

Creating Geographic Rating Area Maps: How to Combine Counties, Split Counties, and use Zip Code Boundaries

Rick Andrews, Centers for Medicare and Medicaid Services, Office of the Actuary
Robert Allison, SAS Institute, Research and Development

ABSTRACT

SAS/GRAPH® will be used to create choropleth maps that identify the geographic rating areas implemented by the Affordable Care Act (ACA). The default areas for each state are Metropolitan Statistical Areas (MSAs) plus the remainder of the State that is not included in a MSA. States may seek approval to base the rating areas on counties or three-digit zip codes, which requires that counties be combined in some states and split in two in others. For the states that use zip codes to identify the rating areas, ZIP code tabulation area (ZCTA) files are used, which are derived from an ESRI® shapefile format (.shp) and are obtained from the U.S. Census Bureau. Also demonstrated will be the utilization of the annotate facility to identify each area and place major cities on the maps.

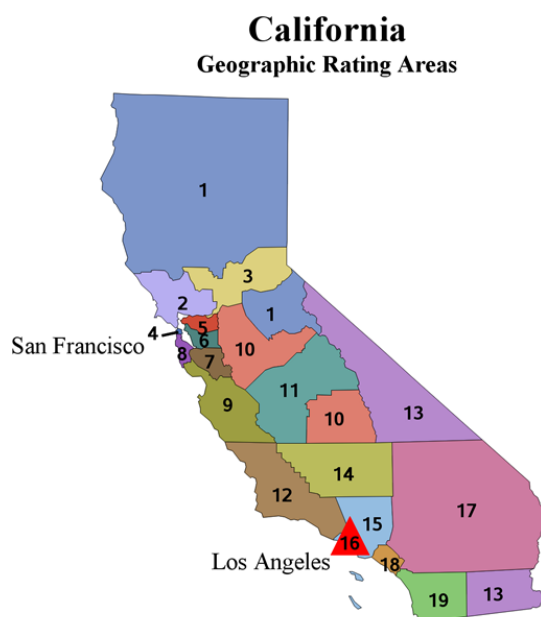
INTRODUCTION

The Market Rules and Rate Review Final Rule (45 CFR Part 147) provides that each state will have a set number of geographic rating areas that all issuers in the state must uniformly use as part of their rate setting (CMS, 2014). Qualified Health Plans are permitted to impose higher rates on individuals and families who live in areas where medical costs are higher, and each state is responsible for establishing the rating areas to be used for these purposes. To graphically represent these areas, SAS/GRAPH procedures such as PROC GMAP, GREMOVE, and GPROJECT are used to create maps where any manner of statistical response data can be represented. Meaning, numerical results that describe the population like the cost per member per month (PMPM), utilization per thousand, or unit cost can be displayed to help geographically illustrate the effects of each rating area.

Other useful SAS® supplied resources will also be discussed, such as SASHELP.ZIPCODE, which provides zip code centroids, MAPS.USCITY, which provides centroids for major U.S. cities, and MAPS.CNTYNAME, which, as the name implies, provides county names. In addition, the MAPLABEL macro will be used to create an annotate data set for labeling the rating areas. A brief description of the annotate facility will be provided in order to describe how to fine-tune the label's position and output additional labels for noncontiguous areas. An example of placing markers on the map to identify a city's location will also be explained. Finally, for states that use zip codes to identify rating areas, a short description of the MAPIMPORT procedure is used to exploit the ZCTA shape files.

BACKGROUND

To identify the geographic rating areas, data were downloaded from the Center for Consumer Information & Insurance Oversight (CCIIO) website (CMS, 2014) and matched to the MAPS.CNTYNAME data set in order to obtain the Federal Information Processing Standard (FIPS) code for each state and county. Those data were then merged with the MAPS.COUNTY data set to acquire the “un-projected” latitude (LAT) and longitude (LONG) in radians, which will become more evident on the next section. Once the rating areas are combined with FIPS codes and the LAT/LONG in radians are acquired, the county boundaries can be removed by utilizing the GREMOVE procedure to reveal only the polygons of the geographic rating areas. After the county boundaries are removed, and a new “Rating-Area-Map” created, then the MAPS.USCITY data set and MAPLABEL macro are used to annotate labels onto the new map.



CREATING THE CODE

This document is not meant to provide a complete process for the creation of rating area maps for all states, but instead, as a training exercise to describe how it can be done. The opinions of the presenters are their own and do not represent the Office of the Actuary, Centers for Medicare and Medicaid Services, or the Department of Health and Human Services and are presented as-is without warranty of any kind.

IMPORT RATING AREAS

Information copied from the CCIIO website (CMS, 2014) was placed in a DATALINES statement and separated by commas to be read into a SAS data set called MD_Rating_Areas. The data are then stored in a temporary area as designated by the keyword, "work".

Keep in mind that some of the county names may be slightly different and will require manual intervention. For instance, the web site may show Prince George's county containing an apostrophe, whereas SAS does not, and also store the county name in uppercase letters.

```
* Code to import MD Ratings;
data work.MD_Rating_Areas;
infile datalines delimiter=',' trunccover;
length Rating_Area $2 County_Name $25;
input Rating_Area $ County_Name $;
datalines;
1, Baltimore
1, Baltimore City
1, Harford
... some data not shown ...
4, Carroll
4, Frederick
4, Washington
;
run;
```

OBTAIN FIPS CODES

The next step is to add the FIPS state and county codes from the MAPS.CNTYNAME data set. These are numeric variables that SAS uses throughout the map making process. The SQL procedure is used to join the two tables together, though keep in mind that SQL can sometimes change the order of the output, which would cause the map to render incorrectly. A DATA step merge could also be utilized instead of an SQL join. A "left" join is being performed in order to determine if a successful match was achieved. A check should be made to ensure that all counties have corresponding FIPS codes. The UPCASE function is used against the County_Name variable because the SAS supplied column is in uppercase letters. In addition, the STFIPS function is used to keep only Maryland counties.

Table 1: MD Rating Area to FIPS Crosswalk

Rating Area	County Name	FIPS State	FIPS County
1	Baltimore	24	5
1	Baltimore City	24	510
1	Harford	24	25
... some data not shown ...			
4	Carroll	24	13
4	Frederick	24	21
4	Washington	24	43

```
* Obtain FIPS Codes;
proc sql;
create table work.MD_Rating_Area_FIPS as
select T1.*, T2.STATE, T2.COUNTY
from work.MD_Rating_Areas as T1
left join
    MAPS.CNTYNAME as T2
on upcase(T1.County_Name)=T2.COUNTYNM
where state = stfips('MD');
quit;
```

CREATE MAP DATA SET

Once the FIPS codes are obtained, a temporary map data set for Maryland counties is created to store the original map latitude and longitude coordinates. The FIPS codes are joined with the MAPS.COUNTY data set in order to obtain the un-projected LAT and LONG, depicted here by the X and Y variables, which are in radians; the reason for doing this will become more evident later in the paper.

```
* Create Map Data Set;
proc sql;
create table work.MD_Temp_Map as
select T1.*, T2.X, T2.Y
from work.MD_Rating_Area_FIPS as T1
inner join
    MAPS.COUNTY as T2
on T1.state = T2.state
and T1.county = T2.county;
quit;
```

REMOVE COUNTY BOUNDARIES

Now that the rating areas have been merged with the MAPS.COUNTY data set, the internal county boundaries can be removed using the GREMOVE procedure. First, sort the input data set by state and rating area and make sure the GREMOVE procedure contains the same BY statement as the sort procedure. The ID statement tells the system to remove the internal “county” boundaries.

PROJECT THE MAP

The GPROJECT procedure converts the spherical longitude and latitude coordinates into Cartesian coordinates, which specifies each point uniquely in a plane by a pair of numerical values. When the un-projected coordinates are plotted using the GMAP procedure, which is designed to plot points on a two-dimensional plane, the resulting map is often reversed and distorted as a result of forcing the spherical map coordinates onto a flat plane (SAS, 2015), hence the need to project the map using Cartesian coordinates.

```
* Remove county boundaries;
proc sort data= work.MD_Temp_Map;
  by state Rating_Area;
run;

proc gremove
  data= work.MD_Temp_Map
  out= work.MD_Gremove_Map;
  by state Rating_Area;
  id county;
run;

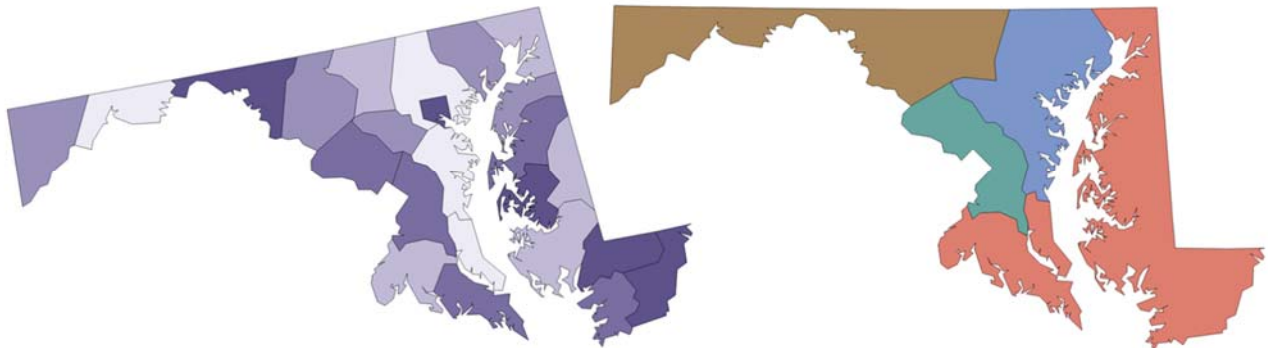
* Project rating area map;
proc gproject
  data= work.MD_Gremove_Map
  out= work.MD_Rating_Area_Map dupok;
  id state Rating_Area;
run;
```

RENDER THE MAP

The examples shown below are meant to shed light on why the un-projected MAPS.COUNTY data set was used in lieu of the already projected MAPS.USCOUNTY data set. The GMAP procedure on the left uses the USCOUNTY data and contains the state and county variables in the ID statement. This data set was projected not only for Maryland but also with all of the other states in the U.S. Because of this inclusion, when only the state of Maryland is requested, the map appears to be slanted at a 30° angle. The map on the right was projected using only Maryland coordinates and therefore renders in a “straighter” manner. Note the use of the state and *rating* area variables in the ID statement on the right, compared to the state and *county* variables in the ID statement on the left.

```
* Example using MAPS.USCOUNTY;
proc gmap
  data= MAPS.USCOUNTY
  map= MAPS.USCOUNTY
  all;
  where state = stfips('MD');
  id state county;
  choro county /
    woutline=1
    nolegend;
run;
quit;
```

```
* Example using new Rating Area Map;
proc gmap
  data= work.MD_Rating_Area_Map
  map= work.MD_Rating_Area_Map
  all;
  id state Rating_Area;
  choro Rating_Area /
    woutline=1
    nolegend;
run;
quit;
```



ANNOTATE LABELS

SAS provides a very useful set of macros for annotating labels and other pieces of information, such as images or symbols, onto a map. The Annotate facility enables the generation of a special data set of graphics commands from which graphical output can be produced. This data set is referred to as an Annotate data set (SAS, 2015). The ANNOMAC macro compiles various macros and makes them available for use. As shown below, the MAPLABEL macro is used to add labels to the approximate centroid of the rating areas.

In the snippet of code below, the MD_Rating_Area_FIPS and Rating_Area_Map data sets are sorted by State and Rating_Area in preparation for the MAPLABEL macro. Note that the map data set is output to a new table called Rating_Area_Sorted to ensure that the original map data order is maintained. This is because the sequence of map data is important to the GMAP procedure. The third parameter of the MAPLABEL macro identifies the annotate data set to store the X and Y coordinates of the labels. The fourth parameter is the variable that contains the desired label. The fifth parameter is the sort order of the incoming data sets. The remaining parameters are self-explanatory.

```
* Macro to add Rating Area labels to the map;

proc sort data= work.MD_Rating_Area_FIPS;
  by state Rating_Area;
run;

proc sort
  data= work.Rating_Area_Map
  out= work.Rating_Area_Sorted;
  by state Rating_Area;
run;

%ANNOMAC;  *<-- Make the annotate macros available;

%MAPLABEL(work.Rating_Area_Sorted,      /* Input Map data set      */
          work.Rating_Area_FIPS,        /* Data set containing label */
          work.Anno_for_Labels,        /* Output data set for GMAP */
          Rating_Area,                 /* Variable to label on     */
          State Rating_Area,           /* List of sort variables   */
          font='Albany AMT/bold',      /* FONT= Font for the label */
          color=black,                 /* COLOR= Color of the label */
          size=3);                    /* SIZE= Size of the label  */
```

The output of the MAPLABEL macro is shown below in Table 2: Annotate Data Set for Rating Area Labels. These variables tell the GMAP procedure where to place the labels. In this case, the “function” variable contains the value of “label”, which as the name implies, will place a label at the x and y coordinates. The “position” variable contains the value of “5”, which means that the text is centered at the x position and slightly lower than the y. The “xsys” and “ysys” variables are used to identify the coordinate system utilized when applying the x and y variables. A value of “2” means that actual values are used based on the dimension of the area needed for the actual map. The “hsys” variable defines the coordinate system used for the “size” variable. In this case, a value of “3” means that the coordinate system will be based on a percentage of the entire graphics area, usually a percentage of the whole page. A complete description of the annotate variables can be found on the SAS documentation website (SAS, 2015).

Table 2: Annotate Data Set for Rating Area Labels

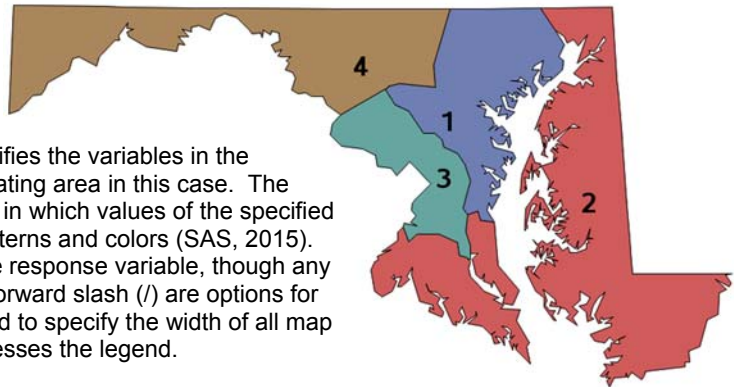
state	<u>Rating Area</u>	<u>position</u>	<u>xsys</u>	<u>ysys</u>	<u>hsys</u>	<u>when</u>	<u>Text</u>	<u>function</u>	<u>size</u>	<u>style</u>	<u>color</u>	<u>x</u>	<u>y</u>
24	1	5	2	2	3	a	1	label	3	'Albany AMT/bold'	black	0.006559	0.006897
24	2	5	2	2	3	a	2	label	3	'Albany AMT/bold'	black	0.018232	0.000151
24	3	5	2	2	3	a	3	label	3	'Albany AMT/bold'	black	0.006235	0.002110
24	4	5	2	2	3	a	4	label	3	'Albany AMT/bold'	black	-0.000664	0.011447

ADD LABELS TO MAP

Once the annotate data set as been created, it can be applied to the GMAP procedure. In this example, the annotate data set is applied using the ANNO= argument. This tells SAS to use the values of Table 2 to apply additional elements to the map.

GMAP DESCRIPTION

The DATA= argument is used to identify the response data, such as PMPM or utilization per thousand. In this case, the Rating Area variable will be used for demonstration purposes only because it is numeric and will render different colors on the map based on the number of areas. The MAP= argument is used to identify the map data set and the ALL argument tells SAS to include all of the map areas from the map data set even if there is no response data. The ID statement identifies the variables in the input data sets that define map areas - state and rating area in this case. The CHORO statement creates two-dimensional maps in which values of the specified response variables are represented by varying patterns and colors (SAS, 2015). As previously stated, the rating area is used as the response variable, though any numeric value can be used. Everything after the forward slash (/) are options for the CHORO statement. The WOUTLINE=1 is used to specify the width of all map area outlines in pixels and the NOLEGEND suppresses the legend.



```
* Add Labels to Rating Area Map;
proc gmap
  data= work.MD_Rating_Area_Map
  map= work.MD_Rating_Area_Map
  anno= work.Anno_for_Labels
  all;
  id state Rating_Area;
  choro Rating_Area /
    woutline=1
    nolegend;
run;
quit;
```

ADD MAJOR CITIES

1 The MAPS.USCITY data set is used to obtain the LAT and LONG coordinates of the major cities desired for the map. Notice that the (x,y) coordinates are dropped when the data set is first referenced because those coordinates were projected using all of the states in the U.S. and would not properly fit in the correct location when only Maryland is projected.

2 This DATA step will be used to create an annotate data set and the variables needed to label the cities are created here. Notice that the RETAIN statement is used to initialize variables like "function" and "style" and as the statement name implies, these variables will be retained on each record. Then the "text" variable is set to the name of the "city" and the "position" is set to the value of "2", which will place the city name at the center of the x coordinate and above the y coordinate.

3 The LAT and LONG variables, which are the un-projected coordinates in degrees, are then converted to radians in order to project them along with the Maryland map coordinates. The Major_City variable will be used later to identify the major city records after projection. Meaning, it will be used within an IF statement to split the map and city coordinates into separate data sets.

```
* Add major cities;

data work.Major_Cities;
  set MAPS.USCITY ( drop= x y );

  where state = stfips('MD')
    and city = 'Baltimore';

  retain function 'label'
    style 'Albany AMT/bold'
    xsys '2' ysys '2' size 2
    color 'black' when 'a';

  text = city; * Text value = city name;
  position = '2'; * Above center of X,Y;

  * Convert degrees to radians *;
  x = atan(1)/45 * long;
  y = atan(1)/45 * lat;

  Major_City = 1; * Flag for projection;

  output; * Output record for city names;

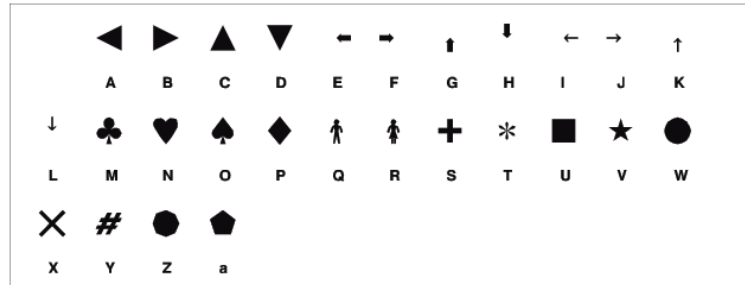
  * Prepare city marker values;
  text='V'; style='marker'; position='5';

  output; * Output record for city markers;
run;
```

4

The records for the city names are then created using the OUTPUT statement and a marker for the cities is also created. The "text", "style", and "position" variables are changed so that a symbol can be used to identify the location of the city. The variable "text" is set to the value of "V" and the "style" variable is set to "marker", a SAS/GRAPH font that will display symbols, or markers, in place of letters. Note the letter "V" will be represented by a star when rendered using the GMAP procedure.

Marker Font



PROJECT MAJOR CITIES

5

The major cities now need to be projected along with the map data set. This is done by combining the two tables using a SET statement within a DATA step.

6

Then the combined data set can be projected just as before by state and rating area.

7

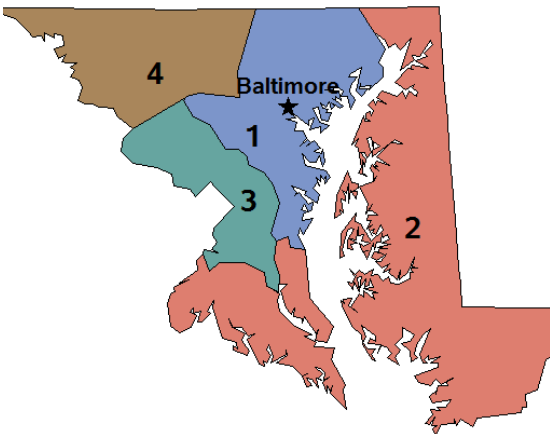
Once both of the data sets are using the same projection, they can be split apart using the Major_City variable, created in the previous step. If Major_City = 1, then output to the Major_Cities data set. Else output the values to the Rating_Area_Map.

8

Now that the major cities have been projected to the map, they can be added to the annotate data set that was created for the rating area labels. This is also done using a SET statement within a DATA step.

9

The resulting data set, called Annotate_Data_Set, will be used in the GMAP procedure to render the major cities.



```
* Project new map with annotations;
data work.Combined;
  set work.Gremove_Map
      work.Major_Cities;
run;
```

5

```
proc gproject
  data= work.Combined
  out= work.Combined dupok;
  id state Rating_Area;
run;
```

6

```
data work.Rating_Area_Map
  work.Major_Cities;
  set combined;
  if Major_City = 1
  then output work.Major_Cities;
  else output work.Rating_Area_Map;
run;
```

7

```
* Put annotate data sets together;
data work.Annotate_Data_Set;
  set work.Anno_for_Labels
      work.Major_Cities;
run;
```

8

```
* Map with new annotate data set;
proc gmap
  data= work.Rating_Area_Map
  map= work.Rating_Area_Map
  anno= work.Annotate_Data_Set
  all;
  id state Rating_Area;
  choro Rating_Area /
    woutline=1
    nolegend;
run;
quit;
```

9

Every state can be done using this method except for Alaska, Massachusetts, Nebraska, and California. Alaska, Massachusetts, and Nebraska, which use the 3-digit zip code (instead of county) for all rating areas, will be discussed later, in the section on ZCTAs. California, which uses the 3-digit zip code to split Los Angeles County, is described below.

CONVERT DEGREES TO RADIANS

In the previous section on adding major cities to the map, MAPS.USCITY was used to obtain the centroids of each locale in order to annotate them onto the map. The (x,y) coordinates in that file have already been projected along with all of the other states in the U.S. Therefore, the cities will not be rendered correctly if used with the Rating Area Map that was projected using only Maryland coordinates.

In order to correctly place them on the map, the process above describes how to convert the LAT and LONG variables from degrees into radians in order to properly project them along with the Maryland coordinates. This section will briefly discuss the reason for the conversion and will provide a description of the math involved in the calculation.

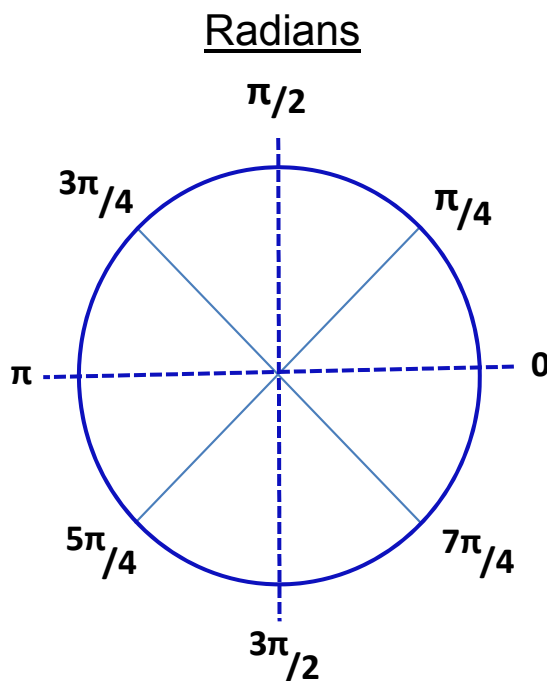
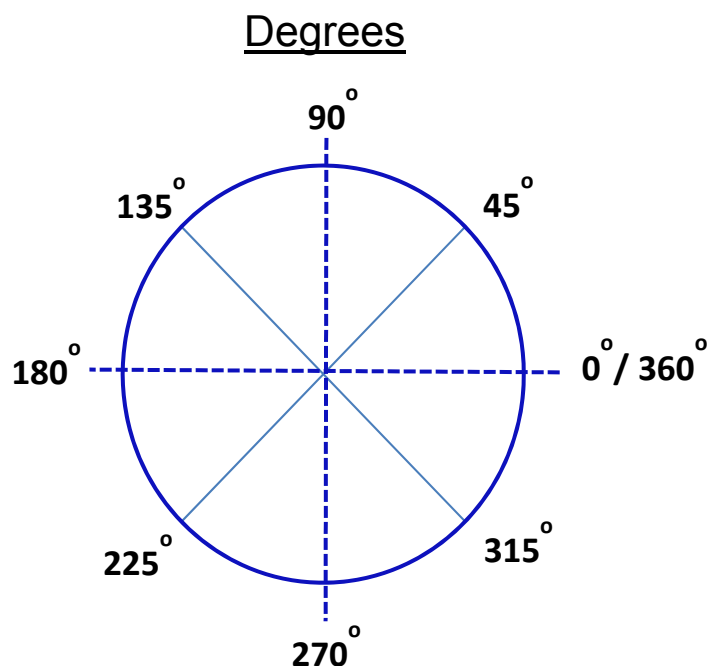
The GPROJECT procedure can process either radians or degrees, though all records need to have the same coordinate units. Since the (LAT, LONG) values in the MAPS.COUNTY data set are in radians, then the major cities data also need to be in radians.

Radians and degrees are two units for measuring angles. Because the ancient Babylonians viewed the numbers 6, 12, and 60 as having particular religious significance, one revolution of a circle is divided into $6 \times 60 = 360$ parts called "degrees". Interestingly, that is also the reason that there are twelve-hour nights and twelve-hour days, with each hour divided into sixty minutes and each minute divided into sixty seconds.

The radian is the standard unit of angular measure, used in many areas of mathematics. An angle's measurement in radians is numerically equivalent to the length of a corresponding arc of a unit circle; one radian is just under 57.3 degrees (when the arc length is equal to the radius). Hence, 90° is $\pi/2$ as shown in the exhibit to the right.

One way to convert any angle from degrees to radians is to take the arc tangent of 1, divide by 45 and multiply by the degrees (Allison, 2012). Another method is to divide the constant value of π by 180 and multiply by the degrees (Zdeb, 2002).

$$\begin{aligned} 90^\circ &= \pi / 2 &&= 1.5707 \text{ radians} \\ 91^\circ &= \text{atan}(1) / 45 \times 91 &&= 1.5882 \\ 92^\circ &= \text{constant}('Pi') / 180 \times 92 &&= 1.6057 \end{aligned}$$



IDENTIFY RATING AREA SPLIT

Since California decided to use 3-digit zip codes to split Los Angeles County, there are two options. One is to add a SAS/GRAPH marker in the form of a triangle, which is similar to using a star to identify a major city. Another is to actually split the county into two parts. Splitting a county is not an easy task and must be done manually since there is no procedure to do it.

To determine how to best split the county, the centroids of each zip code were obtained from the SASHELP.ZIPCODE data set and displayed on a map. The DATA step shown below creates another annotate data set that will generate red dots for rating area 15 and black dots for rating area 16.



```
* Annotate a solid-filled pie at each zip code;

%let ra15 = '906','907','908','910','911',
            '912','915','917','918','935';

%let ra16 = '900','902','903','904','905',
            '913','914','916','923','928','932';

data work.zipcodes;
  length Rating_Area $2 function style color $8
         position $1;
  retain xsys ysys '2' hsys '3' when 'a';

  set SASHELP.ZIPCODE (rename=(x=LONG y=LAT));

  where state = stfips('CA')
        and CountyNM = 'Los Angeles'
        and substr(put(zip,z05.),1,3) in (&ra15,&ra16);

  if substr(put(zip,z05.),1,3) in (&ra15) then
    do;
      Rating_Area = '15';
      color='red';
    end;

  if substr(put(zip,z05.),1,3) in (&ra16) then
    do;
      Rating_Area = '16';
      color='black';
    end;

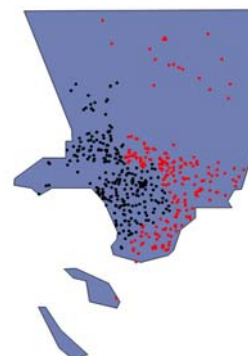
  * Convert degrees to radians *;
  x = atan(1)/45 * LONG * -1;
  y = atan(1)/45 * LAT;

  * Create solid-filled pie;
  function='pie';
  style='psolid';
  position='5';
  rotate=360;
  size=.2;
  anno_flag=1;
run;
```

Since this is a similar process to what was done to annotate the major cities, all of the code will not be presented. First, create a macro variable containing a list of the 3-digit zip codes for each of the rating areas. Then use the macro variables (1) in the WHERE clause to limit the data to only those codes and (2) in the IF statements to identify the rating area codes and the color of the dots.

Next convert the LAT and LONG coordinates to radians and establish the variables needed to create the dots/pies. The method of combining this data with the un-projected map data for use in a GPROJECT and for splitting apart those data sets once projected is not shown.

This exercise is meant to approximate the rating area boundaries. The map depicted below reveals the basic split of Los Angeles County, including rating area 16 (black dots), shown in the southwest area of the map.



PLOT LOS ANGELES COUNTY

The first statement creates a macro variable called "name" that will be used later in a SAS/GRAPH procedure. The FILENAME statement points to the SAS root directory, represented by a period, where the output will be stored.

The first DATA step subsets MAPS.COUNTIES in order to obtain the un-projected map coordinates of Los Angeles County. It also takes the (x,y) values and stores them in the LONG and LAT variables for later use. That subset is then projected to obtain new (x,y) coordinates so they can be plotted.

The second DATA step takes the projected output and creates an Original_Order variable using _N_ to identify the order in which the data are input. The automatic _N_ DATA step variable keeps track of the number of data step loops. In this case, it is used to keep track of the records as they are read. The order is important because the GMAP procedure uses this sequence to create the polygon. If the records are out of sequence, the county will not render correctly. The My_Html variable is used to create "alt text" for each data point, which is useful for identifying the LAT and LONG coordinates of the county boundary.

The Output Delivery System (ODS) statements point to an HTML file that will create a web page containing the "alt text", which creates a popup message to appear when the mouse is hovered over the data point. Note the use of the "name" macro variable that was set at the beginning of the program.

The GOPTIONS statement sets up various graphics options and the SYMBOL statement describes how each data point will be displayed in the output. Each statement is fully described in the SAS documentation (SAS, 2015).

The GPLOT procedure plots the values of two or more variables on a set of X and Y coordinates axes, which, in this case, corresponds to the LONG and LAT coordinates of the map data set. The use of the WHERE statement keeps only the first segment because segment 1 is the only polygon that needs to be split. This will be described later. The PLOT statement "plots" the LAT by (*) LONG and contains three options: HREVERSE, HTML, and NAME. The HREVERSE option specifies that the order of the values on the horizontal axis be reversed in order to present the county as if it were on the map. The HTML option creates the "alt text"; the NAME option specifies the name of the catalog entry and the name of the graphics output file. The ODS statements then tell SAS to create the HTML file.

```
* Plot of Los Angeles County;

%let name = custom_county;

filename odsout '.';

data work.LA_Unprojected;
  set MAPS.COUNTIES;
  where fipstate(state) = 'CA'
    and county = 37;
  LAT=y; LONG=x;
run;

proc gproject
  data= work.LA_Unprojected
  out= work.LA_Project;
  id state county;
run;

data work.LA_Project;
  set work.LA_Project;
  length My_Html $300;
  Original_Order = _N_;
  My_Html =
    'title='||quote('Original_Order: '||
      trim(left(Original_Order))||'0d'x||
    'x='||trim(left(LONG))||
    'y='||trim(left(LAT))||' in radians');
run;

ODS LISTING CLOSE;
ODS HTML path=odsout body="&name..html"
  style=sasweb;

goptions
  device=png noborder gunit=pct
  htitle=4 ftitle="albany amt/bold"
  htext=2.25 ftext="albany amt"
  xpixels=600 ypixels=600;

symbol1 value=circle height=2
  interpol=join color=blue
  pointlabel=(h=8pt "#Original_Order");

title "Coordinates of LA county";

proc gplot data=work.LA_Project;
  where segment=1;
  plot LAT*LONG /
    hreverse
    html=my_html
    name="&name";
run;

ODS HTML CLOSE;
ODS LISTING;
```

DETERMINE COORDINATES

Notice the order of values on the Coordinates of LA County plot. Data points of note include number 1, the starting point; number 168, the ending point, and number 4, which appears to be a good point to use with a new polygon to represent the rating area split because it happens to correspond with the northernmost zip code centroid. This coincidental coordinate corresponds with the black X of the inlay map.

The light colored X's mark points that appear to be good locations for drawing a new line to separate the two rating areas. This line is not the exact location of the actual zip code boundaries, but is a general representation of the borders. Determining the x,y coordinates of these two points can be tedious, but is possible with a bit of trial and error using the LAT and LONG axis of the GPLOT procedure.

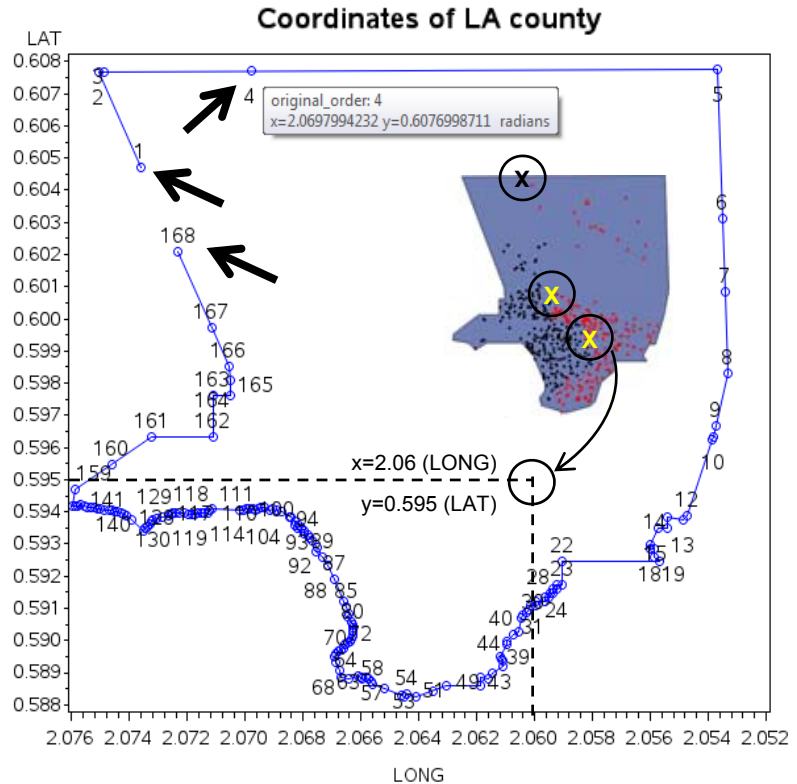


Table 3: Original Order of LA County Map Coordinates depicts the output of the original, projected MAPS.COUNTIES data set. The x,y values are projected and the LONG,LAT values are un-projected. Also note that the values for Original Order 4 for LONG and LAT match the popup of the "alt text" in the output from the GPLOT procedure and that the values are in radians, not degrees.

Table 3: Original Order of LA County Map Coordinates

Original Order	X	Y	State	County	Segment	LONG	LAT
1	0.006660	-0.04579	6	37	1	2.07362	0.60471
2	0.005448	-0.04283	6	37	1	2.07508	0.60768
3	0.005604	-0.04283	6	37	1	2.07489	0.60768
4	0.009782	-0.04278	6	37	1	2.06980	0.60770
... some data not shown ...							
165	0.009265	-0.05238	6	37	1	2.07051	0.59810
166	0.009204	-0.05198	6	37	1	2.07058	0.59850
167	0.008709	-0.05077	6	37	1	2.07117	0.59972
168	0.007738	-0.04842	6	37	1	2.07233	0.60208

INSERT NEW COORDINATES

Once the coordinates have been extracted from MAPS.COUNTIES and the original order has been determined, new coordinates can be added to the data set that corresponds to the approximate border of the two rating areas. Notice the dashed lines that have been superimposed onto the output of the GPLOT procedure intersect at x=2.06 (LONG) and y=0.959 (LAT). These are the estimated coordinates where a new data point could be added. It is also a good point to start the polygon in lieu of the original starting point. The same approximation can be done for the second data point. Therefore, the third data point, which already exists in the original data set, will need to be changed from the number 4 to the number 3. Furthermore, since the polygon will be started in a new location, the first three data points from the original order need to be moved to the end of segment number 1, which is after the original order of 168, in this case.

Four new records (1, 2, 65, and 66), representing the two new x,y coordinates needed to separate the two rating areas, have been added in Table 4: New Order of LA County Map Coordinates.. The reason for four additional records instead of just two is that when a new coordinate is added to a polygon, an additional record should also be added to the neighboring polygon, and they must be in the order that they will be drawn, or connected, by the GMAP procedure. Also note that the Rating_Area variable has been added to designate the split where the new order of 64 is where rating area 15 ends; and the order number 65 is where rating area 16 begins. In addition, the original order records 1 thru 4 have been moved to 171 through 174 so that they are at the end instead of at the beginning of the segment.

Once the new coordinates for LA County have been added to the data set and put into the correct order, they can be re-plotted so the results can be reviewed. The GOPTIONS statement resets the symbols. The SYMBOL1 statement sets the value to a circle, instructs the plotted points to be intersected, and selects the color blue to be used. This will be used to plot rating area 15. SYMBOL2 sets the value to "X" and sets the color to red for rating area 16. Notice the use of the third-variable, "Rating_Area", which is a classification variable against which LAT and LONG are plotted.

Note that in the plot for New Coordinates of LA County, the x and y values for the new order of 1 also equal the values for the new order of 66. Also, both a circle and an "X" exist at that same location since both of those coordinates exist in both polygons. This is true for new order 2 and 3 as well. Once the new records have been verified using the GPLOT procedure, they can be added to the MAPS.COUNTIES data set, but only after the original values for LA County have been removed.

Table 4: New Order of LA County Map Coordinates

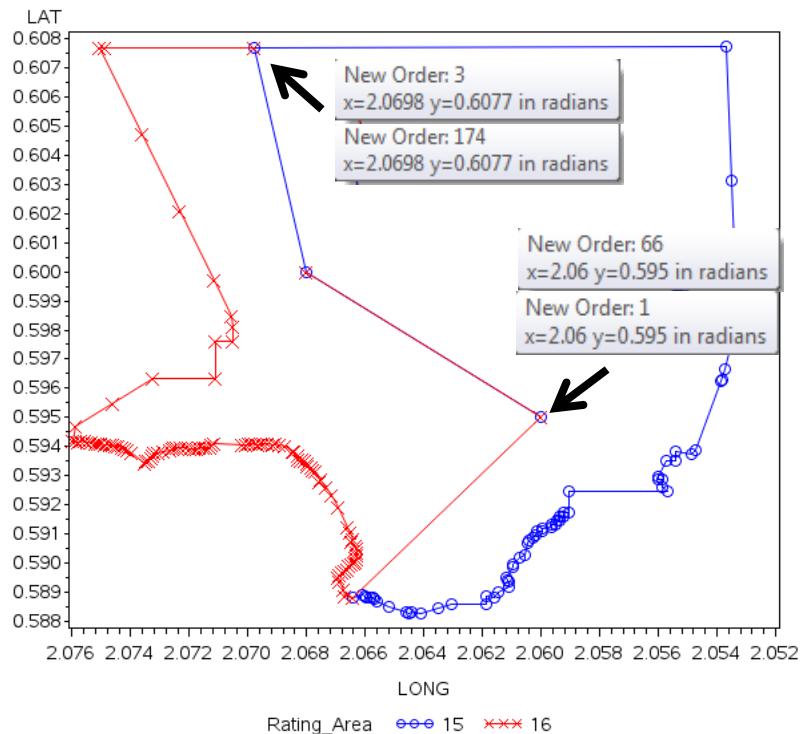
New Order	Original Order	Rating Area	State	County	Segment	LONG	LAT
1	n/a	15	6	37	1	2.06000	0.59500
2	n/a	15	6	37	1	2.06800	0.60000
3	4	15	6	37	1	2.06980	0.60770
... some data not shown ...							
64	65	15	6	37	1	2.06642	0.58880
65	n/a	16	6	37	1	2.06800	0.60000
66	n/a	16	6	37	1	2.06000	0.59500
67	65	16	6	37	1	2.06642	0.58880
... some data not shown ...							
170	168	16	6	37	1	2.07233	0.60208
171	1	16	6	37	1	2.07362	0.60471
172	2	16	6	37	1	2.07508	0.60768
173	3	16	6	37	1	2.07489	0.60768
174	4	16	6	37	1	2.06980	0.60770

```
* Plot New Coordinates for LA County;
options reset=symbol;

symbol1 value=circle height=3 interpol=join c=blue;
symbol2 value=x height=3 interpol=join c=red;

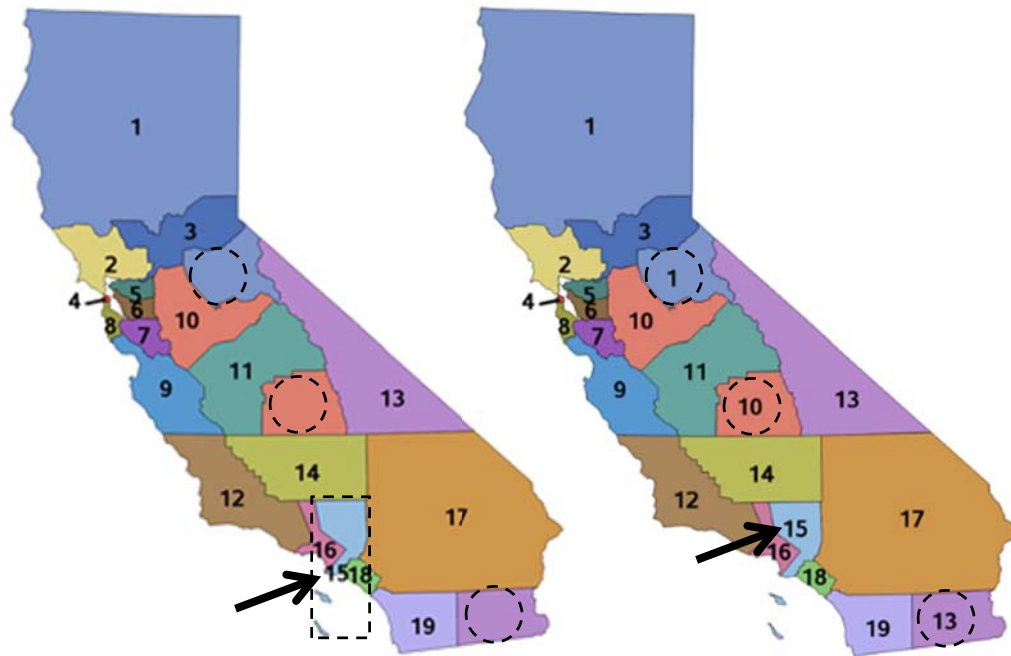
title "New Coordinates of LA county";
proc gplot data=work.New_LA;
  where segment = 1;
  plot LAT*LONG=Rating_Area /
    hreverse
    html=my_html
    name="&name";
run;
```

New Coordinates of LA county



MODIFY ANNOTATE DATA SET

Notice in the first map shown below that the label for rating area 15 is in the center of the dotted rectangular box. This is because the area also contains the two islands, Two Harbors and San Clemente. Therefore, the MAPLABEL macro used the entire area when calculating the centroid. Also, there are three counties that do not have a label because rating areas are not required be contiguous, therefore an additional value must be created to identify these areas. Thus, the labels for rating areas 1, 10, and 13 must be duplicated and manually placed in those areas.



In the code below, the annotate data set is being modified to move rating area 15 and output additional records for rating areas 1, 10, and 13. The first IF statement adds 0.001 to the x value in order to shift the label slightly to the right and adds 0.012 to the y value to move up. Directly after that first IF statement is an OUTPUT statement, which tells SAS to “output” all of the records it has just read without any conditional restraint. The next IF statement outputs a “new” record for rating area 1, but only after setting the x and y values to the centroid of the county. The values, -0.015 and 0.016, were manually obtained by trial and error. The values from a neighboring label were used as a starting point and were incremented by small amounts until the desired location was reached. Similarly, the last two if statements add labels for rating areas 10 and 13.

```
* Modify annotate data set;
data work.annotate_data_set_modified;
set work.annotate_data_set;

* Move label for rating area 15;
if Rating_Area='15' then
do;
x = x + 0.001;
y = y + 0.012;
end;

output; *-- Output original annotation records;

* Add labels for non-contiguous areas;
if Rating_Area='1' then do; x=-0.015; y= 0.016; output; end;
if Rating_Area='10' then do; x= 0.005; y=-0.017; output; end;
if Rating_Area='13' then do; x= 0.055; y=-0.072; output; end;
run;
```

RATING AREAS BY ZIP CODE

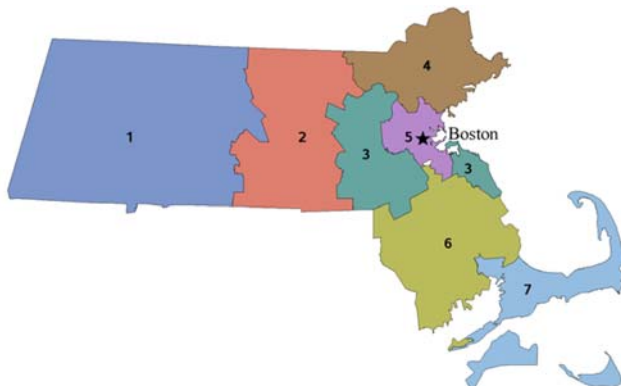
For the three states that use zip codes to define rating areas, Alaska, Massachusetts, and Nebraska, the ZCTA files from the U.S. Census Bureau are used. As previously mentioned, these files are derived from an ESRI® shapefile format (.shp) that need to be downloaded and subsequently imported into SAS map data sets. Information about where to obtain these files can be found on the Census Bureau website (Census, 2010).

This code offers a brief description of importing and processing the ZCTA files, though additional instructions on this subject can be found in other SAS papers (Okerson, 2013). As the name suggests, the MAPIMPORT procedure is used to import the shapefile; the ID statement within it is used to reorder the map polygons; and the RENAME statement renames the ZCTA5CE10 variable to ZipCode.

The FORMAT procedure is used to create a user-defined format called “areas” that identifies the first three digits of the zip codes that make up the Massachusetts rating areas. The PUT statement in the DATA step is used to apply the format to a new variable called Rating_Area.

With the exception of the ZipCode variable in the ID statement of the GREMOVE procedure, which removes the internal zip code boundaries from the rating areas, the remainder of the code is essentially the same as in previous examples. The addition of the EASTLONG option in the GPROJECT procedure specifies that the longitude (X variable) values in the input map data set increase to the east (that is, positive longitude values are east of the prime meridian.) The DUPOK option is used to specify that observations are retained when their projected X and Y values are identical to those in the previous observation while the DEGREES option indicates the data are in degrees in lieu of radians (SAS, 2015).

Note that some of the code required to replicate the Massachusetts map depicted here is not shown because it is essentially the same as the previously described code.



```
* Import ZCTA File into SAS;
proc mapimport
  datafile='tl_2010_25_zcta510.shp'
  out= work.ZCTA_MA;
  id ZCTA5CE10;
  rename ZCTA5CE10 = ZipCode;
run;

proc format;
  value $areas
    '010','011','012','013' = '1'
    '014','015','016'      = '2'
    '017','020'             = '3'
    '018','019'             = '4'
    '021','022','024'       = '5'
    '023','027'             = '6'
    '025','026'             = '7'
    other = 'Missing';
run;

data work.ZCTA_MA_Modified;
  length Rating_Area $2;
  set work.ZCTA_MA;
  Rating_Area =
    put(substr(ZipCode,1,3),$areas.);
run;

proc gremove
  data= work.ZCTA_MA_Modified
  out= work.ZCTA_Remove;
  by Rating_Area;
  id ZipCode;
run;

... some code not shown ...

proc gproject
  data=work.combined
  out=work.projected_map
  dupok eastlong degrees;
  id Rating_Area;
run;

... some code not shown ...

proc gmap
  data= work.projected_map
  map= work.projected_map
  anno= work.annotate_data_set
  all;
  id Rating_Area;
  choro Rating_Area /
  nolegend;
run;
quit;
```

COMBINE COUNTY AND ZCTA

Once the county and ZCTA data sets have been imported for each state, they can be combined into a permanent SAS map data set for use by others. The LIBNAME statement creates a library reference called "geo_data," that points to the location of the folder in which to store the final data set.

The MACRO statement defines a macro called "by_state" and passes one parameter called "state". The APPEND procedure uses the "work.Combined" data set as described earlier to set each state table together. Note the use of the WHERE clause to remove the major city records. This step can be repeated in order to create a data set containing only the desired cities. The MEND statement ends the macro definition and the %by_state calls the macro, passing each two- digit state code.

```
* Combine County and ZCTA;
libname geo_data 'C:\My_Directory';

%macro by_state ( state );

    ... some code not shown ...

proc append
    data= work.Combined
    out= geo_data.Geo_Rating_Area_Map
    force;
    where Major_City ne 1;
run;

%mend by_state;

%by_state ( AK );
%by_state ( AL );

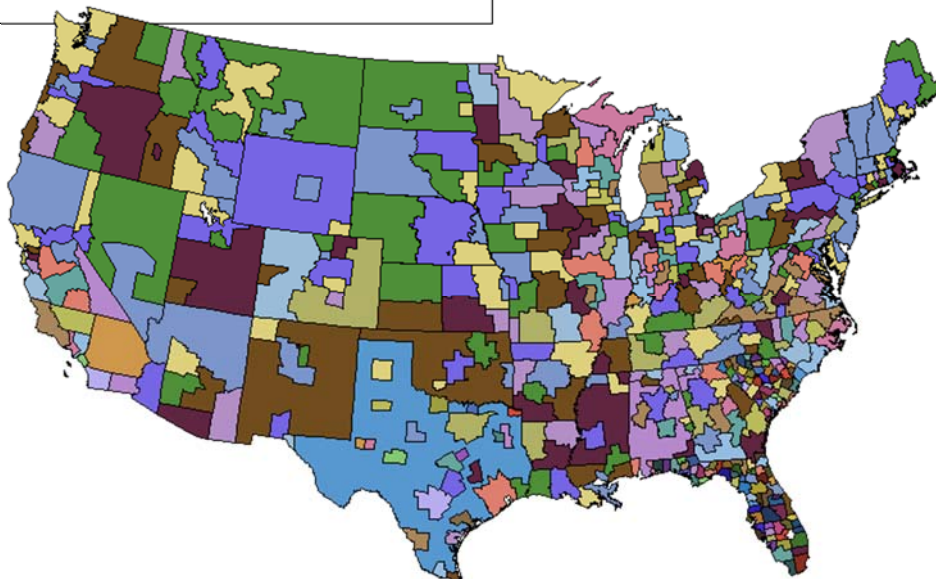
    ... some code not shown ...
```

```
* Create National Rating Area Map;
proc gproject
    data=geo_data.Geo_rating_area_map
    out=work.Geo_rating_area_map
    dupok;
    where state ne stfips('AK')
        and state ne stfips('HI');
    id state Rating_Area;
run;

proc gmap
    data= work.Geo_rating_area_map
    map= work.Geo_rating_area_map all;
    id state Rating_Area;
    choro Rating_Area /
        discrete nolegend;
run;
quit;
```

After the data have been combined, a separate program can be used to project a map for the entire nation, or for each individual state, without having to recreate the data set each time.

Notice, in the code shown here, that Hawaii and Alaska have been removed from the projection. This is because they lie off the mainland and, if projected along with the 50 contiguous states, would cause the map to look awkward. Although not shown, the x,y coordinates for those states can be modified so they appear under the U.S. map. Information on doing this can be found at the SAS Product Documentation website (SAS, 2015).



CONCLUSION

The SAS System offers a myriad of opportunities for creating maps that can be customized in almost any manner imaginable. SAS/GRAPH, ODS, and the Annotate facility are very powerful tools that allow the SAS programmer various options for using SAS supplied map data, importing third party data, and creating user defined maps.

REFERENCES

Allison, Robert, "SAS/GRAPH®: Beyond the Basics", SAS Institute, 2012. Print.

Allison, R (2015), "Census ZIP Code Tabulation Area". Accessed on August 8, 2015 from http://robslink.com/SAS/democd17/zcta_info.htm

Centers for Medicare and Medicaid Services (CMS, 2014). Market Rating Reforms, State Specific Geographic Rating Areas <https://www.cms.gov/CCIIO/Programs-and-Initiatives/Health-Insurance-Market-Reforms/state-gra.html>

Hadden LS, Zdeb M, Allison R (2007). "Wow! You Did That Map with SAS/GRAPH®?" Proceedings of the 31st Annual SAS Users Group International Conference.

Hadden LS, Zdeb M, Allison R (2006). "ZIP Code 411: A Well-Kept SAS® Secret." Proceedings of the 30th Annual SAS Users Group International Conference.

Massengill, D. (2004). "SAS Mapping: Technologies, Techniques, Tips, and Tricks." Available at: <http://support.sas.com/rnd/datavisualization/papers/SASMapping.pdf>.

Massengill, D, Kolovos A (2012). "Together at Last: Spatial Analysis and SAS® Mapping." SAS Global Forum 2012 Proceedings available at <http://support.sas.com/rnd/papers>.

Okerson, B. (2013). "Creating ZIP Code-Level Maps with SAS®" SAS Global Forum 2013. Available at: <http://support.sas.com/resources/papers/proceedings13/214-2013.pdf>

SAS Product Documentation (SAS, 2015). <http://support.sas.com/documentation/>

United States Census Bureau (Census, 2010). ZIP Code Tabulation Areas (ZCTAs). Available at: <http://www.census.gov/geo/reference/zctas.html>.

Zdeb, Mike, "Maps Made Easy Using SAS®", SAS Institute, 2002. Print.

Zdeb, M. "From Shapefiles to Maps Via PROC MAPIMPORT and PROC GMAP," NESUG 2011 Proceedings. "ZIP Code History". Available at: http://zipboundary.com/zipcode_history.html.

Zdeb, M. and Allison, R. "Stretching the Bounds of SAS/GRAPH Software." *Proceedings of the Thirtieth Annual SAS Users Group International Conference*. April 2005.

CONTACT INFORMATION

Rick Andrews
Office of the Actuary
Centers for Medicare and Medicaid Services
7500 Security Boulevard
Baltimore, MD 21244
Phone: (410) 786-6395
E-mail: Richard.Andrews@cms.hhs.gov

Robert Allison
Research and Development
SAS Institute Incorporated
100 SAS Campus Drive
Cary, NC 27513
Phone: (919) 677-8000
E-mail: Robert.Allison@sas.com

The opinions of the presenters are their own and do not represent the Office of the Actuary, Centers for Medicare and Medicaid Services, or the Department of Health and Human Services.

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.

Other brand and product names are trademarks of their respective companies.