Paper 385-2017

Some Tricks in Graph Template Language

Amos Shu, AstraZeneca Pharmaceuticals, LP

ABSTRACT

The SAS® 9.4 Graph Template Language (GTL) Reference book has more than 1300 pages and hundreds of options and statements. It is no surprise that programmers sometimes experience unexpected twists and turns when using the GTL to draw figures. Understandably, it is easy to get frustrated when your program fails to produce the desired graphs despite your best effort. While SAS needs to continue improving the GTL product, this paper offers several tricks that help overcome some of the roadblocks in graphing.

INTRODUCTION

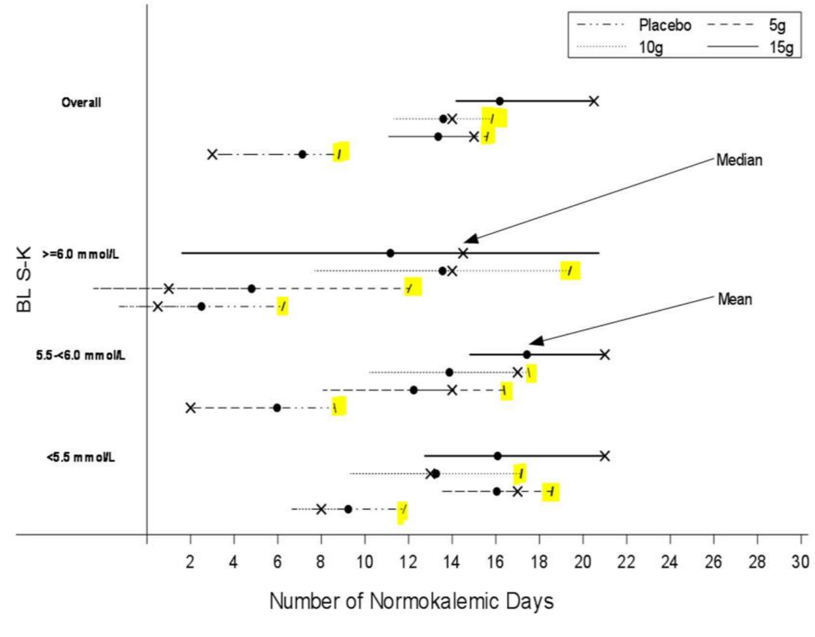
SAS® 9.4 Graph Template Language has room for improvement, which is why many statisticians and programmers have used R and other tools to draw figures instead. Despite its flaws, however, GTL is a powerful graphing tool. This paper discusses some tricks in SAS GTL to help programmers perfect their craft.

All programs presented in this paper were developed on Server SAS® 9.4 in the Windows environment.

1. OPTION THICKNESS

Unappropriated values for Option THICKNESS= could generate unexpected result. Below the unexpected symbol highlighted in yellow is caused by the option THICKNESS = 12PX.

```
SERIESPLOT  X = X_Value  Y = Y_Value / GROUP = blkcat1n
          LINEATTRS = (PATTERN = DashDotDot  COLOR= BLACK THICKNESS = 12PX);
```



The problem can be solved by either setting the option THICKNESS = 1PX or specifying OUTPUTFMT=PNG in the ODS GRAPHICS ON statement.

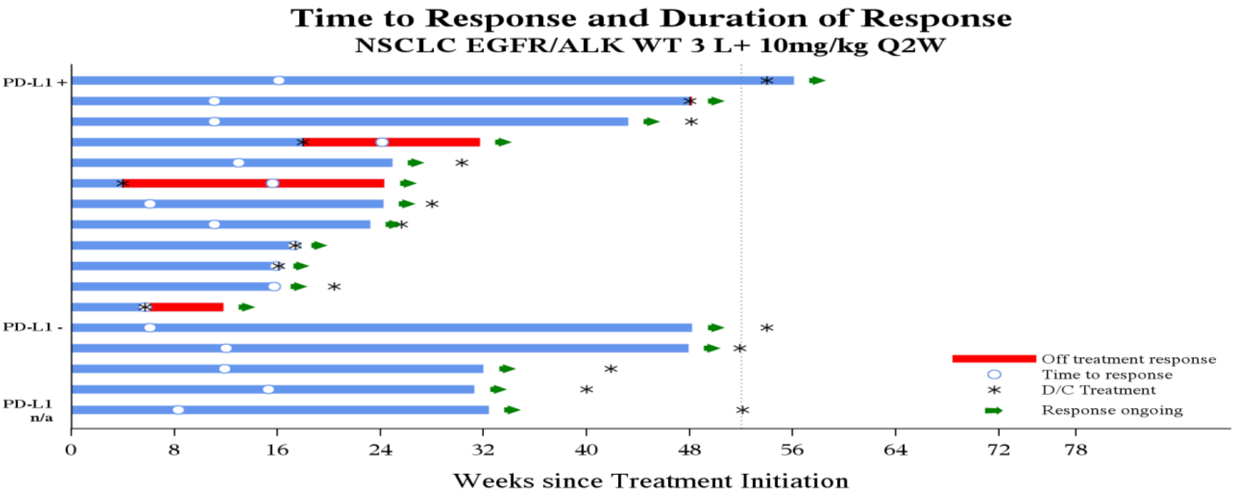
2. FONT TYPE AND UNICODE

It is important to make sure that the font used for generating the legend item supports all the unicode characters. For example, the following LEGENDITEM statement cannot generate the arrow for the legend because the font does not support the unicode “27A1”.

```
   LEGENDITEM  TYPE=text  NAME = 'arrow' / TEXT="(*ESC*) {unicode  '27A1'x}"
                 TEXTATTRS =(SIZE=10pt  WEIGHT=bold  COLOR=green
                 FAMILY="Monotype Sans WT J")  LABEL='Response ongoing' ;
```

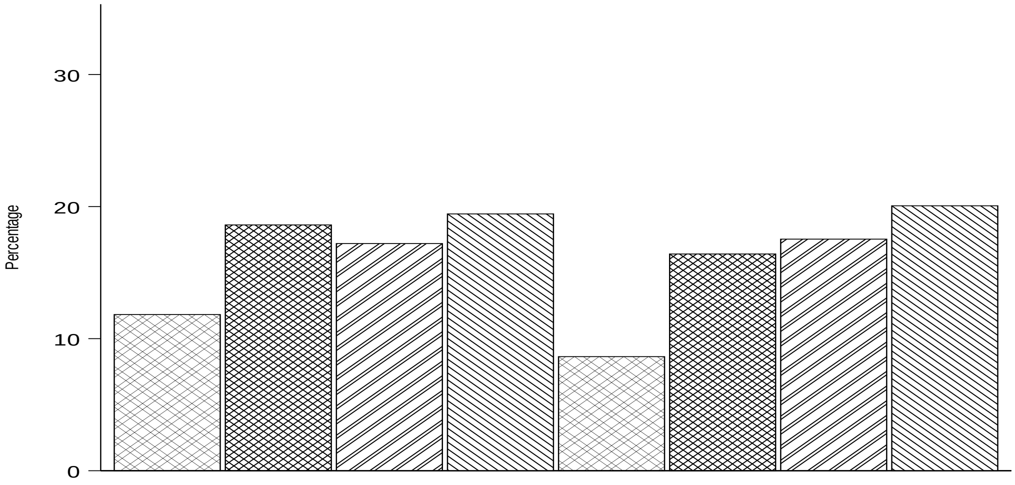
Once changing the unicode to “279E”, the arrow appears in the legend.

```
   LEGENDITEM  TYPE=text  NAME = 'arrow' / TEXT="(*ESC*) {unicode  '279E'x}"
                 TEXTATTRS =(SIZE=10pt  WEIGHT=bold  COLOR=green
                 FAMILY="Monotype Sans WT J")   LABEL='Response ongoing' ;
```



3. BARCHART POSITION AND GROUPING

Every vertical bar chart is evenly distributed in the following barchart. The input dataset and program are below.



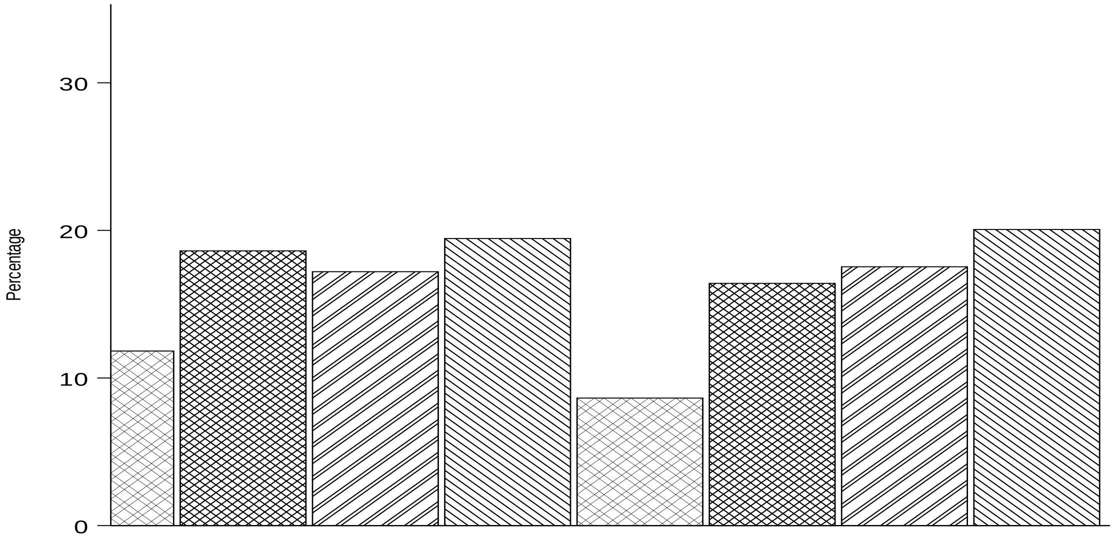
cat1	cat2	x_value	y_value
1	1	2	12
1	2	3	19
1	3	4	17
1	4	5	19.5
2	1	8	9
2	2	9	16
2	3	10	17.5
2	4	11	20

```
LAYOUT OVERLAY / CYCLEATTRS=TRUE
     YAXISOPTS=(LABEL='Percentage' OFFSETMIN=0  TICKVALUEATTRS= (SIZE=12)
        LINEAROPTS=(TICKVALUELIST=(0 10 20 30) VIEWMIN=0 VIEWMAX=35))
     XAXISOPTS=(LABEL=' ' DISPLAY = (LINE)
        LINEAROPTS=(TICKVALUELIST=(1 2 3 4 5 8 9 10 11 15) VIEWMAX=16));

BARCHART X=x_value  Y=EVAL(IFN(cat2=1, y_value, .))/GROUP=cat1
           GROUPDISPLAY=CLUSTER  BARWIDTH=1  CLUSTERWIDTH=0.95
           OUTLINEATTRS=(COLOR=BLACK)  FILLATTRS=(COLOR=WHITE)
           DISPLAY= ALL  FILLPATTERNATTRS =(PATTERN=X5) ;
```

If adding OFFSETMIN=0 to the XAXISOPTS, then the first bar chart only shows in half.

```
XAXISOPTS=(LABEL=' ' DISPLAY = (LINE) OFFSETMIN=0
      LINEAROPTS=(TICKVALUELIST=(1 2 3 4 5 8 9 10 11 15) VIEWMAX=16));
```

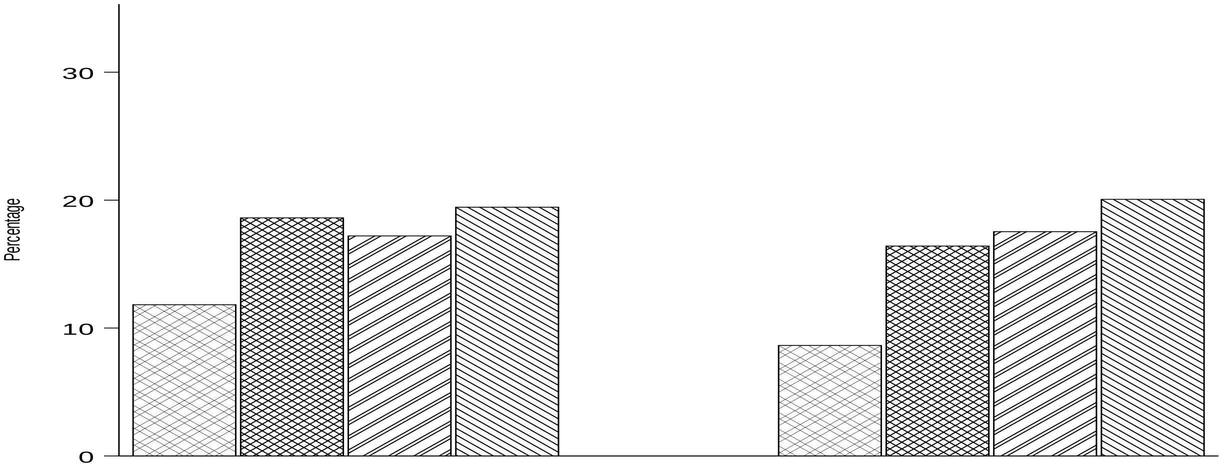


Even though there is the GROUPDISPLAY=CLUSTER option in the BARCHART statement and the values of variable “x_value” for the two cat1 subgroups are significantly different (2, 3, 4, 5 for cat1=1 and 8, 9, 10, 11 for cat1=2), surprisingly they fail to produce the visual of two clusters of bars with sufficient space between them so that the two subgroups can be easily distinguished.

To resolve this issue, some dummy data can be added to the input dataset. The values of dummy data for X-axis should be between the two subgroups, and values of dummy data for Y-axis should be zero. Here is the updated input dataset containing dummy data.

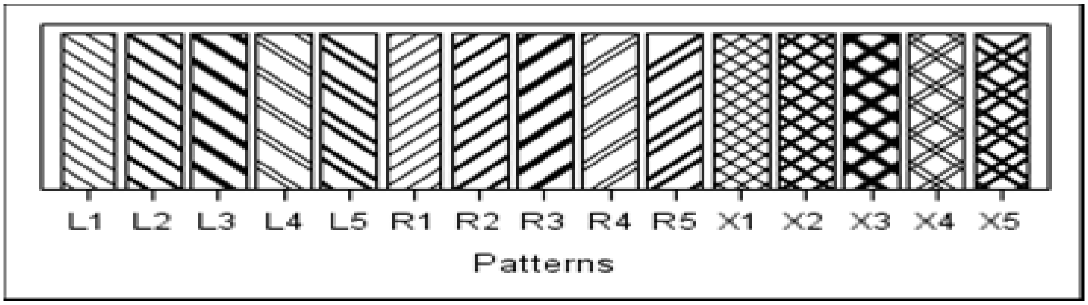
cat1	cat2	x_value	y_value
1	1	2	12
1	2	3	19
1	3	4	17
1	4	5	19.5
		6	0
		7	0
2	1	8	9
2	2	9	16
2	3	10	17.5
2	4	11	20

Without changing the code, we get the figure below that has two clear clusters representing the two subgroups.

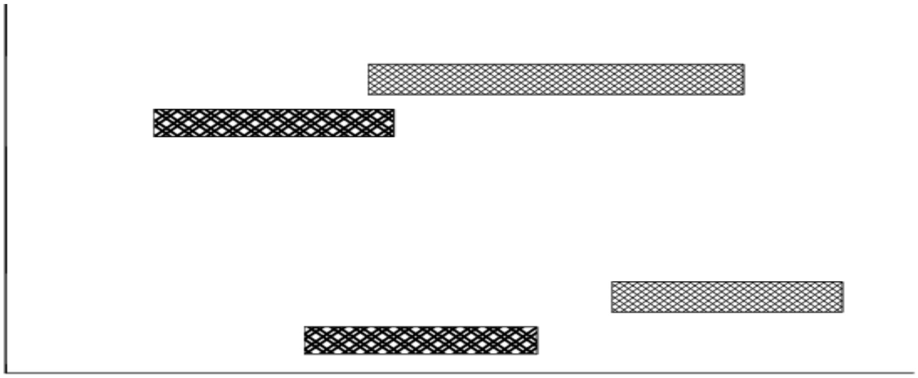


4. BARCHART PATTERN

OPTION FILLPATTERNATTRS = (PATTERN=) is the unique line pattern option for BARCHAT and BARCHATPARM statements. Here are the available pattern types.



If only black and white colors are allowed, people prefer to have barchart plots that look like the one below with different line patterns.

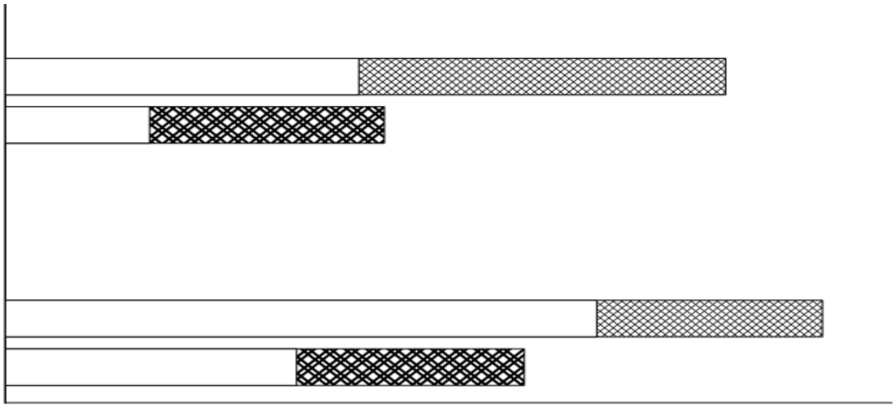


The BARCHART statement can produce this type of line patterns easily but with one catch: all bars have to start from X-axis or Y-axis. To solve this problem, you have to use two BARCHART statements – the first one generating the longer barchart plot with black color and the second one generating the shorter barchart with white color.

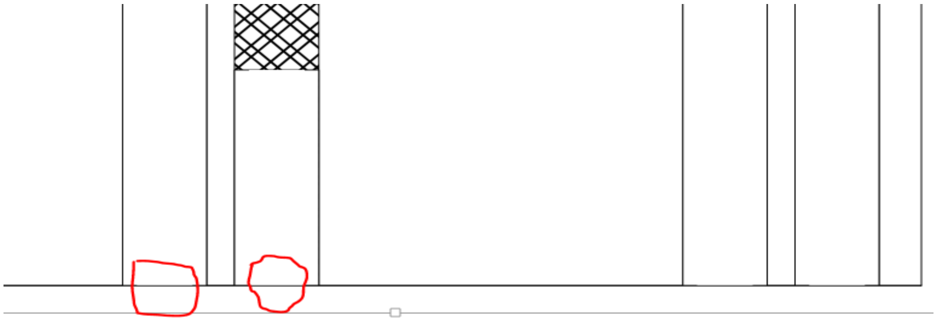
```
BARCHART X=x_value  Y=EVAL(IFN(cat2=1, y_value1, .))/GROUP=cat1
    ORIENT=HORIZONTAL  GROUPDISPLAY=CLUSTER  BARWIDTH=1  CLUSTERWIDTH=0.75
    OUTLINEATTRS=(COLOR=BLACK)  FILLATTRS=(COLOR=WHITE)  DISPLAY= ALL
    FILLPATTERNATTRS =(PATTERN=X5) ;


BARCHART X=x_value  Y=EVAL(IFN(cat2=1, y_value2, .))/GROUP=cat1
    ORIENT=HORIZONTAL  GROUPDISPLAY=CLUSTER  BARWIDTH=1  CLUSTERWIDTH=0.75
    OUTLINEATTRS=(COLOR=WHITE)  FILLATTRS=(COLOR=WHITE)  DISPLAY= ALL
    FILLPATTERNATTRS =(PATTERN=X5) ;
```

At this point, the outlines of the barchart need to be brushed using white paint. The DROPLINE statement can complete this task. However, the white paint causes some damage to Y-axis. For example, the segments inside the red circles below become lighter. The following REFERENCELINE statement will fix the Y-axis.



```
      DROPLINE X=Y_value2  Y=x_VALUE /DROPTO = y
                       LINEATTRS=(COLOR= WHITE  THICKNESS = 45PX);
```



```
REFERENCELINE X=0 /LINEATTRS=(PATTERN=SOLID  THICKNESS = 0.7PX);
```

CONCLUSION

The above four illustrated real examples are frustrating but not always bad from programming skill perspectives. Your programming skills are elevated once the solutions are found, as I have shown in those examples in this paper, by adding appropriate options in the SAS code.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Please contact the author at:

Amos Shu
AstraZeneca Pharmaceuticals, LP
200 Orchard Ridge Dr.
Gaithersburg, MD 20878
Email: amos.shu@astrazeneca.com

TRADEMARK INFORMATION

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

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