Paper 161-2013

Is the Legend in your SAS/Graph® Output Still Telling the Right Story?

Alice M. Cheng, Chiltern International Ltd, Bannockburn, IL
Justina M. Flavin, SimulStat Inc., San Diego, CA

ABSTRACT

In clinical studies, researchers are often interested in the effect of treatment over time for multiple treatments or dosing groups. Usually, in a graphical report, the measurement of treatment effect is on the vertical axis and a second factor, such as time or visit, on the horizontal axis. Multiple lines are displayed in the same figure; each line represents a third factor, such as treatment or dosing group. It is critical that the line appearance (color, symbol and style) are consistent throughout the entire clinical report, as well as, across clinical reports from related studies.

Flavin and Carpenter (2004) showed that the GPLOT procedure, by default, did not guarantee consistency in line appearances. They provided macro and non-macro solutions to this problem. With the introduction of Statistical Graphics (SG) in SAS® v9.3, there are multiple approaches to resolve this problem. In this paper, the authors cover the following topics:

- The Nature of the SGPLOT Procedure (How are Line Attributes Assigned?)
- Re-visit the Line Inconsistency Problem by means of SGPLOT Procedure
- 5 Solutions to Resolve Line Inconsistency Issues
- Discrete Attribute Map and Range Attribute Map
- Maintain Line Consistency in the SGPANEL Procedure

Note that this is the fourth edition of this paper; it is more comprehensive than the former editions in WUSS 2011, WUSS 2012 and PharmaSUG 2012. Detailed discussion on the new features in SAS® v9.3 are provided.

KEYWORDS

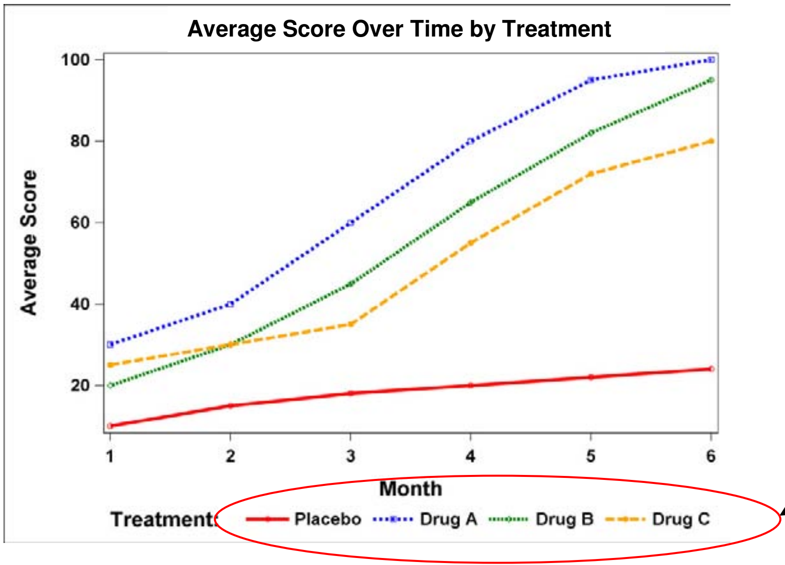
Legend, PROC SGPLOT, Graph Template Language (GTL), Index option, Attribute Maps, PROC SGPANEL

Introduction

In the analysis of clinical data, often times only a subset of data is required. For instance, a treatment group may be excluded from an exploratory analysis. Since SAS assigns line attributes (line color, symbol and style) based on the available data, a subset of the original data may result in inconsistency in the line appearance among treatment groups. This can cause confusion and make the graphic output difficult to interpret. In this paper, the authors illustrate 5 approaches, traditional and non-traditional, to resolve this line inconsistency issue.

THE NATURE OF SGPLOT (HOW ARE LINE ATTRIBUTES ASSIGNED?)

To understand how line attributes are assigned, consider Figure 1A: Average Score over Time for 4 Treatments: Placebo, Drug A, Drug B and Drug C. The result of each treatment is displayed as a line in the graph.



Based on the legend, the order of Treatment Groups are based on the order of group variable (TRT) in the input data DRUG. This also affects how treatment groups are associated with Line Styles.

In GPLOT, the default order is the ASCENDING order of the internal values of the group variable; in SGPLOT (SAS v9.2) the default is the order of group values in the input data.

Figure 1A: Average Scores of 4 Treatments are displayed according to the input DATA order in SGPLOT.

Code for Figure 1A: All 4 Treatments with Line Attributes as Defined in STYLES.MYSTYLE

```
   *--- Read in source data.                                        ---*;
   *--- Note that in input dataset DRUG, the order of treatments is: ---*;
   *--- Placebo, Drug A, Drug B and Drug C.                         ---*;
   data DRUG;
   input TRT & $7. MONTH : 2. AVG_SCORE : 3. @@;
     label TRT='Treatment:'
           MONTH='Month'
           AVG_SCORE='Average Score';
   datalines;
     Placebo 1 10  Placebo 2 15  Placebo 3 18  Placebo 4 20  Placebo 5 22  Placebo 6 24
     Drug A  1 30  Drug A  2 40  Drug A  3 60  Drug A  4 80  Drug A  5 93  Drug A  6 100
     Drug B  1 20  Drug B  2 30  Drug B  3 45  Drug B  4 70  Drug B  5 80  Drug B  6 90
     Drug C  1 25  Drug C  2 30  Drug C  3 35  Drug C  4 55  Drug C  5 70  Drug C  6 80
     ;
    run;
*--- Define Style STYLES.MYSTYLE. ---*;
Proc template;
   define style STYLES.MYSTYLE;  *- STYLES.MYSTYLE is based on STYLES.DEFAULT, -*;
      parent=STYLES.DEFAULT;     *- with some changes specified.               -*;

      *--- Re-define Styles for Line Attributes. ---*;
      style GraphData1 from GraphData1/linestyle=1 contrastcolor=red
         markersymbol='circle';        *- Line Attributes for 1st Treatment -*;
      style GraphData2 from GraphData2/linestyle=2 contrastcolor=blue
         markersymbol='square';        *- Line Attributes for 2nd Treatment -*;
      style GraphData3 from GraphData3/linestyle=3 contrastcolor=green
         markersymbol='diamond';       *- Line Attributes for 3rd Treatment. -*;
      style GraphData4 from GraphData4/linestyle=4 contrastcolor=orange
         markersymbol='circlefilled'; *- Line Attribute for 4th Treatment.  -*/
      class GraphDataDefault/linethickness=4px;

      *--- Redefine other Styles to make the graph more readable. ---*;
      classGraphFonts / 'GraphLabelFont'=("Arial", 18pt, bold)
                        'GraphValueFont'=("Arial", 15pt, bold)
                        'GraphTitleFont'=("Arial", 25pt, bold);
      style graphbackground from graphbackground / color=_undef_;
    end; run;
*--- Use STYLES.MYSTYLE for the plot. ---*;
ods listing style=STYLES.MYSTYLE;
title 'Average Score Over Time by Treatment';
proc sgplot data=DRUG;
   series y=AVG_SCORE x=MONTH / group=TRT markers;
   keylegend/nobordertitle='Treatment: ';run;
```

SAS introduces Statistical Graphics (SG) procedures in SAS v9.2. With these SG procedures, SYMBOL*n* statements in the GPLOT procedure are no longer relevant. One way to define line attributes is through the use of **style GraphData*n* from GraphData*n* statements** within the STYLE definition of PROC TEMPLATE. In addition, the association of line attributes with treatment groups in SG procedures differs from those in traditional SAS/GRAPH procedures. *For the GPLOT procedure, the association of a group variable with SYMBOLn statements is based on the internal values of the group variable in ascending order.* In other words, the group with the lowest internal values is associated line attributes for SYMBOL1; second lowest is associated with SYMBOL2, etc. *With SAS v9.2, the order of treatments in the SGPLOT procedure is based on the order of the treatments listed in the input dataset.* With SAS v9.3, one can use the following statement to ensure that the treatments are displayed in ascending order.

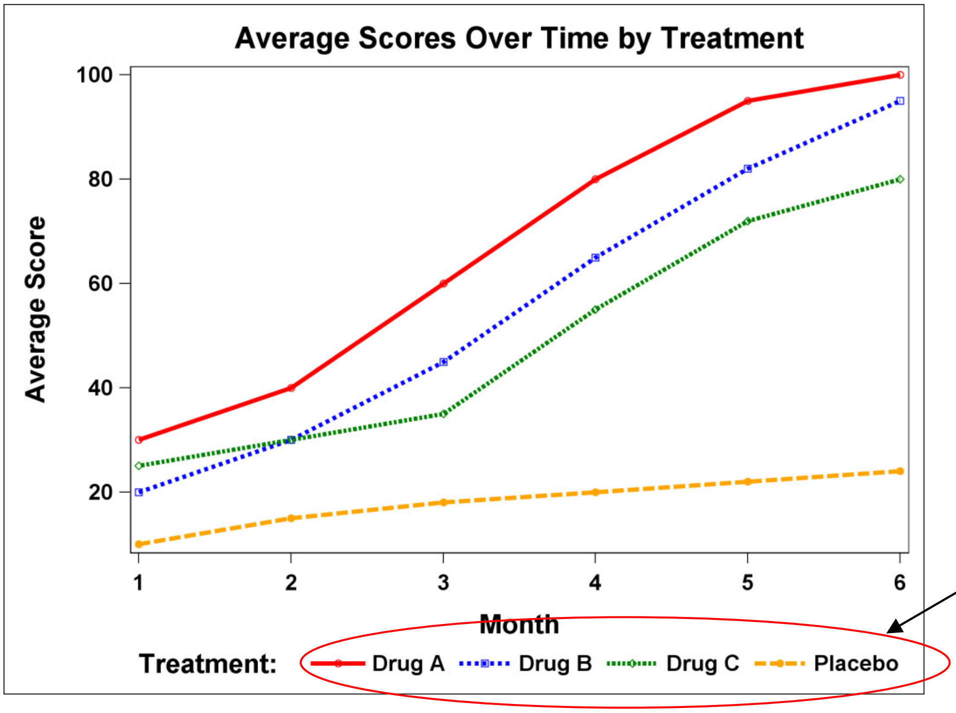
```
series y=AVG_SCORE x=MONTH / group=TRT grouporder=ascending markers;
```

GROUP=*variable* and GROUPORDER = *ascending* | *descending*| *data* options in the SERIES statement specify the ordering of lines based on the value of the group variable. Since this feature is not available prior to SAS v9.3,SAS v9.2 users would have to sort the data based on the internal values of the group variables. It is advisable to explicitly state the value for GROUPORDER option. To be consistent with the GPLOT default results, the authors have sorted the data in ascending order of the variable TRT and produced Figure 1B.

Code for Figure 1B: Sort the data by Treatment Group. Then use SGPLOT procedure.

```
*--- Sort the data by treatment group variable (TRT) so that lines are displayed ---*;
*--- by its internal values, instead of its order in the input dataset.          ---*;
proc sort data=DRUG out=DRUG_BY_TRT;
    by TRT;
  run;

*--- Produce Figure 1B based on the order of the internal value of TRT. ---*;
ods listing style=STYLES.MYSTYLE;
title 'Score Over Time by Treatment';
proc sgplot data=DRUG_BY_TRT;
   series y=AVG_SCORE x=MONTH / group=TRT markers;
   keylegend/noborder title='Treatment: ';
run;
```



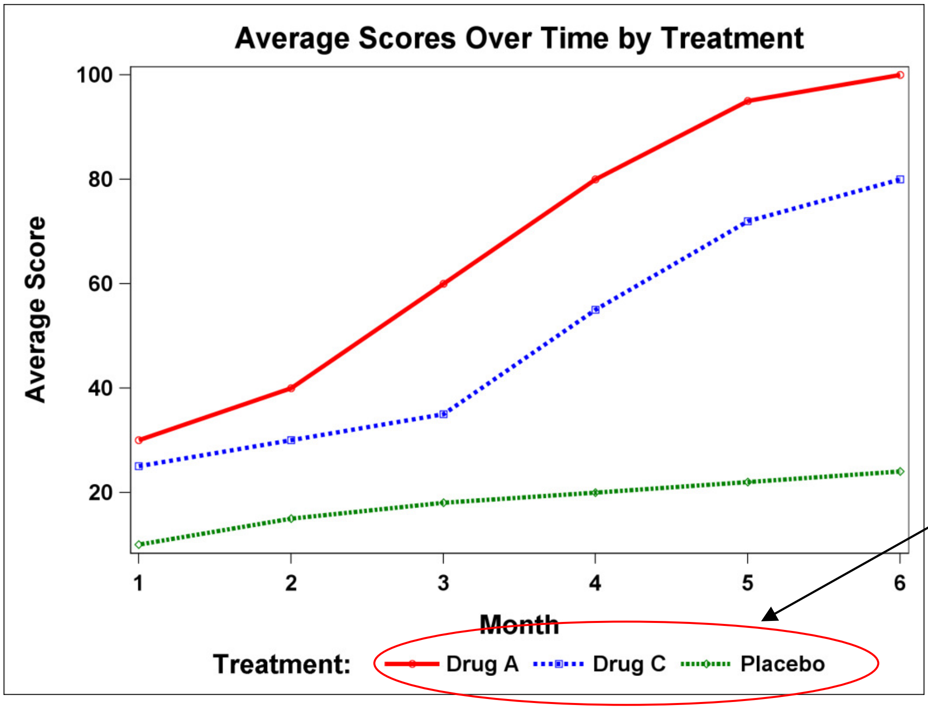
The order here reflects the internal values of the group valuable, TRT. The line attributes are assigned based on this order.

Figure 1B: After sorting by the group variable TRT, the order of treatments in the legend and the line attributes have been changed.

After sorting by group variable TRT, the default order based on data is now the same as the ASCENDING order of treatment groups, This matches the default order of PROC GPLOT. Note that the line attributes for each treatment group in Figure 1B differ from those in Figure 1A.

THE PROBLEM

SGPLOT output if one treatment, say Drug B, is excluded from analysis? In Figure 2, Drug B has been excluded. Note without Drug B, Drug A still has the lowest internal value, namely, ‘Drug A’. Hence, the line attributes for Drug A remain intact. However, Drug C and Placebo, having the second and third lowest internal values, take on the second and third line attributes, instead of the third and fourth line attributes as they did in the full analysis. Inconsistent line attributes are confusing and misleading. It makes interpretation of graphic results difficult as illustrated in Figure 2 below.



Errors!!!
Inconsisitency in line attributes. Drug C should have been **GREEN** and Placebo should have been**ORANGE**!

Figure 2: The same line inconsistency issue in the GPLOT procedure now re-appears in the output for SGPLOT procedure when Drug B is excluded.

Code for Figure 2: SGPLOT Procedure

```
*--- Subset data ordered by TRT to exclude Drug B.  ---*;
Data DRUG_noB;
    set DRUG_BY_TRT (where=(TRT ne ‘Drug B’));
  run;

*--- Produce Figure 2 without Drug B.                ---*;
*--- Order is still based on the internal values of TRT. ---*;
ods listing style=STYLES.MYSTYLE;
title 'Average Score Over Time by Treatment';
proc sgplot data=DRUG_noB;
   series y=AVG_SCORE x=MONTH / group=TRT markers;
   keylegend/noborder title='Treatment: ';
run;
```

Flavin and Carpenter (2004) addressed this issue with the GPLOT procedure. They provided macro and non-macro solutions to the line inconsistency problem. So will the same solutions for the GPLOT procedure be applicable to the problem in the SGPLOT procedure?

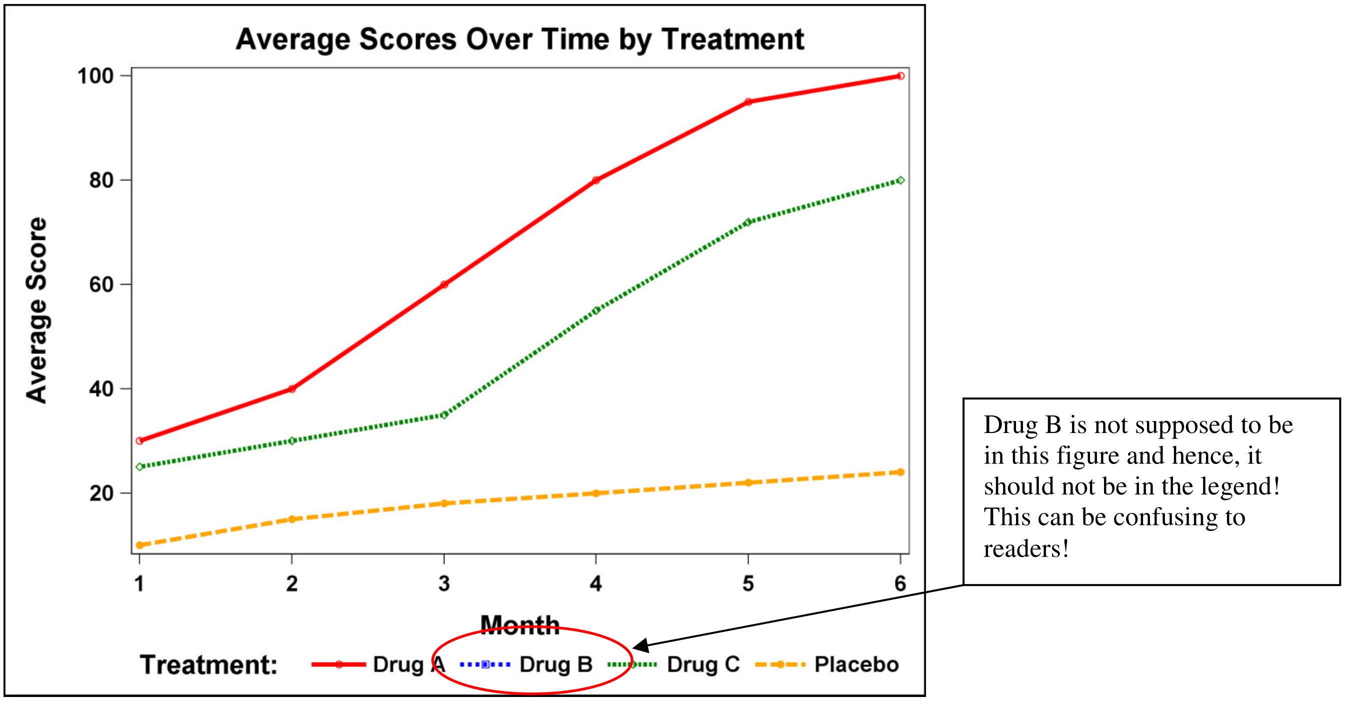
THE SOLUTIONS

5 solutions are provided below. The first 2 are analogous to the solutions suggested by Flavin and Carpenter (2004) to resolve this issue in the GPLOT procedure. A third solution is suggested by the author, Alice Cheng and is also applicable to both the GPLOT and SGPLOT procedures. The last 2 approaches take advantages of new features in Statistical Graphics for SAS v9.2 and beyond.

1. Use Dummy Data

Will the Dummy Data Approach still work for the SGPLOT procedure? In SAS v9.2, treatment groups are to be displayed according to the input data order for treatment groups. So it is important that after adding the dummy data to the original input data set, the combined data set will retain the same DATA order for treatment. While the line attributes now appear to be consistent, the legend still displays the eliminated treatment, Drug B. Same happens with this approach using the

GPLOT procedure. This is appropriate if the intention is to remind readers that there used to be a Drug B, or the intention of the graph is for internal use or for exploratory analysis; otherwise, the Dummy Data Approach is less than desirable! See Figure 3A, an output from the Dummy Data Approach.



Drug B is not supposed to be in this figure and hence, it should not be in the legend! This can be confusing to readers!

Figure 3A: The Add Dummy Data Approach ensures line consistency. However, the excluded treatment group Drug B remains in the legend.

Code for Figure 3A: (Add Dummy Data Approach.)

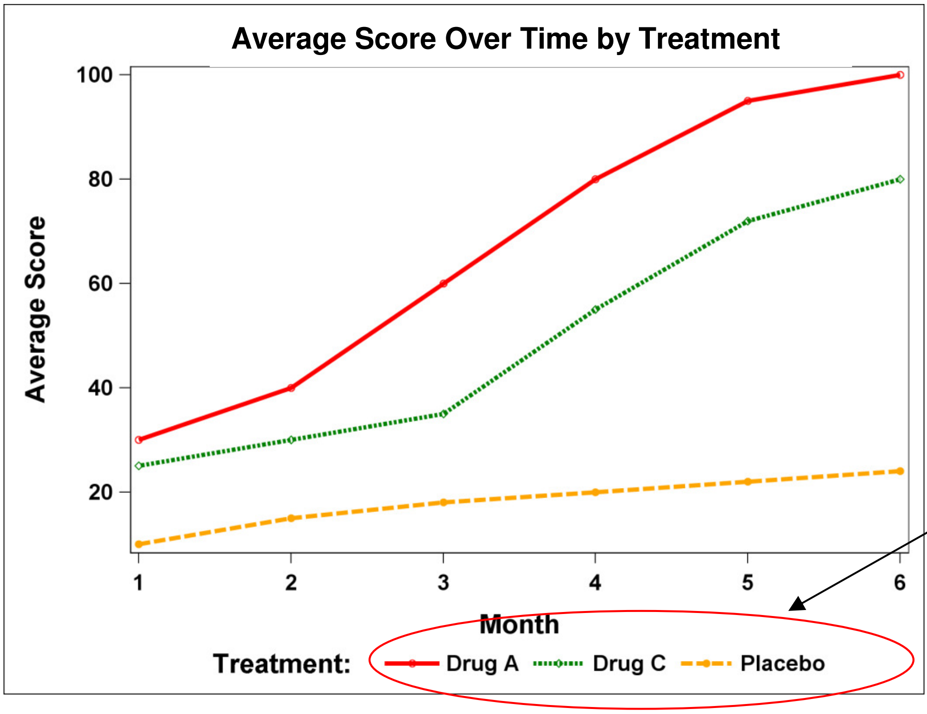
```
*- Create DUMMY dataset so that all treatment groups are represented.-*;
data DUMMY;
  length TRT $7.;
  TRT='Drug A';  MONTH=.; AVG_SCORE=.; output;
  TRT='Drug B';  MONTH=.; AVG_SCORE=.; output;
  TRT='Drug C';  MONTH=.; AVG_SCORE=.; output;
  TRT='Placebo'; MONTH=.; AVG_SCORE=.; output;
run;
*--------------------------------------------------------------------*;
* Note:  There is no need to sort DRUG_WT_DUMMY because:             *;
* As long as Drug A, Drug B, Drug C and Placebo are ordered first,   *;
* second, third and fourth in INPUT data, sorting is not necessary.  *;
* DUMMY dataset has already had the correct order.                   *;
*--------------------------------------------------------------------*;

DATA DRUG_WT_DUMMY;
set DUMMY
    DRUG_noB;
run;

*--- Generate Figure 3A.                                 ---*;
*--- STYLES.MYSTYLE is as defined in Code for Figure 1A. ---*;
ods listing style=STYLES.MYSTYLE;
title 'Average Score Over Time by Treatment';
proc sgplot data=DRUG_WT_DUMMY;
   series y=AVG_SCORE x=MONTH / group=TRT markers;
   keylegend/noborder title='Treatment: '; run;
```

2. Re-map Line Attributes

Flavin and Carpenter (2004) provided detailed examples on how to dynamically modify the SYMBOL*n* statements to ensure line consistency when using the GPLOT procedure. Their approach relies heavily on %SCAN to re-map the treatment attributes to the correct treatment. The same approach can be used to modify the **style GraphData*n* from GraphData*n*statements** within the TEMPLATE procedure. Here, the authors of this paper illustrate how this can also be accomplished by means of a macro named %lineattr. Instead of %scan, CALL SYMPUT will be applied.



Note: Even without Drug B, the 3 other treatments still have the same line attributes. This is the goal of this paper!

Figure 3B: 3 remaining treatments are represented by lines consistent in appearance as those seen in Figure 1B. Legend reflects only the lines displayed in the figure.

Code for Figure 3B: (Re-map Line Attribute Approach)

```
*- Establish a LOOKUP dictionary for line attribute for each drug. -*;
data LOOKUP;
input ID & $7. LINESTYLE : 3. COLOR : $6. SYMBOL : $12.;
cards;
Drug A    1  red     circle
Drug B    2  blue    square
Drug C    3  green   diamond
Placebo   4  orange  circlefilled
; run;

*- Define LINEATTR macro to create Style Template with line attribute. -*;
%macro lineattr (indata=        /* Enter Input Data Set Name.           */
                ,lookup=LOOKUP  /* Enter Lookup Data Set.               */
                ,idvar=ID       /* ID variable in Lookup Data Set.      */
                ,groupvar=TRT   /* Enter Group Variable.                */
                ,style=STYLES.MYSTYLE2 /* Enter Style name to be output */
                ,parent_style=STYLES.MYSTYLE  /* Enter Parent Style.    */
                )/des='To create line attributes in Style Template';
  %local i;
*--- Get the line attribute for treatment groups from LOOKUP.           ---*;
*--- Note: only line attributes for the treatment groups in the dataset ---*;
*--- specified in &indata will be kept.                                  ---*;
```

Code for Figure 3B: (Re-map Line Attribute Approach) (Continued)

```
Proc sql noprint;
    create table LINEATTR as
      select distinct L.*, D.&GROUPVAR
         from &indata D
               left join
              &LOOKUP L
             on D.&GROUPVAR = L.&IDVAR
                 order by D.&GROUPVAR;
quit;
*- Assign value for the line style, colors and symbol of each treatment.-*;
  data _null_;
    set LINEATTR;
    call symput('LINESTYLE'||strip(put(_n_, 3.)), LINESTYLE);
    call symput('COLOR'||strip(put(_n_, 3.)), COLOR);
    call symput('SYMBOL'||strip(put(_n_, 3.)), strip(SYMBOL));
  run;

*--- Re-define your line attributes in a new style. ---*;
proc template;
     define style &STYLE;
       parent=&parent_style;
         %do i=1 %to &sqlobs;
           style GraphData&i from GraphData&i/linestyle=&&LINESTYLE&i
           contrastcolor=&&COLOR&imarkersymbol="&&SYMBOL&i";
         %end;
      end;
   run;
  %mend lineattr;
*--- Invoke %lineattr to create STYLES.MYSTYLES2 with the correct line styles. -*;
*--- Generate line plot using this new style.                                  -*;
%lineattr(indata=DRUG_noB, style=STYLES.MYSTYLE2);

ods listing style=STYLES.MYSTYLE2;
title h=20pt 'Average Score Over Time by Treatment';
proc sgplot data=DRUG_noB;
   series y=AVG_SCORE x=MONTH / group=TRT markers;
   keylegend/noborder title='Treatment: '; run;
```

3. Re-Code Internal Values of Group Variable

In Figure 1B, Placebo is listed last in the legend because its internal value ‘Placebo’ is behind the internal values of other treatments, namely, ‘Drug A’, ‘Drug B’ and ‘Drug C’. If Placebo, instead of Drug B, has been excluded from that figure, the line appearance for the remaining groups would have been unaffected. Hence, a third solution is to re-code so that the treatment groups to be excluded would have higher group order than those for the groups that are to stay. When using PROC GPLOT, the group order is the internal value of the group variable. For PROC SGPLOT in SAS v9.2, the group order is based on the order of the group variable in the input data. As an example, please consider the following code which can also generate Figure 3B.

Code for Figure 3B (Using Re-code Internal Values Approach)

```
*--- Define formats for re-coding group variable, TRT. ---*;
proc format;
invalue INTRTF ‘Drug A’ = 1
               ‘Drug B’ = 4
               ‘Drug C’ = 2
               ‘Placebo’= 3;
     value TRTF 1 = ‘Drug A’
                2 = ‘Drug C'
                3 = ‘Placebo’
                4 = ‘Drug B’; run;
```

Code for Figure 3B (Using Re-code Internal Values Approach) (continued)

```
*-- Re-coded the value from TRT to TRT2 so that treatment groups to be dropped ---*;
*--- have higher internal values and will be listed last.                      ---*;
data DRUG_RECODED;
  set DRUG_noB;
  TRT2=input(TRT, intrtf.);
  label TRT2=”Treatment: “;
  format TRT2 TRTF.;
run;

*--- Sort the data based on the internal value of the re-coded variable TRT2.  ---*;
proc sort data=DRUG_RECODED out=DRUG_RECODED;
    by TRT2; run;

*- Re-order line attributes to match internal values of the group variable, TRT2.-*;
proc template;
  define style STYLES.MYSTYLE3;
    parent=STYLES.MYSTYLE;
    style GraphData1 from GraphData1/linestyle=1 contrastcolor=red
                                     markersymbol='circle';
    style GraphData2 from GraphData2/linestyle=3 contrastcolor=green
                                     markersymbol='diamond';
    style GraphData3 from GraphData3/linestyle=4 contrastcolor=orange
                                     markersymbol='circlefilled';
    style GraphData4 from GraphData4/linestyle=2 contrastcolor=blue
                                     markersymbol='square';
     end; run;

*- Generate Figure 3B that excludes Drug B with Re-coded variable as the group -*;
*- variable.                                                                   -*;
ods listing style=STYLES.MYSTYLE3;
title ‘Average Score Over Time by Treatment’;
proc sgplot data=DRUG_RECODED;
   series y=AVG_SCORE x=MONTH/group=TRT2 markers;
   keylegend/noborder title='Treatment: ';
run;
```

As seen in the Code for Figure 3B, the variable TRT has been re-coded to create another variable, TRT2, which has internal values of 1, 2, 3 and 4. Moreover, TRT2=1 represents Drug A; TRT2=2 represents Drug C; TRT2=3 represents Placebo and TRT2=4 represents Drug B. The legend in Figure 3B is displayed in the order of the input data for TRT2 (which is the same order as the internal value of TRT2) and displayed in TRTF. format. **Style GraphData*n* from GraphData*n* statements** have been re-arranged so that line attributes for the treatments are consistent with the internal values of TRT2. This new style STYLES.MYSTYLE3 is used to create Figure 3B, an ideal graph based on re-coding Internal Value approach. Note that this method can also be applied to PROC GPLOT, in which case **the SYMBOL*n* statements** will be modified.

4. Use INDEX Option in Graph Template Language (GTL)

Graph Template Language (GTL) empowers SAS users with the ability to create high-quality sophisticated statistical graphics with relative ease. While the SGPLOT procedure provides a quick and easy solution to single-cell plot, GTL is designed to create complex multi-cell graphics with overlay of multiple plots. With GTL, one first defines a graphic template to be compiled. Then the graph is rendered by means of the SGRENDER procedure. For the line inconsistency issue, the **INDEX option** in the SERIESPLOT statement of the GTL offers yet another solution.

Consider the code in the following box for this GTL approach. Users first have to create a variable, namely, **DRUG_ID**, to be specified in **the INDEX option of the SERIESPLOT statement**. The values in this DRUG_ID variable are associated with the **style GraphData*n* from GraphData*n* statements** in the statgraph definition. This is a relatively painless approach to resolve the line consistency issue. Unfortunately, the INDEX option is not available in the SGPLOT procedure. If the GTL code appears complex and formidable, don’t panic. If a graph can be created by means of SGPLOT procedure, one can easily use **TMPLOUT option** in SGPLOT to output the corresponding GTL code. Then simply add INDEX option in the SERIESPLOT statement to create the graph template. This template can then be rendered to create a graph without the line inconsistency issue and hopefully, without tears from the developer!

Code for Figure 3B (Using INDEX Option in Graph Template Language (GTL) Approach)

```
data DRUG_IDVAL;
  set DRUG_noB;
  *--- Create variable DRUG_ID to associate the group variables with ---*;
  *--- line attributes defined in STYLES.MYSTYLE.                    ---*;
  if TRT='Drug A' then DRUG_ID=1;
     else if TRT='Drug B'  then DRUG_ID=2;
     else if TRT='Drug C'  then DRUG_ID=3;
     else if TRT='Placebo' then DRUG_ID=4;
run;

proc template;
*--- Template SGPLOT was obtained from TMPLOUT options in PROC SGPLOT.  ---*;
*--- INDEX options in the SERIESPLOT statement has been added manually. ---*;
 define statgraph SGPLOT;
   begingraph;
     EntryTitle "Score Over Time by Treatment" / textattrs=(size=2pct);
     layout overlay;
        SeriesPlot X=MONTH Y=AVG_SCORE /
                   primary=true group=TRT index=DRUG_ID display=(markers)
        LegendLabel="Average Score" NAME="SERIES";
        DiscreteLegend "SERIES" / Location=Outside Title="Treatment: "
                                  Border=false;
      endlayout;
     endgraph;
   end;
run;

*--- Use PROC SGRENDER to render the graph from the stored graphics ---*;
*--- template SGPLOT.                                                ---*;
ods listing;
proc sgrender data=DRUG_IDVAL template=SGPLOT;
run;
```

5. Use of SG Discrete Attribute Map (Available in SAS Version 9.3)

SAS Institute released the 9.3 version of its product on July 11, 2011, its 35th Anniversary Date. This new release has a powerful feature to ensure consistency in line appearance --- Attribute Map. Attribute map is applicable to both SG procedures and GTL, though the coding in SG procedures differs from that using GTL.

5A. SG Discrete Attribute Map

With the SG Attribute Map dataset, users specify ID variable values, group variable values and line attributes in one dataset. This dataset and its ID variables are then referred to in the SGPLOT procedure. As an illustration, the authors provide the following code to demonstrate yet another effortless method to produce Figure 3B in the brave new world of SAS v9.3.

In MYATTRMAP dataset, all 4 observations have ID='DRUG_ID'. These are observations with data for line attributes of different treatment/drug groups. Of course, the ID variable in the MYATTRMAP dataset can contain more that one values. Say, in addition to treatment groups, users are also interested in different dosing levels. Then additional observations with data for line attributes can be added in MYATTRMAP dataset with ID="DOSE_ID". Note that to create a graph with the appropriate line attributes, the SGPLOT procedure is associated with **the dattrmap=MYATTRMAP option**, and the SERIES statement has **ATTRID=a *value in ID variable*** option. In the current example, since the interest is in treatment groups, ATTRID=DRUG_ID option is applied to the SERIES statement.

Code for Figure 3B (Using SG Attribute Map Approach)

```
*-- Create Attribute Map Data Set named MYATTRMAP. --*;
data  MYATTRMAP;
   retain ID "DRUG_ID";
input VALUE $7. @10 LINEPATTERN $3. @14 LINECOLOR $6. @23 MARKERSYMBOL $12.
      @38 markercolor $6.;
cards;
Drug A    1  red     circle        red
Drug B    2  blue    square        blue
Drug C    3  green   diamond       green
Placebo   4  orange  circlefilled  orange
;
run;

*--- Reference to MYATTRMAP to define the line attributes. ---*;
ods listing;
title 'Score Over Time by Treatment';
proc sgplot data=DRUG_noB dattrmap=MYATTRMAP;
   series Y=AVG_SCORE X=MONTH / group=TRT attrid=DRUG_ID marker;
keylegend/noborder title='Treatment: '; run;
```

5B. Discrete Attribute Map with GTL

For more sophisticated graphics, such as multi-cell graphs, most likely GTL will be applied. In that event, attribute maps are also available in GTL. Below is an example of how Figure 3B can be produced by means of discrete attribute map.

Code for Figure 3B (Using Discrete Attribute Map with GTL Approach)

```
*-- Define a template named SGPLOT with Discrete Attribute Map Data Set --*;
proc template;
   define statgraph SGPLOT;
      begingraph;
        Entrytitle “Average Score Over Time by Treatment” / textattrs=(size=2 pct);
        *-- Define the attribute map and assign the name “lineattr”. --*;
        DiscreteAttrMap name=”lineattr” / ignore=true;
          value “Drug A”  / lineattrs=(pattern=1 color=red)
                            markerattrs=(color=red symbol=circle);
          value “Drug B”  / lineattrs=(pattern=2 color=blue)
                            markerattrs=(color=blue symbol=square);
          value “Drug C”  / lineattrs=(pattern=3 color=green)
                            markerattrs=(color=green symbol=diamond);
          value “Placebo” / lineattrs=(pattern=4 color=orange)
                            markerattrs=(color=orange symbol=circlefilled);
        EndDiscreteAttrmap;
      *-- Associate the attribute map with input data column ID and --*;
      *-- assign the name GROUPMARKERS to named association.         --*;
      DiscreteAttrVar attrvar=group markersvar=TRT attrmap=”lineattr”;
      *-- Define the series plot. --*;
      Layout overlay;
|       Seriesplot x=MONTH y=AVG_SCORE / primary=true group=groupmarkers
                                         display=(markers)
                                         Legendlabel=”Average Score”
                                         Name=”SERIES”;
        DiscreteLegend “SERIES”/Location=Outside border=false
                       Title=”Treatment: “;
      Endlayout;
    Endgraph; end; run;
```

Code for Figure 3B (Using Discrete Attribute Map with GTL Approach) (Continued)

```
*Use PROC SGRENDER to render the graph from stored graphics template SGPLOT.*;
ods listing;
proc sgrender data=DRUG template=SGPLOT;
   where TRT ne “Drug B”;
run;
```

So which method should be used?

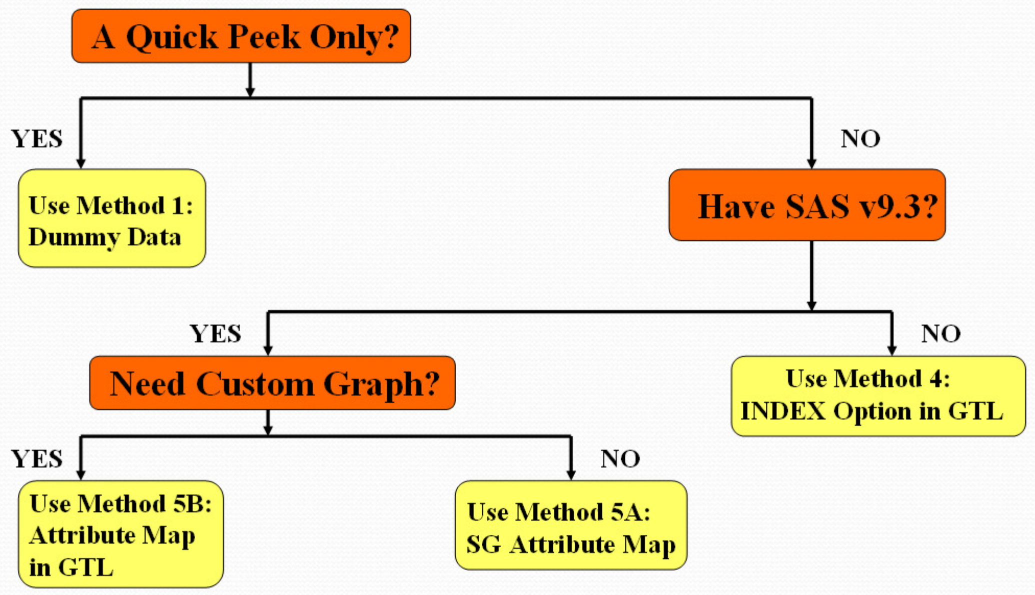


Diagram 1: Which of the 5 approaches should be applied?

A Quick Peek Only?
YES
NO
Use Method 1: Dummy Data
Have SAS v9.3?
YES
NO
Need Custom Graph?
Use Method 4: INDEX Option in GTL
YES
NO
Use Method 5B: Attribute Map in GTL
Use Method 5A: SG Attribute Map

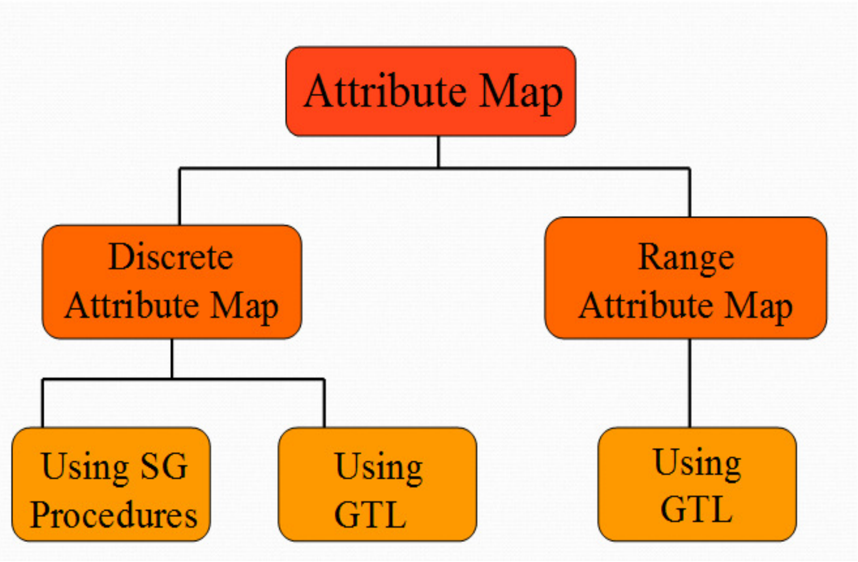
So which of the 5 aforementioned methods should one use? It really depends on the purpose of your graphs, the version of SAS available to you and the degree of customization needed. For a quick peek at the data or for exploratory analysis, Method 1 (Dummy Data Approach) seems to be nice and easy to use, despite it displays the eliminated group in the legend. If you have only SAS v9.2, you may want to consider Method 4 (Index Option in GTL). If SAS v9.3 is available and you don’t need much customization, Method 5A (SAS Attribute Map) should be your choice. For those with SAS v9.3 and need more customization of your graph, Method 5B (Attribute Map in GTL) will be an ideal solution. The authors would not recommend Methods 2 (Re-Map Line Attributes Approach) and 3 (Re-code Internal Values Method) because both methods need re-coding and are not taking advantages of the new features in Statistical Graphics. In addition, Method 3 is not very practical because often times, one will not know which treatment will be dropped in the future.

ADDITIONAL COMMENTS

Range Attribute Map (Available in SAS Version 9.3)

The attribute maps we have discussed so far are discrete attribute maps. For discrete attribute maps, discrete values are associated with certain attributes. In our previous examples, the distinct values are the 4 treatment groups. We have also seen that for discrete attribute maps, we can use either SG procedures or GTL to produce the output. However, clinical data can be also continuous and grouped into categories. For instance, based on a subject’s body mass index (BMI) at baseline, a subject can be classified as underweight (< 19 kg/m^2), normal (19 - <26 kg/m^2), overweight (26 - <30 kg/m^2) and obese (≥ 30 kg/m^2). In the following example, the authors have provided an illustration of the range attribute map by adding a scatterplot of individual scores at different time points. The colors of the dots in the scatterplot are based on the BMI at baseline of the corresponding subject. (Please see Figure 4.) Note that as of SAS v9.3, range attribute maps are only available in GTL and not in SG procedures.

Diagram 2: Two Types of Attribute Maps



Example for Range Attribute Map

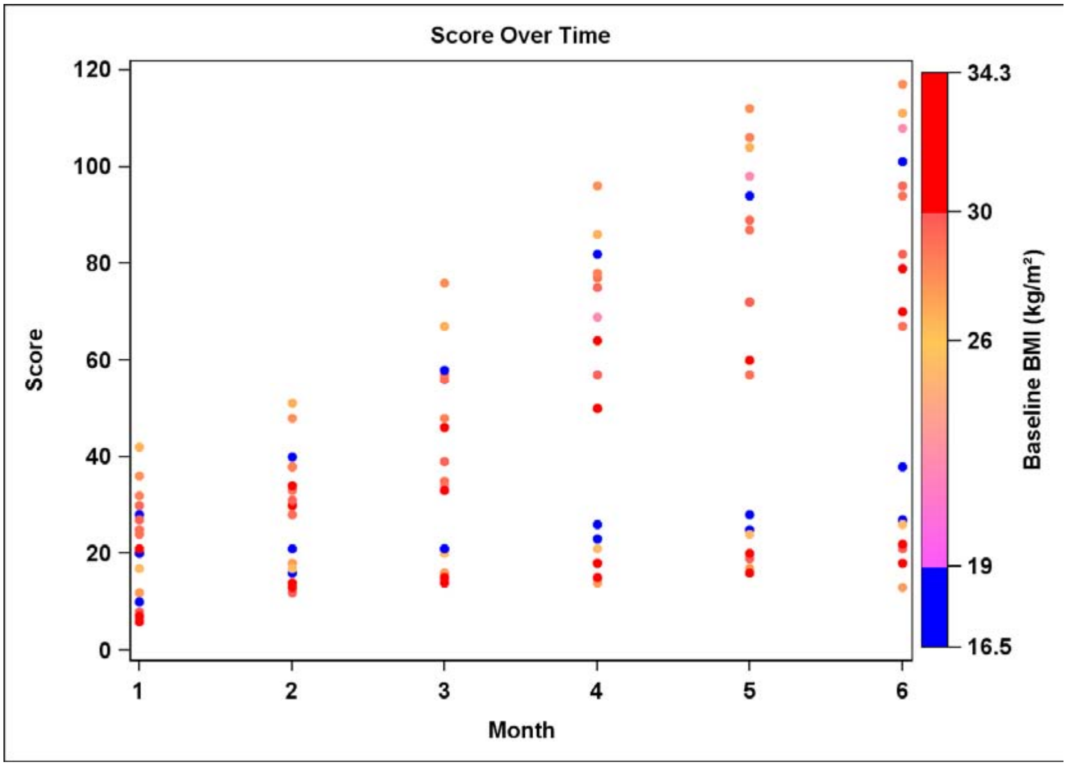


Figure 4: A scatterplot of score over month with the color of the dots representing the value of BMI at baseline.

Code for Figure 4 (Using Range Attribute Map with GTL Approach)

```
data SCORE;
   input USUBJID 1-3 TRT $ 7-13 @16 BBMI 4.1 @24 MONTH 1. SCORE 28-30;
   label BBMI='Baseline BMI(kg/m^{unicode ‘00b2’x})';
 datalines;
 100   Placebo  29.8    1     8
 100   Placebo  29.8    2    12
 100   Placebo  29.8    3    16
 … more data …;
 run;
*--- Define Style STYLES.MYSTYLE4. ---*;
proc template;
  define style STYLES.MYSTYLE4;
    parent=STYLES.DEFAULT;  *- STYLES.MYSTYLE is based on STYLES.DEFAULT, -*;
                            *- with some changes specified.                -*;

       *--- Re-define other Styles to make the graph more readable. ---*;
       class GraphFonts / 'GraphLabelFont'=("Arial", 10pt, bold)
                          'GraphValueFont'=("Arial", 10pt, bold)
                          'GraphTitleFont'=("Arial", 10pt, bold);
       style graphbackground from graphbackground / color=_undef_;
   end;
run;

*--- Define Statgraph template SGPLOT_SCAT. ---*;
ods escapechar='^';
proc template;
  define statgraph SGPLOT_SCAT;
    begingraph;
      entrytitle 'Score Over Time'/ textattrs=(size=10pt);

      *-- Define the attribute map and assign the name “BBMI_Range” --*;
      RangeAttrMap name="BBMI_Range";
        range MIN - < 19 / rangecolor=blue;
        range 19 - < 26  / rangecolormodel=(lightpurple lightorange);
        range  26 - <30  / rangecolormodel=(lightorange lightred);
        range  30 -  MAX / rangecolor=red;
      EndRangeAttrMap;

      *-- Associate the attribute map with the variable BBMI and --*;
      *-- assign variable name RANGEVAR for this association.    --*;
      RangeAttrVar attrvar=rangevar var=BBMI attrmap="BBMI_Range";
      layout overlay;
         scatterplot x=MONTH y=SCORE / markercolorgradient=rangevar
           markerattrs=(symbol=circlefilled size=6px)name="scatter";
           continuouslegend "scatter" / orient=vertical halign=right
              title="Baseline BMI (kg/m^{unicode '00b2'x})";
      endlayout;
     endgraph;
   end;
run;

*--- Render the scatterplot by using STYLES.MYSTYLE4 and the SGPLOT_SCAT ---*;
*--- template.                                                           ---*;
ods graphics on /height=5in width=7in;
ods rtf file="c:\WUSS2012\correction2.rtf";
ods rtf style=STYLES.MYSTYLE4; *--- Use STYLES.MYSTYLE4 for the plot. ---*;

proc sgrender data=SCORE template=SGPLOT_SCAT;
run;
ods rtf cloae;
```

MAINTAIN LINE CONSISTENCY IN THE SGPANEL PROCEDURE

In early phase clinical trials, researchers are often interested in various lab test results of individual subjects over time. Since subjects in early phase clinical trials are few, such results can be nicely displayed in a panel of graphic cells. Each of these graphics cells contains data for a subject, as seen in Figures 5A and 5B below.

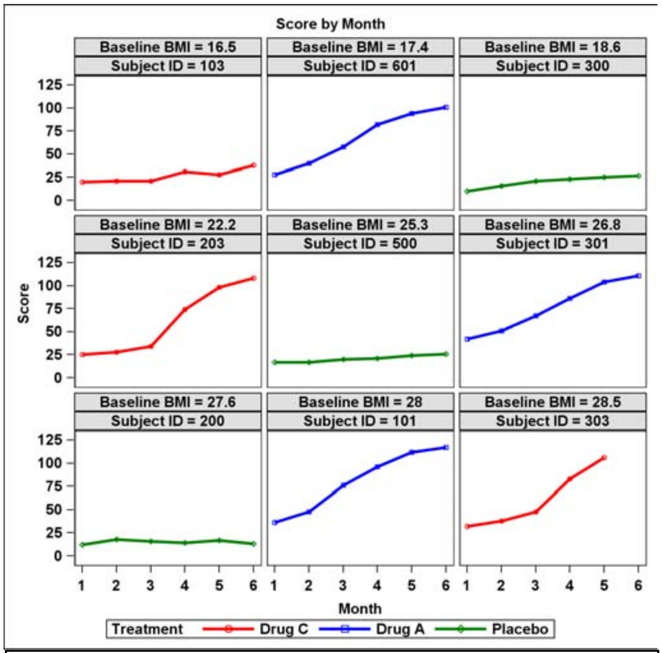


Figure 5A. Lines are inconsistent with Figure 3B. Treatment attributes are based on the order of the PANELBY variables (Baseline BMI and Subject ID) and Treatment in input data.

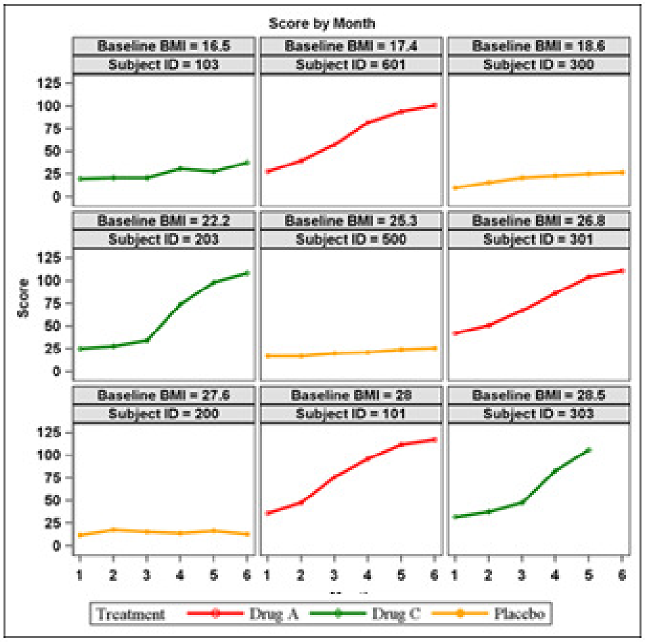


Figure 5B. Lines are consistent with Figure 3B after applying Discrete Attribute Map in PROC SGPANEL or Discrete Attribute Map within GTL and then rendered by PROC SGRENDER.

In Figures 5A and 5B, cells are ordered from left to right and from top to bottom according to the values of Baseline BMI and Subject ID. In this example, Baseline BMI and Subject ID are the PANELBY variables in the SGPANEL procedure. Since there is only one treatment per subject in this example, each cell can be regarded as having the rest of the treatment groups missing. In Figure 5A, no effort has been made to ensure line consistency. Based on the value of Baseline BMI and Subject ID, Subject 103 is the first subject; hence, the treatment for this subject, Drug C is assigned to the first set of line attributes defined by STYLES.MYSTYLE, namely a red line of linestyle 1 and a circle marker. By the same token, the second subject, the Subject 601, took Drug A and hence, Drug A is assigned to the second set of line attributes from STYLES.MYSTYLE (blue color, linestyle=2 and square symbol). Subsequent set of line attributes will be assigned to each treatment group when SAS first encountered a subject in that treatment group. Note the set of line attributes in Figure 5A is not consistent with the correct set of line attributes seen in Figure 3B. This, however, can easily be fixed by the attribute map methods (Method 5A and 5B) as we saw in the examples for the SGPLOT procedures.

Code for Figure 5A (The SGPANEL Procedure by Default)
To Create Multi-Cell Graph with Inconsistency in Line Attributes

```
ods _all_ close;
ods escapechar='^';

ods listing style=STYLES.MYSTYLE;

ods rtf file="C:\SGF_2013\Figure5A.rtf";
title h=20pt 'Score by Month';
proc sgpanel data=SCORE; *Note: SCORE is the same dataset in Range Attribute Map Example.*;
    panelby BBMI USUBJID / layout=panel columns=3 rows=3 colheaderpos=top
                           rowheaderpos=right uniscale=row start=topleft spacing=5;
    series x=MONTH y=SCORE /group=TRT  attrid=MYID name='SCORE'
           lineattrs=(pattern=solid thickness=3) markers;
    colaxis values=(1 2 3 4 5 6) label='Month';  rowaxis  label='Score'; run;
```

Code for Figure 5B (The SGPANEL Procedure Using Attribute Map)
To Create Multi-Cell Graph with Consistency in Line Attributes

```
*-- Create Attribute Map Data Set named MYATTRMAP. --*;
data  MYATTRMAP;
   retain ID "DRUG_ID";
input VALUE $7. @10 LINEPATTERN $3. @14 LINECOLOR $6. @23 MARKERSYMBOL $12.
      @38 markercolor $6.;
cards;
Drug A    1  red     circle        red
Drug B    2  blue    square        blue
Drug C    3  green   diamond       green
Placebo   4  orange  circlefilled  orange
; run;

ods _all_ close;
ods escapechar='^';
ods listing style=STYLES.MYSTYLE;

ods rtf file="C:\SGF_2013\Figure5b_1.rtf";


title h=20pt 'Score by Month';
proc sgpanel data=SCORE dattrmap=MYATTRMAP;
    panelby BBMI USUBJID / layout=panel columns=3 rows=3 colheaderpos=top
                           rowheaderpos=right uniscale=row start=topleft spacing=5;
    series x=MONTH y=SCORE /group=TRT  attrid=DRUG_ID name='SCORE'
           lineattrs=(pattern=solid thickness=3) markers grouporder=ascending;
    colaxis values=(1 2 3 4 5 6) label='Month';  rowaxis  label='Score';

run;
```

There are several interesting points in regard to the code above to create Figure 5B. (1) Note that an Attribute Map Dataset named MYATTRMAP has been created. This attribute map dataset was referenced in the SGPANEL procedure through **dattrmap=MYATTRMAP option**. (2) The value of the ID variable in MYATTRMAP dataset is referenced by **attrid=DRUG_ID** in the SERIES statement. (3) Despite lines are defined within STYLES.MYSTYLE, the attribute map defined within the SGPANEL procedure has taken precedence over the definition within STYLES.MYSTYLE referred to here by the ods rtf statement. (4) **grouporder=ascending** is essential to ensure that within the treatment listed in the legend is in the ascending order (i.e. in the order of Drug A, Drug C, Placebo, as seen in Figure 5B), instead of in the order decided by the data (i.e., the order of Drug C, Drug A, Placebo, as seen in Figure 5A).

Another approach to create Figure 5B is by means of GTL and SGRENDER procedure. The GTL code may appear to be formidable at first glance. However, it follows the basic GTL structure and can allow more customization in the graph. (Please see the box below.) Unfortunately, the TMPLOUT option, which outputs GTL code from the SGPLOT procedure, is not supported in the SGPANEL procedure in SAS v9.3. As a result, users cannot simply output the GTL code for the SGPANEL procedure and perform some slight modification to the code. Users have to supply their own code.

Basic Graph Template Language (GTL) Structure

```
proc template;
   define statgraph template-name;
     begingraph / < designheight=h >
       <designwidth=w> <options>;
            GTL - global-statements;
            GTL - layout-block;
     endgraph;
  end;
run;
```

(From Graph Template Language Tip Sheet, listed in the reference.)

Code for Figure 5B (Use of Discrete Attribute Map with the GTL)
To Create Multi-Cell Graph with Consistency in Line Attributes

```
proc template;
  define statgraph SGPANEL;
  dynamic _xviewmin_ _xviewmax_ _yviewmin_ _yviewmax_;
  dynamic _panelnumber_ _byline_;
    begingraph / designwidth=640 designheight=640;
    EntryTitle "Score by Month" / textattrs=( size=10pt);

    *-- Define the attribute map and assign the name "lineattr". --*;
      DiscreteAttrMap name="lineattr" / ignorecase=true;
         value "Drug A" / lineattrs=(pattern=1 color=red)
                          markerattrs=(color=red symbol=circle);
         value "Drug B" / lineattrs=(pattern=2 color=blue)
                          markerattrs=(color=blue symbol=square);
         value "Drug C" / lineattrs=(pattern=3 color=green)
                          markerattrs=(color=green symbol=diamond);
         value "Placebo"/ lineattrs=(pattern=4 color=orange)
                          markerattrs=(color=orange symbol=circlefilled);
      EndDiscreteattrmap;

    *-- Associate the attribute map with input data column ID and --*;
    *-- assign the name GROUPMARKERS to named association.         --*;
      Discreteattrvar attrvar=groupmarkers var=TRT attrmap="lineattr";

      layout gridded / rowgutter=5;
        layout datapanel classvars=( BBMI USUBJID) /
          sparse=false includeMissingClass=false rowgutter=5 columngutter=5
          rowDataRange=unionall columnDataRange=union panelNumber=_panelnumber_
          cellHeightMin=50px cellWidthMin=50px start=TopLeft columns=3 rows=3
          rowAxisOpts=( display=all altdisplay=all Label="Score" type=auto
          linearopts=( viewmin=_yviewmin_ viewmax=_yviewmax_ ) )
          columnAxisOpts=( display=all altdisplay=all Label="Month" type=auto
          linearopts=( tickvaluelist=( 1 2 3 4 5 6) viewmin=0 viewmax=6 ) );
           layout prototype / __SGPROC;
             SeriesPlot X=MONTH Y=SCORE/primary=true display=(markers) group=TRT
                                       Lineattrs=( Pattern=1 Thickness=3)
                                       LegendLabel="Treatment" NAME="Treatment"
                                       grouporder=ascending;
           endlayout;
         endlayout;
         DiscreteLegend "Treatment" / Title="Treatment: " Border=false;
        endlayout;
       endgraph;
     end;
  run;

ods graphics on;
ods rtf file="C:\SGF_2013\Figure5b_2.rtf";
ods rtf style=STYLES.MYSTYLE;

proc sgrender data=SCORE template=SGPANEL; run;
ods rtf close;
```

Based on the code above, a template named **SGPANEL** has been created. Within this template, **a Discrete Attribute Map named *"lineattr"*** has been defined to ensure the same line attributes will always be assigned to the same treatment. **group=TRT** identifies TRT as the variable to be used for grouping. **grouporder=ascending** ensures that the legend to the graphic panel are listed based on the ascending order of the treatment group, instead of the order of the data. After SGPANEL template has been defined. SGRENDER procedure applies the SGPANEL template and renders the multi-cell graph, as seen in Figure 5B.

ASIDE

It is interesting to observe that the authors have decided to leave out the unit of measurement for BMI (kg/m^{2}) in Figures 5A and 5B. This is done intentionally so as not to crowd the label at the top of each panel. Our label for BBMI in data step SCORE is as follows:

```
Label BBMI='Baseline BMI (kg/m^{Unicode '00b2'x})';
```

This code got truncated in Figures 5A and 5B while the following code would work:

```
Label BBMI='Baseline BMI (kg/m**2)';
```

Hence, truncation is based on the length of the code (i.e., `'Baseline BMI (kg/m^{Unicode '00b2'x})'` vs. `'Baseline BMI (kg/m**2)'`, not the length of actual text displayed (i.e., 'Baseline BMI (kg/m^{2})' vs 'Baseline BMI (kg/m**2)'.)

ADDITIONAL REMARKS

Note that the usage of INDEX option in GTL has not been mentioned in the creation of multi-cell graph such as Figure 5B in this paper because SAS currently does not support this method in the generation of multi-cell graph.

In future, the SG procedures will be available through Base SAS. Users do not need to have a special license for SAS/GRAPH in order to execute these SG procedures. This can serve as an incentive to transition your graphical production from traditional methods to the SG procedures.

CONCLUSION

In the analysis of clinical data, often times only a subset of data is of interest. Special attention has to be taken in order to guarantee that the line appearance for the treatments stay intact in subset analyses. In this paper, the authors illustrated 5 different approaches to ensure consistency in line attributes for the SGPLOT procedure. These methods include (1) adding dummy observations to input data, (2) re-mapping line attributes, (3) re-ordering of group variable internal values, (4) use of INDEX option in GTL and (5) use of Discrete Attribute Maps in SG procedures and in Graph Template Language (GTL). Users should select the appropriate method based on the purpose of the graph, the version of SAS available and the amount of customization needed.

In addition, readers have been introduced to Discrete Attribute Maps and Range Attribute Maps, as well as, the usage of Attribute Maps with SG procedures and Graph Template Language (GTL). The authors have demonstrated that the Attribute Map methods used for SGPLOT procedure can also be applied to the SGPANEL procedure to ensure line consistency in a panel of line plots. The SG procedures are user-friendly and can produce quality graphics within a short period of time. GTL may take a little bit more time to master, but can be useful when more customizations are needed.

SLIDE PRESENTAION

Slide presentation of this paper will be available in http://www.slideshare.net/alice_m_cheng

REFERENCES

- Carpenter, Arthur L. (2004). *Carpenter's Complete Guide to the SAS® Macro Language, Second Edition.* SAS Institute, Cary, NC.
- Cheng, Alice M. (2011). "Is the Legend in your SAS/Graph® Output Still Telling the Right Story?" *WUSS 2011. http://www.lexjansen.com/cgi-bin/xsl_transform.php?x=wuss2011&s=proceedings&c=wuss#sdp11*
- Flavin, Justina M. and Carpenter, Arthur L. (2004). "Is the Legend in your SAS/Graph® Output Telling the Right Story?" *SUGI 29.* http://www2.sas.com/proceedings/sugi29/086-29.pdf
- Kuhfeld, Warren F. (2010). *Statistical Graphics in SAS®): An Introduction to the Graph Template Language and the Statistical Graphics Procedures.* SAS Institute, Cary, NC.
- Matange, Sanjay and Health, Dan (2011). *Statistical Graphics Procedures by Example: Effective Graphs Using SAS.* SAS Institute, Cary, NC.
- SAS Institute Inc. (2011). Graph Template Language Tip Sheet. http://support.sas.com/rnd/base/topics/statgraph/GTL_Tips.pdf
- SAS Institute Inc. (2011). *SAS/GRAPH® 9.3 Graph Template Language Reference, Second Edition.* SAS Institute, Cary, NC. http://support.sas.com/documentation/cdl/en/grstatgraph/64764/HTML/default/viewer.htm#p0891gx3y0z8xqn1k9ijhv5xughi.htm
- SAS Institute Inc. (2011). *SAS® 9.3, ODS Graphics: Procedures Guide.* SAS Institute, Cary, NC. http://support.sas.com/documentation/cdl/en/grstatproc/62920/HTML/default/viewer.htm#titlepage.htm

ACKNOWLEDGMENTS

Alice Cheng was first inspired to explore this topic after a brief conversation with Shawn Hopkins, of Seattle Genetics. Thank you, Mr. Hopkins, for sharing your insight and expertise. Both authors are grateful to Susan Morrison, Lelia McConnell and Marcia Surratt of SAS Technical Support and Sanjay Matange of the SAS Institute for sharing their expertise on statistical graphics, as well as, Adam Sharp and Steve Michaelson of SimulStat, Inc. for providing us with SAS v9.3. A special thank you is extended to to Jim Szymeczek and Boyd Roloff of Chiltern International Ltd for their support of this paper. Last but definitely not least, we would like to express our sincere gratitude to Art Carpenter for his valuable feedback on the predecessor of this paper. Any errors and oversight in this article are the sole responsibility of the authors.

CONTACT INFORMATION

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

Alice M. Cheng
E-mail: alice.cheng@chiltern.com
alice_m_cheng@yahoo.com
http://www.linkedin.com/in/alicemcheng

Justina M. Flavin
E-mail: justina.flavin@gmail.com