SESUG Paper 292-2019

What's in the pipeline for the SGMAP Procedure?!?

Robert Allison, SAS Institute

ABSTRACT

SGMAP is a fairly new procedure in BASE SAS, and is undergoing rapid development and enhancements. It currently has rich capabilities for plotting data on tile-based maps, such as OpenStreetMaps® (as of release 9.4 maintenance 6), but only basic capabilities for creating choropleth (polygon-based) maps. In this paper I describe several new features under development that will help round out the choropleth mapping capabilities. These new features are "all but done" and should be available in a SAS release in the near future.

INTRODUCTION

If you’ve tried creating choropleth maps in version 9.4 maintenance 6 (9.4m6) of the SGMAP procedure, you might have noticed it only has basic functionality, which makes it a bit cumbersome to use. In future releases, we will be greatly enhancing the functionality, and making it much easier to use (with options more like the other ODS Graphics procedures, such as SGPLOT). In the examples below, I show two versions of each map – first the current 9.4m6 version of the code, and then the code you will be able to use in a future release.

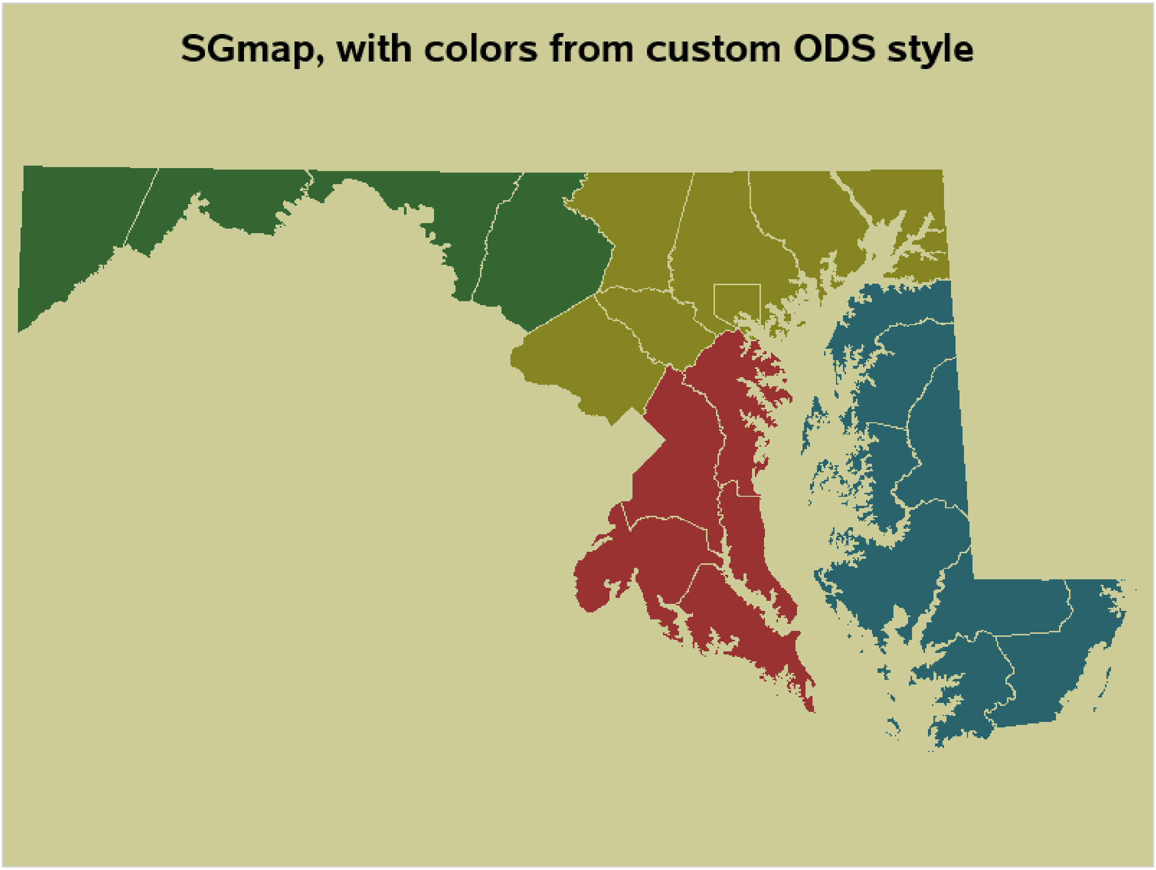
SPECIFYING DISCRETE COLORS FOR MAP AREAS

This example demonstrates how to use color in a choropleth map containing discrete values (either discrete numeric values, or character values). This is the kind of map I use most often. Let’s start by creating the data (my_data) and map (my_map) datasets.

```
data my_data; set mapsgfk.us_counties_attr (where=(statecode="MD"));
length Region $20;
if idname in ('Garrett' 'Allegany' 'Washington' 'Frederick')
 then region='Western';
if idname in ('Carroll' 'Baltimore' 'Baltimore City' 'Harford'
 'Cecil' 'Howard' 'Montgomery') then region='Central';
if idname in ('Anne Arundel' "Prince George's" 'Calvert'
 'Charles' "St. Mary's") then region='Southern';
if idname in ('Kent' "Queen Anne's" 'Caroline' 'Talbot' 'Dorchester'
 'Wicomico' 'Somerset' 'Worcester') then region='Eastern';
run;

data my_map; set mapsgfk.us_counties (where=(statecode="MD"));
run;
proc gproject data=my_map out=my_map (drop = lat long)
 latlong eastlong degrees;
id county;
run;
```

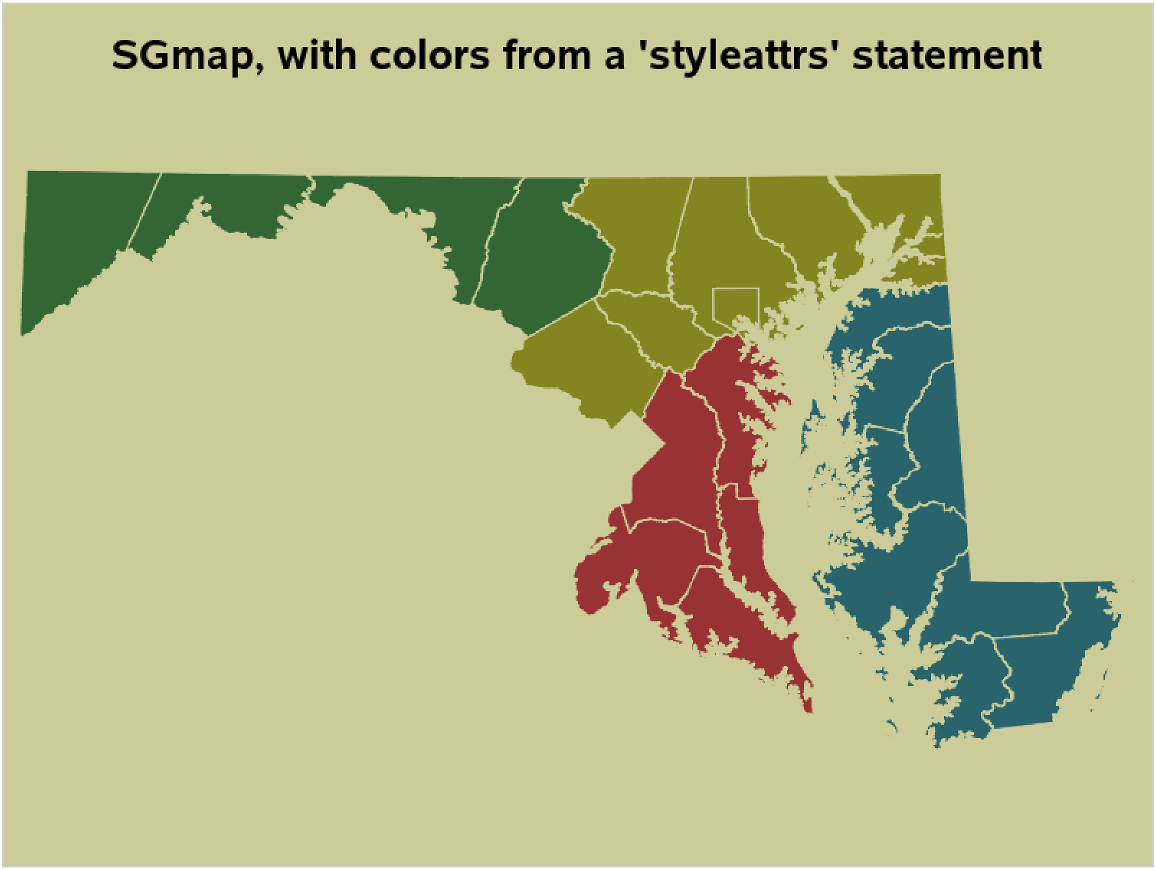
When plotting a choropleth map in 9.4m6, if you don't want the default colors, then you must modify the ODS style to change the colors. The following code modifies the htmlblue style to change the colors of the map polygons, and the background behind the map. I'm using hexadecimal RGB color codes, but you could also use color names (such as red, purple, or dodgerblue). I name my modified style 'md_style' - 'MD' is the abbreviation for Maryland.

```
ods path(prepend) work.templat(update);
proc template;
define style styles.md_style;
 parent=styles.htmlblue;
 style graphbackground / color=cxcccc99;
 class graphcolors /
  'gdata1'=cx336633
  'gdata2'=cx993333
  'gdata3'=cx848421
  'gdata4'=cx29636b
  ;
 end;
run;

ODS HTML style=md_style;

title1 "SGmap, with colors from custom ODS style";
proc sgmap maprespdata=my_data mapdata=my_map noautolegend;
choromap region / mapid=county lineattrs=(color=cxcccc99);
run;
```



Modifying ODS styles is a bit cumbersome, but in the future you will be able to control the colors more easily/directly using a styleattrs statement (just like you use in SGPLOT). This will shorten your code considerably!

```
title1 "SGmap, with colors from a 'styleattrs' statement";
proc sgmap maprespdata=my_data mapdata=my_map noautolegend;
styleattrs datacolors=(cx336633 cx993333 cx848421 cx29636b)
 backcolor=cxcccc99;
choromap region / mapid=county lineattrs=(color=cxcccc99);
run;
```



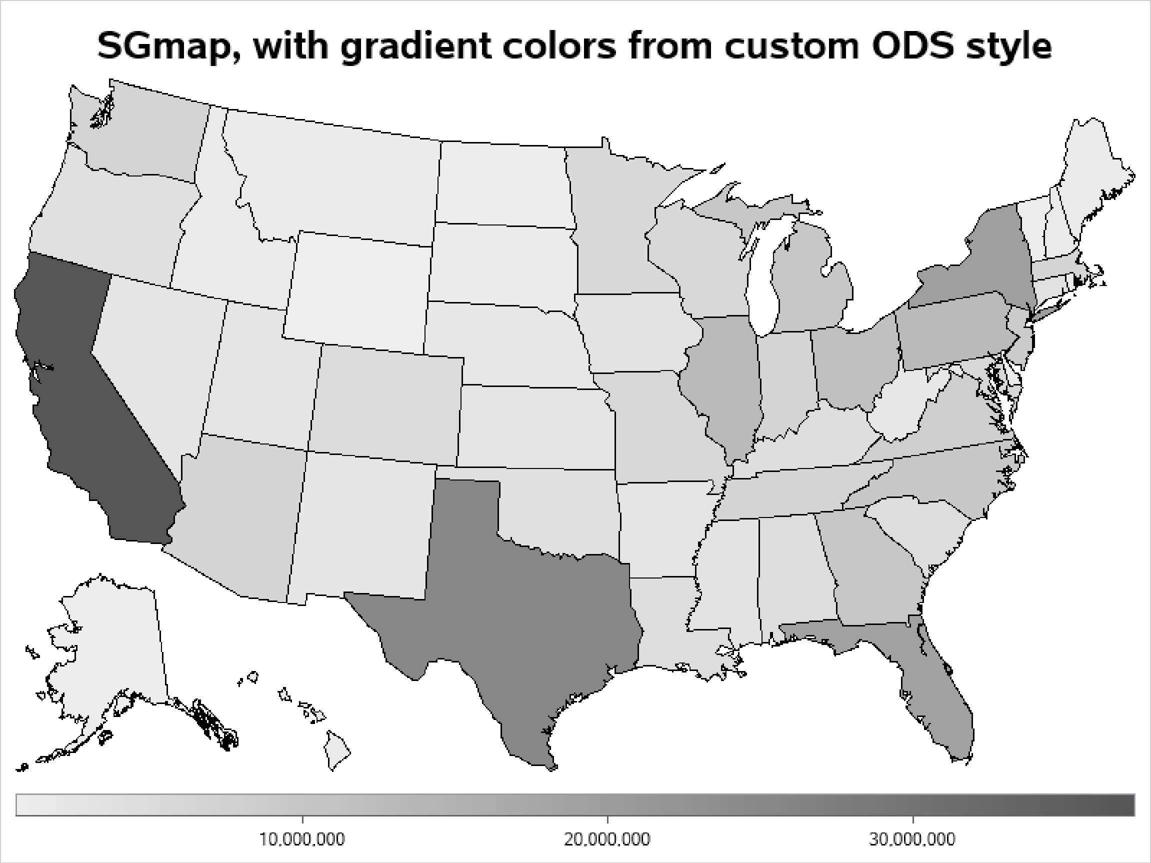
SPECIFYING GRADIENT COLORS FOR MAP AREAS

In 9.4m6, you must modify the ODS style, and change the startcolor and endcolor, to control the colors used in a choropleth map with gradient/continuous shading.

```
ods path(prepend) work.templat(update);
proc template;
define style styles.my_grad;
 parent=styles.htmlblue;
  style twocolorramp / startcolor=grayee endcolor=gray55;
 end;
run;
```

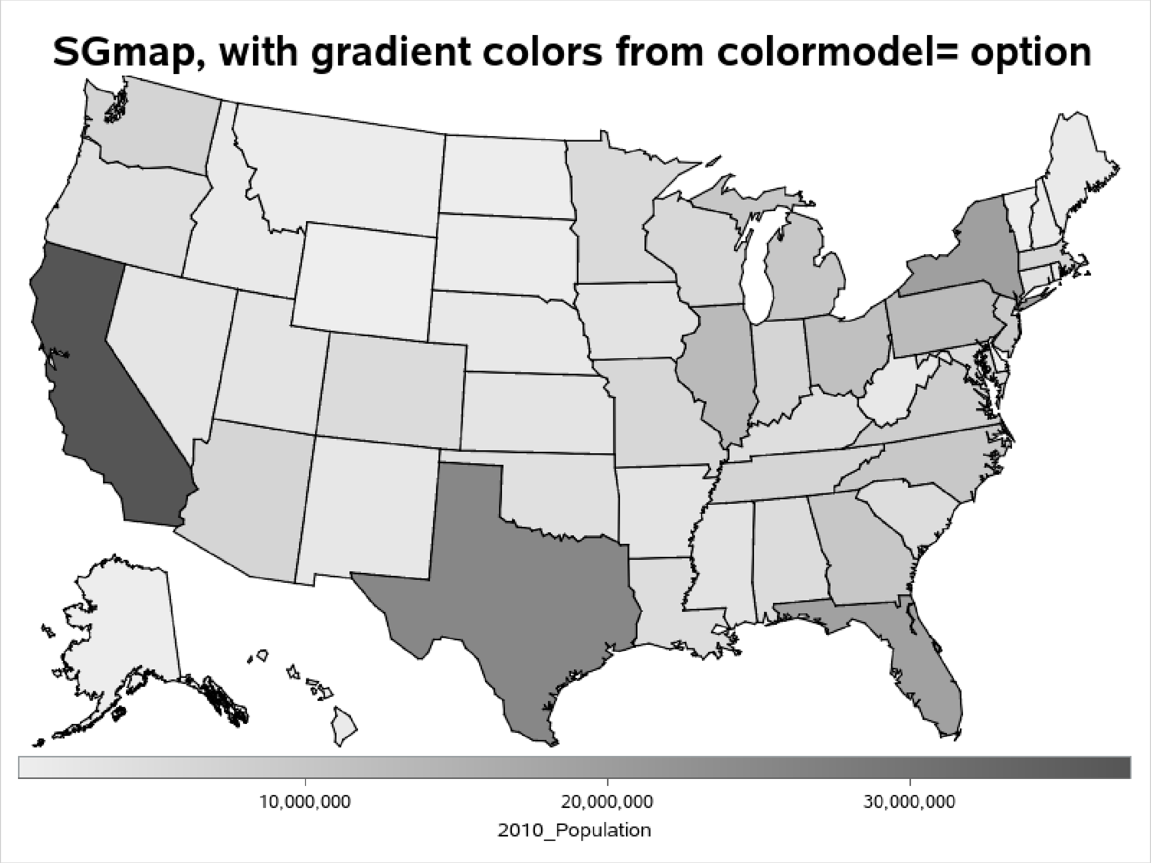
```
ODS HTML style=my_grad;
```

```
title1 "SGmap, with gradient colors from custom ODS style";
proc sgmap maprespdata=sashelp.us_data mapdata=mapsgfk.us;
choromap population_2010 / mapid=statecode id=statecode;
run;
```



In a future release, you will be able to specify the colors for the ends of the gradient via a simple colormodel= option in SGMAP.

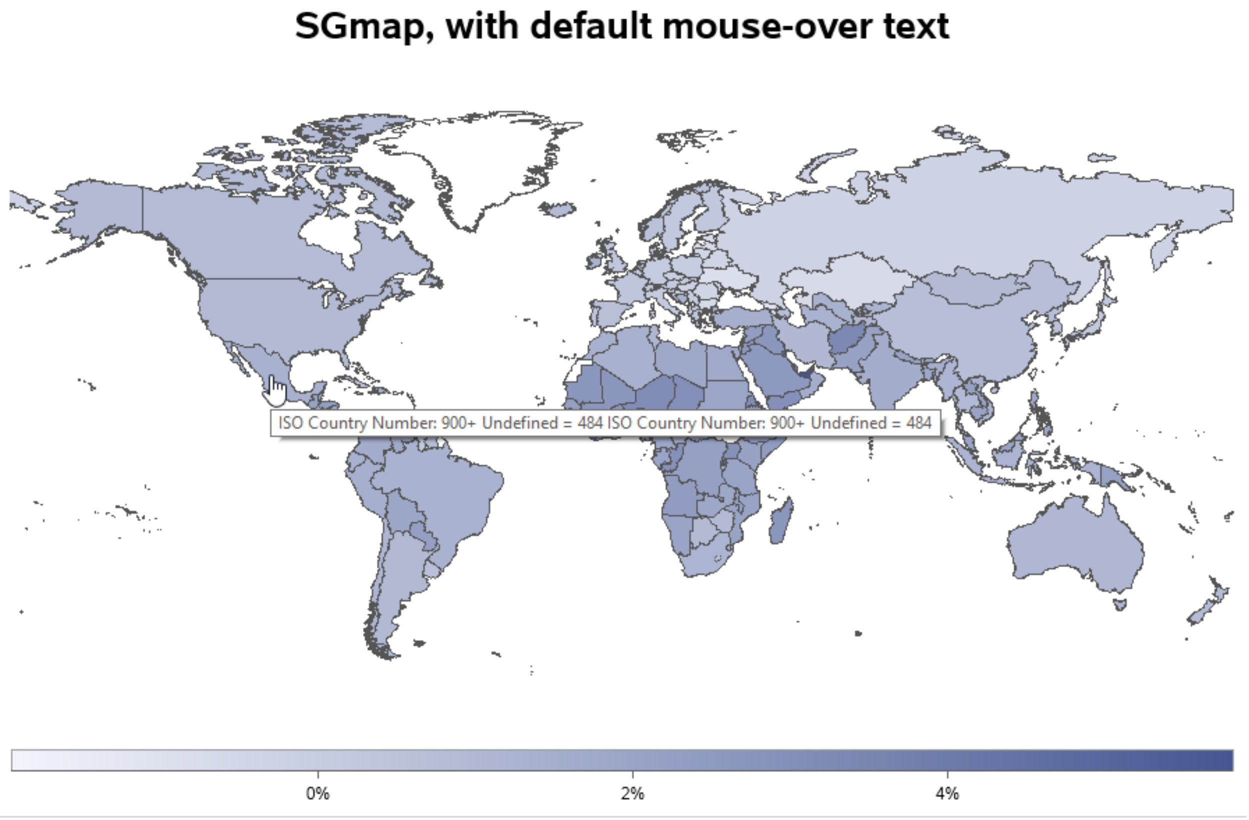
```
title1 "SGmap, with gradient colors from colormodel= option";
proc sgmap maprespdata=sashelp.us_data mapdata=mapsgfk.us;
choromap population_2010 / mapid=statecode id=statecode
 colormodel=(grayee gray55);
run;
```



DEFAULT MOUSE-OVER TEXT

In 9.4m6, the default mouse-over text is all on one line, when you turn on the ‘imagemap’ ODS graphics option in SGmap. It also shows the id variable twice - once for the map dataset, and once for the response dataset. Notice in the screen-capture below, that it is difficult to discern where the label stops and the value starts (especially in this case, where the label also contains numbers):

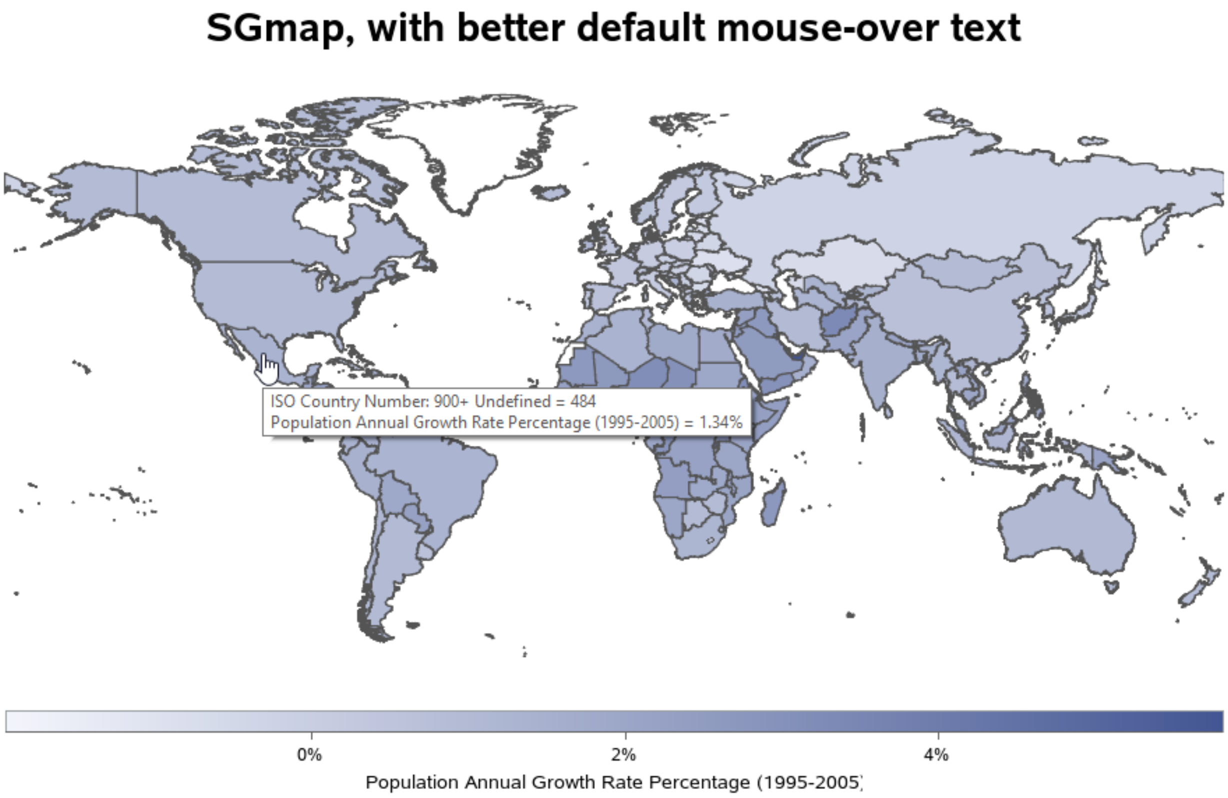
```
data my_map; set mapsgfk.world
  (where=(idname^='Antarctica' and density<=2) drop=lat long);
iso_num=.; iso_num=iso;
run;
ods graphics / imagemap tipmax=2500;

title1 "SGmap, with default mouse-over text";
proc sgmap maprespdata=sashelp.demographics mapdata=my_map;
choromap / mapid=iso_num id=iso lineattrs=(color=gray55);
choromap popAGR / mapid=iso_num id=iso lineattrs=(color=gray55);
run;
```



ISO Country Number: 900+ Undefined = 484 ISO Country Number: 900+ Undefined = 484

In a future release, the default mouse-over text will be split onto multiple lines (one line per variable). And since (by definition) the id variables have the same value in the map and response datasets, that value will only be shown once in the default mouse-over text. Notice in the screen-capture below, it is **much** easier to read the values now:

```
ods graphics / imagemap tipmax=2500;

title1 "SGmap, with better default mouse-over text";

proc sgmap maprespdata=sashelp.demographics mapdata=my_map;
choromap / mapid=iso_num id=iso lineattrs=(color=gray55);
choromap popAGR / mapid=iso_num id=iso lineattrs=(color=gray55);
run;
```

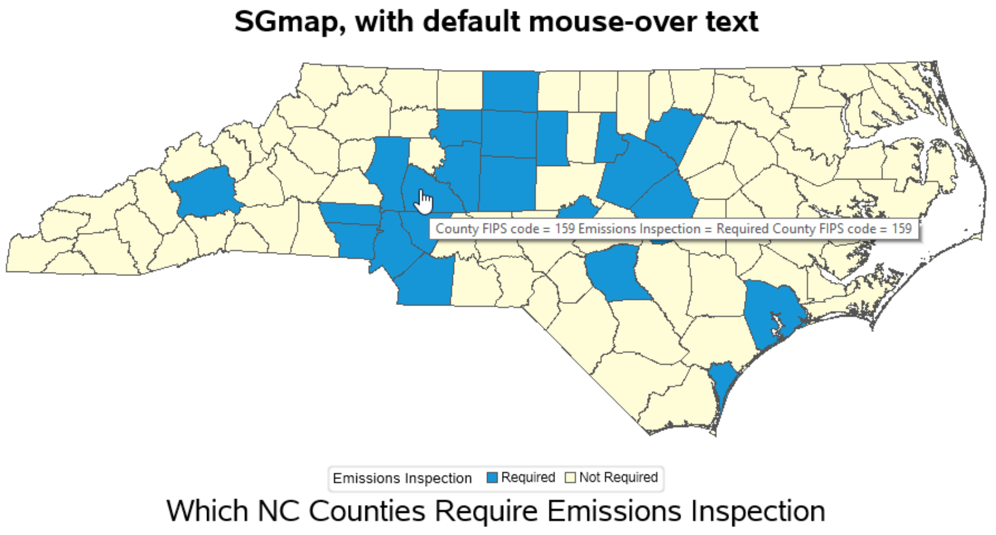

SGmap, with better default mouse-over text
0%
2%
4%
Population Annual Growth Rate Percentage (1995-2005)

ISO Country Number: 900+ Undefined = 484
Population Annual Growth Rate Percentage (1995-2005) = 1.34%

CUSTOM MOUSE-OVER TEXT AND DRILL-DOWN

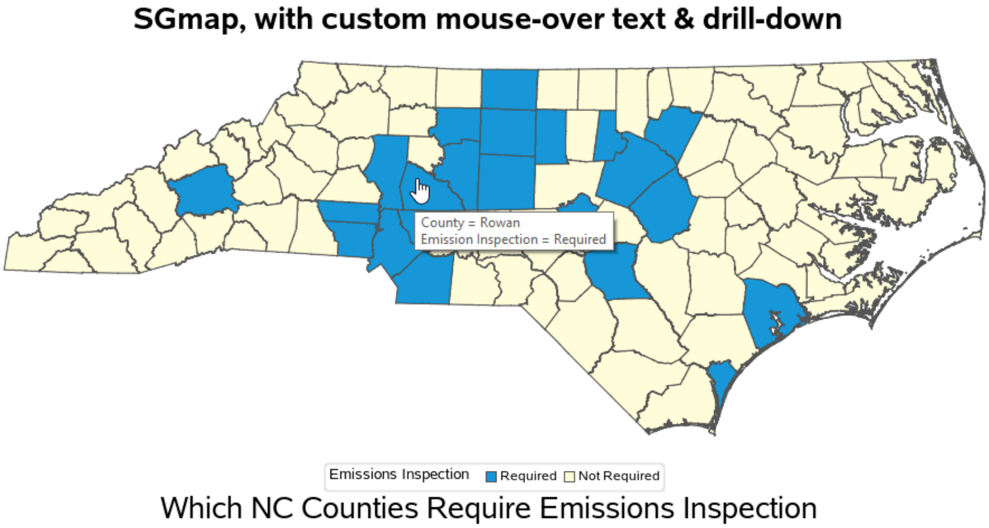
In 9.4m6, you can only get the default mouse-over text, and drill-downs are not supported.

```
data my_map; set mapsgfk.us_counties (where=(statecode="NC"));
run;
proc gproject data=my_map out=my_map (drop = lat long)
 latlong eastlong degrees;
id county;
run;

data my_data; set mapsgfk.us_counties_attr (where=(statecode="NC"));
length Inspection $20;
label Inspection='Emissions Inspection';
if idname in ('Alamance' 'Buncombe' 'Cabarrus' 'Cumberland' 'Davidson'
 'Durham' 'Forsyth' 'Franklin' 'Gaston' 'Guilford' 'Iredell' 'Johnston'
 'Lee' 'Lincoln' 'Mecklenburg' 'New Hanover' 'Onslow' 'Randolph'
 'Rockingham' 'Rowan' 'Union' 'Wake') then Inspection='Required';
else Inspection='Not Required';
run;

ods path(prepend) work.templat(update);
proc template;
define style styles.nc_style;
 parent=styles.htmlblue;
 class graphcolors / 'gdata1'=cx0095da 'gdata2'=cxfffcd6;
 end;
run;

ODS HTML style=nc_style; ods graphics / imagemap tipmax=2500;

title1 "SGmap, with default mouse-over text";
footnote "Which NC Counties Require Emissions Inspection";
proc sgmap maprespdata=my_data mapdata=my_map;
choromap Inspection / mapid=county lineattrs=(color=gray55);
run;
```


SGmap, with default mouse-over text
County FIPS code = 159 Emissions Inspection = Required County FIPS code = 159
Emissions Inspection ■ Required □ Not Required
Which NC Counties Require Emissions Inspection

In a future release, you will be able to specify which variables to show in the mouse-over text via the tip= and tiplabel= options (similar to SGPLOT). For example, although you might be using the numeric county FIPs code as the map ID, you might want to show the county name in the mouse-over text. You can also specify a drill-down variable via the url= option.

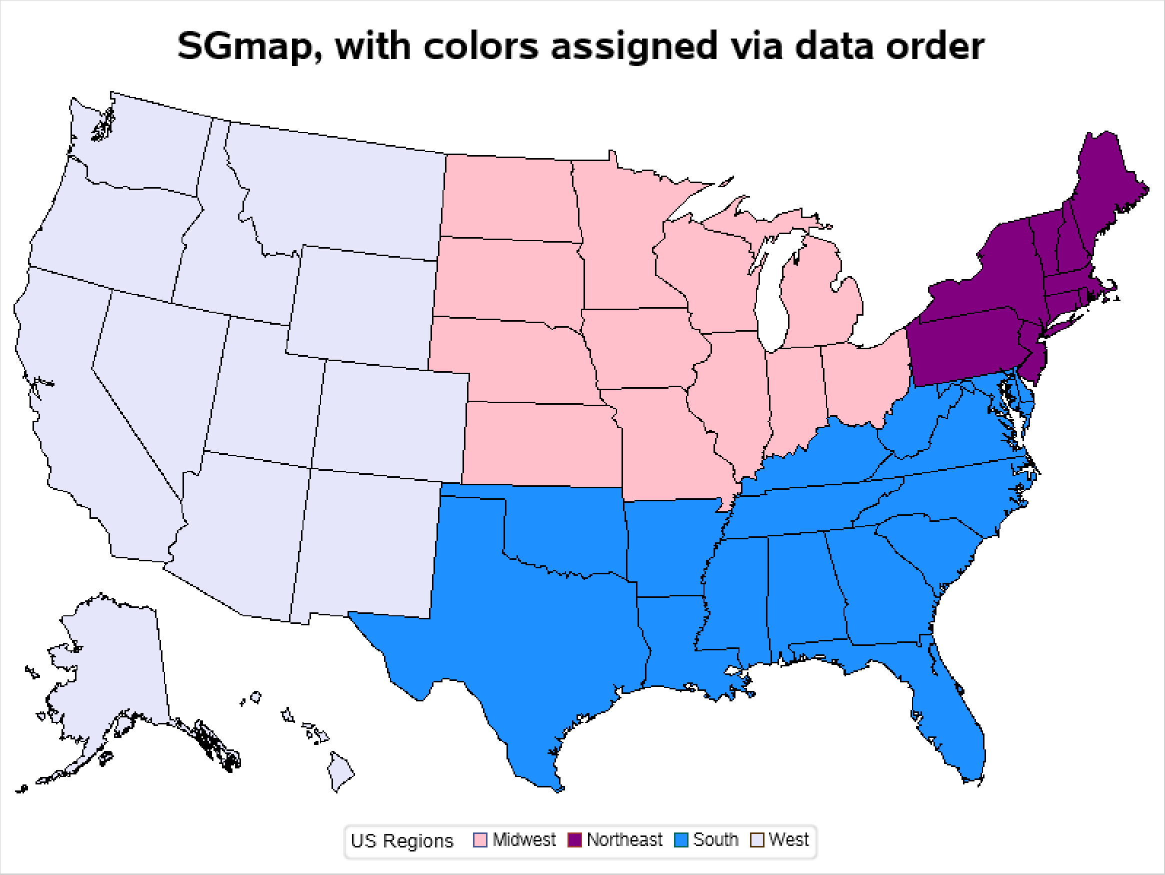
```
data my_data; set my_data;

length my_url $300;

my_url='https://www.ncdot.gov/dmv/title-registration/emissions-
safety/Pages/inspection-stations.aspx?term='||trim(left(idname))||'
county&field=county';

run;


title1 "SGmap, with custom mouse-over text & drill-down";

footnote "Which NC Counties Require Emissions Inspection";


proc sgmap maprespdata=my_data mapdata=my_map;

styleattrs datacolors=(cx0095da cxfffcd6);

choromap Inspection / mapid=county lineattrs=(color=gray55)

 tip=(idname inspection)

 tiplabel=('County' 'Emission Inspection')

 url=my_url;

run;
```


SGmap, with custom mouse-over text & drill-down
County = Rowan
Emission Inspection = Required
Emissions Inspection
Required
Not Required
Which NC Counties Require Emissions Inspection

CONTROL COLORS VIA ATTRIBUTE MAP

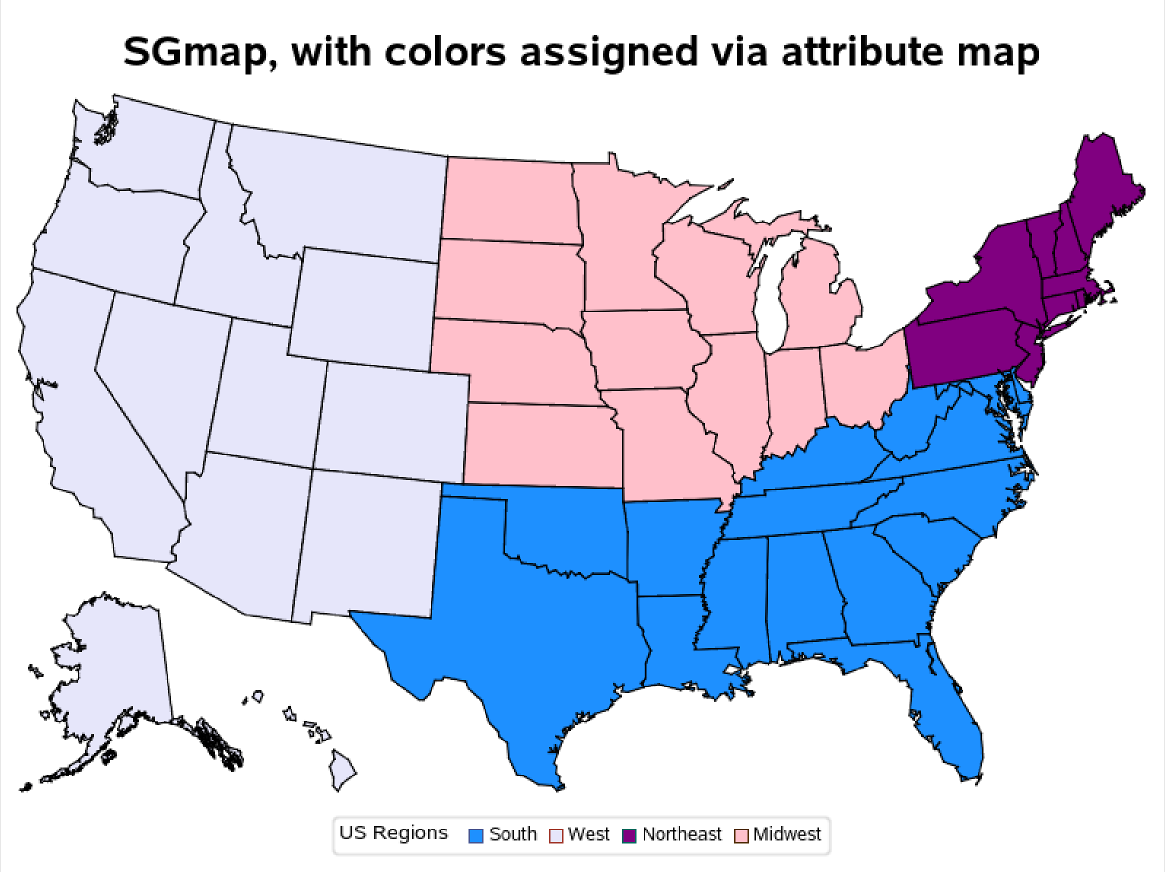
In 9.4m6, the colors are assigned based on the order the values are found in the dataset. Therefore, if it is important for specific colors to be assigned to specific values, you have to pay very close attention to your data order. Also, there are several caveats (for example, if your data changes the next time you run your code, and has fewer unique values to map to the colors).

```
proc sort data=sashelp.us_data out=my_data;
by region;
run;

ods path(prepend) work.templat(update);
proc template;
define style styles.us_style;
 parent=styles.htmlblue;
 class graphcolors /
  'gdata1'=pink 'gdata2'=purple 'gdata3'=dodgerblue 'gdata4'=lavender;
 end;
run;
ODS HTML style=us_style;


title1 "SGmap, with colors assigned via data order";
proc sgmap maprespdata=my_data mapdata=mapsgfk.us;
choromap region / mapid=statecode id=statecode;
run;
```



In a future release, you will be able to use attribute maps to assign specific colors to specific values (just like SGPLOT). This gives you much better control, when your colors are important.

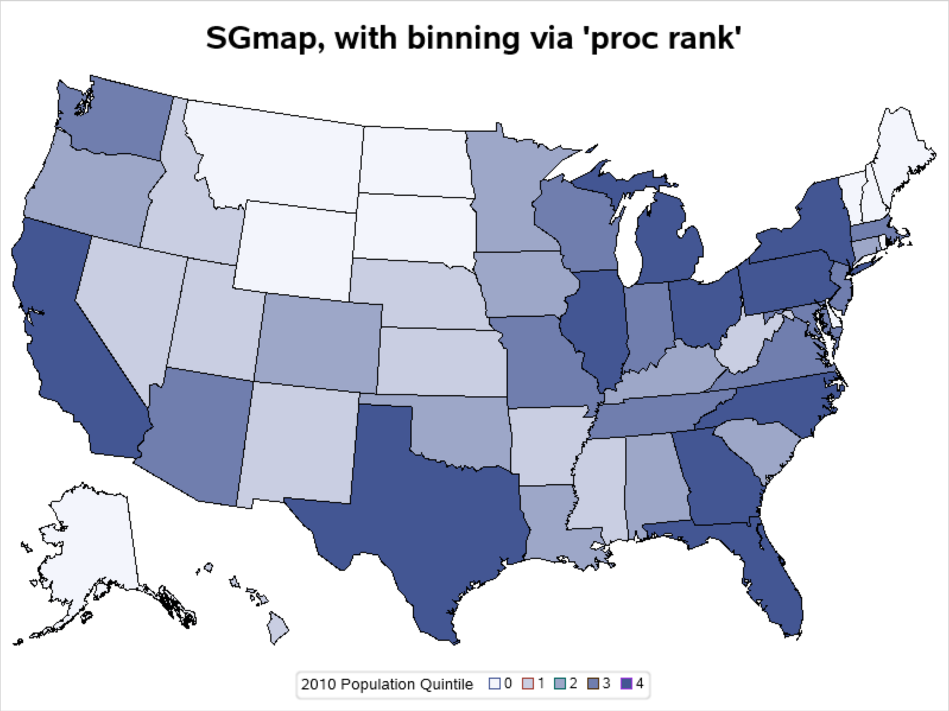
```
data my_attrmap;
length value fillcolor $20;
input value fillcolor;
id='regioncolors';
datalines;
West      lavender
South     dodgerblue
Northeast purple
Midwest   pink
;
run;

title1 "SGmap, with colors assigned via attribute map";

proc sgmap maprespdata=sashelp.us_data mapdata=mapsgfk.us
dattrmap=my_attrmap;
choromap region / mapid=statecode id=statecode attrid=regioncolors;
run;
```



BINNING WITH NUMLEVELS=

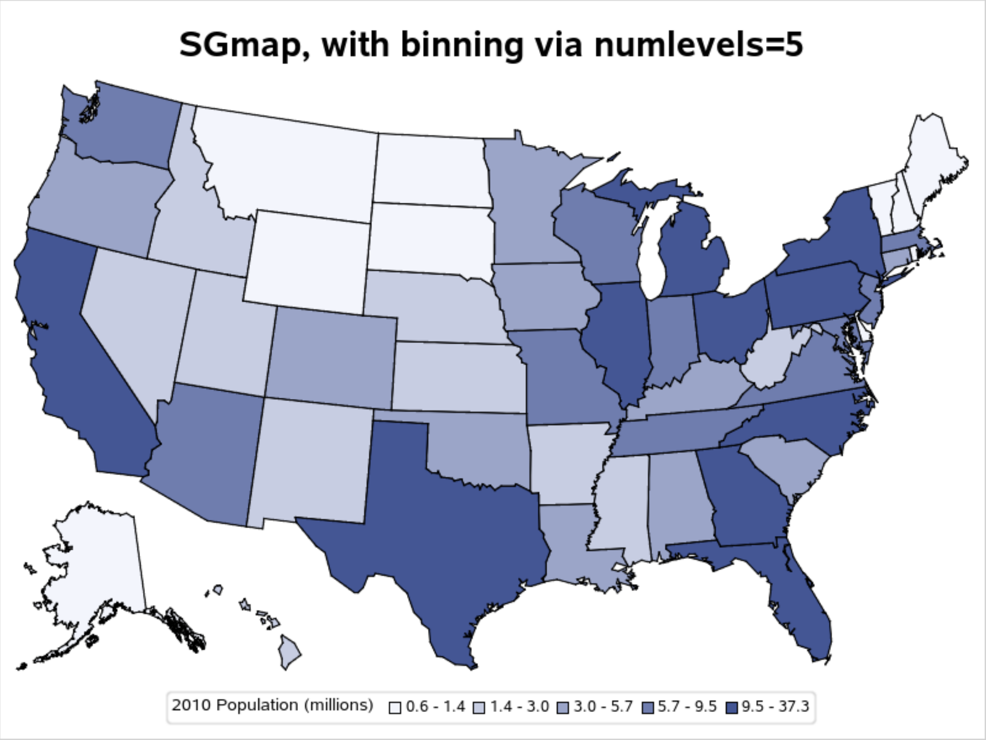
In 9.4m6, SGMAP has no built-in binning. Therefore, if you want to color your map using quantile binning, you have to pre-process your data and create a variable with your own bins. In this example I use the RANK procedure to calculate the quintiles (5 bins) for the population.

```
proc rank data=sashelp.us_data (where=(statecode not in ('DC' 'PR')))
out=my_data groups=5 ties=low;
 var population_2010; ranks quintile_var;
run;

proc sort data=my_data out=my_data;
by quintile_var;
run;

ods path(prepend) work.templat(update);
proc template;
define style styles.my_grad;
 parent=styles.htmlblue;
 class graphcolors / 'gdata1'=cxf3f5fc 'gdata2'=cxc9cee2
  'gdata3'=cx9da7c8 'gdata4'=cx707eae 'gdata5'=cx435693;
 end;
run;
ODS HTML style=my_grad;


title1 "SGmap, with binning via 'proc rank'";
proc sgmap maprespdata=my_data mapdata=mapsgfk.us;
label quintile_var='2010 Population Quintile';
choromap quintile_var / discrete mapid=statecode id=statecode;
run;
```



In a future release, SGMAP will allow you to specify the numlevels= (similar to levels= in GMAP), and the leveltype= (in this case, quantile binning). This will allow you to easily create a map without having to pre-summarize your data. Another benefit is that it automatically creates a legend showing the numeric ranges (whereas the RANK procedure shown above only told you which quintile the values were in – not the range of values in each quintile).

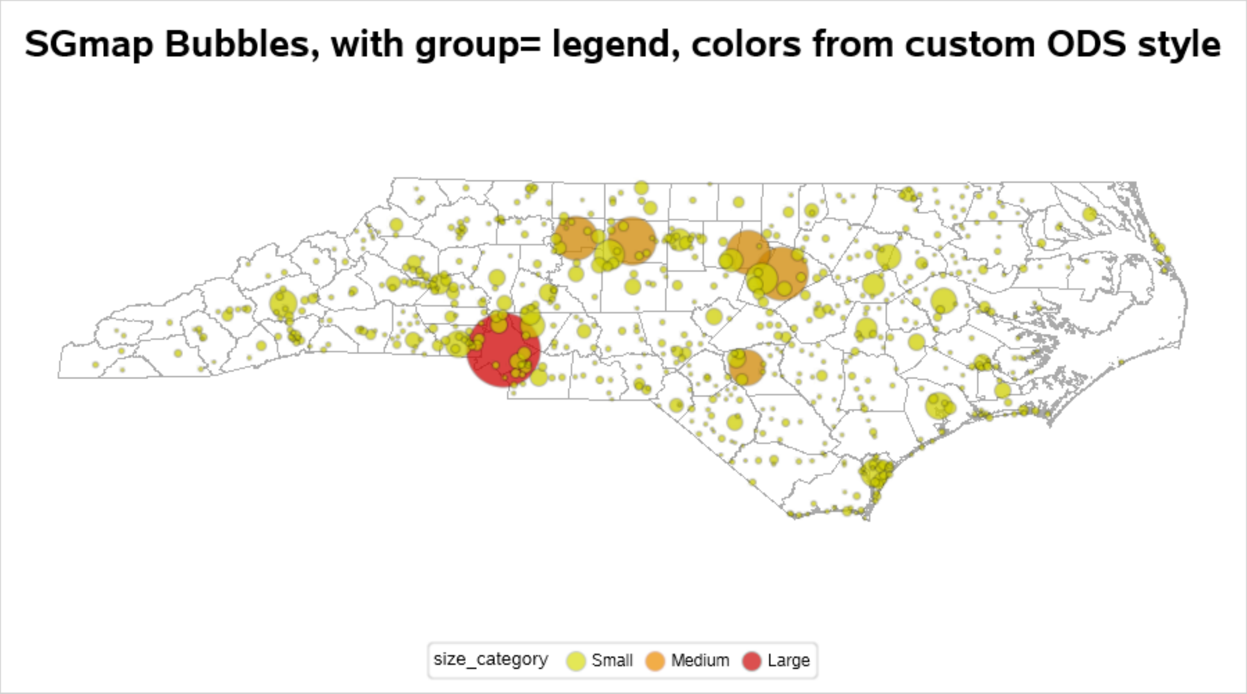
```
data my_data; set sashelp.us_data (where=(statecode not in ('DC' 'PR')));
population_2010=population_2010/1000000;
run;


title1 "SGmap, with binning via numlevels=5";

proc sgmap maprespdata=my_data mapdata=mapsgfk.us;
label population_2010='2010 Population (millions)';
format population_2010 comma5.1;
choromap population_2010 / mapid=statecode id=statecode
 numlevels=5 leveltype=quantile;
run;
```


SGmap, with binning via numlevels=5
2010 Population (millions)
0.6 - 1.4
1.4 - 3.0
3.0 - 5.7
5.7 - 9.5
9.5 - 37.3

SPECIFYING GRADIENT COLORS FOR BUBBLES

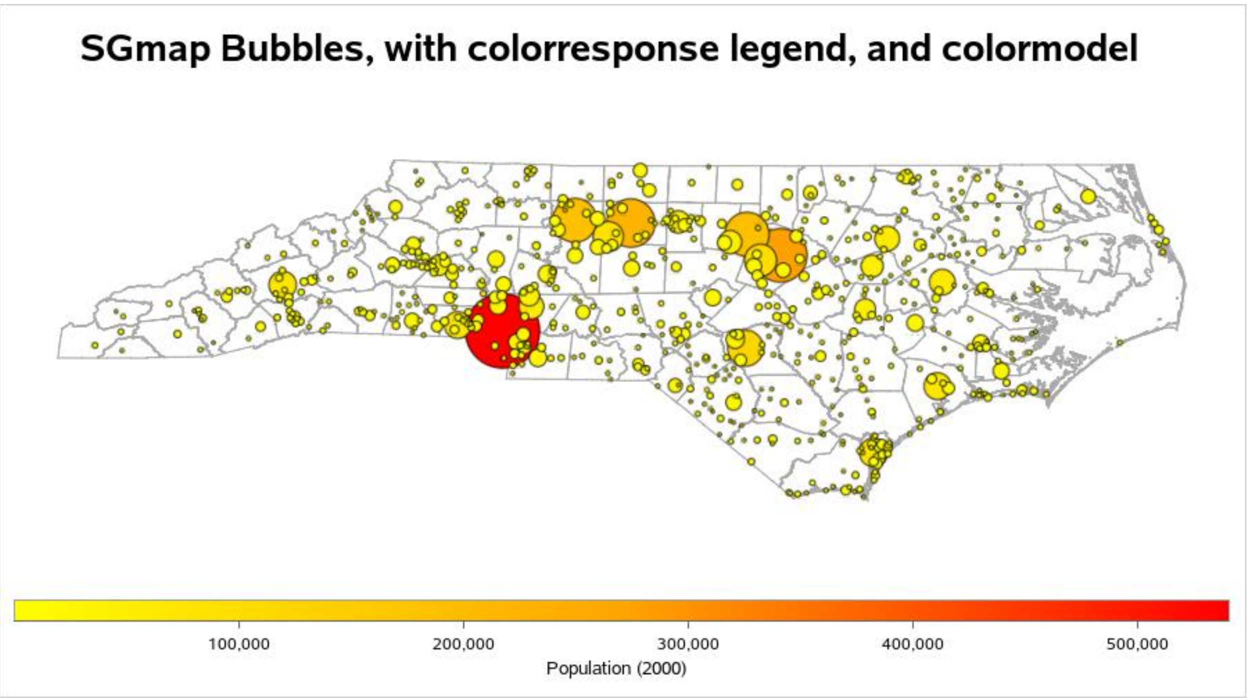
In 9.4m6, you can have all your bubbles a single color, or you can assign discrete colors based on the group= values.

```
data my_data; set maps.uscity (where=(statecode='NC'));
long=-1*long;
length size_category $20;
if pop>500000 then size_category='Large';
else if pop>100000 then size_category='Medium';
else size_category='Small';
run;

proc sort data=my_data out=my_data; by pop; run;

data my_map; set mapsgfk.us_counties (where=(statecode='NC') drop=x y); run;

ods path(prepend) work.templat(update);
proc template;
define style styles.bub_styl;
 parent=styles.htmlblue;
  class graphcolors / 'gdata1'=yellow 'gdata2'=orange 'gdata3'=red; end;
run;
ODS HTML style=bub_styl;

title1 "SGmap Bubbles, with group= legend, colors from custom ODS style";
proc sgmap plotdata=my_data mapdata=my_map;
choromap / mapid=county lineattrs=(color=grayaa);
bubble x=long y=lat size=pop / bradiusmin=1 bradiusmax=30
 group=size_category transparency=.40;
run;
```


SGmap Bubbles, with group= legend, colors from custom ODS style
size_category
Small
Medium
Large

In a future release, you will be able to have a gradient color ramp, and specify the colors using the colormodel= option (you will no longer need to modify the ODS style!)

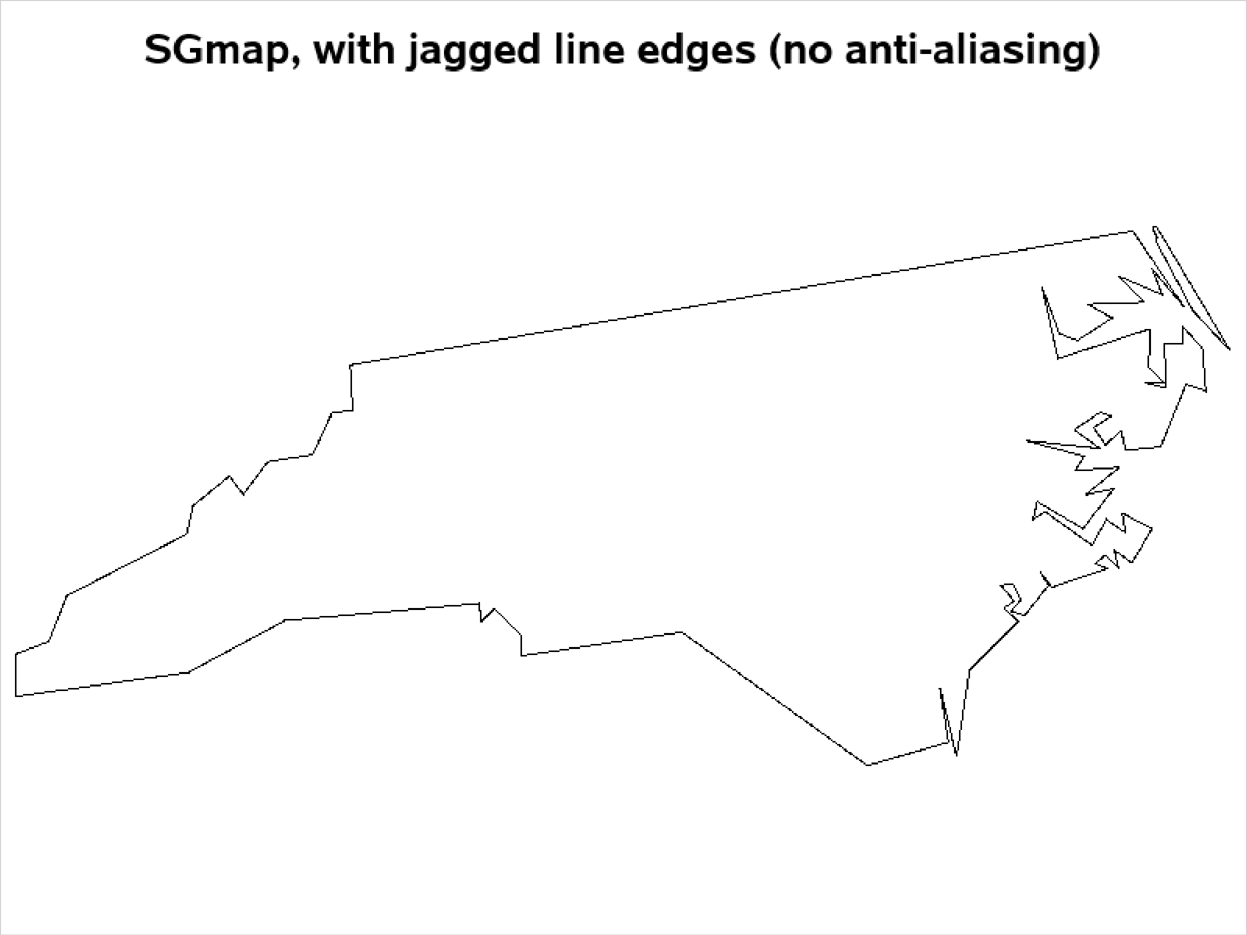
```
data my_data; set maps.uscity (where=(statecode='NC'));
long=-1*long;
format pop comma12.0;
run;

proc sort data=my_data out=my_data;
by pop;
run;

data my_map; set mapsgfk.us_counties (where=(statecode='NC') drop=x y);
run;


title1 "SGmap Bubbles, with colorresponse legend, and colormodel";

proc sgmap plotdata=my_data mapdata=my_map noautolegend;
choromap / mapid=county lineattrs=(color=grayaa);
bubble x=long y=lat size=pop / bradiusmin=1 bradiusmax=30
 colorresponse=pop colormodel=(yellow orange red) name='bub';
gradlegend 'bub';
run;
```



SMOOTH EDGES / ANTI-ALIASING

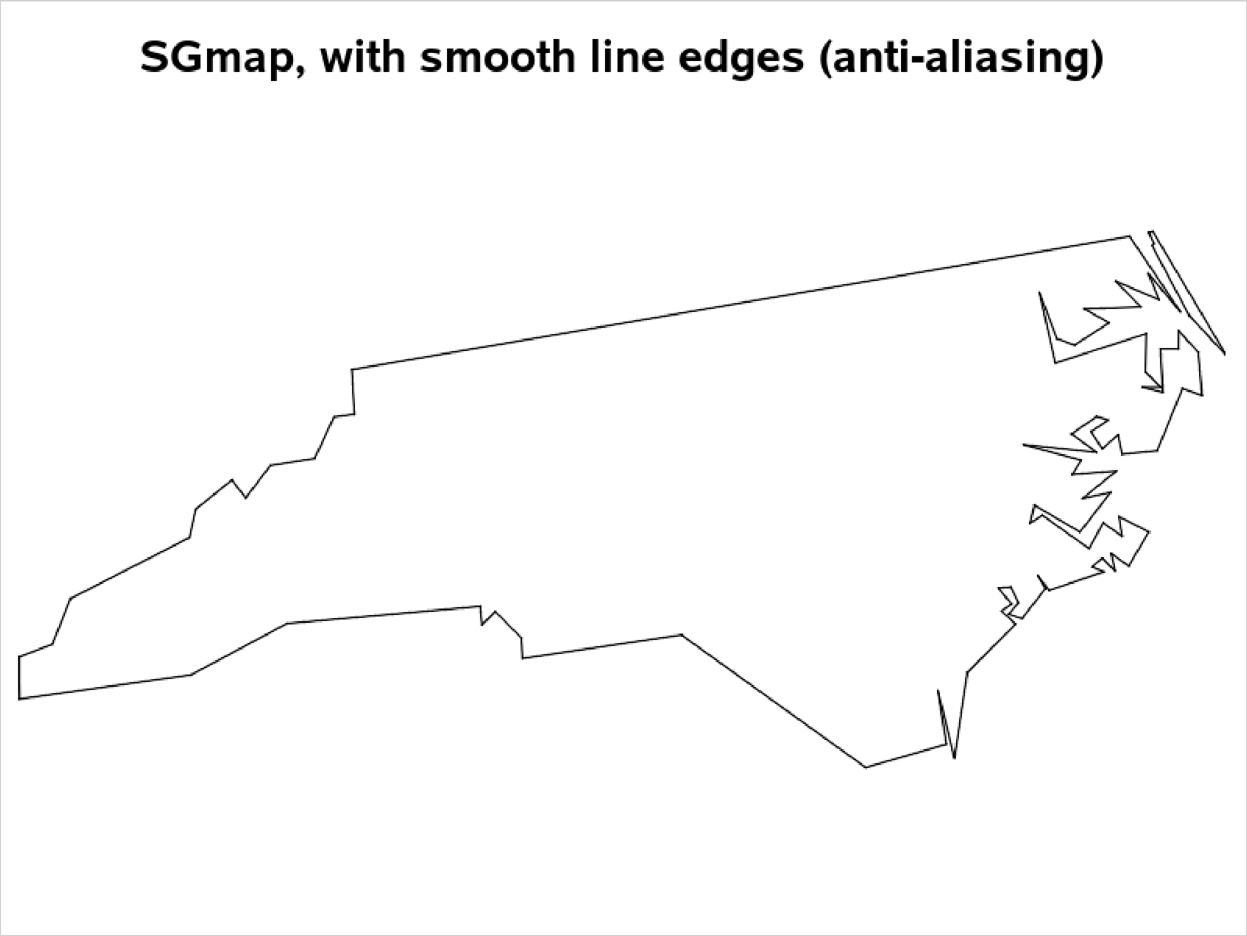
In 9.4m6, SGMAP does not support anti-aliasing for png output. Therefore, if you look very closely, you can see jagged edges along the angled lines (such as the northern border of North Carolina, in this example).

```
title1 "SGmap, with jagged line edges (no anti-aliasing)";

proc sgmap mapdata=mapsgfk.us (where=(statecode='NC')) noautolegend;
choromap / mapid=statecode;
run;
```



In a future release, SGMAP will support anti-aliasing. Anti-aliasing basically fills in extra pixels along the jaggies, and makes these pixels a slightly lighter color, and your eyes visually interpret it as a smooth line. This will happen automatically – you will not need to specify any extra options!

```
title1 "SGmap, with smooth line edges (anti-aliasing)";

proc sgmap mapdata=mapsgfk.us (where=(statecode='NC')) noautolegend;
choromap / mapid=statecode;
run;
```



CONCLUSION

The SGMAP procedure in BASE SAS has a lot of neat enhancements coming up in future releases. These enhancements will allow you to use less code, and/or provide you with enhanced functionality. I can't tell you exactly when – but it will be worth the wait!

CONTACT INFORMATION

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

Robert Allison
SAS Institute
Robert.Allison@SAS.com
https://blogs.sas.com/content/author/robertallison/