PharmaSUG 2020 - Paper DV-158

Enhanced Visualization of Clinical Pharmacokinetics Analysis by SAS GTL

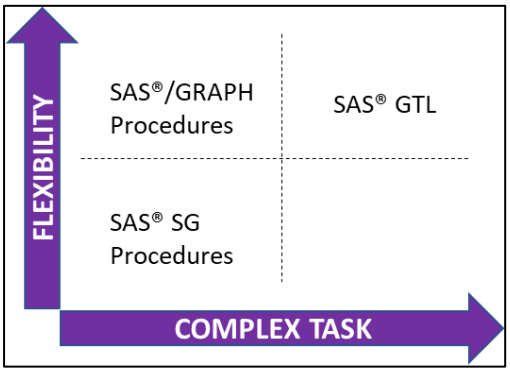
Min Xia, PPD

ABSTRACT

Graphs are essential tools to comprehensively represent data and raise the readability of analysis. They are an integral part of Pharmacokinetic (PK) analysis and the high-quality PK graphs are very important for Clinical Study Report (CSR) and a key task for regulatory submissions. This paper provides instructions to create PK analysis graphs powered by SAS Graph Template Language (GTL) such as multiple panels series plot and multiple columns forest plot. In addition, DYNAMIC variables and other advanced techniques are introduced to increase the flexibility of GTL and the data visualization accuracy.

INTRODUCTION

Pharmacokinetics is the study of how drug behaves in the body through the mechanisms of absorption, distribution metabolism and excretion. During clinical pharmacological development, this crucial application investigates the correlations between drug concentrations and pharmacological responses. The results lead to enhance the safety and the efficacy of patient's drug therapy. In order to increase the analysis accuracy, it is common that a large amount of records is stored and examined. Therefore, it is critical to properly visualize the data and the analytical results during PK study.



Display 1. Complexity – Flexibility Matrix

Although GTL has been introduced in SAS ODS graphics since SAS 9.2, traditional SAS/GRAPH and SAS SG procedures are still widely used. As shown in Display 1, SAS SG procedure can produce impressive graphs by using pre-defined templates with little inputs, but the output might not fully meet with PK scientist's expectation. Because all templates have been pre-defined, few options can be used for the customization through SAS SG procedure. That is why programmers still use traditional SAS/GRAPH with self-developed macros and annotate facility to increase flexibility to accomplish the requests. However, this procedure sacrificed the graph quality and the production efficiency. Furthermore, it would result in erroneous outputs and difficulties for debugging because of convoluted procedures within the custom-built tools.

With the release of SAS 9.4, GTL has been powered with more features and enhancements. It has become one-step solution for all kinds of graphs with unlimited panels, layouts and fully customizable options in styles and it even supports user-defined formats with Unicode values.

In order to build GTL plot, three key components are required:

1. Style template. It defines the appearance of graph such as font, size, color, marker etc.

2. Graph template. This key component is defined by TEMPLATE procedure and is enclosed within BEGINGRAPH and ENDGRAPH statements. In the block, multiple elements can be used to define the structure of graph such as: LAYOUT statement to manage the graph area, PLOT statement to overlay desired plots on each area and other supplementary statements for more precise controls.

3. SGRENDER procedure. It outputs the graph with combination of data and template objects to the active ODS destination.

This paper is intended to present the procedures to create two common graphs in PK analysis by using SAS GTL.

MULTI-PANEL SERIES GRAPH

In Pharmacokinetics the kinetic profile of many drugs is not mono-exponential versus time. However, in the terminal phase, log-concentrations of analytes in plasma decline linearly. In order to obviously visualize the full data plots over the huge concentration ranges and enhance the evaluation of PK models, semi-logarithmic scale is frequently plotted side by side with linear scale on the same page.

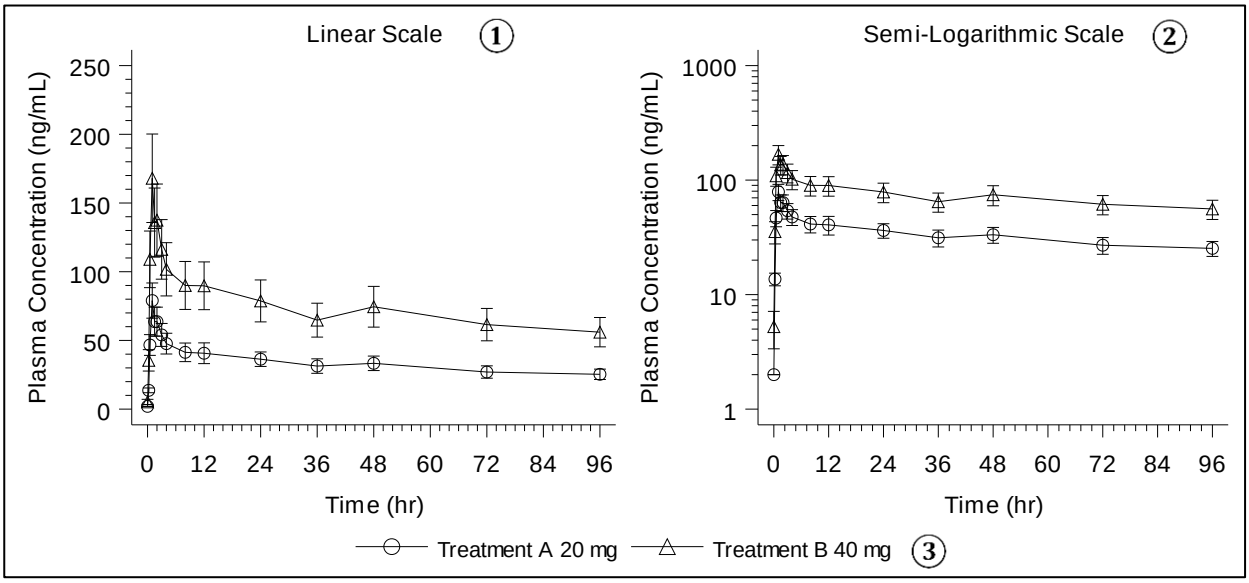


Figure 1. Mean(±SD) Plot with Linear Scale and Semi-Logarithmic Scale Side by Side

As shown in Figure 1, this example plot presents the mean plasma concentrations with standard deviation (SD) versus time profile grouped by treatments. It is divided into 3 portions: Linear scale at left portion, Semi-logarithmic scale at right portion and the global legend on the middle bottom of graph. Below are the codes to generate this graph with minimum modifications (SAS Program 1).

```
proc template;
   define statgraph mean_errbar_2col_Graph / store = work.template;
      begingraph;

         /* Divides overall layout into two sections side by side */
     (a) layout lattice / columns = 2 columngutter = 10;

             /* Left Portion contains Linear mean line and error-bars */
         (b) cell;

               /* The header for on the top plot */
         (c)   cellheader;
                 entry 'Linear Scale' / border = false;
               endcellheader;

               /* Define layout with axis options for linear Y-axis */
         (d)   layout overlay / xaxisopts = (label = "Time (hr)" type = linear)
①                              yaxisopts = (label = "Plasma Concentration (ng/mL)"
                               type = linear linearopts = (viewmin=0
                               thresholdmax = 1 thresholdmin = 1));

         (e)     seriesplot x = time
                            y = mean / group = treatment index = treatn
                                       yerrorlower = lower yerrorupper = upper
                                       display = (markers) name = 'linear';
               endlayout;
             endcell;

             /* Right Portion contains Semi-Logarithmic mean line and error-bars */
         (f) cell;

               /* The header for on the top plot */
         (g)   cellheader;
                 entry 'Semi-Logarithmic Scale' / border = false;
               endcellheader;

               /* Define layout with axis options for log Y-axis */
         (h)   layout overlay / xaxisopts = (label = "Time (hr)" type = linear)
②                               yaxisopts = (label = "Plasma Concentration (ng/mL)"
                               type = log logopts = (base = 10 thresholdmax = 1
                               thresholdmin = 1));

         (i)     seriesplot x = time
                            y = mean_log / group = treatment index = treatn
                                           yerrorlower = lower yerrorupper = upper
                                     display = (markers) name = 'semilog';
               endlayout;
             endcell;

         endlayout;

         /* Shared Lengend for two plots on the bottom */
③       layout globallegend / border = False;
           discretelegend "linear" / autoitemsize = true;
         endlayout;

     endgraph;
   end;
run;
```

SAS Program 1. PROC TEMPLATE to Create Mean (±SD) plot with Linear Scale and Semi-Logarithmic Scale side by side

(a) LAYOUT LATTICE controls the distribution of blocks in whole graph area. In this case, two portions by column are created evenly side by side with option *columns = 2*. If we can want distribute graphs vertically, option can be modified with *rows = 2*.

(b) The first block (left portion) starts with CELL statement for the linear scale as shown in Figure 1 ①. CELL statement is to define cell which is the container for each GTL graph outputs. It by fault is not required if LAYOUT statements are used. In order to precisely define the header for each cell, CELL statement is recommended to create the in-graph header for each portion.

(c) CELLHEADER block reserves area on the top of cell for the header. ENTRY statement is used to add header text 'Linear Scale'. Although it is common to directly create header description with the option in LAYOUT OVERLAY statement, this extra step provides the better alignment option of header within the plotting region.

(d) LAYOUT OVERLAY statement initiates the plot with definition of X and Y axes by using option *xaxisopts=(axis-options)* and *yaxisopts=(axis-options)* respectively. Y-axis is tuned by additional options *type=linear linearopts=(viewmin=0 thresholdmax=1 thresholdmin=1)* so that the linear scale is ensured, and all tick marks are presented within full data range.

(e) SERIESPLOT statement is selected to draw the graph of mean concentrations versus time group by treatment. In the options, *yerrorlower* and *yerrorupper* are assigned to the standard deviation for vertical bars and *display = (markers)* is selected to display symbols at mean value. In order to safeguard the group sorting order both in graph and in legend, it is always recommended to add option *index* = referring to the numeric version of group sorting order variable.

Like (b) - (e), statements (f) - (i) are used for the right portion to display semi-logarithmic scale as shown in Figure 1 ②. Key updates are to change the header as 'Semi-Logarithmic Scale' in statement (g); to update Y-axis type as log scale by using option *type = log logopts = (base = 10 thresholdmax = 1 thresholdmin = 1)* in statement (h) and to modify the Y-axis value assigned to mean_log which only holds positive values to avoid warnings during the semi-log plotting in statement (i).

At the last step, LAYOUT GLOBALLEGEND statement is applied to insert the legend block on the bottom as shown in Figure 1 ③. The legend from plot 'linear' is loaded by DISCRETELEGEND statement. The symbols are referred by the different treatments. Finally, side by side two panels graph can be executed by PROC SGRENDER to load the data with GTL onto ODS destination.

In traditional SAS/GRAPH and SAS SG procedures, PROC GREPLAY is always used to create multi-panels graphs onto ODS destination by loading pre-built graphs from SAS catalog. Thanks to the one step solution powered by GTL combined with PROC SGRENDER, we can skip this step to save a lot of I/O processing time and maintain graphs high quality without manually assigning layouts in TEMPLATE.

Furthermore, GTL provides us flexible ways to lay out more graphs on one page. For example, as we can observe in Figure 1 that concentrations in the early phase of profile are too clustered to differentiated, so a picture-in-picture (PIP) plot of the first 4 hours profile would be very helpful to obviously distinguish the difference between treatments as shown below in Figure 2. In order to embed another independent plot inside graph, LAYOUT GRIDDED is considered following after SERIESPLOT statement in each CELL block. Inside the LAYOUT GRIDDED block, similar series of statements LAYOUT OVERLAY and SERIESPLOT are used to create insert plot as shown in SAS Program 2.

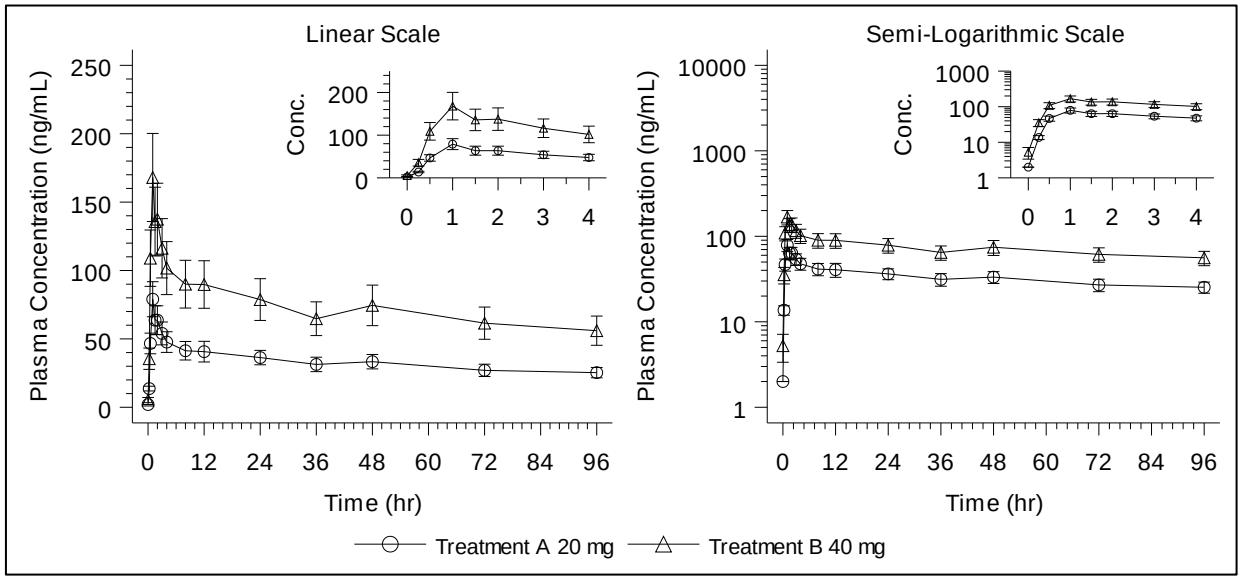


Figure 2. Mean(±SD) Plot with Linear Scale and Semi-Logarithmic Scale with PIP Graph for the First 4 Hours Profile

```
/* Define gridded layout for the nested plot */
layout gridded / width = 30pct height = 22pct valign = top halign = right;

   layout overlay / xaxisopts = (display=(line ticks tickvalues)   type = linear)
                    yaxisopts = (label = "Conc."
                                 type = linear linearopts = (viewmin=0
                                    thresholdmax = 1 thresholdmin = 1));
      seriesplot x = time_pip
                 y = mean / group = treatment index = treatn
                                    yerrorlower = lower yerrorupper = upper
                            display = (markers) name = 'linear_pip';
   endlayout;
endlayout;
```

SAS Program 2. LAYOUT GRIDDED Block to Create PIP Graph

MULTI-COLUMN FOREST PLOT

Forest plot is an essential tool to summarize the impact of covariates on PK parameters such as Cmax (Maximum concentration) and AUC0-t (Area under the curve from 0 to the last non-zero concentration). It illustrates the treatment ratios of PK parameters combined with geometric least square mean ratios along with corresponding 90% confidence intervals. As well suggested in FDA's Population Pharmacokinetics Guidance for Industry, this plot is one of key to support decision making during clinical pharmacology development. For the complicated studies, it is remarkable to visualize covariates at multiple levels among treatments, analytes and PK parameters all in one forest plot across whole study as shown in Figure 3. Different analyte names are listed on the left panel and the horizontal forest plots are drawn for each PK parameters as sub-category listed along the Y-axis. For the better representation, analyte name is only displayed once for each group. Therefore, firstly FORMAT dataset is created to display PK parameter short name along the Y-axis as shown in Display . Then after applying linear mixed-effects models by using PROC MIXED procedures, statistic variables are derived with some other key variables. Those key variables will be used later in plot for the purpose of display as shown in Display . The explanation for those variables is descripted as below:

• x_label: Label of X-axis directly from the output of ANOVA analysis without hard coding.

• order: Order number as sorting variable on the Y-axis.

• ppcat_label: The unique name of analyte derived from ppcat used on the panel ① in Figure 3.

• param_label: The display of Y-axis value with PK parameter short name.

start	label	fmtname	type
1	AUC0-t	paramlst	C
2	AUC0-inf	paramlst	C
3	Cmax	paramlst	C
4	AUC0-t	paramlst	C
5	AUC0-inf	paramlst	C
6	Cmax	paramlst	C

Display 2. FORMAT Dataset for PK Parameter Display

x_label	order	ppcat	ppcat_label	param_label
Treatment B / Treatment A	1	Analyte 1	Analyte 1	AUC0-t
Treatment B / Treatment A	2	Analyte 1		AUC0-inf
Treatment B / Treatment A	3	Analyte 1		Cmax
Treatment B / Treatment A	4	Analyte 2	Analyte 2	AUC0-t
Treatment B / Treatment A	5	Analyte 2		AUC0-inf
Treatment B / Treatment A	6	Analyte 2		Cmax

Display 3. Dataset with Additional Variables for Data Driven Graph Display

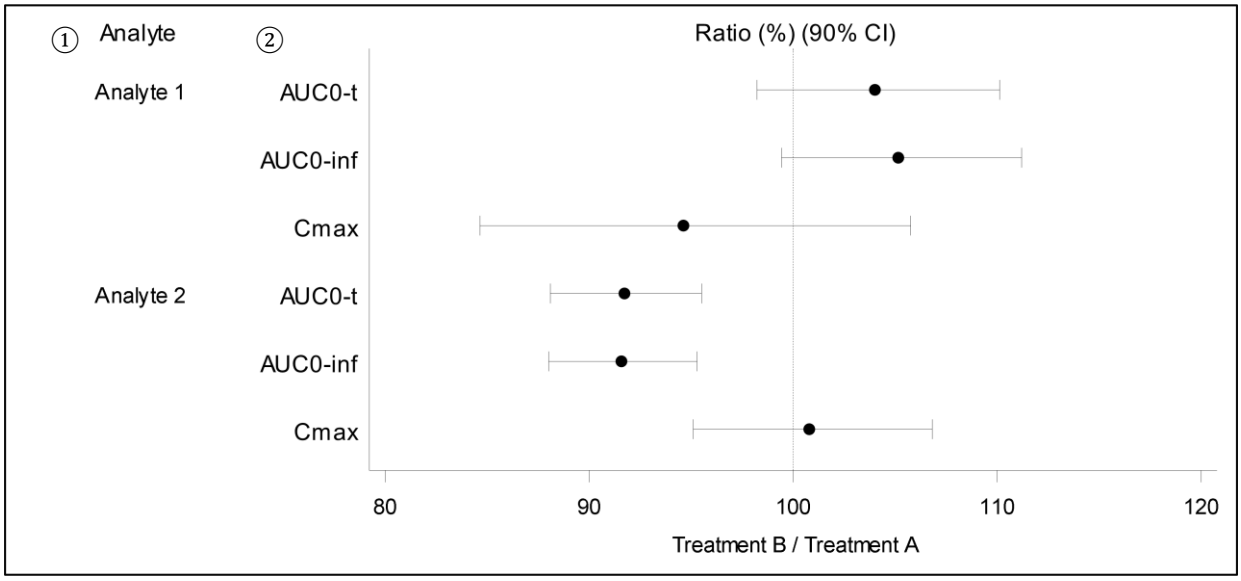


Figure 3. Forest Plot with Linear Scale and Semi-Logarithmic Scale Side by Side

In this graph, we want to display the X-axis label driven by variable x_label in dataset. For this purpose, BY statement in PROC SGRENDER and DANAMIC statement in PROC TEMPLATE are introduced as shown in SAS Program 3. Detail explanation for key statements are also demonstrated.

```
proc template;
   define statgraph Forest / store = work.templat;

  (a) dynamic _byval_;

       begingraph;

         /* Divides overall layout into two sections with different size */
      (b) layout lattice / columns=2 columnweights = (0.2 0.8) columngutter = 0;

           /* Left portion contains analyte name list */
      (c) cell;

             /* The header for on the top plot */
      (d)    cellheader;
               entry 'Analyte' / border = false;
             endcellheader;

①            /* Define layout without axis display */
      (e)    layout overlay / walldisplay = none xaxisopts = (display = none)
                              yaxisopts = (display = none reverse = true);

      (f)      axistable y = order value = ppcat_label / display = (values);
             endlayout;
           endcell;

           /* Right portion contains forest plot */
      (g)  cell;

             /* The header for on the top plot */
      (h)    cellheader;
               entry 'Ratio (%) (90% CI)' / border = false;
             endcellheader;

             /* Define layout with axis options */
      (i)    layout overlay / walldisplay = none
                              xaxisopts = (label = _byval_
                                linearopts = (tickvaluelist = (80 90 100 110 120)))
                              yaxisopts = (type = discrete reverse = true
②                                          display = (line tickvalues) discreteopts
                                           = (tickvalueformat = $paramlst.));

      (j)      scatterplot y = order
                           x = ratio /  xerrorlower = cilo xerrorupper = cihi
                                        markerattrs = (symbol = circlefilled);

      (k)      referenceline x = 100 / lineattrs = (pattern = 2 thickness = 1);
             endlayout;
           endcell;

         endlayout;
       endgraph;
   end;
run;

options nobyline orientation=landscape;

(l) proc sgrender data = graph template = Forest;
      by x_label;
   run;
```

SAS Program 3. PROC TMPLATE and PROC SGRENDER to Create Two Panels Forest Plot

(a) Before BEGINGRAPH, DYNAMIC statement is used to declare the dynamic variable _byval_ initialized by BY statement in SGRENDER procedure. This variable can be executed in the later GTL statements.

(b) LAYOUT LATTICE splits graph area into 2 portions horizontally by *option columns = 2*. Option *columnweights = (0.2 0.8)* specified that left portion occupies 20% of graph area and right portion occupies 80%.

(c) The first CELL statement creates the left portion to display analytes list as shown in Figure 3 ①. Analyte name is only displayed once for each group.

(d) CELLHEADER and ENTRY statements set up the listing header as 'Analyte'.

(e) LAYOUT OVERLAY statement establishes the plot area without any axis display by using option *display = none*. Option *reverse = true* is also turned on for Y-axis to ensure the display order from top to bottom.

(f) AXISTABLE statement provides an easy option to display the value by referring the variable to correspondent axis. Option *display = (values)* is checked to only show values.

(g) This CELL statement reserves right portion to draw forest plot as shown in Figure 3 ②.

(h) CELLHEADER and ENTRY statements set up the plot header as 'Ratio (%) (90% CI)'.

(i) LAYOUT OVERLAY statement frames the plot area with definition of both axes. In particular, the label of X-axis is driven by dynamic variable with option *label = _byval_* and the value of Y-axis is displayed as the PK parameter short names by using format $paramlst. defined as in Display 3.

(j) SCATTERPLOT statement draws the symbols for ratio and the horizontal error bars for low and high 90% CI correspondingly by assigning the CI values to option *xerrorlower* and *xerrorupper*.

(k) A vertical dash reference line is created at 100% by using REFERENCELINE statement.

(l) During SGRENDER procedure, BY statement initiates variable x_label as dynamic variable used in the GTL as descripted in (a) statement.

By using the similar approach, more columns can be added for further level analysis. CELL blocks benefit the accurate alignment of layout header within the data plotting area. Together with BY statement in PROC SGRENDER and DYNAMIC statement in PROC TEMPLATE, we have more efficient ways to display data-driven values in graphs. Thanks for option *tickvalueformat=$paramlst*, it provides flexible ways to label the display on the axis without changing the sorting order. In addition, SAS 9.4 supports Unicode in the inline format, it enables us to display more special characters on the axis. However, {sub} and {sup} makeups are still not supported for the inline format. If the subscripted or superscripted alphabets are needed on the axis, GTL in SAS 9.4 provides a great alternative way by using DRAWTEXT statement inside the LAYOUT block. Although SAS SG procedure can produce similar plot, more codes and pre-computing in dataset are required because of its inflexibility.

CONCLUSION

During the demonstration of two example plots, SAS GTL has showed us limitless options to create stat-of-the-art graphs. Learning SAS GTL is one of the best investments in improving visualization of clinical pharmacokinetics analysis.

REFERENCES

Harris, K. 2018. “Great Time to Learn GTL”, PharmaSUG 2018 - Paper EP18. Available at https://www.lexjansen.com/pharmasug/2018/EP/PharmaSUG-2018-EP18.pdf

Mistry, J. 2019. “Learning SAS® GTL Basics with Practical Examples”, PharmaSUG 2019 - Paper AD111. Available at https://www.lexjansen.com/pharmasug/2019/AD/PharmaSUG-2019-AD-111.pdf

McCarthy, A. 2017. “Using Animated Graphics to Show PKPD Relationships in SAS 9.4®”, PharmaSUG 2017 - Paper DV07. Available at https://www.lexjansen.com/pharmasug/2017/DV/PharmaSUG-2017-DV07.pdf

US Food and Drug Administration. July 2019. “Population Pharmacokinetics Guidance for Industry”. Available at https://www.fda.gov/regulatory-information/search-fda-guidance-documents/population-pharmacokinetics. Accessed January 3, 2020.

Srivastva, A. 2017. “Special Symbols in Graphs: Multiple Solutions”, PharmaSUG 2017 - Paper TT05. Available at https://www.pharmasug.org/proceedings/2017/TT/PharmaSUG-2017-TT05.pdf

SAS® 9.4 Graph Template Language: User’s Guide, Fifth Edition. Available at https://documentation.sas.com/?docsetId=grstatug&docsetTarget=titlepage.htm&docsetVersion=9.4&locale=en Accessed January 3, 2020.

ACKNOWLEDGMENTS

I would like to sincerely thank my managers Bo Wei, Shallabh Mehta and programming director Edward Elam for reviewing this paper and providing valuable inputs.

CONTACT INFORMATION

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

Min Xia
PPD
min.xia@ppdi.com