SAS® GLOBAL FORUM 2020 #SASGF

Paper 4228-2020

Using Heat Map or GTILE: Does Size Matter in Your Graphics?

Devi Sekar, RTI International, North Carolina

ABSTRACT

With SAS® 9.2 and beyond, ODS Graphics brings in a new way of generating high quality graphs. Many users still find themselves at the crossroads, trying to decide what path to follow – the traditional SAS/GRAPH or ODS Graphics. Both can produce most of the common types of graphs, such as scatter plots, regression and box plots, line graphs, bar charts, and histograms. In this paper we will share examples to generate a heat map using SGPLOT in ODS Graphics and a GTILE graph with SAS/GRAPH and discuss the advantages of each. With heat maps, you can display patterns in the data for a chosen response variable for one-dimensional data (in a map) or two-dimensional data (in a table or graph). Heat maps make tables and graphs easier to interpret, by shading the background color based on the frequency of observations in each cell of the graph or table. With GTILE, the response values are shown by both color gradient and area. We will also show how to customize a heat map by making use of the SAS® Graph Template Language (GTL) and SGRENDER procedure. Users will appreciate how quick and easy it is to generate sophisticated graphs with ODS graphics.

BACKGROUND

Our department generates several reports and maps for the Center for Medicare and Medicaid Services (CMS) using Medicare claims data. One of the projects involved analyzing the Medicare population that overlaps across different measures. This project gave us experience with the SGPLOT and GTILE procedures.

While RTI is involved in several studies related to opioid consumption, none of these data have been used for this presentation. For demonstrating the techniques in this paper, we used opioid data provided by the Drug Enforcement Administration's Automation of Reports and Consolidated Orders System, known as ARCOS. This database allows researchers to learn how much hydrocodone and oxycodone went to individual states and counties, and from which companies and distributors. Examples in this paper are limited to North Carolina counties. In this database, data is available only for the years 2006 through 2012. We have used county level population data for the state of North Carolina from the census.gov website to calculate opioid doses per person.

From 2006 to 2012, there were 2,552,612,498 prescription pain pills supplied to North Carolina, which amount to 268 pills for every man, woman, and child in the state. In this paper, we will analyze this data through the heat maps in SGPLOT and GTILE graphs.

SAS® 9.4 M3 was used to run the code in this paper in SAS®/Enterprise Guide on SAS Linux Grid.

Figure 1 below is a county level map with opioid usage per person for the year 2012. This is included to illustrate geographic distribution of opioid consumption in the state of North Carolina. While neither GTILE or heat map table can fully display the spatial relationship as well as a map, we would need six additional maps like Figure 1 to display the data for the years 2006 through 2011.

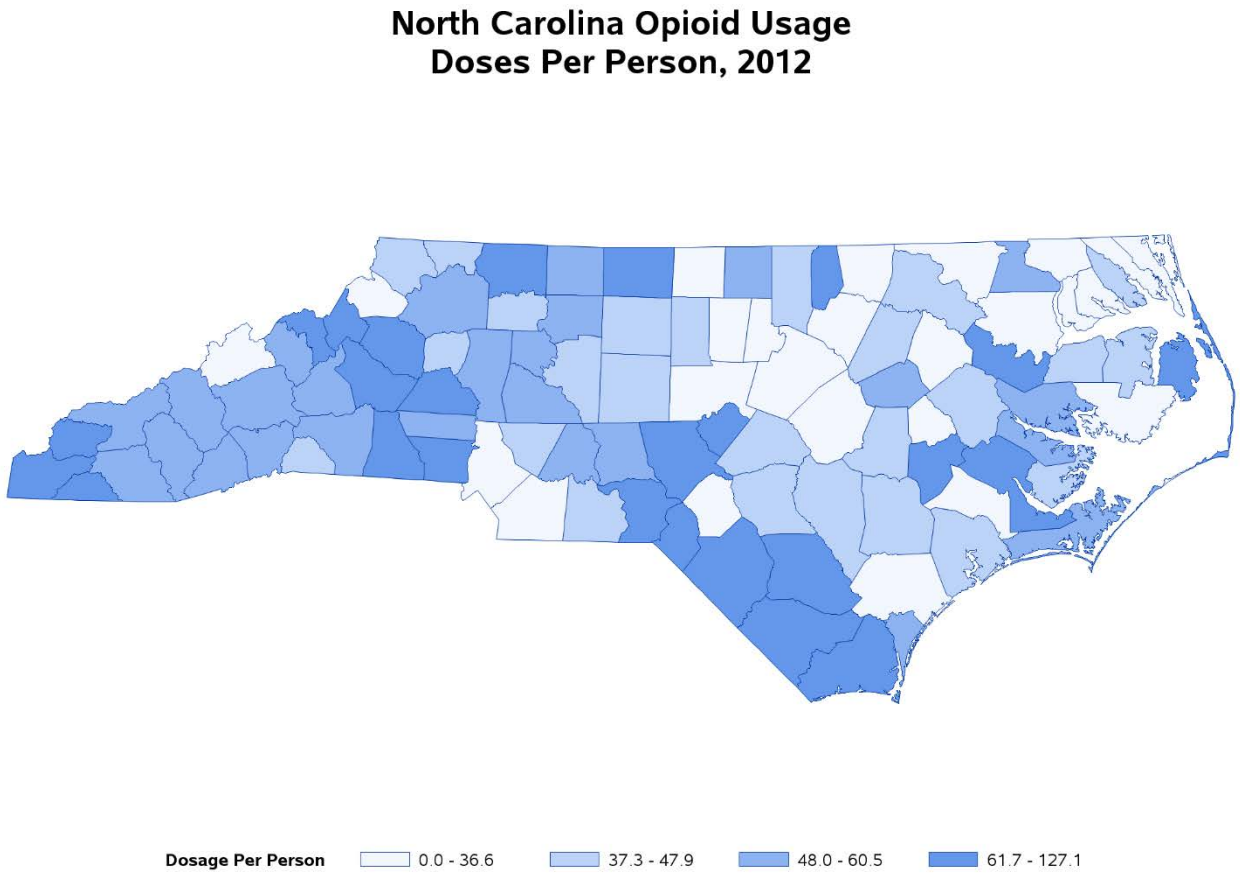

Figure 1 County level opioid consumption per person, for the year 2012

TOPICS COVERED IN THIS PAPER

- SGPLOT heat maps with code via ODS GRAPHICS
- GTILE graphs with code via SAS®/GRAPH
- Customized heat map using Graph Template Language (GTL) and SGRENDER Procedure

WORKING WITH ODS GRAPHICS

ODS Graphics is available in Base SAS® (version SAS 9.3 and beyond). In ODS Graphics, each graph is created based on a compiled graph template – a SAS® program written in GTL.

You can generate graphs using ODS Graphics in three ways:

1. Graphs from SAS®/STAT, SAS®/ETS, and SAS®/QC procedures and some Base SAS® Procedures such as FREQ, UNIVARIATE, and CORR
2. SG Procedures
    - SGPLOT
    - SGPANEL
    - SGSCATTER
3. Customized Graph Template Language (GTL) and SGRENDER Procedure

Of these, we will demonstrate SGPLOT and GTL techniques.

SGPLOT - HEAT MAPS VIA ODS GRAPHICS

SGPLOT supports HEATMAP and HEATMAPPARM statements starting with the release SAS® 9.4M3. Heat maps use colors to communicate numeric data by varying the underlying values that represent red, green, and blue (RGB) as a linear function of the data. Heat maps make tables easier to interpret since the cells are shaded with a color gradient based on the data value behind it. You can use heat maps to display spatial data, plot big data, and enhance tables. With big data, patterns in heat maps are clear, because colors are used to display the frequency of observations in each cell of the graph. Figure 2 below uses color to highlight opioid usage across all seven years of data, displaying the highest quartile counties. This helps replace the need for seven maps, as in Figure 1 and we can see the opioid usage trend rising over the years in the heat map.

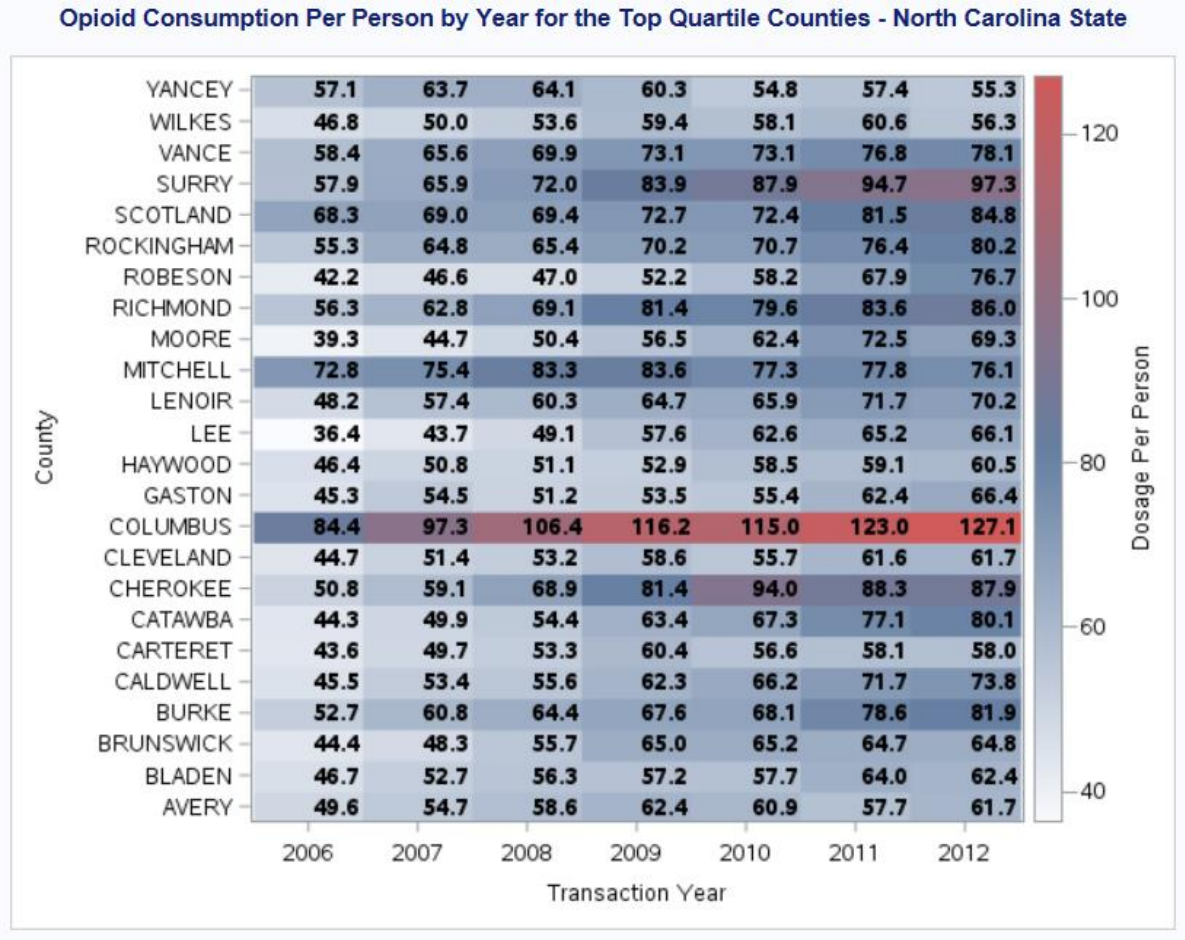

Figure 2 County level Heat Map for the top quartile counties

Dataset to create the heatmap above is generated from the program code in Appendix I. A few observations from the county_opioid_sales_by_quartile dataset is displayed below:

	BUYER_COUNTY	BUYER_STATE	quartile	trans_yr	dosage_sold	rsdnt_popcnt_yr2010	dosage_sold_pp	url1
1	ALAMANCE	NC	2	2006	4177597	151131	27.642	./ALAMANCE.html
2	ALAMANCE	NC	2	2007	5045760	151131	33.387	./ALAMANCE.html
3	ALAMANCE	NC	2	2008	5302980	151131	35.089	./ALAMANCE.html
4	ALAMANCE	NC	2	2009	5518180	151131	36.513	./ALAMANCE.html
5	ALAMANCE	NC	2	2010	5529390	151131	36.587	./ALAMANCE.html
6	ALAMANCE	NC	2	2011	6057440	151131	40.081	./ALAMANCE.html
7	ALAMANCE	NC	2	2012	6259710	151131	41.419	./ALAMANCE.html
8	ALEXANDER	NC	2	2006	1104800	37198	29.701	./ALEXANDER.html
9	ALEXANDER	NC	2	2007	1229380	37198	33.05	./ALEXANDER.html
10	ALEXANDER	NC	2	2008	1330050	37198	35.756	./ALEXANDER.html

Output 1: county_opioid_sales_by_quartile dataset

Below is the code to generate the heat map above (in Figure 2):

```
%let colormodel = (cxFAFBFE cx667FA2 cxD05B5B);
proc sgplot data=county_opioid_sales_by_quartile(where=(quartile=4))
      ❸tmplout="&pgmpath/opioidslstmplt_htmapparm.sas";
  ❶heatmapparm y=buyer_county x=trans_yr colorresponse=dosage_sold_pp /
                  name='lgnd' colormodel=&colormodel discretex NOMISSINGCOLOR
                              ;
  text y=buyer_county x=trans_yr text=dosage_sold_pp/
            textattrs=(size=10 weight=bold) ❹url=url1;
  ❷gradlegend 'lgnd'/integer title='Dosage Per Person' extractscale;
  title1 "Opioid Consumption Per Person by Year for Counties with high
      dosage per person - North Carolina State";
  format dosage_sold_pp comma12.1;
  label trans_yr="Transaction Year"   buyer_county= "County";
run;
```

In the code above, we used the COLORMODEL= option to specify the three colors as a color ramp from the htmlblue style: from the lowest levels of dosage per person in the neutral color (gray), to (blue), and then the highest levels of dosage per person in the color (red).

Explanation of the above code:

1. HEATMAPPARM statement generates table of cells based on the trans_yr and county variables. It colors the cells base on the response variable, dosage_sold_pp values.

2. GRADLEGEND statement generates gradient scale on the right side of the graph.

3. Tmplout =*"filename"* option in the PROC SGPLOT statement generates a SAS program with the template code behind this SG procedure.

4. Variable url1 specified in the HEATMAPPARM statement makes the cells as hyperlinks in the heat map for drill down capability, when used with the ODS HTML destination. This variable is created in the dataset used for the heat map and has the value *county name*.html. Each county needs to have its own .html file with its name.

GTILE GRAPH

We can drill down into each cell in the heat map table in Figure 2 above and analyze the data for each county and the year using GTILE graphs.

The GTILE procedure creates charts that consist of rectangles or squares that are divided into tile-shaped segments. These charts are sometimes referred to as rectangular tree maps. The GTILE charts represent the relative sizes of tiles to one another and to the whole. The GTILE procedure provides three statements that you can use to define the layout in order to visualize your data − FLOW, TILE, and TOGGLE. FLOW creates a chart that honors data order and can be read from left to right, while TILE creates a chart that orders data by value, descending from bottom left to top right. TOGGLE creates a chart that honors data order and can be read from left to right. When changing levels, the display toggles from one row to one column; with two TILEBY variables, TOGGLE resembles a stacked bar chart. We show FLOW as an example, since it provides the most intuitive layout.

The code below will generate a GTILE chart for each county in our dataset, using data on opioid doses sold by each distributor. Within the FLOW statement, two parameters are required: a numeric variable to determine the size of the tiles (DOSAGE_SOLD in our example), and a TILEBY parameter with at least one variable to order the layout. Since our code specifies TILEBY (TRANS_YR, DISTRIBUTOR), our graphs will consist of a major tile for each year, subdivided by the company distributing the opioid doses. The number of doses sold determines the size of both the major and minor tiles.

```
%let colormodel = (cxFAFBFE cx667FA2 cxD05B5B);
options device=javaimg;
proc gtile data=SASdataset;
  flow dosage_sold
       tileby=(trans_yr,distributor)  /
       colorramp=&colormodel    colorvar=trans_yr
       labellevel=2      detaillevel=2;
by county_name;
label trans_yr="Year";
run;
```

While only those two parameters (TRANS_YR, DISTRIBUTOR), are required to structure a GTILE chart, much of the impact comes from the optional parameters. COLORRAMP determines which colors will be used in our chart, and we have been consistent with the SGPLOT heat maps shown earlier. COLORVAR defaults to the numeric size variable; by specifying TRANS_YR, the groupings are more distinct, and we overcome the lack of size specificity by including the dosage sold as part of the distributor's name. Since DISTRIBUTOR is the second variable in our TILEBY parameter, LABELLEVEL and DETAILLEVEL are both set to 2 to give us the maximum amount of information for this chart; setting either of them to 1 result in a chart that is much less useful.

GTILE Graph for Columbus county is shown below in Figure 3.

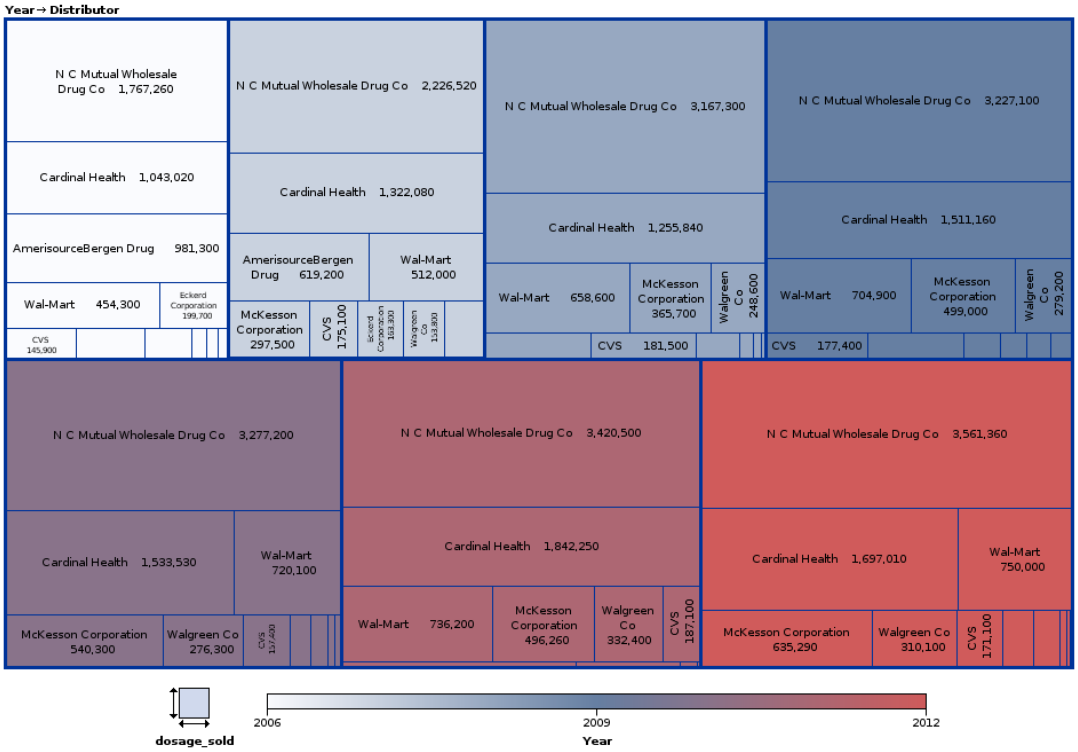

Figure 3 GTILE Graph for Columbus County

ENHANCED TABLES

What if we want to highlight the cells with top five and bottom five values in the above heat map in Figure 2? We can accomplish this task by overlaying two scatter plots, one for the top five counties and another one for the bottom five counties to the table in Figure 2.

This task is accomplished with the code below:

First, identify the top five and bottom five cells in the data and then create two variables top_dose_cnty and bottom_dose_cnty in the dataset.

```
proc sort data=county_opioid_sales_by_quartile(where=(quartile=4)) out=
                                        top_quartile_counties;
  by dosage_sold_pp;
run;

data _null_;
  set top_quartile_counties nobs=n;

   if _n_ = 5 then do;
      call symput("bottom5" , put(dosage_sold_pp, 5.2));
   end;
   if _n_ = (n-5) then do;
      call symput("top5" , put(dosage_sold_pp, 5.2));
   end;
```

```
 run;

/* Create two variables top_dose_cnty and bottom_dose_cnty in the dataset */

data counties_with_top_bottom_doses;
  set county_opioid_sales_by_quartile(where=(quartile=4));

  if dosage_sold_pp >= &top5 then top_dose_cnty = buyer_county;
  else top_dose_cnty = " ";


  if dosage_sold_pp <= &bottom5 then bottom_dose_cnty = buyer_county;
  else bottom_dose_cnty = " ";

run;

/* Now overlay two scatter statements to the SGPLOT procedure as  below*/

%let colormodel = (cxFAFBFE cx667FA2 cxD05B5B);

proc sgplot data=counties_with_top_bottom_doses(where=(quartile=4))
             tmplout="&pgmpath/tmplt_htmap_scatter.sas" noautolegend;
  heatmapparm  y=buyer_county x=trans_yr colorresponse=dosage_sold_pp /
            name='lgnd'
            colormodel=&colormodel discretex NOMISSINGCOLOR;
  ❶scatter y=top_dose_cnty x=trans_yr/ filledoutlinedmarkers
      markerfillattrs=(color=red)
      markeroutlineattrs=(color=red thickness=2)
            markerattrs=(symbol=starfilled size=15);

  ❷scatter y=bottom_dose_cnty x=trans_yr/filledoutlinedmarkers
            markerfillattrs=(color=yellow)
            markeroutlineattrs=(color=yellow thickness=2)
            markerattrs=(symbol=squarefilled size=15 );

  text    y=buyer_county x=trans_yr text=dosage_sold_pp /
                  textattrs=(size=10 weight=bold)  url=url1;

  gradlegend 'lgnd'/integer title='Dosage Per Person' extractscale;

  title1 "Opioid Consumption Per Person by Year for the Top Quartile Counties
- North Carolina State";
  format dosage_sold_pp comma12.1;
  label trans_yr="Transaction Year"
        buyer_county= "County";

run;
```

By inserting the two scatter statements to the SGPLOT procedure, a heat map table highlighted with top five and bottom five cells is generated as below:

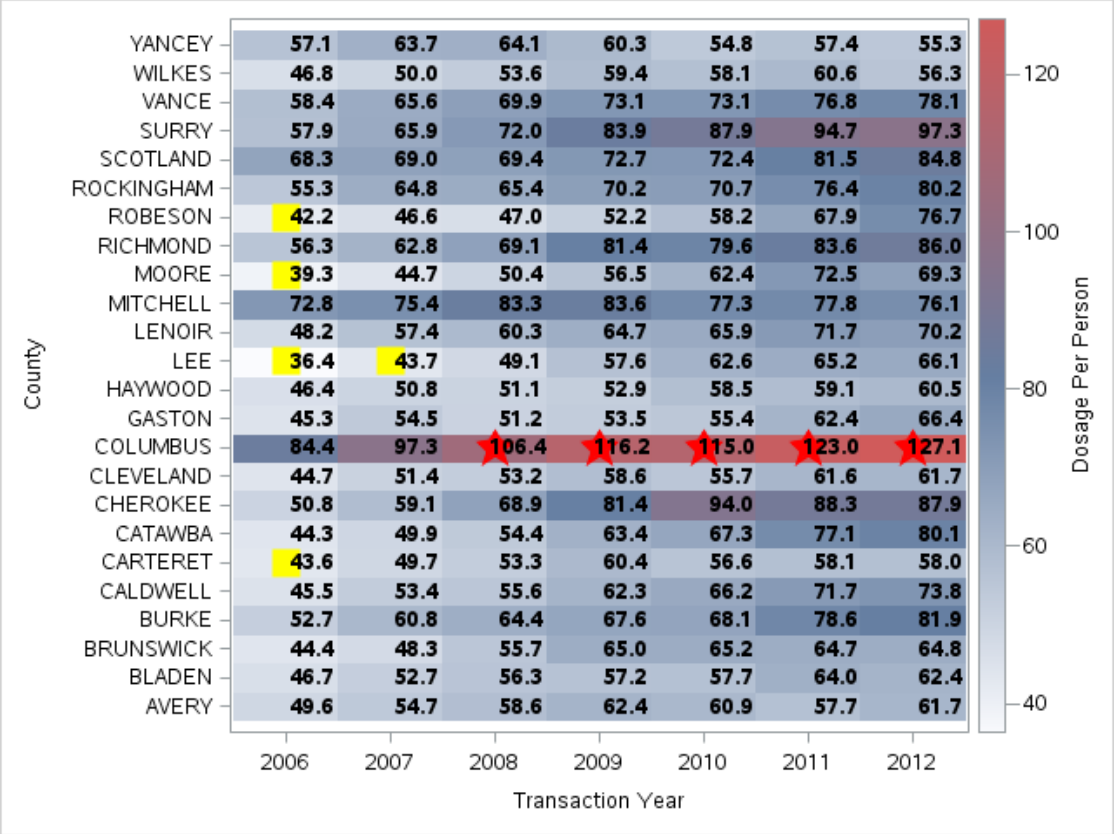

Figure 4 Heat map with squares and stars for the bottom and top cells, respectively

GRAPH TEMPLATE LANGUAGE (GTL)

In the above examples, as we have seen with the SGPLOT procedure, we can do a simple plot (Figure 2) or overlay multiple plots (Figure 4). Once we start building graphs and tables with the SG procedures, we quickly realize the need to include additional features to our graphs and tables and we realize that the SG procedure options are not enough. This requires learning Graph Template Language (GTL).

Graph Template Language (GTL) in ODS Graphics enables you to have total control over the layout and attributes of the graph. One can generate customized, sophisticated graphs easily with the help of GTL. Coding style in GTL is very different from SAS/GRAPH. GTL is the heart of ODS Graphics. Graphs in SG procedures are generated using GTL behind the scenes. GTL is structured, powerful, flexible, and intuitive.

In GTL, graphs are built by using plot and layout statements. The plot statements determine how data are represented in the graph, and the layout statements determine where the plots are drawn on the graph. More advanced layouts can be used to divide the plot area into multiple independent cells. In addition, statements can be nested so multiple plots can be arranged to create elegant visuals.

Creating a graph using GTL is a two-step process -

- Define the structure of the graph using the GTL syntax in a STATGRAPH template. Compile and save the template.

- Generate the graph using the SGRENDER procedure by specifying the appropriate data set and the template.

GTL CODE BEHIND AN SGPLOT PROCEDURE

In the program code for the heat map (Figure 4. above), tmplout=*"filename"* option saved the GTL code behind the heat map as a SAS program : tmplt_htmap_scatter.sas. This program is listed below:

```
proc template;
  define statgraph sgplot;
  begingraph / collation=binary discreteAxisOffsetPad=false;
  EntryTitle "Opioid Consumption Per Person by Year for the Top Quartile
             Counties - North Carolina State" /;
  layout overlay / xaxisopts=(labelFitPolicy=Split)
  x2axisopts=(labelFitPolicy=Split);
  HeatMapParm X='trans_yr'n Y='BUYER_COUNTY'n
  ColorResponse='dosage_sold_pp'n / xBinAxis=false yBinAxis=false
  includemissingcolor=false ColorModel=(CXFAFBFE CX667FA2 CXD05B5B )
  primary=true discretex=true NAME="lgnd";
  ScatterPlot X='trans_yr'n Y='top_dose_cnty'n / subpixel=off Markerattrs=(
  Symbol=STARFILLED Size=15) filledOutlinedMarkers=true
  MarkerFillattrs=(Color=CXFF0000)
                 MarkerOutlineattrs=( Color=CXFF0000 Thickness=2)
  LegendLabel="top_dose_cnty" NAME="SCATTER";
  ScatterPlot X='trans_yr'n Y='bottom_dose_cnty'n / subpixel=off
  Markerattrs=( Symbol=SQUAREFILLED Size=15) filledOutlinedMarkers=true
  MarkerFillattrs=(Color=CXFFFF00)
   MarkerOutlineattrs=( Color=CXFFFF00 Thickness=2)
   LegendLabel="bottom_dose_cnty" NAME="SCATTER1";
   TextPlot X='trans_yr'n Y='BUYER_COUNTY'n text='dosage_sold_pp'n /
   TextAttrs=( Size=10 Weight=bold ) URL='url1'n LegendLabel="County"
   NAME="TEXT";
   ContinuousLegend "lgnd" / Location=Outside halign=right Title="Dosage Per
   Person" extractscale=true Integer=true;
endlayout;
endgraph;
end;
run;
```

Explanation of the above code:

This program consists of a PROC TEMPLATE statement followed by three or more nested blocks of statements:

- DEFINE/END block. It begins with a DEFINE STATGRAPH statement followed by a template name and ends with an END statement. This step creates a graph template called sgplot.
- BEGINGRAPH / ENDGRAPH block. It provides a place for you to specify options that affect the graph size, the graph border, and the graph background color. Most statements that construct the graph are specified inside this block. The graph title is specified in the ENTRYTITLE statement is placed right after the BEGINGRAPH statement.
- LAYOUT OVERLAY/ENDLAYOUT block. Statements that create the graph are placed in this block. There is no limit to the number of graph statements that you can place, in this block. The HEATMAPPARM statement, two SCATTERPLOT statements for top five counties and bottom five counties, and TEXTPLOT statement are placed within the LAYOUT OVERLAY block in the above code.

Now, use the template sgplot in PROC SGRENDER with the dataset counties_with_top_bottom_doses as below, to create the heat map in Figure 4.

```
proc sgrender data= counties_with_top_bottom_doses(where=(quartile=4))
                    template=sgplot;
  format dosage_sold_pp comma12.1;
  label trans_yr="Transaction Year"
        buyer_county= "County";
run;
```

INCLUDE BORDERS AND LEGEND FOR THE TOP AND BOTTOM ENTRIES

Let us make some modifications to the above GTL code above to include

- Borders for each cell in the table
- Instead of hard coding the colors for the color ramp, use the style-element: attribute to derive the colors from the style colors.
- Include legend for the bottom and top cells.

```
proc template;
  define statgraph mytmplt;
  begingraph / collation=binary discreteAxisOffsetPad=false;

❶* Add legend items;
   legendItem type=marker name="b_marker" /
        markerattrs=(color=yellow symbol=squarefilled)
        label="bottom 5 " ;
   legendItem type=marker name="t_marker" /
        markerattrs=(color=red  symbol=starfilled)
        label="top 5 " ;

  layout overlay / xaxisopts=(labelFitPolicy=Split)
```

```
                  x2axisopts=(labelFitPolicy=Split);
   HeatMapParm X='trans_yr'n Y='BUYER_COUNTY'n
ColorResponse='dosage_sold_pp'n / xBinAxis=false yBinAxis=false
includemissingcolor=false
❷ColorModel=(GraphWalls:Color ThreeColorRamp:StartColor) display=all
includemissingcolor=false
❸outlineattrs=graphdata2(thickness=1) primary=true discretex=true
NAME="lgnd";

   ScatterPlot X='trans_yr'n Y='top_dose_cnty'n / subpixel=off
Markerattrs=( Symbol=STARFILLED Size=15) filledOutlinedMarkers=true
MarkerFillattrs=(Color=CXFF0000) MarkerOutlineattrs=( Color=CXFF0000
Thickness=2) LegendLabel="top_dose_cnty" NAME="SCATTER";

   ScatterPlot X='trans_yr'n Y='bottom_dose_cnty'n / subpixel=off
Markerattrs=( Symbol=SQUAREFILLED Size=15) filledOutlinedMarkers=true
MarkerFillattrs=(Color=CXFFFF00) MarkerOutlineattrs=( Color=CXFFFF00
Thickness=2) LegendLabel="bottom_dose_cnty" NAME="SCATTER1";

   TextPlot X='trans_yr'n Y='BUYER_COUNTY'n text='dosage_sold_pp'n /
TextAttrs=( Size=10 Weight=bold ) URL='url1'n LegendLabel="County"
NAME="TEXT";

   ContinuousLegend "lgnd" / Location=Outside halign=right
Title="Dosage Per Person" extractscale=true Integer=true;

  * include the legend;
  ❹discretelegend "b_marker" "t_marker" /
          autoitemsize=true;

  endlayout;
  endgraph;
  end;
run;
```

In the above code:

- Statements marked 1 inserted the legend items.
- Option marked 2 used colors from white (GraphWalls:Color) to blue (ThreeColorRamp:StartColor)
- Option marked 3: outlineattrs=graphdata2(thickness=1) outlined the borders for each cell.

Now run the SGRENDER procedure using the template mytmplt to generate the table in Figure 5.

```
proc sgrender data= counties_with_top_bottom_doses(where=(quartile=4))
                    template=mytmplt;
  title1 "Opioid Consumption Per Person by Year for the Top Quartile Counties
- North Carolina State";
```

```
  format dosage_sold_pp comma12.1;
  label trans_yr="Transaction Year"
        buyer_county= "County";
run;
```

The code above generates the heat map table shown in Figure 5 below.

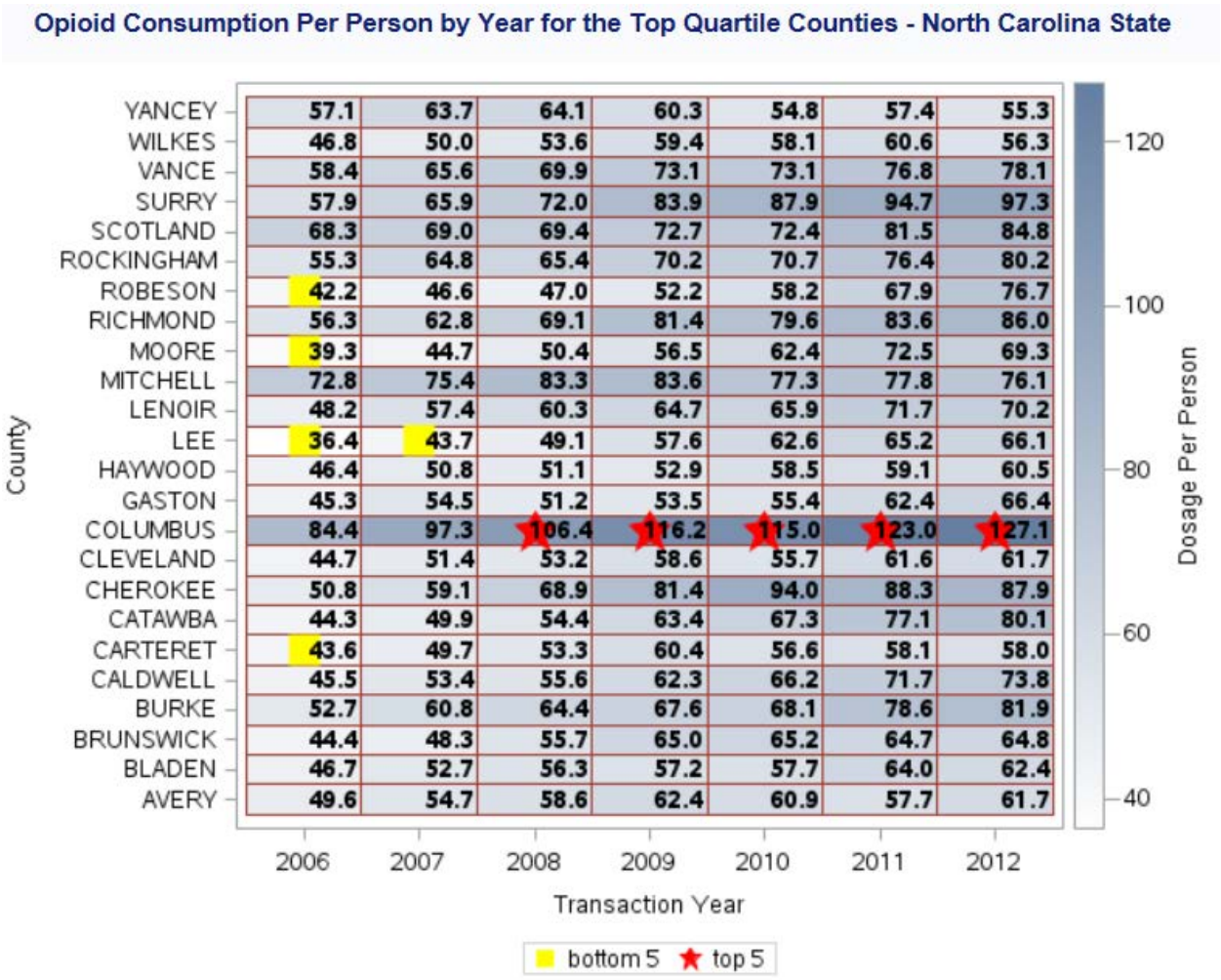

Figure 5: Heat map with squares and stars for the bottom and top cells, respectively

CONCLUSION

Datasets are getting larger and heat maps are a great way to display a large amount of data either in a map or a table. With the GTILE graphs, the response values are shown by both color gradient and area. It is easy to identify key values or problematic values in the data through heat maps and GTILE graphs. Each of these techniques, heat maps using ODS graphics as well as GTILE, has its place. Heat maps make tables and graphs interpret with the shaded background color, whereas the GTILE graphs make use of shaded color and area to display the response values. Graph Template Language (GTL) with the SGRENDER procedure makes these graphs even more sophisticated and powerful. In this paper, we have only scratched the surface of GTL.

REFERENCES

Kufeld, Warren F. (2017) "Heat Maps: Graphically Displaying Big Data and Small Tables" *Proceedings of the 2017 SAS Global Forum Conference*. Available at https://support.sas.com/resources/papers/proceedings17/SAS0312-2017.pdf

Matange Sanjay, "Getting Started with the Graph Template Language in SAS, Examples, Tips, and Techniques for Creating Custom Graphs"

Kufeld, Warren F. "Basic ODS Graphics Examples" (http://support.sas.com/documentation/prod-p/grstat/9.4/en/PDF/odsbasicg.pdf).

Kufeld, Warren F. "Advanced ODS Graphics Examples" (http://support.sas.com/documentation/prod-p/grstat/9.4/en/PDF/odsadvg.pdf).

Matange, Sanjay's blog Graphically Speaking on tips, tricks, and the latest developments in ODS Graphics (http://blogs.sas.com/content/graphicallyspeaking/) .

SAS® 9.4 ODS Graphics: Procedures Guide

ACKNOWLEDGMENTS

The authors are grateful to Sarah Lein, RTI and Kavya Sekar, Analyst in Health Policy, Congressional Research Service for their valuable time in editing this paper and providing helpful suggestions.

CONTACT INFORMATION

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

Devi Sekar
Center for Health Data Analytics
RTI International
3040 East Cornwallis Road
Research Triangle Park, NC 27709
Email: dsekar@rti.org

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

APPENDIX I

Program code to generate the heat map in Figure 2:

```
libname sesug19 "/pgm/data/dsekar/SESUG19";

%let Pathout=/pgm/data/dsekar/SESUG19/;

%let Pgmpath=/pgm/dsekar/SESUG19/opioid_programs;

data nc_population;
   set sesug19.population_data;
   if substr(stcou,1,2) = "37";
   county_name = substr(area_name,1,index(area_name,",")-1);
   county_name = upcase(county_name);

run;

data nc_opioid_data;
   set sesug19.nc_opioid_data;

   trans_yr = year(transaction_date);

   if dosage_unit ne "null" then
        dosage = input(dosage_unit, 8.);
   else
      dosage = 0;

run;


proc sql;
   create table county_opioid_sales as
   select buyer_county, buyer_state, sum(dosage) as dosage_sold
   from nc_opioid_data
   group by buyer_county, buyer_state;

   create table county_opioid_sales_pp as
   select a.*, b.rsdnt_popcnt_yr2010,
               round(dosage_sold/rsdnt_popcnt_yr2010,0.001) as dosage_sold_pp
   from county_opioid_sales a left join nc_population b
   on a.buyer_county=b.county_name;

   create table county_opioid_sales_by_yr as
   select buyer_county, buyer_state, trans_yr, sum(dosage) as dosage_sold

   from nc_opioid_data
   group by buyer_county, buyer_state, trans_yr;

   create table county_opioid_sales__by_yr_pp as
   select a.*, b.rsdnt_popcnt_yr2010,
          round(dosage_sold/rsdnt_popcnt_yr2010,0.001) as dosage_sold_pp
```

```
   from county_opioid_sales_by_yr a left join nc_population b
   on a.buyer_county=b.county_name;
quit;


/* Generate Quartiles based on the dosage_sold_pp value */

proc rank data=county_opioid_sales_pp groups=4 out=dosage_pp_quartiles;
   var dosage_sold_pp;
   ranks quartile;
run;

data dosage_pp_quartiles;
   set dosage_pp_quartiles;

   quartile = quartile+1;
run;

/* Append quartile value to the county_year sales data */

data county_opioid_sales_by_quartile;
   merge dosage_pp_quartiles(in=a keep=buyer_state buyer_county quartile)
         county_opioid_sales__by_yr_pp(in=b);
   by buyer_state buyer_county;
   if a and b;
   length url1 $30;

   /* Create variable url1 for generating hyperlinks in the heat map */

   url1="./" || compress(buyer_county) || '.html';
run;

%let colormodel = (cxFAFBFE cx667FA2 cxD05B5B);

ods graphics on / imagemap=on border=off;

ods html file="SESUG19_opioid_Sales_HeatMap.html" path="&pathout." (url=none)
nogtitle;

proc sgplot data=county_opioid_sales_by_quartile(where=(quartile=4))
tmplout="&pgmpath/opioidslstmplt_htmapparm.sas";
heatmapparm  y=buyer_county x=trans_yr colorresponse=dosage_sold_pp /
name='lgnd'
            colormodel=&colormodel discretex NOMISSINGCOLOR;

text        y=buyer_county x=trans_yr text=dosage_sold_pp /
textattrs=(size=10 weight=bold)  url=url1;
gradlegend 'lgnd'/integer title='Dosage Per Person' extractscale;
title1 "Opioid Consumption Per Person by Year for the Top Quartile Counties -
North Carolina State";
format dosage_sold_pp comma12.1;
label trans_yr="Transaction Year"
      buyer_county= "County";

run;
ods html close;
ods graphics off;
```

Program code to generate the GTILE graph in Figure 3:

Note: This code produces an HTML page for each of North Carolina's 100 counties.

```
proc sort data=sesug19.top10_distributor_sales_by_yr
          out= top10_distributor_sales_by_yr;
by buyer_county trans_yr descending dosage_sold;
run;

data top10_distributor_sales_by_yr;
set top10_distributor_sales_by_yr;
by buyer_county;
if first.buyer_county then county_num+1;
county_name=propcase(buyer_county);
if county_name='Mcdowell' then county_name='McDowell';
run;

%macro gtile1;
%do i=1 %to 100;
data gmap_temp;
set top10_distributor_sales_by_yr(where=(county_num=&i));
distributor=trim(distributor) || ' - ' ||
left(put(dosage_sold,comma12.));
call symput('county',compress(trim(buyer_county)));
run;

ods html file="&county..html" path="&pathout." (url=none) nogtitle
style=seaside;
%let colormodel = (cxFAFBFE cx667FA2 cxD05B5B);
options device=javaimg;
run;
proc gtile data=gmap_temp;
  flow dosage_sold  tileby=(trans_yr,distributor)  /
    colorramp=&colormodel colorvar=trans_yr
    labellevel=2
    detaillevel=2;
  by county_name;
  label trans_yr="Year";
title1 h=5 "#byval(county_name) County Opioid Sales";
;
  run;
ods html close;
%end;
%mend gtile1;
%gtile1;
```