

Creating Regional Maps with Drill-Down Capabilities

Deb Cassidy

Cardinal Distribution, Dublin, OH

ABSTRACT

SAS/GRAPH® includes many maps which are very useful. But what if you need a map that truly represents your company? Many companies combine several states into a region. My company does not even follow state boundaries so some states are split across more than one region. This presentation will show you how to turn the county-level map into a region-level map. Of course, with the advent of ODS, no presentation is complete without discussing ODS. The presentation will also cover the extra steps so you can have drill-down capabilities with your new regional map. This presentation uses SAS/GRAPH. It is assumed the audience has some SAS knowledge but may not have used SAS/GRAPH maps or ODS.

CREATING A REGIONAL MAP

My company currently divides the United States into 12 regions based on sales territories. Some states are split across regions. In reality there are some areas of the country which are shared by two regions. It was determined that all counties were to be assigned to a single, primary region for analytical purposes. However, when it came time to assign the states of New York and New Jersey, it was determined that it was easier to create a separate region for those states that to decide which was the primary region. Hence, our regions are unique so a pre-existing map would not work.

I already had a dataset with each state and county combination although I could have used the COUNTIES dataset provide with SAS/GRAPH as my starting point. The following code would create a unique list of states and counties from the map dataset.

```
PROC SQL;
  CREATE TABLE COUNTY_LIST AS
  SELECT UNIQUE STATE, COUNTY
  FROM MAPS.COUNTIES;
```

The dataset you use must use the same numeric state and county codes used in the SAS/GRAPH COUNTIES dataset. You simply need to add a

variable indicating the region. As explained below, I used a numeric variable to represent region.

The COUNTIES dataset included with SAS/GRAPH is merged with my region dataset. You may think that is all you have to do but if you try to use this dataset in PROC GMAP you will discover a problem - the map is reversed. Some of the SAS/GRAPH maps are projected and others are not. If they are not, you need to run PROC GPROJECT which will convert latitude and longitude into Cartesian coordinates which are used by PROC GMAP. This PROC also has several options for controlling the angle of the maps.

In most cases, you'll also want to use PROC GREMOVE. This will remove the county and state boundaries within each region. When mapping the counties in the entire United States, the map will be very cluttered if you don't perform this step. There is one other step that you may want to consider. The map dataset is very detailed and shows the many curves the borders of the states have. This is particularly noticeable on states which have a body of water as a boundary. The COUNTIES dataset includes a density variable which can be used to eliminate some of the detail. This will smooth out the map which can be beneficial which when showing a large area. The code to do these steps is shown at the end of the paper.

ADDING DRILL-DOWN CAPABILITY

The JAVA applet requires a "jar" file or a text version of the map. Since you have modified the original map dataset, the supplied file is not usable. Therefore you need to create your own. Fortunately, SAS provided a macro for this conversion. The macro will create a text version of the map for use by the applet. The macro is shown at the end of the paper. The text file ends up in the following format.

```
REGION,SEGMENT,X,Y
numeric,numeric,numeric,numeric
1,1,0.2770831409,-0.005568895
...rest of coordinates for Region 1
2,1,0.2827390409,0.0396897128
...rest of coordinates for Region 2
...rest of coordinates for other Regions
```

So far you have created the custom map you need to show regions and saved it as a text version. The next step is to generate the map using ODS. The options/parameters on the ODS statement provide the information needed so it can be used with the JAVA applet. The **DRILLPATTERN** parameter specifies where to look for the web pages. There needs to be a separate page for each value on your map. The example shown uses files on your pc. On a web site, this would be a web address.

There were two problems I encountered when specifying the **DRILLPATTERN**. At the time this paper was written, I had not confirmed if these are indeed problems or if there was simply something about my data or my system causing the problems. First, when I used a character variable for my regions, the drill-down did not work. Switching to a numeric variable solved that problem. Second, I couldn't simply use `c:\{regname}.htm`. I had to have a prefix before the region value in the file names of the web pages.

I also encountered another problem with the map. Some of the states have lakes within the state boundaries. Depending upon the states selected, the output was either blank where the map should be or a distorted map appeared. Reducing the density eliminated the distortion problem. However, the output was still blank if I left in the states of Florida, Utah, New Jersey and Oregon regardless of what I used for density. At the time this paper was written, I was still investigating ways to eliminate the lakes in these states. So contrary to any ideas you might have based on the final map, Florida has not been dropped from the United States!

FINAL MAP

The final map is shown on the next page. Also shown is the original map with state boundaries to show you how things have changed (the one with county boundaries was far too cluttered to attempt to print here). When you move your cursor over each region you will get a pop-up box giving you the values of the variables used to create your map. When you click on a given region, you will go to the web page for that region.

CONCLUSION

SAS/GRAPH provides you with a variety of maps which can then be used to create your own custom map. ODS offers additional functionality beyond simply displaying the graph.

TRADEMARK INFORMATION

SAS is a registered trademark of SAS Institute, Inc. in the USA and other countries. ® indicates USA registration.

REFERENCES

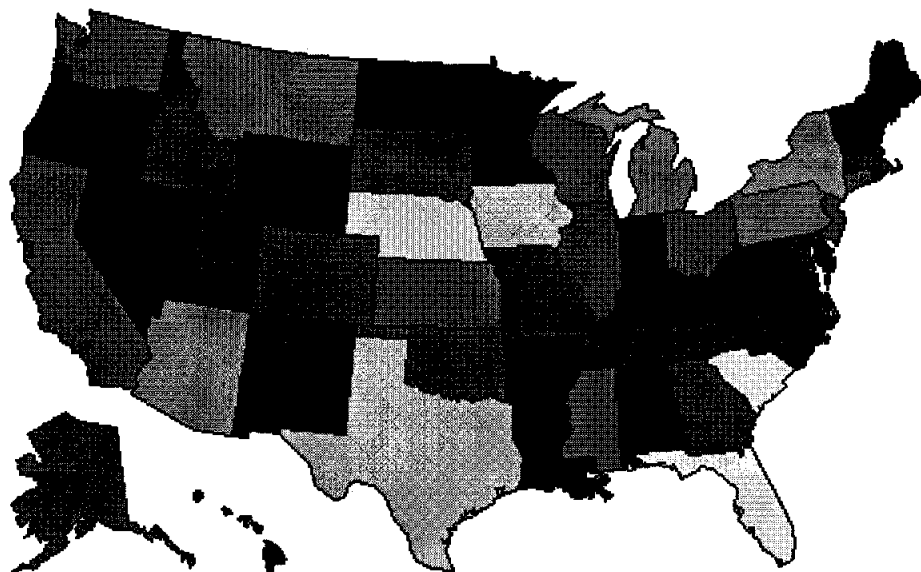
SAS Institute Inc., SAS/GRAPH® Software: Reference, Version 6, First Edition, Volume 2, Cary, NC: SAS Institute Inc., 1990. 664 pp.

SAS Institute Inc., TS-625, Cary, NC: SAS Institute Inc., 2000.

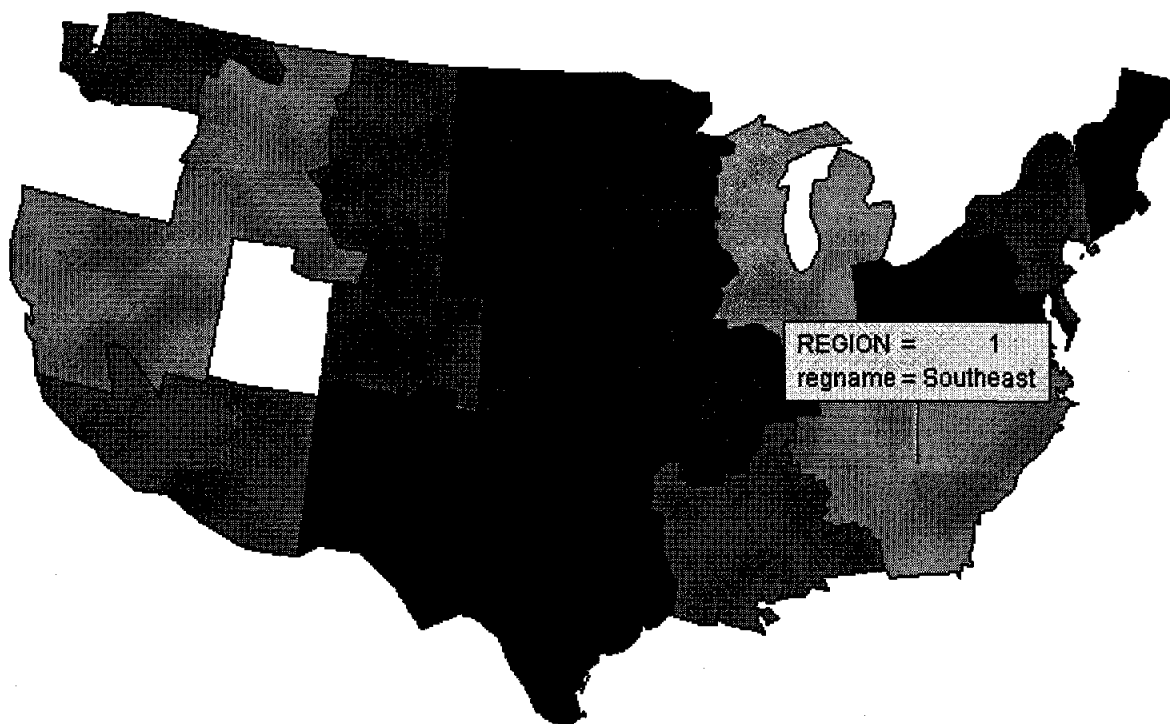
CONTACT INFO

Deb Cassidy
Cardinal Distribution
7000 Cardinal Place
Dublin, OH 43017
deb.cassidy@cardinal.com
614-757-7136

ORIGINAL STATE-LEVEL MAP



FINAL REGION-LEVEL MAP



CODE

*** ASSIGN ANY NECESSARY LIBNAME STATEMENTS TO ACCESS YOUR SAS/GRAPH MAPS OR YOUR DATA;

*** Specify the file location to store the text version of the map. This must be the same path as the JAR files which are supplied with SAS/GRAPH. A macro variable is used since this location is used more than once in the program;

```
%let dataloc=c:\program files\sas institute\shared files\applets\SSU_Paper.txt;
```

**** This is the macro which will create the text version of the map;

```
%macro exportds(dataset,file);
```

```
%let delimiter=,;
```

```
proc contents data=&dataset out=temp;
```

```
run;
```

```
/*write out the variable names first*/
```

```
data a;
```

```
set temp end=last;
```

```
file "&file" dsd dlm="&delimiter";
```

```
put name @;
```

```
run;
```

```
/*write out the variable types next*/
```

```
data a;
```

```
set temp end=last;
```

```
length ctype $9;
```

```
file "&file" mod dsd dlm="&delimiter";
```

```
if type=1 then ctype='numeric';
```

```
else ctype='character';
```

```
put ctype @;
```

```
run;
```

```
/*write out the data*/
```

```
data a;
```

```
set temp end=last;
```

```
file 'temp.out';
```

```
if _n_ = 1 then do;
```

```
put "data a; set &dataset;";
```

```
%let out="file '&file' mod dsd dlm='&delimiter';";
```

```
put &out;
```

```
put 'put ' @;
```

```
end;
```

```
put name @;
```

```
if last then do;
```

```
put ';run;';
```

```
end;
```

```
run;
```

```
%inc 'temp.out';
```

```
%mend;
```

*** PROC FORMAT is used to assign names to each region I have created;

```
PROC FORMAT;
```

```
VALUE REGNAME
```

```
1='Southeast'
```

```
2='Mid-Atlantic'
```

```
3='Mid-America'
```

```
4='Gulf'
```

```
5='Northwest'
```

```
6='Pacific Northwest'
```

```
7='Pacific Southwest'
```

```
8='North'
```

```
9='Northeast'
```

```
10='Southwest'
```

```
11='Midwest'
```

```
12='NY/NJ';
```

*** INN.MY_REGIONS is my dataset which contains the region assignment for each state and county;

*** For my purposes, Hawaii, Alaska and Puerto Rico were to be eliminated. Florida, Utah, New Jersey and;

*** Oregon were also deleted due to the problems encountered with the lakes. These last 4 states will be included ;

*** once that problem is resolved. The numeric codes for each state are in the SAS/GRAPH manual.;

```
data work.mydata ;
```

```
length regname $1020;
```

```
set inn.my_regions;
```

```
if state not in (2,15,72,12,49,41,34);
```

```
regname=put(region,regname.);
```

```
run;
```

*** This step merges the SAS/GRAPH supplied COUNTIES map with my data to add the region variable;

*** As noted above, some states were deleted. The DENSITY was restricted to have a cleaner looking map;

*** I experimented with different density cutoffs before deciding to use <1;

```
data mymap;
```

```
merge maps.counties inn.my_regions;
```

```
by state county;
```

```
if state not in (2,15,72,12,49,41,34);
```

```
if density<1;
```

```
run;
```

*** The map dataset must be sorted before using GREMOVE to eliminate the county and state boundaries;

*** In PROC GREMOVE, the BY statement specifies the boundaries you want to keep while the ID;

*** statement specifies the original boundaries;

```
proc sort data=mymap;
```

```
by region;
```

```
run;
```

```
proc gremove data=mymap out=mymapR;
```

```
by region;
```

```
id county state;
```

```
run;
```

***PROC GPROJECT will convert latitude and longitude to Cartesian coordinates. If you are using maps other;

SERENDIPITY

```
*** than the US, you may wish to use some of the options to modify the projection;
proc gproject data=mymapR out=mymapP;
  id region;
run;
```

```
*** This macro will convert the map dataset to the text file which is required for the JAVA applet.
%exportds(mymapP,&dataloc)
```

```
*** Close ODS listing to be able to route the output to the ODS HTMLfile;
*** The HTML file and the DRILLPATTERN files can be a web address.;
*** The DRILLDOWNMODE is set to HTML so a click will drive the drilldown;
*** MapLocation is the location of the text version of the map;
ods listing close;
ods html file="f:\dcassidy\DRILL_DOWN_MAP.htm"
  parameters=("DRILLDOWNMODE"="HTML")
  parameters=("DRILLPATTERN"='file:///c:\REG{&regname}.htm')
  parameters=("BACKCOLOR"="WHITE")
  parameters=("MapLocation"="&dataloc");
```

```
***This statement sets the device to Java to produce an HTML file that that will appear in your web browser.;
***It also sets up customizations for the graph.;
```

```
goptions reset=all
  dev=java
  cback=white
  border
  gunit=pct
  htext=3
  colors=( green, red, orange, pink, blue, cyan, magenta, lime, purple, yellow, violet, indigo);
```

```
***This proc produces the graph.;
title1 "Acme Distribution Regions";
proc gmap map=mymapP data=mydata all;
  id region;
  choro regname / discrete
    coutline=black
    nolegend
    des='Acme Distribution Regions'
    name='Regions';
footnote 'Click on region to go to regional page';
run;
quit;
```

```
*** These statements close the HTML file and open the channel to the graph output windows.;
ods html close;
ods listing;
```