ABSTRACT
SAS/GRAPH® is a very powerful data analysis and presentation tool. Creating information-rich graphics facilitates understanding the underlying data. Creating these graphics in SAS® can be time-consuming and challenging. This paper introduces some tips and tricks in using PROC GPLOT to generate plots from clinical data in ODS RTF format, including methods to:
1. Restructure data to create error bars using interpol=hiloctj option
2. Separate error bars that overlap at the same tick mark
3. Display irregular numeric data in x-axis with unequal intervals
4. Dynamically assign the y-axis tick marks with frequently updated data
5. Build a common reference line for a plot with two different y-axes
6. Control the position and alignment of legends for plots with two y-axes
7. Align the value of BY variables in the title of plot by using #BYVAL()
8. Insert Greek letters in the title, footnote or legend

This paper is intended for those who are familiar with PROC GPLOT procedure, and all figures in this paper are created as an ODS RTF output.

INTRODUCTION:
Producing informative graphics is a common task for SAS users. SAS graphics are generated by writing codes to build elements such as defining axes, legends and title etc. Creating an informative graph in SAS can sometimes be very time-consuming. However, there are tips and tricks that can make this task easier. This paper summarizes some tips and tricks in generating informative graphs in SAS, which may save you time in developing your graphics creation as well as enhance your graphic output.

TIPS & TRICKS 1:
Restructure data to create error bars using interpol=hiloctj option

Figure 1

In clinical studies, there is often interest in examining the drug effect over time, so creating plots with error bars as shown in Figure 1 is often requested by clients. How can you display the three values (upper limit, lower limit, and mean) in one line or symbol? SAS has the solution for you by using the INTERPOL=HILOCY option in the SYMBOL statement to generate such plots. This option specifies that a solid vertical line connect the high and low Y values for each X time point. In order to use this option, it needs at least two values of Y for every value of X, otherwise, the single value is displayed without the vertical line. Before using this option, you must restructure the data. By using a simple data step (see code below), you can assign three different variables into one variable. After this step, you can specify the option in the symbol statement. The error bars will be generated as in Figure 1.
TIPS & TRICKS 2: Separate error bars that overlap at the same tick mark

In a plot like Figure 1, if you have three or four error bars (3 or 4 treatments) to be displayed at the same time point, the error bars may easily overlap. As shown in Figure 2, it is hard to distinguish the difference between groups even if they are colored. In order to clearly display values for each group, you can use the trick of adding a small value to the variable in X-axis for each of the treatment such that the value will not overlap. The difference between groups will become clear (i.e. Figure 3). Sample code 2 shows you how to do this.

### Sample Code 1

```plaintext
*** assign three different values ***;
*** into one variable ***;
data test;
  set result;
  amean-tuclm;
  seq=1;
  output;
  amean-tlclm;
  seq=2;
  output;
  amean-tmean;
  seq=3;
  output;
  drop tuclm tlclm tmean nobs;
run;

symbol2 color=blue
  interpol=hiloctj
  cv=blue
  ci=blue
  line=1
  value=none
  ......
```

### Sample Code 2

```plaintext
/*******************************************/
reset the time point for other groups, so
the symbol will not be overlapped on the
same time point
/*****************************/
data final;
  set test;
  newtpt1=tptnum+0.2;
  newtpt2=tptnum+0.4;
run;
```

```plaintext
proc gplot data=final;
  plot trt1*tptnum
    trt2*newtpt1
    trt3*newtpt2/overlay legend=legend
    noframe vaxis= axis2 haxis=axis1
    vref=0 cvref=black;
run;
```
TIPS & TRICKS 3:
Display irregular numeric data in x-axis with unequal interval

In making plots for clinical studies, it is common to display the x-axis with irregular numeric data with uneven intervals like the example shown in Figure 1: time point=0.5, 1, 1.5, 2, 3, 4, 6, 8, 10, 12, 16, 24, instead of a scale such as 2, 4, 6, 8, 10, 12 etc. You may think that you can use the ORDER option in AXIS1 statement (order=(0.5 1 1.5 2 3… .)), but the time point will be displayed in even intervals. For this case, you really want to show the time point at the true interval. A quick and easy solution is to use the tick option to specify the label as blank. This will omit all the unnecessary time points for display purposes. By using the empty quotation marks (""), the label will be set as blank so that it looks like it skipped the time point as needed in the figure. Alternatively, you may use the do loop to specify which tick marks to be omitted within some range (see sample code 3a). There are other methods (e.g. the ANNOTATION dataset method) to get the same result, which I will not go into details in this paper. The following is the sample code for a trick that does not require complex programming.

Sample Code 3

```plaintext
axis1 order=(0 to 24 by 0.5 )
   major= none minor = none
   label =(h= 1.0 'Time post dose (hrs)')
value= (h=0.7 t=1 '' t=6 '' t=8 '' t=10 '' t=12 '' t=14 '' t=15 ''
   t=16 '' t=18 '' t=19 '' t=20 '' t=22 '' t=23 '' t=24 '' t=26 ''
   t=27 '' t=28 '' t=29 '' t=30 '' t=31 '' t=32 '' t=34 '' t=35 ''
   t=36 '' t=37 '' t=38 '' t=39 '' t=40 '' t=41 '' t=42 '' t=43 ''
   t=44 '' t=45 '' t=46 '' t=47 '' t=48 '' t=49 '')
```

Sample Code 3a

```plaintext
axis1 ......

/* or you may use the do loop to specify which tick mark to be omitted*/
value=(h=0.7 f=swiss1
%do i=&stick %to &etick;
t=&i ''
%end;)
```

TIPS & TRICKS 4:
Dynamically assign the y-axis tick marks with frequently updated data

It is common that data used for plotting will be frequently refreshed. It is ideal to have the tick marks for axes automatically updated every time when data are updated. How do you dynamically assign the y-axis tick marks without having to modify the program whenever new data are loaded? (See solution in the sample code 4). First, you need to find the maximum and minimum value every time new data are loaded. Set the minimum and maximum value as macro variables so that you can use them in the ORDER statement later when you specify these graph options. Secondly, you need to define the interval between the minimum and maximum value. This determines how much you want to increment each tick mark. The following are some examples of ranges. If you predict the value range to be relatively small, you may want to use &step=0.2. If you know the range will be large, you may consider to set &step=10 or even 100. The final step is to ascertain that all the values are within the range by using &step.
TIPS & TRICKS 5:  
Build a common reference line for a plot with two different y-axes

In preparing a plot, a reference line is often added to make the plot more informative. However, when there are two y-axes in a plot (each y-axis represents different variable), adding a reference line can be very challenging especially if one y-axis always has positive values (eg. Plasma concentration), and another y-axis can have both negative and positive values (eg. QTcB changes). You may consider specifying these two y-axis tick marks using two AXIS statements and having different ORDER options. This will give you two reference lines. However, this will make the plot very confusing because readers will not be sure which reference line is associated with which axis. The trick is to find the common value range for both y-axes and use the VREF=0 option (see sample code 5). It will set the common reference line at value=0 for both axes. Before using this trick, you may need to use TIP & TRICK #3 and #4 to dynamically assign the tick marks and omit the negative tick marks for the right y-axis (see previous sample code 4 for the two y-axes case). If you use TIP & TRICK #5, the plot will present the information as needed (see figure 4).

Sample Code 4

```plaintext
......
......

*** Find the min and max from the data ***;
if _stat_='MIN' then min=min(a, b, c, d);
else if _stat_='MAX' then max=max(a, b, c, d);
if eof then do;
call symput('min',compress(put(floor(min),best12.)));
call symput('max',compress(put(ceil(max),best12.)));
end;
......

*** set the axis increment dynamically ***;
%let step=3;
%if 100<=&max-&min<=50 %then %let step=10;
data vmax;
** reset the max to cover all values;**
vmax=ceil((&max-&min)/&step)*&step - abs(&min);
call symput('vmax',compress(put(vmax,best12.)));** find tick mark start and end # to be set as missing label for two y-axes case **;
** for left y-axis **;
stick=ceil((&qtcmax-&qtcmin)/&step) +2;
etick=ceil((vmax-&min)/&step) +1;
call symput('stick',compress(put(stick,best12.)));
call symput('etick',compress(put(etick,best12.)));
** for right y-axis: only show positive values as needed **;
stick2=ceil((0-&min)/&step);
call symput('stick2',compress(put(stick2,best12.)));
run;
......

*** code for PROC GPlot ***;
......
axis2 order=(&min to &vmax by &step);
```

Sample Code 5

```plaintext
proc gplot data=final;
plot &trtname*tptnum / overlay noframe legend=legend1 haxis=axis1
    vaxis=axis2 vref=0 cvref=black;
plot2 a*tptnum
    b*tptnum /overlay noframe legend=legend2 vaxis=axis3;
run;
```
TIPS & TRICKS 6:
Control the position and alignment of legends for plots with two y-axes

You may need to create two y-axes plot in order to display the changes made by two parameters over the same time period as figure 4 shows, and you may want both legends to be displayed in the plot area (not under the plot x-axis). In order to properly align and correctly position those legends, specify the ORIGIN= option (the POSITION= option is ignored).

Sample Code 6

```latex
options reset=all;
  legend1 origin=(70,90) pct
    mode=share;
  legend2 origin=(70,85) pct
    mode=share;
proc gplot data=test;
  plot  test1*timept / overlay legend=legend1;
  plot2 test2*timept / overlay legend=legend2;
run; quit;
```

From the above sample code 6, there are two legends produced for your plot, one from the PLOT statement and another from the PLOT2 statement. Each legend is treated separately, so the entries in each legend may not line up. If you use the ORIGIN= option on a LEGEND statement to specify the origin of the legend, the legends will align properly. When specifying the origin, you will need to use the Y value to keep the legends from overlapping. You need to specify MODE=SHARE on the LEGEND statement to allow the legend to be placed inside the graph area when using the ORIGIN= option.

TIPS & TRICKS 7:
Align the value of BY variable in the TITLE statement of the plot by using #BYVAL()

Often, the client may want to see figures for each patient and request all titles be aligned to the left to be consistent with table titles. In order to fulfill this requirement, specify the BY statement in PROC GPLOT. Be aware that if you apply the BY statement in the figure title, the subject ID will be automatically displayed and aligned in the center as shown in the following:

Figure 5

```
Figure 12: Individual Time-Matched Changes in Test1 Over Time – Treatment 1
Unique Subject Identifier = 001002
```
To align the title to the left, you may try to use options like J=L in the TITLE statement. This does not work since there is not an option to left justify the BY line. However, you can display the BY variable value using the #BYVAL() option on a TITLE statement that supports the JUSTIFY option. First, you need to specify the NOBYLINE option on an OPTIONS statement to suppress the default by-line. Secondly, do the alignment in the TITLE statement. Also, by using the #BYVAL(), you may specify the text label as what you want (e.g. see Figure 6), otherwise, it will automatically display the label of the variable.

Sample Code 7

```
options nobyline;

   title4 j=l "Individuaal Time-Matched Changes in Test1 Over Time-Treatment 1";
   title5 j=l "USUBJID= #byval(usubjid)";

   proc gplot data=test;
      plot a*b;
      by usubjid;
      run;
      quit;
```

Figure 6

**Figure 12: Individual Time-Matched Changes in Test1 Over Time – Treatment 1**

USUBJID = 001002

TIPS & TRICKS 8:
Insert Greek letters in title, footnote or legend

There are occasions when a figure requires the insertion of Greek letters (e.g. alpha, beta) in the title, footnote, or legend etc. By using ODS RTF code to generate figures, the output (graph) is a single image, which means that when it converts the output into RTF file, it won’t decode the unicode (e.g. “u945”) to display the Greek letter α. However, you can use the Greek font to specify the alpha and beta characters in the TITLE or FOOTNOTE statement. For example:

Sample Code 8

```
   title1 h=2 f=swiss 'This is the ' f=greek '61'x f=swiss 'character';
   title2 h=2 f=swiss 'This is the ' f=greek '62'x f=swiss 'character';
```

If you want to know which hex values you need to specify for a particular Greek character, you can use the GFONT procedure to see a list. For example:

Sample Code 9

```
   proc qfont name=greek nobuild romhex;
   run;
```

Similarly, you can apply this font specification in labels, legends and anywhere you want by using the font option.
CONCLUSION

SAS is very powerful in defining graphic elements if you know the tips and tricks. If you are not familiar with those options, it may take time to figure out the shortcuts. I hope the tips and tricks in this paper can make your life a little easier when you create informative and specially formatted graphics using PROC GPLOT. I am open to get your feedback and welcome you to share your experiences with me in creating complex graphs.

ACKNOWLEDGEMENTS

I would like to acknowledge the programming team at Applied Clinical Intelligence for their support, especially thank Tim Kelly, Jeffrey Davidson, and Alan Smith for their suggestions and comments.

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.

REFERENCES


CONTACT INFORMATION

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

Qin Lin
Applied Clinical Intelligence, LLC
225 City Avenue, Suite 15
Bala Cynwyd, PA 19004
Tel: 484-429-7200 ext. 116
Email: qlin@a-ci.com
www.a-ci.com