Paper 233-2010

Customizing ODS Statistical Graphs Step-by-Step

Dongsheng Yang, Cleveland Clinic Foundation, Cleveland, OH
Anne S. Tang, Cleveland Clinic Foundation, Cleveland, OH

ABSTRACT

Different from the traditional SAS/GRAPH® in SAS® 9.2, ODS statistical graphics from procedures provide an easy and quick way to explore and visualize data. However, if you want to generate high-quality plots for reports and publications, Graph Template Language (GTL) is a powerful tool to customize and build standalone ODS statistical graphs. This paper describes how to output SAS® codes and data objects of an ODS statistical graph generated from a statistical procedure, and then use GTL to customize graph templates and style templates. Some examples on customizing ODS graphics step-by-step are provided.

INTRODUCTION

Many statistical procedures in SAS® 9.2 automatically provide ODS graphics. Those ODS graphics are generated by the Graph Template Language (GTL) using the default template. By using GTL, templates can be modified in an easy and flexible way in order to have desired graphs. This presentation shows how to create high quality graphs which are independent of statistical procedures step-by-step.

Figure 1 presents a general scheme to modify an ODS statistical graph. We can modify appearance of an ODS graph by using either ODS style template or graph template. ODS graph styles control the overall appearance of an ODS graph such as graph elements (lines, markers, fonts and colors, fitted lines, and confidence bands etc) and attributes (line style, marker sizes etc). Graph templates control the layout and details of an ODS graph such as graph elements, attributes, and etc. SAS® recommends try to control overall appearance (by modifying "style") first so that it can be used across many graphs.

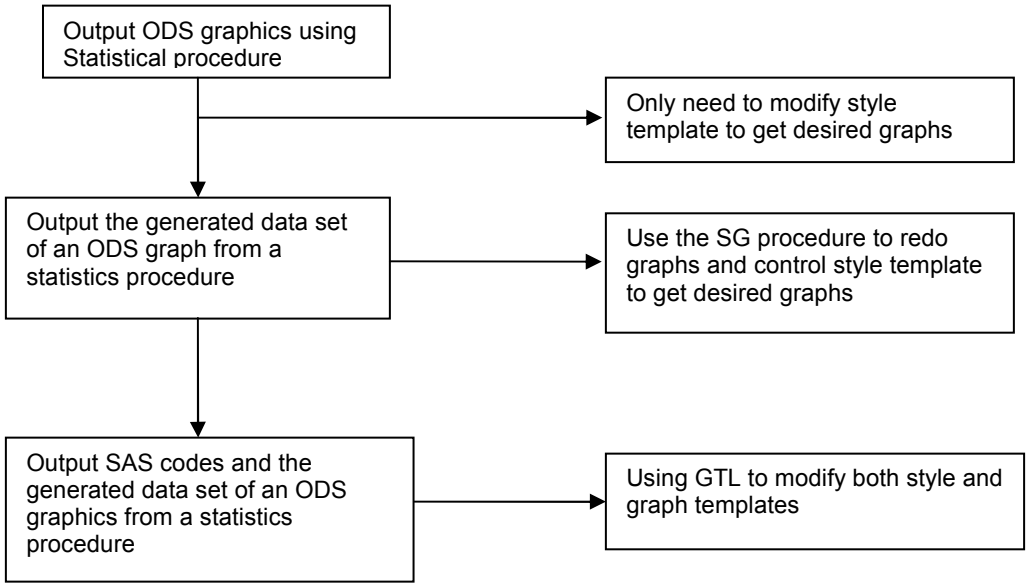

Figure 1 A schema to modify ODS statistical graphics

CONTROLING STYLE TEMPLATE

The following codes showed a real example from a client who wanted an effect plot of the diabetic group but not the non-diabetic group from the logistic regression. The default output included both diabetic and non-diabetic group. By

controlling the style template, we set the non-diabetic group to be white color and black in diabetic group (Figure 2). However, at this point, we could not control the legend position.

```
proc template ;
    define style styles.mystyle; parent = styles.Journal;
    style  GraphFonts /
                'GraphValueFont'     = ("<MTserif>, Times New Roman", 14pt)
                'GraphLabelFont'     = ("<MTserif>, Times New Roman", 16pt, bold)
                'GraphTitleFont'     = ("<MTserif>, Times New Roman", 18pt, bold);
     style GraphDataDefault /  linethickness = 3px;
     style GraphAxisLines   /  linethickness = 2px;
     style GraphWalls       /  lineThickness = 2px FrameBorder = on;
     style graphdata1       /  linestyle = 1   ContrastColor = white; ❶
     style graphdata2       /  linestyle = 1   ContrastColor = black; ❷
end;
ods listing   style   = mystyle;  Ods graphics on;
Proc logistics data = mydata plots = effect descending;
    Class group;
    Model y = group num_drug;
run;
```

❶ Set the non-diabetic group to be white
❷ Set the diabetic group to be black

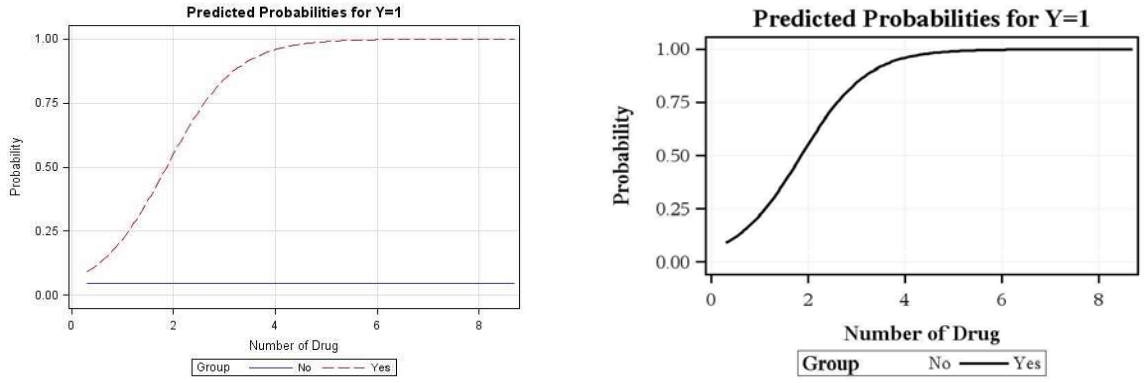

Figure 2 Left panel: default output; right panel: control the style template and only show the diabetic group

USING SG STATISTICAL PROCEDURES

Since the client wanted to remove legend and insert “Diabetics” by the curve, we have to output the generated data of the effect plot and make a graph using the SGPLOT procedure (Figure 3).

```
ods graphics on;
ods output effectplot = mydata; ❶
proc logistic data = combat_1 plots=EFFECT
descending;
Class group;
    Model y = group num_drug;
ods listing     style     = mystyle   ; ❷
proc sgplot data = mydata NOAUTOLEGEND ;
    title "Predicted Probabilities for
           Pain = Yes ";
series x = _xvar y = _PROB/group = _INDEX;
    xaxis label = "Number of Drug";
    yaxis label = "Probability";
    KEYLEGEND “Diabetics” / POSITION=topright;
run;
```

❶ Output the generated data of the effect plot
❷ Use the previous modified style template procedure

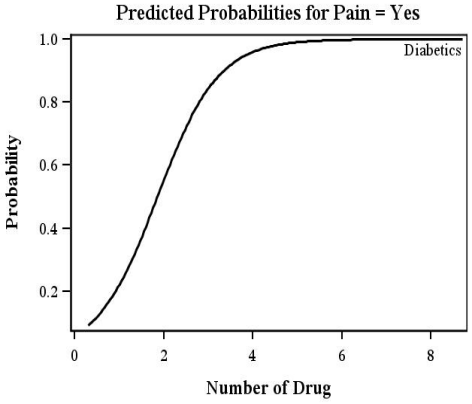

Figure 3 An effect plot created by Proc SGplot

USING GRAPH TEMPLATE LANGUAGE

Here we showed how to use GTL to customize a complicated ODS graph step-by-step. We use the example "The example 51.8 Comparing Receiver Operating Characteristic Curves in the PROC Logistic from SAS9.2 help documents".

The main steps include:

1. Find the ODS graph template and the data set it used from a statistical procedure using ODS Trace On.
2. Output codes of the interested graph template from ODS graph and its related data object
3. Manipulate the output data set to add any desired points, lines or others which you want to be shown in the graphs
4. Change the style template to control overall appearance
5. Modify the graph template for specific and detailed changes
6. Set up the ODS Destination and ODS graphic statement
7. Run the SGRENDER procedure to associate the appropriate data with the template

ODS TRACE ON

By using ODS TRACE ON, we can easily find the name of the interested graph template which was used to create an ODS graphics and the name of related data object in the SAS® log window.

Below is ODS TRACE ON output for ROCOverlay object:

```
Output Added:
-------------
Name:        ROCOverlay  ❶
Label:       ROC Curves
Template:    Stat.Logistic.Graphics.ROCOverlay  ❷
Path:        Logistic.ROCComparisons.ROCOverlay
-------------
```

❶ Object name of an ODS statistical graph. It is the name of the ROC overlay graph and the corresponding data. So later we can use ODS select ROCOverlay to select and output a single graph object. We can output the corresponding data object using statement: ODS output ROCOverlay = data set name

❷ The name of graph template for ROC overlay curves.

SELECT DEFAULT TEMPLATE CODES

Since SAS® has defined default graph template codes for each ODS statistical graph, we can use proc template to display codes of the interested templates either in the log window or by writing to an external file from running ODS TRACE ON. The following statements outputs codes of the ROCoverlay template:

```
proc template;
      source Stat.Logistic.Graphics.ROCOverlay/store = sashelp.tmplmst;
run;
```

The codes output in the log window:

```
proc template;
define statgraph Stat.Logistic.Graphics.ROCOverlay;
   notes "Receiver Operating Characteristic Overlaid Curves";
   dynamic _TITLE;
   (omit some codes ..........)
              discretelegend "Step" / title="ROC Curve (Area)";
      endlayout;
   EndGraph;
end;
run;
```

As we see, codes are well-defined, easy to read and easy to modify. Here is the code and default graph output for a ROC overlay statistical graph. We are going to try to customize axis, fonts, labels, lines, legend and add some specific information (Figure 4).

```
ods graphics on/;
ods trace on; ❶
ods select ROCOverlay ; ❷
ods output
ROCOverlay=ROCOverlaydata ROC =
myROCdata; ❸
proc logisticata=roc ;
  model popind(event='0') =
  alb tp totscore / nofit;
     roc 'Albumin' alb;
     roc 'K-G Score' totscore;
     roc 'Total Protein' tp;
     roccontrast reference('K-G
          Score')/ estimate e;
run;
ods trace off;
ods graphics off;
```

❶ Represent ODS TRACE ON

❷ Represent selecting single statistical graph

❸ Represent output the generated data set

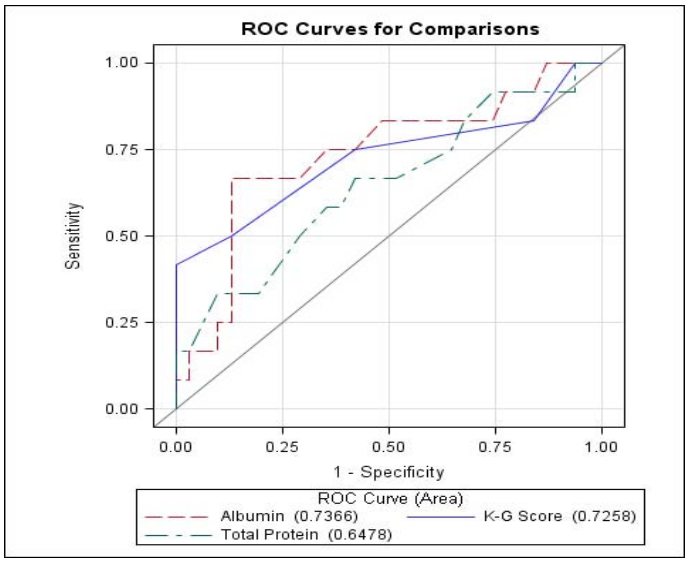

Figure 4 the default output of ROC Curves for Comparisons

MANIPULATE DATA

We may want to make three main changes in the graph:

1. Display only to the third decimal place for the C-statistics in the legend

2. Present the ROC curve which had the largest C-statistics in a bold line if those individual curves are intertwined together

3. Make the cutoff point with data label along the ROC curve for the variable with largest C- statistics

However, only one data set is allowed to pass into GTL each time. If we want to add more details such as points, lines, arrows and plots to the ODS statistical graph, we need to manipulate our data object in the data step. Here is part of the generated data set of the ROC overlay which has been used to create the above default ROC overlay plots:

Obs	_SENSIT_	_1MSPEC_	_ROC_		_GROUP_
1	0.00000	0.000000	Albumin	(0.7366)	2
2	0.08333	0.000000	Albumin	(0.7366)	2
3	0.08333	0.032258	Albumin	(0.7366)	2
4	0.16667	0.032258	Albumin	(0.7366)	2
5	0.16667	0.064516	Albumin	(0.7366)	2

Below is a portion of new data after we made changes in the data step to overlay a thicker line onto the Albumin ROC curve (by using new variables: _x_ and _y_). We also created another 3 new variables “x_cutoff”, “y_cutoff “ and “cutoff” to mark the cutoff point in the graph.

SENSIT	_1MSPEC_	_GROUP_	_roc_		_x_	_y_	y_cutoff	x_cutoff	cutoff
0.00000	0.000000	2	Albumin	(0.737)	0.000000	0.00000	0.66667	0.12903	3
0.08333	0.000000	2	Albumin	(0.737)	0.000000	0.08333	.	.	
0.08333	0.032258	2	Albumin	(0.737)	0.032258	0.08333	.	.	
0.16667	0.032258	2	Albumin	(0.737)	0.032258	0.16667	.	.	
0.16667	0.064516	2	Albumin	(0.737)	0.064516	0.16667	.	.	

CHANGE ODS STYLE TEMPLATE

The following PROC TEMPLATE code shows the beginning of the style definition on top of journal style. We changed font, font sizes, axis lines, graph boundary lines and line style of individual ROC curve. Only modified codes are shown here.

```
ods path   work.templat(update      ❶
        sashelp.tmplmst  (read);    ❷

proc template;
define style styles.MyROCoverStyle; ❸
          parent = styles.Journal; ❹

style GraphFonts  /  ❺
      'GraphDataFont'           = ("<MTserif>, Times New Roman", 10pt)
      'GraphUnicodeFont'        = ("<MTserif>, Times New Roman", 9pt)
      'GraphValueFont'          = ("<MTserif>, Times New Roman", 12pt)
      'GraphLabelFont'          = ("<MTserif>, Times New Roman", 16pt,  bold)
      'GraphFootnoteFont'       = ("<MTserif>, Times New Roman", 10pt,  bold)
      'GraphTitleFont'          = ("<MTserif>, Times New Roman", 18pt,  bold)
      'GraphAnnoFont'           = ("<MTserif>, Times New Roman", 10pt);

style GraphAxisLines   /  Linethickness = 2px;    ❻
style GraphWalls       /  LineThickness = 2px ;   ❻
style Graphdata1       /  Linestyle = 4;          ❼
style Graphdata2       /  Linestyle = 1;          ❼
style Graphdata3       /  Linestyle = 27;         ❼

end;
```

We will make the following changes:

❶ Save the customized Style template into work library

❷ Default template library for ODS statistical graphics. This template will be invoked if the first one does not exist

❸ Assign a name to the new style

❹ Use Journal style as its parent style

❺ Modify fonts or font sizes for axis labels, title, footnote, legend titles, axis tick values and legend entries etc.

❻ Modify the line thickness of axis lines and graph boundaries

❼ Select types of lines.

MODIFY ODS GRAPH TEMPLATE

An ODS graph style definition provides other detailed formatting information of a graph output. After we defined our new ODS style, we now want to customize specific visual display on the existing graph template. You can use the graph template language to specify plot layouts (such as lattice or overlays), plot types (such as scatter plots and series plots), text elements (such as titles, footnotes, and insets), and other plot features (such as lines, marker symbols, and colors).

The following statements display the graph template of ROC curves:

```
define statgraph Graphics.ROCOverlay; ❶
     notes "Receiver Operating Characteristic Overlaid Curves";
     dynamic _TITLE ;
     BeginGraph;
        entrytitle _TITLE;
        layout overlayequated /equatetype = square
           yaxisopts = (       gridDisplay    = off  ❷
                               label          = "Sensitivity"
                               shortlabel     = "TPR"
                               offsetmin      = 0.05
                               offsetmax      = 0.05)

           xaxisopts = (       gridDisplay    = off ❷
                               label          = "1 - Specificity"
                               shortlabel     = "FPR"
                               offsetmin      = 0.05
                               offsetmax      = 0.05)

            commonaxisopts = (  tickvaluelist   =(0 .25 .5 .75 1)
```

```
                              viewmin         = 0
                              viewmax          = 1
                            );

            lineparm x=0 y=0 slope=1 / extend    = true
                  lineattrs =( color = grey thickness = 2px); ❸

            seriesplot y =_SENSIT_ x=_1MSPEC_ /
                        connectorder = xaxis
                        tip          = (group y x)
                        group        = _ROC_
                        index        = _GROUP_
                        name         = "Step"
                        primary      = true
                        lineattrs    = (thickness = 2px); ❸

          seriesplot y = _y_ x = _x_ /  LINEATTRS  = ( PATTERN = 4 thickness = 3px) ❹
                                        Connectorder = xaxis;

          scatterplot y = y_cutoff x = x_cutoff/ datalabel = cutoff
                                       markerattrs  = (size = 10pt
                                       symbol       = circle weight = bold); ❺

          discretelegend "Step" / Title      = "ROC Curve (Area)"  ❻
                                  LOCATION    = INSIDE
                                   HALIGN     = RIGHT
                                   VALIGN     = BOTTOM
                                   PAD        = (RIGHT = 2PX bottom = 2px)
                                   ORDER      = ROWMAJOR
                                   ACROSS     = 1
                                   BORDER     = FALSE
                                   TITLEATTRS = (family = 'times' size = 12pt weight =
BOLD)
                                   VALUEATTRS = (family = 'times' size = 10pt);
          endlayout;
       EndGraph;
    end;
run;
```

❶ Assign a name to the new graph template

❷ Do not display grids

❸ Change default line attributes

❹ Overlay a new curve and change the default line attributes by using the new variables created in data object

❺ Overlay a scatter plot and change the default marker attributes by using the new variables created in data object

❻ Modify legend location and its attributes

The DEFINE STATGRAPH statement in PROC TEMPLATE assigns a name to the new graph template definition. DYNAMIC statement defines one dynamic variable which creates the title of the graph. The whole GTL syntax block begins from BEGINGRAPH to ENDGRAPH. The LAYOUT block begins from LAYOUT to ENDLAYOUT to define overlaid in the graph, and it is the main component of a graph template. Couple things we could do within the LAYOUT block include creating a graph overlaid by another graph, changing attributes of AXISOPTS according to our needs, and specifying the color and size of a symbol, text, and legend.

ODS DESTINATION STATEMENT

The ODS destination statements will indicate output directory and type of the image. ODS HTML, ODS LISTING, ODS PDF, ODS PS, and ODS RTF are often used here.

```
ods listing
style      = MyROCoverStyle ❶
image_dpi  = 150            ❷
gpath      = "h:\Temp";     ❸
```

❶ Specify the style we defined

❷ Specify the dots per inch (DPI), which is the image resolution for graphical output

❸ Specify the directory where the graphics files are saved

ODS GRAPHICS STATEMENT

The ODS GRAPHICS statement enables ODS to create graphics

```
ods graphics on/reset                                    ❶
                  imagefmt  = jpg                        ❷
                  imagename = "MyROCoverlay"             ❸
                  border    = off                        ❹
                  height    = 6.5in                      ❺
                  width     = 6.5in                      ❻
                  SCALE     = on;                        ❼
```

❶ Reset ODS GRAPHICS options to their default settings

❷ Specify type of the image file

❸ Specify the image file name

❹ Specify whether to draw the graph with a border

❺ Specify the height of the graph

❻ Specify the width of the graph

❼ Specify whether the fonts and symbol markers are scaled proportionally with size of the graph

ODS RUN THE SGRENDER PROCEDURE

The SGRENDER procedure produces a graph from an input SAS data set and an ODS graph template:

```
proc sgrender data = rocdata template = Graphics.ROCOverlay;
run;
```

The final customized graph output:

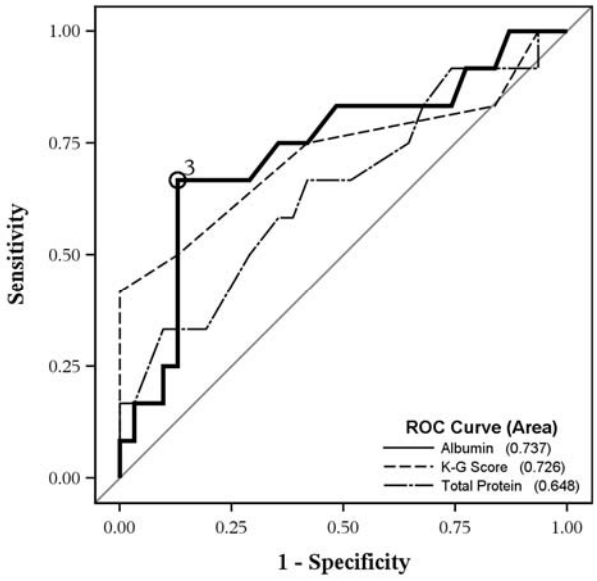

Figure 5 The customized graph output of the ROC Curves for Comparisons

CONCLUSION

ODS statistical graphics in SAS 9.2 provides most common plots within procedures. By using the graph template language, it is convenient to modify and extend default statistical templates to create high quality of graphs.

REFERENCES

■ Cartier, J. (2006), “A Programmer’s Introduction to the Graphics Template Language,” Proceedings of the Thirty-first Annual SAS Users Group International Conference. Cary, NC: SAS Institute Inc.

■ SAS Institute Inc. SAS/GRAPH SAS®9.2: Graph Template Language User's Guide

■ SAS Institute Inc. SAS/GRAPH ® 9.2: Graph Template Language Reference

■ SAS Institute Inc. SAS/STAT® 9.2 User's Guide

CONTACT INFORMATION

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

Name: Dongsheng Yang
Enterprise: Cleveland Clinic
City, State ZIP: Cleveland, OH 44195
Work Phone: (216) - 6365418
E-mail: yangd@ccf.org

Name: Anne S. Tang
Enterprise: Cleveland Clinic
City, State ZIP: Cleveland, OH 44195
Work Phone: (216) - 4440639
E-mail: tanga@ccf.org

Web: http://www.lerner.ccf.org/qhs/