Paper 495-2013
Automated and Customized Reports as a Single Image File Using Graph Template Language (GTL): A Case Study of Benchmarking Reports in Medical Research

Monarch Shah, Ginny P. Lai, Eric P. Elkin
ICON Late Phase & Outcomes Research, San Francisco, CA

ABSTRACT
Site benchmarking reports give us an overview of basic demographic, clinical, and disease characteristics for the individual site with comparison to the study as a whole. A solution was needed for on-going reporting to over 250 study sites participating in a long-term study. The report needed to be concise, present data in both tables and figures, and provide information from both the individual study site and the entire study. (This objective could also arise in other fields, for example comparing each store's performance to the entire chain or each classroom's performance to the school district.) However, creating and combining tables and figures into an Excel or PowerPoint file can be challenging and time consuming. Alternatively, through a new graphical procedure available in SAS® 9.2, one can customized reports and automate them to generate for each site and for future use. This paper will focus on using Graph Template Language (GTL) to create panels comprising of descriptive table and multiple graphs as a single image file for site benchmarking reports.

INTRODUCTION
In a multi-site, multi-year disease registry, analytical graphs and tables are often an integral part of the study. Specifically, study sites can benefit from a summarized report comparing their patients to the overall registry on a regular basis. The site benchmarking report gives us an overview of basic patient information as well as descriptions regarding the disease of interest.

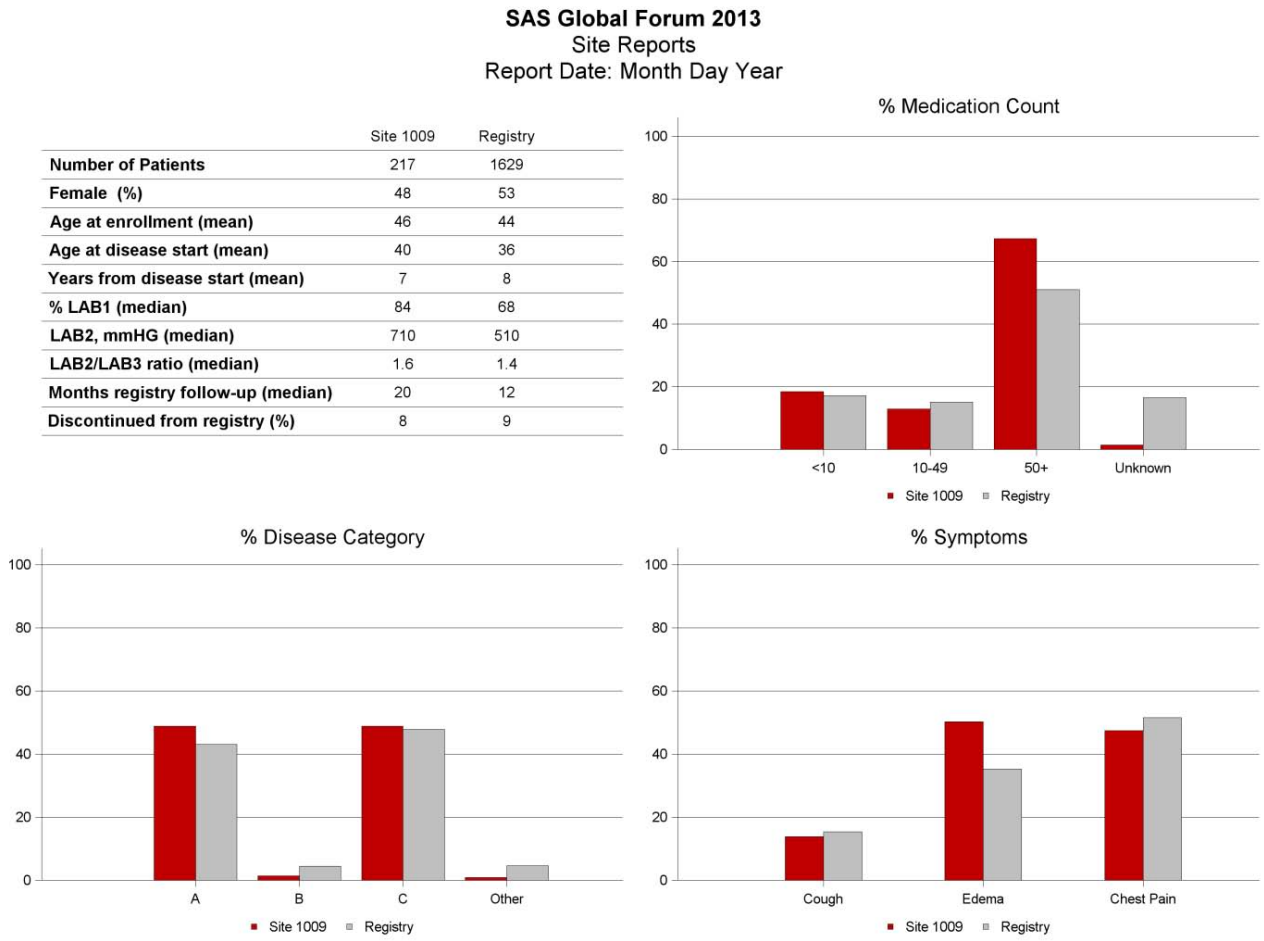
These demographics, clinical, and disease characteristics can be summarized into tables and/or graphs. In most instances, producing these results involves multiple procedures and DATA steps which are repeated for each site. However, the number of sites may range from 10 to the thousands. In our particular case, we had to create site-specific benchmarking reports for over 250 sites. In addition, creating and combining tables and graphs into a single Excel, PowerPoint, or Word file may be challenging and time consuming. Although many of these tasks may be accomplished using SAS/GRAPH and DATA steps to some extent, they do not provide a straightforward solution because of some limitations in layout capabilities.

Therefore, we needed a more flexible tool to go beyond SAS/GRAPH in order to create our own customized reports which could be automated to produce reports for over 250 study sites. As a result, we used a new graphical procedure available in SAS 9.2. Graph Template Language (GTL) presents a powerful alternative which is more time-efficient and reproducible for combining panels including multiple tables and graphs as an image/plot for site reports.

GRAPH TEMPLATE LANGUAGE (GTL)
GTL is a subset of the PROC TEMPLATE statements used in the STATGRAPH templates. All of the default graphs that are created by SAS/STAT and by SAS/GRAPH are generated from compiled templates written in GTL. Programmers who need to go beyond the graphs created by these SAS procedures can use GTL to design their graphs by modifying the predefined STATGRAPH template. After creating the template, programmers can associate the modified template with PROC SGRENDER procedure to generate desired graphs.

REPORT EXAMPLE
In this paper, we use a sample dataset comprising patient data regarding demographics, medical, and disease information to create a single display, four-panel summary of tables and figures for each study site. The first panel of the report will be a descriptive table that compares patient characteristics of an individual site to the overall registry. The next three panels will include three side-by-side bar charts of comparing an individual site and the overall registry for a variety of clinical information. Figure 1 shows a sample of the final benchmarking report for a hypothetical site.

FIGURE 1 (SEE APPENDIX FOR CODE)



DEFINE THE STRUCTURE OF THE IMAGE

In order to achieve the goal, we will need to treat each of the four panels as an individual and independent plot. One of the most powerful aspects of the GTL is the syntax build around hierarchical LAYOUT statement blocks. A layout is a container that arranges its contents as *cells*. A cell can include a plot, a title, a legend, or even another layout. The layout arranges the cells in a predefined manner into rows, columns, or a single cell. All STATGRAPH templates definitions must begin with a LAYOUT statement block.

We decided to use the most flexible and appropriate layout option LAYOUT LATTICE to produce the site reports. LAYOUT LATTICE allows us to create a multi-cell grid of graphs where plot areas and tick display areas across grid cells are automatically aligned on a lattice for better comparison. See the SAS User's Guide for GTL for more details on the other LAYOUT statements.

Note that one of the very useful qualities of GTL is the DYNAMIC statement and/or macro variables that resolve when the template is executed by PROC SGRENDER. This will help facilitate processing data for lots of study sites. In this example, there are five DYNAMIC variables title1, tbl1, med, disease and symp, which will get referenced in PROC SGRENDER procedure.

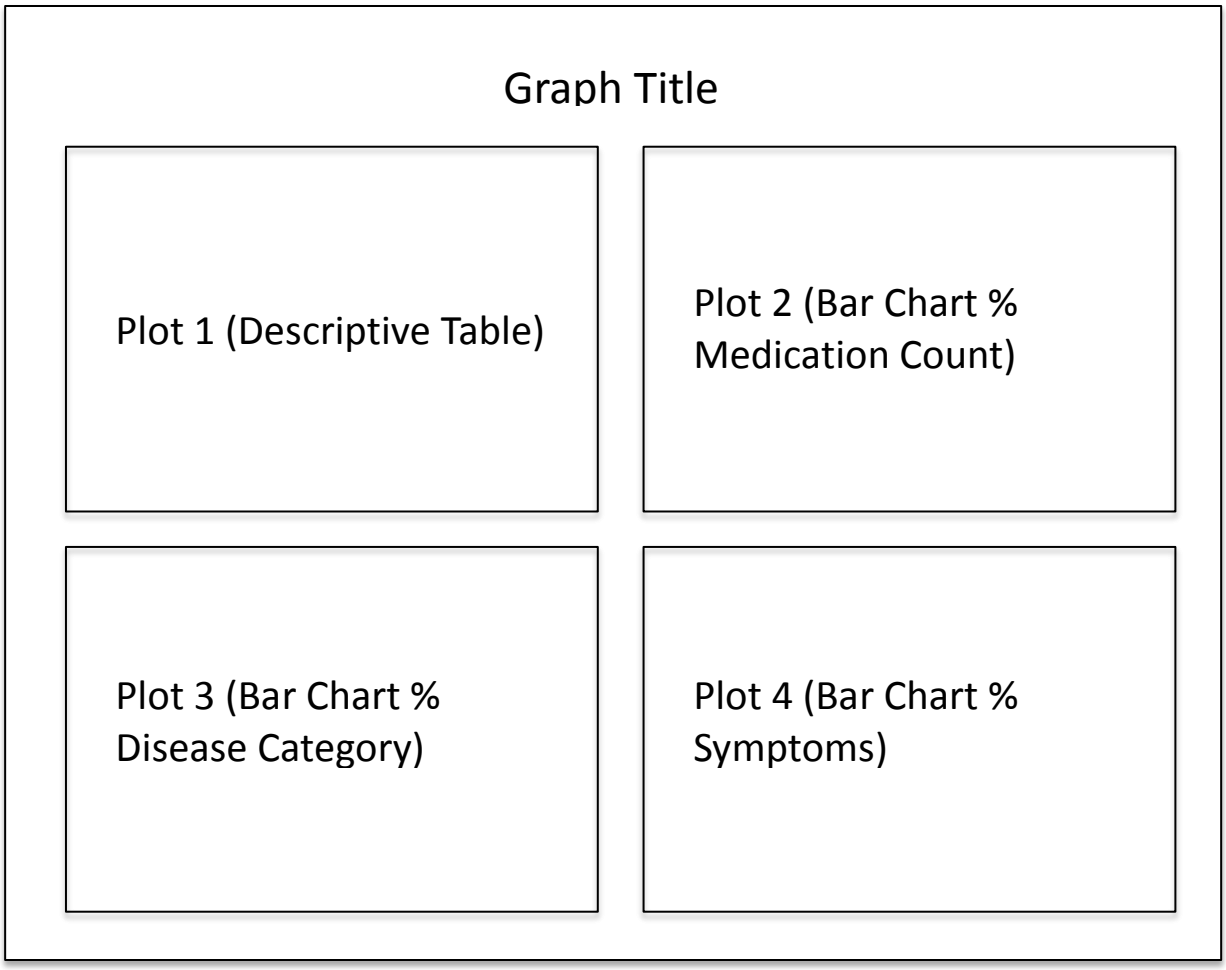
```
proc template;
  define statgraph SGF2013;
  dynamic title1 tbl1 med disease symp;
  begingraph;

  entrytitle textattrs=(size=30spx weight=bold) "SAS Global Forum 2013";
  entrytitle textattrs=(size=30spx) "Site Reports";
  entrytitle textattrs=(size=30spx) title1;

  layout lattice / columns=2 rows=2 shrinkfonts=true rowweights=(.5 .5)
                   columnweights=(.5 .5) rowgutter=10 columngutter=10;
```

After assigning the specific attributes for the graph title, the code presented above will create a multi-cell grid containing four panels on our layout by using the COLUMNS=2 and ROWS=2 option. One can think of it as a big box which is divided into four compartments where each compartments will be populated with its own graph. Options ROWWEIGHTS and COLUMNWEIGHTS allow us to specify the size of each individual plots. For the purpose of this site report, each of the four panels will take up equal space on our layout. Fifty percent of the horizontal and vertical space is assigned to each individual cell. The ROWGUTTER=10 and COLUMNGUTTER=10 options will put space between each of the cells. Figure 2 shows the final specified LATTICE layout.

FIGURE 2



Note that an ENDGRAPH statement must have a corresponding BEGINGRAPH statement and an END statement (not shown in the sample code above) needs to go with the DEFINE statement at the end of each TEMPLATE procedure. For the next few sections, multiple LAYOUT OVERLAY statements will be used within the LAYOUT LATTICE statement.

PLOTTING DATA

CREATE TABLE 1 AS A 'GRAPH'

In this section, we will explain how to create the first table as a graph placed in the top-left panel.

Data to produce a descriptive table for Site 1009 will be structured as shown in Figure 3. Descriptive statistics for the site and the overall registry have been calculated ahead of time using PROC MEANS for continuous data and PROC FREQ for categorical data. These statistics were then converted from numeric to character values for plotting purposes. Additional variables are also created to produce the desired table as a plot by using available options in SAS GTL.

FIGURE 3

id	label	var_name	blank_lbl	blank	site_lbl	site1009	registry_lbl	registry
1	label	Discontinued from registry (%)			Site i	8	Registry	9
2	label	Months registry follow-up (median)			Site i	20	Registry	12
3	label	LAB2/LAB3 ratio (median)			Site i	1.6	Registry	1.4
4	label	LAB2, mmHG (median)			Site i	710	Registry	510
5	label	% LAB1 (median)			Site i	84	Registry	68
6	label	Years from disease start (mean)			Site i	7	Registry	8
7	label	Age at disease start (mean)			Site i	40	Registry	36
8	label	Age at enrollment (mean)			Site i	46	Registry	44
9	label	Female (%)			Site i	48	Registry	53
10	label	Number of Patients			Site i	217	Registry	1629
11	label				Site i	Site 1009	Registry	Registry

The LAYOUT OVERLAY statement overlays multiple graph statements which are contained within an overlay container. For example, we'd like to plot a table with values comparing Site 1009 to the overall registry. We'd also like a label associated with each of the values. LAYOUT OVERLAY allows us the flexibility to produce such a table as a plot as long as the data are structured accordingly.

```
layout overlay / yaxisopts=(offsetmin=0 offsetmax=0.05 display=none
                            linearopts=(viewmin=0 viewmax=11))
                 xaxisopts=(offsetmin=0.25 offsetmax=0.05 display=none)
                 cycleattrs=true;

    scatterplot y=id x=label / markercharacter=var_name
                               markercharacterattrs=(size=25spx weight=bold);
    scatterplot y=id x=blank_lbl / markercharacter=blank
                                   markercharacterattrs=(color=white size=23spx);
    scatterplot y=id x=site_lbl /  markercharacter=tbl1
                                   markercharacterattrs=(size=23spx);
    scatterplot y=id x=registry_lbl / markercharacter=registry
                                      markercharacterattrs=(size=23spx);

        referenceline y=.5 / lineattrs=(thickness=1px color=black);
        referenceline y=1.5/ lineattrs=(thickness=1px color=black);
        referenceline y=2.5/ lineattrs=(thickness=1px color=black);
        referenceline y=3.5/ lineattrs=(thickness=1px color=black);
        referenceline y=4.5/ lineattrs=(thickness=1px color=black);
        referenceline y=5.5/ lineattrs=(thickness=1px color=black);
        referenceline y=6.5/ lineattrs=(thickness=1px color=black);
        referenceline y=7.5/ lineattrs=(thickness=1px color=black);
        referenceline y=8.5/ lineattrs=(thickness=1px color=black);
        referenceline y=9.5/ lineattrs=(thickness=1px color=black);
        referenceline y=10.5/lineattrs=(thickness=1px color=black);

endlayout;
```

For the descriptive table, we do not want any default y-axis or x-axis information to show up on the plot. The DISPLAY=NONE option allows us to remove default axis label, axis tick marks, and axis tick values. The OFFSETMAX option will put 5% space at the end of the last tick mark and end of the axis that will put small gaps between the next plot and the title of the plot. Option CYCLEATTRS=TRUE is used to change the visual properties of each plot without explicitly setting it. Any plots that derive their visual properties from one of the graph elements will be cycled through and applied.

Each descriptive statistics row has a unique ID variable in a specific order. The LAYOUT OVERLAY statement will cycle through all the non-missing IDs (1 to 11 in our example) and each record will be plotted as an individual plot. Next, all the plots in one single column will be stacked with the first on the bottom and then record after record within a SCATTERPLOT statement. Finally, SAS nests or superimposes each of the 'columns' described in each SCATTERPLOT statement from left to right within the plotting cell.

For our descriptive table, we want a column for the statistics label, a column for Site 1009 statistics and a column for overall registry statistics. The graphing statement SCATTERPLOT within GTL allows us to plot text values. The first SCATTERPLOT statement will plot a dummy variable ID as the y-axis and a dummy variable LABEL as the x-axis. The MARKERCHARACTER option allows us to plot desired text strings or values instead of markers. In our example, the MARKERCHARACTER=VAR_NAME option allows us to put descriptive statistics labels as a "column" for our descriptive table. We specify the table label font to be bold and 25 spx in size.

The second SCATTERPLOT statement will put a blank column next to the first one to create a more visually appealing table. The third SCATTERPLOT statement will plot the descriptive data for Site 1009. Again, using the MARKERCHARACTER option, we can use the values as a text string to be plotted. Similarly the SCATTERPLOT statement is used for plotting the overall registry column. Finally, by applying the REFERENCELINE option, we draw the reference line with 1px in thickness and black in color under each record associated with the ID. The final 'table' plotted as a graph is shown below (Figure 4).

Figure 4

	Site 1009	Registry
Number of Patients	217	1629
Female (%)	48	53
Age at enrollment (mean)	46	44
Age at disease start (mean)	40	36
Years from disease start (mean)	7	8
% LAB1 (median)	84	68
LAB2, mmHG (median)	710	510
LAB2/LAB3 ratio (median)	1.6	1.4
Months registry follow-up (median)	20	12
Discontinued from registry (%)	8	9

BAR CHART 1: MEDICATION COUNT

In this section, we will explain how to create the side-by-side bar chart placed in the top-right panel. The other two bar charts are created in a similar fashion.

Data to produce a side by side bar chart are shown in Figure 5. The percentage of each medication count category ('<10', '10-49', '50+', and 'unknown') are calculated for Site 1009 and for the overall registry. Because we would like to create a side-by-side bar chart to compare site 1009 with the overall registry, the medication data for Site 1009 are summarized as one column and medication data for the overall registry are summarized as another. The medication category labels are included in the first column.

Figure 5

ct1_lbl	ct1_registry	ct1_1009
<10	17.189	18.4332
10-49	15.163	12.9032
50+	51.074	67.2811
Unknown	16.575	1.3825

First, we'd like to add titles to the three bar charts in the same report. By putting the LAYOUT OVERLAY statement inside the CELL – ENDCELL block, we can customize our plot to have titles by adding the CELLHEADER – ENDCELLHEADER block. Here, we add "% Medication Count" after the ENTRY statement as the chart title and with the option TEXTATTRS=(size=30spx) to specify the title to be 30 spx in size.

```
cell;

cellheader;
  entry "% Medication Count"/ textattrs=(size=30spx);
endcellheader;

layout overlay / yaxisopts=(display=(tickvalues ticks line) offsetmax=0.05
                            linearopts=(viewmin=0 viewmax=100) griddisplay=on
                            gridattrs=(thickness=1px color=black))
                 xaxisopts=(display=(line ticks tickvalues)
                            offsetmin=.25 offsetmax=0.2)
                 cycleattrs=true;

    barchart x=ct1_lbl y=med / discreteoffset=-0.2 barwidth=0.4 name='A1';
    barchart x=ct1_lbl y=ct1_Registry / discreteoffset=0.2 barwidth=0.4 name='B1';

    discretelegend 'A1' 'B1' / border=false;

 endlayout;
 endcell;
```

Again, we start with a LAYOUT OVERLAY statement to create the bar chart. First, the axis attributes are assigned with option YAXISOPTS and XAXISOPTS. Option DISPLAY=(TICKVALUES TICKS LINE) allows us to show tick values, ticks and a y-axis line for the y-axis. We are leaving 5 % space at the end of the y-axis with the OFFSETMAX option. By using VIEWMIN and VIEWMAX options, we are forcing GTL to populate the data ranging from 0 to 100 on the y-axis. The GRIDDISPLAY=ON option turns on the grid display. In other words, we will see a grid line for every tick values. We also assign specific attributes to the grid using a GRIDATTRS option. By using the OFFSETMIN and the OFFSETMAX option, we can leave enough space at the start of the X-axis and at the end of the X-axis to force bars to appear close to each other.

Our goal is to compare Site 1009 with overall registry using a side-by-side bar chart in this cell. We can accomplish this task by plotting two separate bar charts with a common x-axis categorical variable. The BARCHART statement with the same X= attribute (variable CT1_LBL) and a separate Y= attribute will plot bars based on their y-axis data values. If we do not specify the DISCRETOFFSET option, bars for each category from both Site 1009 and the overall registry will be overlapping. The DISCRETOFFSET option puts the bars for Site 1009 0.2 pts left from the center of the categorical x-axis tick values, and puts the bars for the overall registry 0.20 pts right from the center of the categorical x-axis tick values. The width of the bars is set to be 0.4 pts. Finally, we assign both BARCHART statement as ‘A1’ and ‘B1’ in the NAME option so we can reference them in the legend statement. This will allow viewers to easily identify which bar is for Site 1009 and with is for the overall registry.

The DISCRETELEGEND statement will put legends for Site 1009 and the overall registry just below the x-axis. The BORDER=FALSE option removes borders around the legends. By default, GTL displays the label associated with the variable when the DISCRETELEGEND statement is used. We label our variable MED (macro resolves to CT1_1009 when called at the end) as “Site 1009” and variable CT1_REGISTRY as “Registry” in the previous DATA Step to get the desired result.

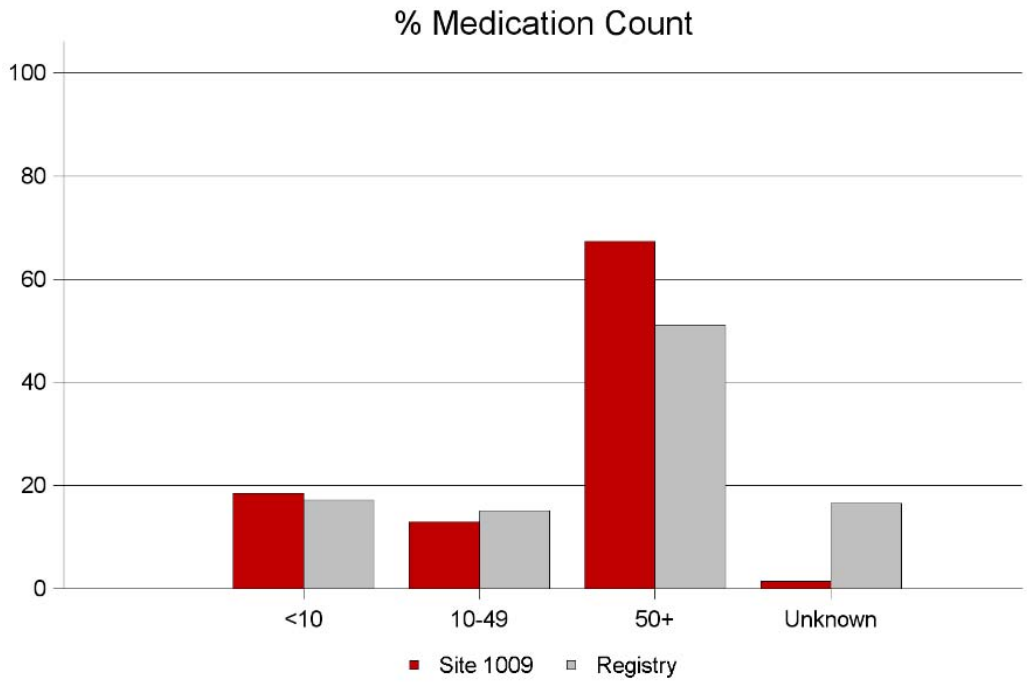
We run the code below in advance to modify the default template (defined as MYFONT) and to apply the registry’s color scheme to the bar charts produced by GTL.

```
proc template;
  define style myfont;
  parent=styles.default;
   style GraphFonts /
         'GraphDataFont'     = ("Arial",18spx)
         'GraphUnicodeFont'  = ("Arial",6spx)
         'GraphValueFont'    = ("Arial",18spx)
         'GraphLabelFont'    = ("Arial",14spx,bold)
         'GraphFootnoteFont' = ("Arial",16spx,bold)
         'GraphTitleFont'    = ("Arial",24spx)
         'GraphAnnoFont'     = ("Arial",8spx)
         'GraphValueText'    = ("Arial",16spx,bold);
   style GraphWalls /LineThickness = 2spx FrameBorder = off;
   style GraphBackground/BackGround=white;
   style GraphColors from GraphColors /
          "gdata1" = #C00000
          "gdata2" = #BFBFBF
```

```
            "gdata3" = blue;
  end;
run;
```

The final side-by-side bar chart of percent medication count is shown in Figure 6.

Figure 6



BAR CHART 2 AND 3: DISEASE CATEGORY AND SYMPTOMS

The datasets and codes are set up similarly for creating bar chart 2 (percent disease category) and 3 (percent symptoms).

```
cell;

cellheader;
  entry "% Disease Category"/ textattrs=(size=30spx) ;
endcellheader;

layout overlay / yaxisopts=(display=(tickvalues ticks line) offsetmax=0.05
                            linearopts=(viewmin=0 viewmax=100) griddisplay=on
                            gridattrs=(thickness=1px color=black))
                 xaxisopts=(offsetmin=.25 offsetmax=0.2
                            display=(line ticks tickvalues))
                 cycleattrs=true;

    barchart x=ct2_lbl y=disease/ discreteoffset=-0.2 barwidth=0.4 name='A2';
    barchart x=ct2_lbl y=ct2_registry/ discreteoffset=0.2 barwidth=0.4 name='B2';

    discretelegend 'A2' 'B2' / border=false;

endlayout;

endcell;
```

```
cell;

cellheader;
  entry "% Symptoms"/ textattrs=(size=30spx);
endcellheader;

layout overlay / yaxisopts=(display=(tickvalues ticks line) offsetmax=0.05
                            linearopts=(viewmin=0 viewmax=100) griddisplay=on
                            gridattrs=(thickness=1px color=black))
                 xaxisopts=(offsetmin=.25 offsetmax=0.2
                            display=(line ticks tickvalues))
                 cycleattrs=true;

    barchart x=ct3_lbl y=symp/ discreteoffset=-0.12 barwidth=0.24 name='A3';
    barchart x=ct3_lbl y=ct3_registry/ discreteoffset=0.12 barwidth=0.24 name='B3';

    discretelegend 'A3' 'B3' / border=false;

endlayout;

endcell;
```

EXECUTE THE TEMPLATE AND CREATE THE FINAL REPORT

Once we have a template ready, the SGRENDER procedure will execute the template and create the multi-cell site report. Because the 250 site reports we need to produce encompass a uniform style, we can accomplish this cumbersome and repetitive task by defining macro variables in the DYNAMIC statement mentioned in the beginning of this paper. In our example, we set up our dataset beforehand in a way that site numbers is associated with data from each specific site. As a result, we can cycle through all the site numbers in the macro call to produce 250 reports in the final step.

```
%macro report (reportDT=, SiteNum=, ImageName=);
  ods listing gpath= "C:\" style=myfont;
  ods graphics / reset noborder width=4800spx height=3600spx imagename=&ImageName.;

  proc sgrender data=sitedata template=SGF2013;
    dynamic title1       = "Report Date: &reportDT."
            tbl1         = "site&SiteNum."
            med          = "ct1_&SiteNum."
            disease      = "ct2_&SiteNum."
            symp         = "ct3_&SiteNum.";

  run;

  ods graphics off;
%mend report;

%macro ds;
  %do site = 1001 %to 1250;
    %report (reportDT=Month Day Year, SiteNum=&site., ImageName="Site&site.");
  %end;
%mend ds;
%ds;
```

Again, see Figure 1 for an example of the site benchmarking report.

CONCLUSION

In this paper, we have shown several examples of the flexibility and power of the Graph Template Language. We have shown how to create a table as a figure so it can be included with other figures in a multi-cell report. The report provides a lot of information on one page, while maintaining a good amount of space (the page is not too cluttered or overwhelming). We have also shown how to use macros and the DYNAMIC statement for an elegant solution to a repetitive task. This alternate and automated approach will make the process less time consuming and more accurate. Although the example here is study sites in a large disease registry for medical research, the concepts describe in this paper can be applied to other “benchmarking” situations.

APPENDIX

PROC TEMPLATE code used to create site reports.

```
proc template;
  define statgraph SGF2013;
   dynamic title1 tbl1 med disease symp;
  begingraph;

  entrytitle textattrs=(size=30spx weight=bold) "SAS Global Forum 2013";
  entrytitle textattrs=(size=30spx)"Site Reports";
  entrytitle textattrs=(size=30spx) "Report Date: Month Day Year";

  layout lattice / columns=2 shrinkfonts=true rowweights=(.5.5) columnweights=(.5 .5)
                   rowgutter=10 columngutter=10;

  layout overlay / yaxisopts=(offsetmin=0 offsetmax=0.05 display=none
                  linearopts=(viewmin=0 viewmax=11))
                   xaxisopts=(offsetmin=0.25 offsetmax=0.05 display=none)
                   cycleattrs=true;

     scatterplot   y=id x=label /markercharacter=tbl_label  markercharacterattrs=(size=25spx weight=bold );
     scatterplot   y=id x=blank_lbl / markercharacter=empty_    markercharacterattrs=(color=white size=23spx);
     scatterplot   y=id x=site_lbl  / markercharacter=site_1009 markercharacterattrs=(size=23spx );
     scatterplot   y=id x=registry_label / markercharacter=registry markercharacterattrs=(size=23spx );

     referenceline y=.5 / lineattrs=(thickness=1px color=black);
     referenceline y=1.5/ lineattrs=(thickness=1px color=black);
     referenceline y=2.5/ lineattrs=(thickness=1px color=black);
     referenceline y=3.5/ lineattrs=(thickness=1px color=black);
     referenceline y=4.5/ lineattrs=(thickness=1px color=black);
     referenceline y=5.5/ lineattrs=(thickness=1px color=black);
     referenceline y=6.5/ lineattrs=(thickness=1px color=black);
     referenceline y=7.5/ lineattrs=(thickness=1px color=black);
     referenceline y=8.5/ lineattrs=(thickness=1px color=black);
     referenceline y=9.5/ lineattrs=(thickness=1px color=black);
     referenceline y=10.5/ lineattrs=(thickness=1px color=black);
  endlayout;

cell;
  cellheader;
        entry "% Medication Count"/ textattrs=(size=30spx);
  endcellheader;
  layout overlay / yaxisopts=( display=(tickvalues ticks line)
                   offsetmax=0.05 linearopts=(viewmin=0 viewmax=100)
```

```
                    griddisplay=on gridattrs=(thickness=1px color=black))
                   xaxisopts=(offsetmin=.25 offsetmax=0.2 display=(line ticks tickvalues))
                    cycleattrs=true;

      barchart x=ct1_lbl y=med / discreteoffset=-0.2 barwidth=0.4 name='A1';
      barchart x=ct1_lbl y=ct1_registry / discreteoffset= 0.2 barwidth=0.4 name='B1';
      discretelegend 'A1' 'B1' / border=false;
endlayout;
endcell;

cell;
  cellheader;
        entry "% Disease Category"/ textattrs=(size=30spx) ;
  endcellheader;
layout overlay / yaxisopts=(display=(tickvalues ticks line) offsetmax=0.05
                 linearopts=(viewmin=0 viewmax=100) griddisplay=on
                 gridattrs=(thickness=1px color=black))
                 xaxisopts=(offsetmin=.25 offsetmax=0.2 display=(line ticks tickvalues))
                  cycleattrs=true;
      barchart x=ct2_lbl y=disease / discreteoffset=-0.2 barwidth=0.4 name='A2';
      barchart x=ct2_lbl y=ct2_Registry / discreteoffset= 0.2 barwidth=0.4 name='B2';
  discretelegend 'A2' 'B2' / border=false;
endlayout;
endcell;

cell;
  cellheader;
        entry "% Symptoms"/ textattrs=(size=30spx) ;
  endcellheader;
layout overlay / yaxisopts=(display=(tickvalues ticks line) offsetmax=0.05
                 linearopts=(viewmin=0 viewmax=100) griddisplay=on
                 gridattrs=(thickness=1px color=black))
                 xaxisopts=(offsetmin=.25 offsetmax=0.2 display=(line ticks tickvalues))
                  cycleattrs=true;

   barchart x=ct3_lbl y=symp / discreteoffset=-0.12 barwidth=0.24 name='A3';
   barchart x=ct3_lbl y=ct3_registry  / discreteoffset= 0.12 barwidth=0.24 name='B3';
  discretelegend 'A3' 'B3' / border=false;
endlayout;
endcell;

endlayout;
endgraph;
end;
run;
```

CONTACT INFORMATION

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

Monarch Shah
ICON Late Phase & Outcomes Research
188 Embarcadero, Suite 200
San Francisco, CA 94105
(415) 371-2139
monarch.shah@iconplc.com
www.iconplc.com/services/late-phase/

Ginny P. Lai
ICON Late Phase & Outcomes Research
4350 La Jolla Village Dr., Suite 350
San Diego, CA 92122
(858) 795-8232
ginny.lai@iconplc.com
www.iconplc.com/services/late-phase/

Eric Elkin
ICON Late Phase & Outcomes Research
188 The Embarcadero, Suite 200
San Francisco, CA 94105
Work Phone: (415)-371-2153
E-mail: eric.elkin@iconplc.com
www.iconplc.com/services/late-phase/