Line Types” on page 248 for representations of line types.
size	numeric constant or numeric variable name; the width of the line.
style	literal; the pattern that fills the area bounded by the frame. Values for *style* are the following bar or block patterns:

SOLID S	a solid fill.
EMPTY E	an empty fill.
style<*density*>	a shaded pattern: *style* can be R | X | L *density* can be 1...5.

See also the discussion of fill patterns for bars and blocks in VALUE= on page 214.

Details See “%SYSTEM” on page 499 for information on setting the reference system.

%LABEL

Places a text label at the (*x*, *y*) position.

Variables affected: ANGLE, COLOR, FUNCTION, POSITION, ROTATE, SIZE, STYLE, TEXT, X, Y

Updates: XLSTT, YLSTT

Syntax

%LABEL (*x*, *y*, *text-string*, *color*, *angle*, *rotate*, *size*, *style*, *position*);

Parameter	*Description*
x,y	numeric constant or numeric variable name; the *x* and *y* coordinates of the location of the label in the graphics output.
text-string	character string enclosed in quotation marks or character variable name; the text that you want to display.
color	literal; the color of the line. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. Use an asterisk (*) to indicate the previously defined value for *color*.
angle	numeric constant or numeric variable name; the angle of the character baseline with respect to the horizontal. The pivot point is at (*x*, *y*).
rotate	numeric constant or numeric variable name; the rotation angle of each character in the string. It is equivalent to the ROTATE= option in the FOOTNOTE, NOTE, and TITLE statements.
size	numeric constant or numeric variable name; the height of the text string being drawn. The height units are based on the value of the HSYS variable. See “HSYS Variable” on page 461 for an explanation of the values used with this variable.
style	literal; the character font that draws the text. Do not enclose *style* in quotation marks unless you are specifying a hardware font description. See Chapter 6, “SAS/GRAPH Fonts,” on page 125 for more information about specifying fonts.
position	literal; the placement and alignment of the text string. *Position* can take values from 0 to F (hexadecimal numbering) and <, +, and > and must not be enclosed in quotation marks. Invalid or missing values default to 5. For an explanation of the values of *position*, see “POSITION Variable” on page 466.

%LINE

Draws a line from (*x*1, *y*1) to (*x*2, *y*2) coordinates.

Variables affected: COLOR, FUNCTION, LINE, SIZE, X, Y

Updates: XLAST, YLAST

Syntax

%LINE (*x*1, *y*1, *x*2, *y*2, *color*, *line*, *size*);

Parameter	*Description*
*x*1, *y*1	numeric constants or numeric variable names; the *x* and *y* coordinates of the beginning point of the line.
*x*2, *y*2	numeric constants or numeric variable names; the *x* and *y* coordinates of the ending point of the line.
color	literal; the color of the bar. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. Use an asterisk (*) to indicate the previously defined value for *color*.
line	numeric constant or numeric variable name; the type of line to draw. Values for *line* are integers from 1 to 46. See “Specifying Line Types” on page 248 for representations of line types.
size	numeric constant or numeric variable name; the width of the line. The HSYS variable controls the units that *size* uses. See also “HSYS Variable” on page 461.

%MOVE

Moves to the (*x*, *y*) coordinate.

Variables affected: FUNCTION, X, Y

Updates: XLAST, YLAST

Syntax

%MOVE (*x*, *y*);

Parameter	*Description*
x	numeric constant or numeric variable name; the horizontal coordinate of the position to move to.
y	numeric constant or numeric variable name; the vertical coordinate of the position to move to.

%PIEXY

Calculates a point on the outline of the slice arc.

Variables affected: ANGLE, FUNCTION, SIZE, X, Y

Syntax

%PIEXY (*angle, size*);

Parameter	*Description*
angle	numeric constant or numeric variable name; the angle of rotation when moving around the perimeter of a pie clockwise. The default is 0.0 (horizontal).
size	numeric constant or numeric variable name; the radius multiplier. If you specify 1.5, the Annotate facility calculates 1.5 times the radius that you specified with the *size* parameter in the SLICE macro.

Details The calculated coordinates are placed in the internal coordinate pair (XLAST, YLAST). This macro is useful when you want to position text around a pie (circle).

When you use this macro, the Annotate facility expects a slice to have been previously drawn. If a slice has not been drawn or FUNCTION='PIECNTR' has not been processed, you can get erroneous results.

See Also

"PIEXY Function" on page 445

%POLY, %POLY2

Begins drawing a polygon at (*x*, *y*). It also determines the color, fill pattern, and type of line that draws the polygon.

Variables affected: FUNCTION, X, Y, COLOR, LINE, STYLE

Syntax

%POLY (x, y, *color*, *style*, *line*);

%POLY2(*x*, *y*, *color*, *style*, *line*, *width*);

Parameter	*Description*
x, *y*	numeric constant or numeric variable name; the *x* and *y* coordinates of the beginning point of the polygon.
color	literal; the color of the interior of the polygon unless *style* is EMPTY. *Color* can be any SAS/GRAPH color name. Use an asterisk (*) to indicate the previously defined value for *color*.
style	literal; specifies the pattern that fills the polygon. Values for *style* are the following map patterns:

MSOLID MS	a solid pattern.

MEMPTY ME	an empty pattern.
M*density<style<angle>>*	a shaded pattern: *density* can be 1...5 *style* can be X | N *angle* can be 0...360.

See also the discussion of the fill patterns for maps in VALUE= on page 215.

line	numeric constant or numeric variable name; the type of line to draw. Values are 1 through 46. See “Specifying Line Types” on page 248 for representations of the line types.
width	sets the size variable and controls the line thickness of the outline around the bar or polygon.

See Also

“POLYCONT Function” on page 449

%POLYCONT

Continues drawing the polygon to the point (*x*, *y*).

Variables affected: COLOR, FUNCTION, X, Y

Syntax

%POLYCONT (*x*, *y*, *color*);

Parameter	*Description*
x, *y*	numeric constants or numeric variable names; the *x* and *y* coordinates of the ending point of the line.
color	literal; the color of the outline of the polygon when you specify color in the first POLYCONT macro. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. *Color* is ignored in subsequent POLYCONT macros in the POLYCONT sequence.

Details The first POLYCONT macro in the POLYCONT sequence determines the outline color of the polygon being defined.

%POP

Removes the coordinates (XLAST, YLAST) and (XLSTT, YLSTT) from the LIFO system stack and updates the internal coordinate pairs with these retrieved values.

Variable affected: FUNCTION

Updates: XLAST, YLAST, XLSTT, YLSTT

Syntax

%POP;

Details Use the POP macro when you want to access the values of (XLAST, YLAST) and (XLSTT, YLSTT) that you previously stored with the PUSH function or macro.

%PUSH

Enters the coordinates (XLAST, YLAST) and (XLSTT, YLSTT) in a LIFO system stack.

Variables affected: FUNCTION, internal coordinates

Syntax

%PUSH;

Details The LIFO stack is a way to save previously calculated coordinates. It enables you to retain coordinate values for later use by utility functions without recalculating those values. In order to save coordinate values in the stack, you must explicitly push them onto the stack. See Chapter 10, "The Annotate Data Set," on page 403 for a description of the LIFO stack.

%RECT

Draws a nonfillable rectangle with opposite corners at (*x*1, *y*1) and (*x*2, *y*2).

Variables affected: COLOR, FUNCTION, LINE, SIZE, X, Y

Updates: XLAST, YLAST

Syntax

%RECT (*x*1, *y*1, *x*2, *y*2 , *color*, *line*, *size*) ;

Parameter	*Description*
*x*1, *y*1	numeric constant or numeric variable name; the *x* and *y* coordinates of the lower-left corner of the rectangle.

*x*2, *y*2	numeric constant or numeric variable name; the *x* and *y* coordinates of the upper-right corner of the rectangle.
color	literal; the color of the outline of the rectangle. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. Use an asterisk (*) to indicate the previously defined value for *color*.
line	numeric constant or numeric variable name; the type of line to draw. Values for *line* are integers from 1 to 46. See "Specifying Line Types" on page 248 for representations of line types.
size	numeric constant or numeric variable name; the width of the line. The HSYS variable controls the units that *size* uses. See also "HSYS Variable" on page 461.

Details The BAR macro produces filled rectangles.

%SCALE

Scales input coordinates (*ptx*, *pty*) from a range of the minima (*x*1, *y*1) and maxima (*x*2, *y*2) to a range of minima (*vx*1, *vy*1) and maxima (*vx*2, *vy*2).

Variables affected: X, Y

Syntax

%SCALE (*ptx*, *pty*, *x*1, *y*1, *x*2, *y*2 , *vx*1, *vy*1, *vx*2, *vy*2);

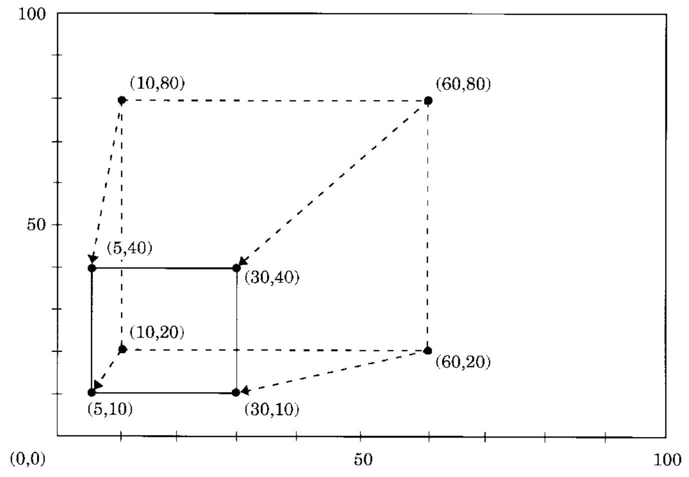
Parameter	*Description*
ptx, *pty*	numeric variables; the coordinates to scale.
*x*1, *y*1	numeric constants; the minima of the original range.
*x*1, *y*2	numeric constants; the maxima of the original range.
*vx*1, *vy*1	numeric constants; the minima of the new range.
*vx*2, *vy*2	numeric constants; are the maxima of the new range.

Details The SCALE macro reduces or enlarges Annotate graphics. It scales only graphics elements that use two-dimensional, numeric coordinates. The SCALE macro does not affect graphics elements that are drawn with text functions.

The difference between the SCALE and SCALET macros is that the SCALE macro always places the origin at (0, 0) and plots the new coordinates with respect to that origin. The SCALET macro plots the new coordinates with respect to (*vx*1, *vy*1). This SCALE macro reduces *x* and *y* coordinates by 50 percent as shown in Figure 11.22 on page 496:

```
%SCALE(x, y, 0, 0, 100, 100, 0, 0, 50, 50);
```

Figure 11.22 Using the SCALE Macro to Reduce the Size of a Box



%SCALET

Scales input coordinates (*ptx*, *pty*) from a range of the minima (*x*1, *y*1) and maxima (*x*2, *y*2) to a range of minima (*vx*1, *vy*1) and maxima (*vx*2, *vy*2).

Variables affected: X, Y

Syntax

%SCALET (*ptx*, *pty*, *x*1, *y*1, *x*2, *y*2 , *vx*1, *vy*1, *vx*2, *vy*2);

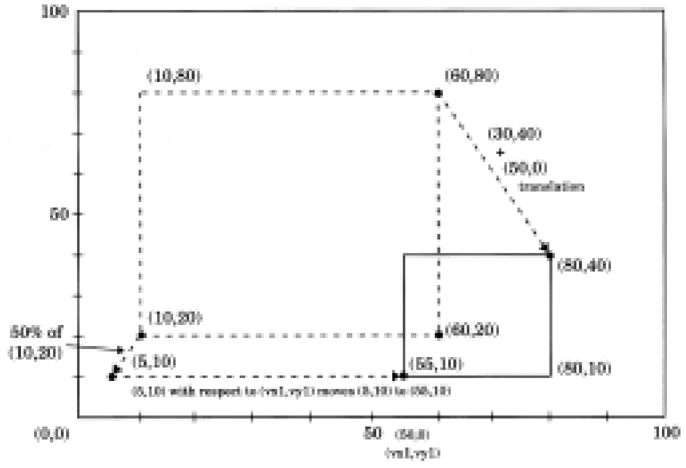
Parameter	*Description*
ptx, *pty*	numeric variables; the coordinates to scale.
*x*1, *y*1	numeric constants; the minima of the original range.
*x*1, *y*2	numeric constants; the maxima of the original range.
*vx*1, *vy*1	numeric constants; the minima of the new range; the origin from which to plot the new coordinates.
*vx*2, *vy*2	numeric constants; the maxima of the new range.

Details The SCALET macro reduces or enlarges Annotate graphics. This macro can only scale graphics elements that use two-dimensional, numeric coordinates. The SCALET macro does not affect graphics elements that are drawn with text functions.

The difference between the SCALET and SCALE macros is that the SCALET macro plots the new coordinates with respect to (*vx*1, *vy*1). The SCALE macro always places the origin at (0, 0) and plots the new coordinates with respect to that origin. This SCALET macro reduces *x* and *y* coordinates by 50 percent and plots the new coordinates with respect to (50, 0), as shown in Figure 11.23 on page 497:

```
%SCALET(x, y, 0, 0, 100, 100, 50, 0, 50, 100);
```

Figure 11.23 Using the SCALET Macro to reduce the Size of a Box



%SEQUENCE

Specifies when to draw Annotate graphics elements, relative to the procedure's graphics output or relative to the other Annotate graphics drawn.

Variable affected: WHEN

Syntax

%SEQUENCE (*when*);

The value of *when* is not case sensitive, must be one of these values, and must not be enclosed in quotation marks:

AFTER	literal; specifies that subsequent Annotate graphics will overlay procedure graphics. A is an alias for AFTER.
BEFORE	literal; specifies that subsequent Annotate graphics will be overlaid by procedure graphics. This value is the default. B is an alias for BEFORE.

%SLICE

Draws an arc or section of a fillable circle with the center at (*x*1,*y*1).

Variables affected: ANGLE, COLOR, FUNCTION, LINE, ROTATE, SIZE, STYLE, X, Y

Updates: XLAST, YLAST

Syntax

%SLICE (x1, y1, *angle*, *rotate*, *size*, *color*, *style*, *line*);

Parameter	*Description*
*x*1,*y*1	numeric constants or numeric variable names; the *x* and *y* coordinates of the center of the pie in the graphics output.
angle	numeric constant or numeric variable name; the starting angle of the slice arc. The default is 0.00 (at the three o'clock position) if *angle* is not specified for the first slice. After the first slice, the default is the ending angle of the slice arc that was just drawn. Therefore, you can specify an entire pie more easily by omitting the start and end calculations otherwise required. If you want the next slice to start at an angle that is different from the ending angle of the previous slice, you must specify a value for *angle*. The pivot point for *angle* is at (*x*, *y*).
rotate	numeric constant or numeric variable name; the angle of rotation of the slice arc that begins at the angle that is specified by *angle*. The default is 0.00. For an example of using rotate, see "PIE Function" on page 442.
size	numeric constant or numeric variable name; the radius of the circle. The units of radius are determined by the HSYS variable. See "%SYSTEM" on page 499 for information on setting the value of the HSYS variable when you use Annotate macros.
color	literal; the color of the slice, either the fill or outline. If *style* is EMPTY, *color* only determines the color of the outline. If *style* is a pattern, *color* determines the color of the fill. *Color* can be any SAS/GRAPH color name and must not be enclosed in quotation marks. Use an asterisk (*) to indicate the previously defined value for *color*.
style	literal; specifies the pattern that fills the pie slices. Values for *style* are the following pie and star patterns:

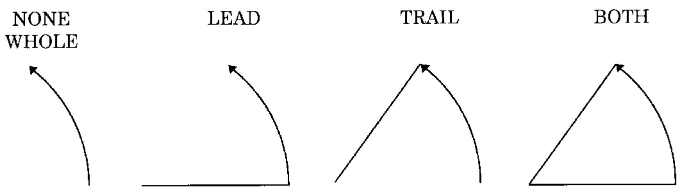
PSOLID PS	a solid fill.
PEMPTY PE	an empty fill.
P*density<style<angle>>*	a shaded pattern: *density* can be 1...5 *style* can be X | N *angle* can be 0...360

See also the discussion of fill patterns for pie and star charts in VALUE= on page 217.

line	literal; specifies the slice lines to draw. See Figure 11.24 on page 499 for values and their actions. LINE=NONE draws just the outside of the arc. Values for *line* are

WHOLE	draws no radius lines.
NONE	draws no radius lines.
LEAD	draws a radius from lead point to center.
TRAIL	draws a radius from trail point to center.
BOTH	draws both lead and trail radius lines.

Figure 11.24 Using LINE Values with the SLICE Macro



%SWAP

Exchanges control between (XLAST, YLAST) and text (XLSTT, YLSTT) coordinates.

Variable affected: FUNCTION

Updates: XLAST, YLAST, XLSTT, YLSTT

Syntax

%SWAP;

%SYSTEM

Defines the Annotate reference systems and the XSYS, YSYS, and HSYS variables.

Variables affected: HSYS, XSYS, YSYS,

Syntax

%SYSTEM (*xsys, ysys, hsys*);

xsys, ysys, hsys

specify values that represent a coordinate system and an area of the output. The following table explains these values:

Absolute Systems	Relative Systems	Units
1	7	percentage of data area
2	8	data values
3	9	percentage of graphics output area
4	A	cell in graphics output area
5	B	percentage of procedure output area
6	C	cell in procedure output area

The default is %SYSTEM (4, 4, 4).

Details

Note: Not all coordinate systems are valid with all Annotate variables or all SAS/GRAPH procedures. See "Annotate Functions" on page 429 for any restrictions that apply to the variable that you want to use. △

The ZSYS variable cannot be set through this macro. Use an explicit variable assignment instead:

```
zsys='value'; output;
```

where *value* can be 1, 2, 7, or 8.

See "Coordinate Systems" on page 411 for a description of the areas and coordinate systems.

%TXT2CNTL

Assigns the values of the text (XLSTT, YLSTT) coordinates to the control (XLAST, YLAST) coordinates.

Variable affected: FUNCTION

Updates: XLAST, YLAST

Syntax

%TXT2CNTL;

Details Use the TXT2CNTL macro when you want nontext functions to use the ending position of a text string as a starting or ending point.

Using Annotate Macros

Macro Structure

The general form of an Annotate macro is

```
%MACRO (parameters);
```

In general, the macro name represents a function and the parameters contain the values for the variables that can be used with the function. All macros except DCLANNO, SYSTEM, and SEQUENCE output an observation.

The parameters are either numeric or character. Numeric parameters can be numeric constants or numeric variable names that have been initialized to the appropriate value. Most character parameters must be expressed as literals, that is character strings without quotation marks. Exceptions are the text values that are used with the COMMENT and LABEL macros, which can be expressed as character strings enclosed in quotation marks or as character variable names.

The Annotate facility assigns the parameter values to the corresponding Annotate variables. Therefore, the observations in an Annotate data set that is created with macros that look the same as the ones that you created with assignment statements.

Making the Macros Available

To use Annotate macros, you must provide your program with access to the data set that contains the macros, and you must compile the macros before you use them. Check with your SAS Software Consultant to find out if the fileref for the data set that contains the Annotate macros that are supplied with SAS/GRAPH software is allocated automatically at your site. Then access the Annotate macros in one of these ways:

- If the fileref is not set automatically, find out where the Annotate macros are stored and allocate a fileref that points to the data set:

  ```
  filename fileref 'external-file';
  ```

  Then include the Annotate macros in your session:

  ```
  %include fileref (annomac);
  ```

- If the fileref is set automatically, compile the Annotate macros and make them available by simply submitting the ANNOMAC macro:

  ```
  %annomac;
  ```

  Note: The ANNOMAC macro must be run before any other Annotate macros are used in a SAS session. You will see a message in the SAS log that indicates that the Annotate macros are now available. The message also shows you how to get help for using the macros. △

Annotate Macro Task Summary

The following table summarizes the tasks performed by the Annotate macros.

Table 11.1 Tasks with Annotate Macros

If you want to...	Use this macro...
assign values of (XLSTT,YLSTT) to (XLAST,YLAST)	%TXT2CNTL;
begin drawing a polygon	%POLY(*x, y, color, style, line*);
continue drawing a polygon	%POLYCONT(*x, y, color*);

If you want to...	Use this macro...
copy (XLAST,YLAST) to (XLSTT,YLSTT)	%CNTL2TXT;
declare all variables	%DCLANNO;
draw a bar	%BAR(*x*1, *y*1, *x*2, *y*2, *color, line, style*);
draw a circle	%CIRCLE(*x, y, size, color*);
draw a frame	%FRAME(*color, line, size, style*);
draw a line from (XLAST,YLAST) to (XLSTT,YLSTT)	%DRAW2TXT(*color, line, size*);
draw a line from previous point	%DRAW(*x, y, color, line, size*);
draw a line	%LINE(*x*1, *y*1, *x*2, *y*2, *color, line, size*);
draw a pie slice or arc	%SLICE(*x, y, angle, rotate, size, color, style, line*);
draw a rectangle	%RECT(*x* 1,*y* 1,*x* 2,*y* 2, *color, line, size*);
draw text	%LABEL(*x, y, text, color, angle, rotate, size, style, position*);
exchange the values of (XLAST,YLAST) and (XLSTT,YLSTT)	%SWAP;
move to a point near a pie slice	%PIEXY(*angle, size*);
move to a point without drawing	%MOVE(*x, y*);
put values into a stack	%PUSH;
retrieve values from a stack	%POP;
scale and move input	%SCALET(*ptx, pty, x*0, *y*0, *x*1, *y*1, *x*0, *vy*0, *vx*1, *vy*1);
scale input	%SCALE(*ptx, pty, x*0, *y*0, *x*1, *y*1, *x*0, *vy*0, *vx*1, *vy*1);
set the coordinate system for the observation	%SYSTEM(*xsys, ysys, hsys*);
set when to draw an observation	%SEQUENCE(*when*);
write a comment to the data set	%COMMENT(*text*);

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/GRAPH® Software: Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS/GRAPH® Software: Reference, Version 8

ISBN 1-58025-525-6

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999