

AN INTRODUCTION TO CREATING MAPS WITH SAS/GRAPH

Mike S. Zdeb, New York State Department of Health

INTRODUCTION

Maps can be created via SAS by using the GMAP procedure, one of the PROCs available within SAS/GRAPH. Four different map types can be created using GMAP: choropleth, prism, surface, and block. This paper will discuss creating choropleth maps, i.e. "...2-dimensional maps that represent data values as combinations of pattern and color that fill map areas" (definition from *SAS/GRAPH Software Usage Version 6*). Once you understand how to create a map via a series of examples, you will also learn how to customize the output of GMAP via an annotate data set. The maps produced by the various examples are shown at the end of the paper. They were all produced using a SAS-supplied postscript printer device driver (if 'device driver' has already caused confusion, don't worry since they are discussed in a later section).

Choropleth maps are produced by using a combination of a map data set and a response data set. A map data set contains the information needed to draw map boundaries, e.g. countries, states, counties, zip codes, etc. A response data set contains the information that is to be displayed on the map, e.g. birth rates, death rates, number of AIDS cases, etc.

If you have licensed SAS/GRAPH, you have access to a number of map data sets that are provided with the software. Most of the map data sets provided with SAS/GRAPH contain geographic areas (boundaries) represented in terms of longitude and latitude, x and y coordinates respectively. The x and y coordinates are usually in radians, not in the more familiar metric of degrees. This distinction, radians versus degrees, will normally be of no concern when using SAS-supplied map data sets. However, on those occasions when labels or symbols are to be added to a map via an annotate data set, it is important to understand the metric being used for the map's x-y coordinates.

In addition to x-y coordinates, map data sets also contain an identification (ID) variable. The ID identifies the geographic area associated with each pair of x-y coordinates. In many of the SAS-supplied map datasets, the name of the identification variable is ID. However, when the term ID variable in this paper, it is a generic reference to the variable that identifies a map area. Typical ID variables for maps of areas of the United States are state, county, zip code, census tract, etc. In order to map the response data, the response data set must

also contain an ID variable that allows SAS/GRAPH to match response data to the map data set. The ID variable in the map and response datasets must have the same name and type (i.e. character or numeric). The action of GMAP is somewhat analogous to a match-merge performed in a data step. A match-merge matches observations from two or more datasets via one or more by-variables. PROC GMAP matches observations from a response data set to those in a map data set via one or more ID variables. A data step match-merge results in a new dataset, while PROC GMAP results in a geographical display of your response data.

There may also be other variables in a map dataset. All of the SAS-supplied map datasets contain the variable SEGMENT, used by PROC GMAP when a map area comprises more than one polygon. An example of such an area is the state of Hawaii where multiple polygons (islands) make up an area with a single ID variable, STATE. Several of the datasets also contain the variable DENSITY. That variable is useful in controlling the amount of detail to be displayed in map boundaries. You will probably never concern yourself with map segments. If you use any of the SAS-supplied maps of the United States or Canada, you will learn that the density variable is quite useful.

MAKE YOUR OWN MAP

Though you might associate map boundaries with coordinates in terms of longitude and latitude, PROC GMAP will draw any shapes that you create in a map data set. Example 1 constructs a map dataset that contains four rectangles. A response data set is created with one observation matched to each map area (i.e. rectangle). PROC GMAP creates a map.

...Example 1...

```
*the response dataset;
data poly_dat;
input rectangl z @@;
datalines;
1 10 2 20 3 30 4 40
;
run;
```

```
*the map dataset;
data poly_map;
input rectangl x y @@;
datalines;
1 0 0 1 1 0 1 1 1 0 1
2 1 0 2 2 0 2 2 1 2 1 1
3 0 1 3 1 1 3 1 2 3 0 2
4 1 1 4 2 1 4 2 2 4 1 2
;
run;
```

```
*fill patterns for map areas;
pattern1 c=black v=m3n0;
pattern2 c=black v=m3n45;
pattern3 c=black v=m3n135;
pattern4 c=black v=m3n90;
title 'EXAMPLE #1 - MAKE YOUR OWN MAP';
```

```
*create the map;
proc gmap
data=poly_dat
map=poly_map;
id rectangl;
choro z/discrete;
run;
quit;
```

The map dataset contains the minimum set of variables needed to draw a map. There are x-y coordinates and an ID variable, i.e. RECTANGL. The patterns used to fill each rectangle are specified in a set of pattern statements. More about patterns available within PROC GMAP can be found in volume one of the *SAS/GRAPH Software Reference*. The various portions of the pattern definition (v=) control line density, angle, and crosshatching. It is important to specify a color (c=) within each pattern. Without a color, each pattern statement defines as many patterns as there are colors available on the output device being used. Thus, if no colors were specified, each rectangle would be filled with the first pattern in a series of different colors. If you have ever used PROC GPLOT, you'll notice that this is the same behavior as occurs with SYMBOL statements.

Within PROC GMAP, the map and response (data=) datasets are specified. An ID statement specifies that the variable that links the data to the various map areas is RECTANGL. The option CHORO requests a choropleth map (as opposed to a PRISM, SURFACE, or BLOCK map). Only one PROC GMAP option is used, i.e. DISCRETE. This option tells the procedure to treat the numeric variable Z as discrete rather than continuous. Each of the four values of the response variable (10, 20, 30, and 40) is displayed with a different pattern, and a legend is produced at the bottom of the page. Without the DISCRETE option, PROC GMAP uses its own set of rules to determine the values and number of levels of the response variable to be grouped and displayed. The default behavior is equivalent to using the option MIDPOINTS. The LEVEL and MIDPOINTS options can also be used to control the display of data, but the DISCRETE option will be used for any maps produced in this paper.

USING A SAS-SUPPLIED MAP

There are several SAS-supplied maps that can be used to produce maps of all of the United States, of groups of states, or of individual states. Example 2 constructs an outline map of the United States,

using the dataset US as both the map and response dataset.

...Example 2...

```
pattern v=e;
title 'EXAMPLE #2 - US MAP DATASET';
```

```
proc gmap
map=us
data=us (obs=1)
all;
id state;
choro state/nolegend;
run;
quit;
```

Since we want an outline map (no fill patterns), an empty pattern is chosen (v=e). The map dataset US is specified. We also need a response dataset with an ID variable named STATE. We know the dataset US contains the variable STATE, so it is also used as the response dataset. A dataset option is used to limit the response dataset to one observation. PROC GMAP only displays those areas that are found in both the map and response datasets unless the ALL option is used. If that option had been left out, our map would have displayed one state, Alabama, the first observation in the US dataset. The ALL option tells GMAP to draw all of the geographic areas in the map dataset, even when there is no data for an area in the response dataset. The NOLEGEND option is used to suppress display of the legend. Since only one value of the variable STATE is displayed, the DISCRETE option is not needed.

USING A SUBSET OF A SAS-SUPPLIED MAP

Several of the map datasets provided with SAS/GRAPH allow you to produce county maps of any state. The next example uses a portion of one of the datasets to produce an outline map showing New York State counties (notice the author's affiliation).

...EXAMPLE 3...

```
*create a 'no fill' pattern, repeated 62 times;
pattern c=black v=e r=62;
title1 'EXAMPLE #3 - COUNTIES MAP DATASET';
title2 'NEW YORK STATE COUNTIES';
```

```
proc gmap
data=counties
map=counties;
id county;
choro county/nolegend;
where state eq 36;
run;
quit;
```

Once again, the same dataset is used as both the map and response dataset. However, since a WHERE statement is used to select only New York State counties via the FIPS (Federal Information Processing System) county number, a dataset option specifying OBS=1 (as used in example 2) cannot be used on the same dataset. Therefore,

there will be a number of different values for the variable being mapped, i.e. COUNTY. New York has 62 counties, so the repeat option (r=) is used in the pattern statement to have the same pattern used for each county. The ALL option is not needed since each county is present in both the map and response datasets. The DISCRETE option is not needed since the same pattern is used no matter what value GMAP decides to map via the default MIDPOINTS option.

You may have noticed (then again, if you are not familiar with New York, you may not have) that the map produced by example 3 is backward and distorted (then again, if you are familiar with New Yorkers...). Why did that occur if it did not in example 2? The answer lies in the type of x-y coordinates present in the US versus the COUNTIES dataset. Both datasets define areas in terms of longitude and latitude. In the US dataset, the values have already been projected, while the COUNTIES dataset contains unprojected points.

When you draw a graph, both the x and y coordinates normally begin in the lower left corner of the page. X-values get larger as you go from left-to-right, and y-values get larger as you go up. Latitude increases (in the northern hemisphere) as you move up from the equator, but longitude increases as you travel from right-to-left (east-to-west). The reason that the map is backward is that longitude increases from right-to-left and GMAP draws the map using a standard x-y coordinate system where x increase from left-to-right. The map is distorted since the longitude/latitude coordinate system is based on spherical coordinates, not a flat surface. Example 4 shows how to correct the map produced in example 3. PROC GPROJECT is used to correct both the direction and distortion

...Example 4...

```
*create a map dataset containing projected
coordinates - limit the map to New York state;
proc gproject
data=counties (where=(state eq 36))
out=nys;
id state;
run;

pattern v=e;

proc gmap
data=nys (obs=1)
map=nys
all;
id county;
choro county/nolegend;
note 'EXAMPLE #4 - COUNTIES MAP DATASET';
note 'NEW YORK STATE COUNTIES (CORRECTED VIA
PROC GPROJECT)';
run;
quit;
```

New York State counties are chosen in PROC GPROJECT via a dataset option. A map dataset is

produced containing projected longitude and latitude. We can revert to the method used in example 2 (no repeated patterns, and an OBS=1 dataset option on the response dataset) since STATE selection was already done in PROC GPROJECT (not in GMAP as done in example 3). The ALL option is needed. Without it, only one county would be displayed (remember, OBS=1).

Rather than use TITLE statements, NOTES are used to label the map. In examples 2 and 3, the title 'consumed' a portion of the display area. By default, NOTES begin in the upper left corner of the display area. Since they 'share' rather than 'consume' the display area, more area is available for the map. A map of New York state offers a lot of room for text in the upper left side of the display area.

USING 'REAL' DATA

Thus far, we have produced maps that displayed geographic areas, but no real data. The next example will use the NYS dataset produced via PROC GPROJECT in example 4 and display some 'real' data. The appendix shows the data step that produces dataset NEWBORNS containing one observation for each New York State county and two variables, a FIPS county number and the number of births in each county during 1996.

...Example 5...

```
*create a format to group observations;
proc format;
value birthfmt
low-499      = '<500'
500-999     = '500-999'
1000-9999  = '1,000-9,999'
10000-high  = '10,000+';
run;

*fill patterns for the map areas;
pattern1 v=s c=grayff;
pattern2 v=s c=grayee;
pattern3 v=s c=grayaa;
pattern4 v=s c=gray68;

proc gmap
data=newborns
map=nys;
id county;
choro births/discrete;
format births birthfmt.;
note 'EXAMPLE #5 - BIRTHS IN NEW YORK STATE,
1996';
run;
quit;
```

PROC FORMAT is used to create a format that can be used to group the counties by the number of births. Since four groups are defined in the format, four patterns are created to fill the areas (counties) on the map with solid (v=s) shades of gray (c=). The various grays are defined via hexadecimal notation, with the 256 available shades ranging from GRAYFF (white) to GRAY00 (black). Note that not all output devices can use gray scale shading. The DISCRETE option is needed to control how GMAP forms groups, and the format BIRTHFMT is used to

define groups based on the number of births in each county. The DISCRETE option is needed even though the format will create four groups of counties.

One problem with this map is that impossible to distinguish adjacent counties that are shaded with the same gray fill. When a solid pattern is specified, it is outlined in the same color used to fill the area. The white box outlined in white in the legend blends in with the background and adjacent counties in the map sometimes blend together. This problem can be corrected via the COUTLINE= option. Since there is room in the lower right side of the display, the legend will be moved to create more area for displaying the map.

...Example 6...

```
legend1
across=2
origin=(15,10)
label=none
shape=bar(3,4)
mode=share;

proc gmap
data=newborns
map=nys (where=(density lt 6));
id county;
choro births/
discrete
legend=legend1
coutline=black;
format births birthfmt.;
note 'EXAMPLE #6 - BIRTHS IN NEW YORK STATE,
1996';
run;
quit;
```

The legend statement offers control over both the appearance and placement of the legend. The ORIGIN= option takes advantage of the empty area in the lower left side of the display area (origin coordinates are expressed in terms of percentage of display area). The MODE= option tells GMAP that the legend will SHARE the display area rather than force the map to appear above the legend. The other legend options control appearance. PROC GMAP must be told to use the legend via an option (legend=). The three GMAP options result in an acceptable display of the data.

In the introduction to this paper, it was stated that the variable DENSITY is present in some map datasets. It is present in the COUNTIES dataset that was used to produce the dataset NYS via PROC GPROJECT. The values of density range from 0 to 6. X-y coordinates with a density of 6 provide the most detail to area boundaries. At the scale of a state, the observations with a density of 6 can be left out of map creation without losing much detail. Subtle differences can be seen in the output from example 6 when compared to example 5. They are most evident in looking at the islands in the lower right and top center of the maps.

The elimination of the 'extra coordinates' will reduce CPU time in creating a map and will speed up map creation on slow output devices. If the variable DENSITY is not present in a map dataset, PROC GREDUCE can be used to create it.

COMBINING MAP AREAS

There are times when it is necessary to display data in areas that are combinations of smaller geographic areas. For example, in New York state, counties can be aggregated into larger health service areas. There is a SAS/GRAPH procedure that allows you to remove the shared borders between areas that are combined. Example 7 shows how PROC GREMOVE can be used to convert a county-based map dataset into a health service area map. First, a new geographic identifier (HSA) must be added to each observation in the dataset. The value of HSA assigned to each observation is based upon the value of the variable COUNTY. The variable HSA will group counties into health service areas and then be used to identify new areas to be mapped. The data step that assigns an HSA to each observation is shown in the appendix.

...Example 7...

```
proc sort data=hsatemp;
by hsa;
run;

*eliminate county boundaries with HSA areas;
proc gremove
data=hsatemp
out=hsamap;
by hsa;
id county;
run;

proc gproject
data=hsamap
out=hsaproj;
id hsa;
run;

pattern v=e;

proc gmap
data=hsaproj (obs=1)
map=hsaproj
all;
id hsa;
choro hsa/discrete nolegend;
note 'EXAMPLE #7 - HEALTH SERVICE AREAS';
note 'AREAS FORMED FROM COUNTIES VIA GREMOVE';
run;
quit;
```

Once the new variable HSA has been added to each observation in the map dataset, PROC GREMOVE is used to create a new map dataset with HSA rather than county boundaries. Since HSA is used as a by-variable within GREMOVE, the dataset is first sorted by HSA. Within GREMOVE, the by-variable identifies the observations to be combined into new areas, while the ID variable identifies the areas to be combined.

ADDING AREA LABELS VIA ANNOTATE

The map produced in example 9 would be more informative if there were labels that showed the location of each of the eight health service areas. Labels of map areas are not automatically provided within PROC GMAP. However, they can be added via an annotate dataset. If you have ever used a drawing program, you know that adding text to a graphic is usually as simple as: choosing the 'add text' tool, using a mouse to move the cursor to the point on the screen where the label is to be located, typing the text. In addition to these tasks, you also make a decision as to text appearance (font and size).

SAS/GRAPH offers a graphics editor if you are using SAS via the display manager. It can be used to place text on a graphic produced by any of the SAS/GRAPH procedures. If you have a lot of labels to add, this can be tedious. If you are using SAS in batch mode, the editor is not available. Adding labels to a graphic via an annotate dataset can be accomplished in either display manager or batch mode. The annotate dataset can be thought of as a script that contains a literal description of all the actions you would perform and decisions you would make if you were altering a graphic interactively, i.e. via editing with a mouse. Annotation allows you to customize any output produced with SAS/GRAPH. The degree of customization is limited only by your knowledge of annotate datasets.

In example 8, labels are added to the HSA-based map that was produced in example 7. The labels are placed at the center of each of the areas. The x-y coordinates in the map dataset are longitude and latitude expressed in radians. The easiest way to place a label at the center of each is to express the location of the HSA centers in the same metric, i.e. longitude and latitude in radians.

...Example 8...

```
data labels;
length text $5;
retain
xsys ysys '2'
position '5'
function 'label'
size 2.5
style 'swiss'
when 'a'
;
input text x y;
*the longitude/latitude of HSA area centroids;
datalines;
HSA-1 1.371803 0.742941
HSA-2 1.348815 0.744540
HSA-3 1.321340 0.756707
HSA-4 1.323836 0.738309
HSA-5 1.294420 0.756696
HSA-6 1.295074 0.726438
HSA-7 1.288815 0.710749
HSA-8 1.273433 0.713304
;
run;
```

```
*combine the map dataset with map labels;
data map_labl;
set hsmap labels;
run;

*project all observations (map and labels);
proc gproject
data=map_labl
out=hsaproj;
id hsa;
run;

*separate the labels from the map dataset;
data hsaproj labels;
set hsaproj;
if function eq 'label' then output labels;
else output hsaproj;
run;

pattern v=e;

*label the map via the annotate option;
proc gmap
data=hsaproj (obs=1)
map=hsaproj
all;
id hsa;
choro hsa/
discrete
nolegend
annotate=labels;
note 'EXAMPLE #7 - HEALTH SERVICE AREAS';
note 'LABELS ADDED TO AREAS VIA ANNOTATION';
run;
quit;
```

The first data step creates the annotate dataset. In the dataset, there are a number of variables that answer the questions: what, where, how, and when. The variable FUNCTION defines 'what' - annotation will create a label. Variables X, Y, and POSITION define 'where' - X and Y are map coordinates and POSITION indicates where on the x-y location the label is to be placed (a value of 5 calls for a label centered on the x-y location). The variables STYLE and SIZE define 'how' - STYLE chooses the font and SIZE the text height. Finally, a value of 'a' for WHEN places the label on the map after the map is drawn. With unfilled areas, WHEN is not important. If areas are filled, a value of 'b' (before) will make the labels invisible since they will be under the filled areas.

The only variables left to be explained are XSYS, YSYS, and HSYS. For now, all you need to know about XSYS and YSYS is that they are both assigned a value of two when the x-y coordinates in the annotate dataset are in the same metric as the map dataset. In this example, both the map and annotate datasets are in terms of longitude and latitude in radians. If the variable HSYS is assigned a value of three, the value of the variable SIZE represents a percentage of the display area. You should not expect to understand all the nuances of annotation from one example. What you should realize though is that annotation gives you a lot of power to control the appearance of SAS/GRAPH output.

Once the dataset LABELS is created, it is combined with the map dataset HSAMAP to form MAP_LABEL. The combined dataset is then projected via GPROJECT. You must project the map and labels together to insure that the labels will be placed in the correct location on your output. Just accept this notion for now. Once you have more experience with SAS/GRAPH mapping and annotation, you will reach that '...eureka, I understand...' moment. The final step prior to mapping is to separate the projected map dataset from the projected annotate dataset. This is done in a data step. Since only observations from the original dataset LABELS have a value for the variable FUNCTION, the if-then-else statement will correctly separate the datasets. PROC GMAP looks much the same as it did in example 8, except for the additional option ANNOTATE= that uses the dataset LABELS to label the map areas.

PROC GMAP OUTPUT

When running SAS/GRAPH in display manager mode, the output from PROC GMAP is directed to the output window. If you are using SAS/GRAPH in batch mode or want to send output directly to a laser printer, plotter, etc., you must use a GOPTIONS statement and specify the output device. SAS/GRAPH provides drivers for a large number of output devices. As stated in the introduction, all the graphics shown at the end of this paper were produced using a SAS-supplied postscript printer driver, i.e. PS300. The GOPTIONS statement used to produce all the output was:

```
goptions dev=ps300 noprompt rotate=landscape
border ftext=hwpsl009 htext=2.5
gunit=pct;
```

The value of 'hwpsl009' for FTEXT results in a HELVETICA font. A value of 2.5 for HTEXT in combination with GUNIT=PCT results in text whose height is 2.5 percent of the display area.

FINAL THOUGHTS

Just as with other SAS/GRAPH procedures, PROC GMAP can produce 'publication quality' results. If you want to display data on a state or county basis, the needed maps are supplied with SAS/GRAPH, as are those needed to produce maps of many different countries. Description of SAS-supplied map datasets can be found in *SAS Technical Reports* P-196 and P-208.

If you want to create maps of other areas, e.g. zip-based maps, you will have to create your own map datasets or purchase them. Some vendors sell files already in SAS map dataset format. If you want to find such vendors, you can use a web browser and

go to the SAS Institute web site, www.sas.com. If you use the search engine and specify 'I need a map', you will find vendor names. Since the format for map datasets is quite easy to understand, you might prefer to convert mapping coordinates from other formats to SAS map datasets.

The most difficult part of this paper is the section on annotation. As was stated in that section, the intent of the example is not to make you an annotate guru. Rather, it is intended to show you how to use annotation to overcome one of the deficiencies of PROC GMAP, i.e. the lack of area labeling.

APPENDIX

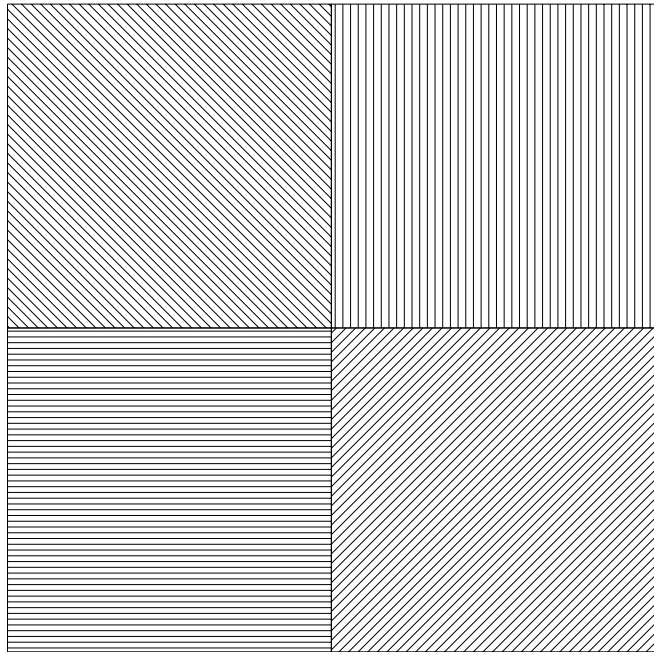
The following data step produces dataset BIRTHS, used in examples 5 and 6.

```
data newborns;
input county births @@;
datalines;
001 3307 003 581 005 23404 007 2258
009 1109 011 991 013 1689 015 1078
017 621 019 891 021 670 023 583
025 454 027 3348 029 12031 031 413
033 510 035 641 037 753 039 499
041 43 043 704 045 1793 047 40928
049 369 051 712 053 858 055 9669
057 575 059 17722 061 20045 063 2744
065 2702 067 6283 069 1146 071 4893
073 526 075 1509 077 549 079 1282
081 32691 083 1945 085 5882 087 4239
089 1242 091 2523 093 1777 095 364
097 206 099 395 101 1136 103 19953
105 839 107 630 109 851 111 1976
113 673 115 695 117 1217 119 12696
121 471 123 327
;
```

The following data step adds the variable HSA (health service area) to a subset of the COUNTIES map dataset. The value of the variable HSA is determined by the value of the variable COUNTY. The new dataset, HSATEMP, is used in examples 7 and 8.

```
data hsatemp;
set counties;
where state eq 36 and density lt 6;
if county in (3 9 13 29 37 63 73 121)
then hsa='1';
else
if county in (15 51 55 69 97 99 101 117 123)
then hsa='2';
else
if county in (11 23 43 45 49 53 65 67 75 89 109)
then hsa='3';
else
if county in (7 17 107)
then hsa='4';
else
if county in (1 19 21 25 31 33 35 39 41 57 77
83 91 93 95 113 115)
then hsa='5';
else
if county in (27 71 79 87 105 111 119)
then hsa='6';
else
if county in (5 47 61 81 85)
then hsa='7';
else
if county in (59 103)
then hsa='8';
run;
```

EXAMPLE #1 - MAKE YOUR OWN MAP

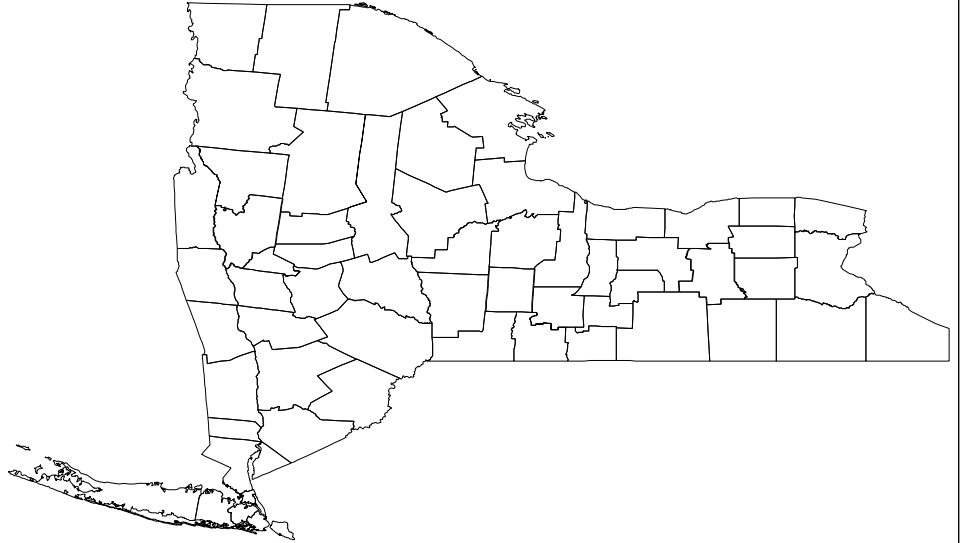


Z 10 20 30 40

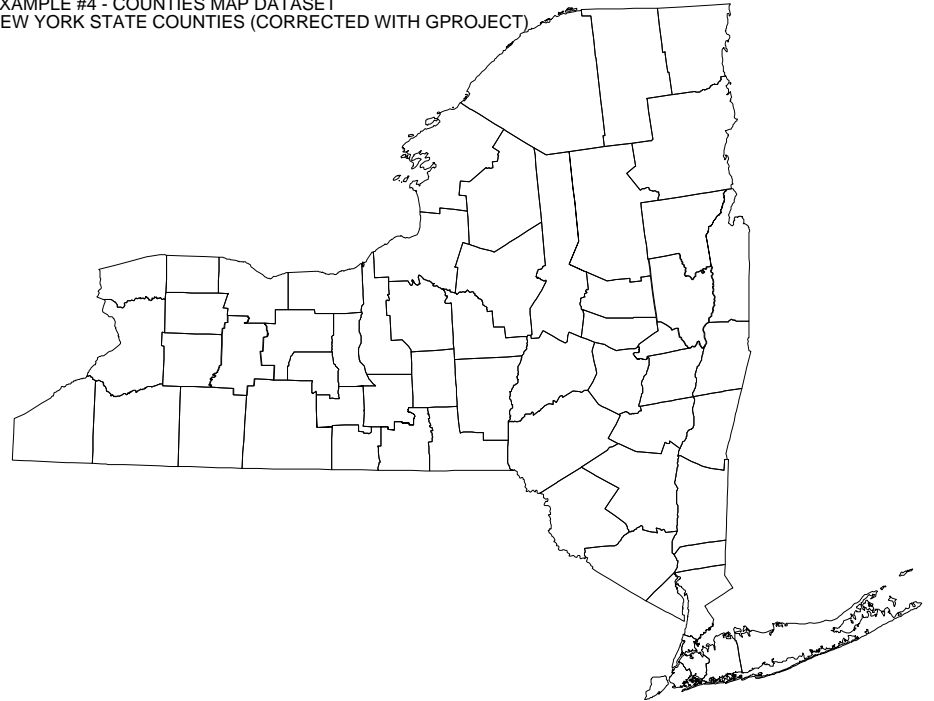
EXAMPLE #2 - US MAP DATASET



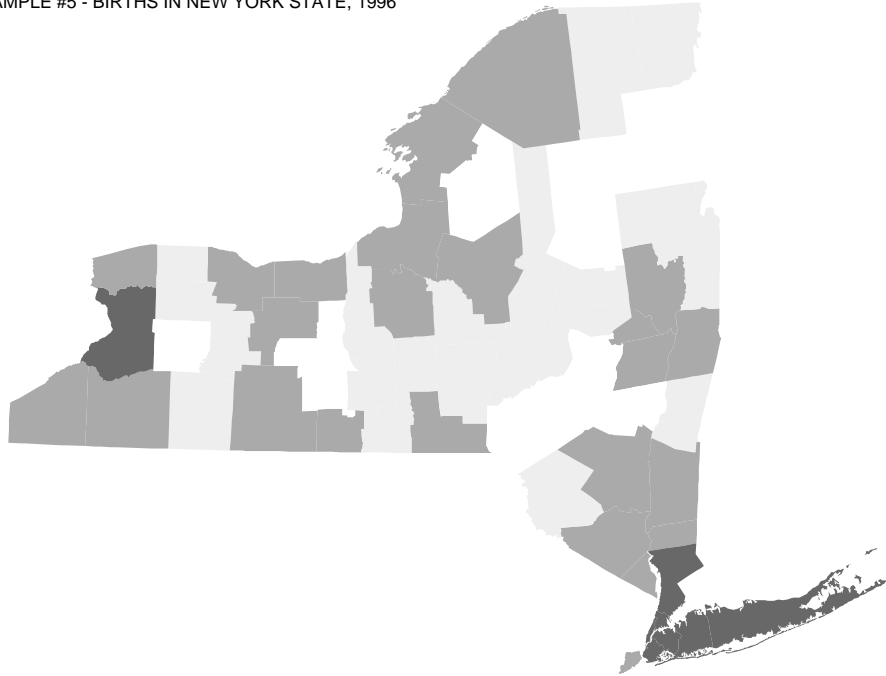
EXAMPLE #3 - COUNTIES MAP DATASET
NEW YORK STATE COUNTIES



EXAMPLE #4 - COUNTIES MAP DATASET
NEW YORK STATE COUNTIES (CORRECTED WITH GPROJECT)

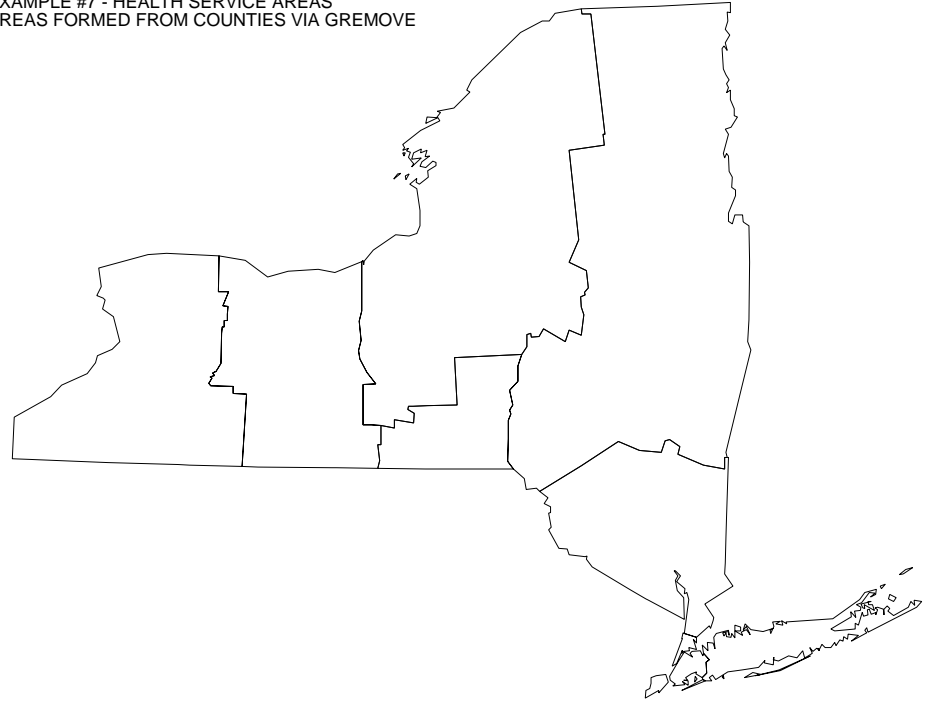


EXAMPLE #5 - BIRTHS IN NEW YORK STATE, 1996

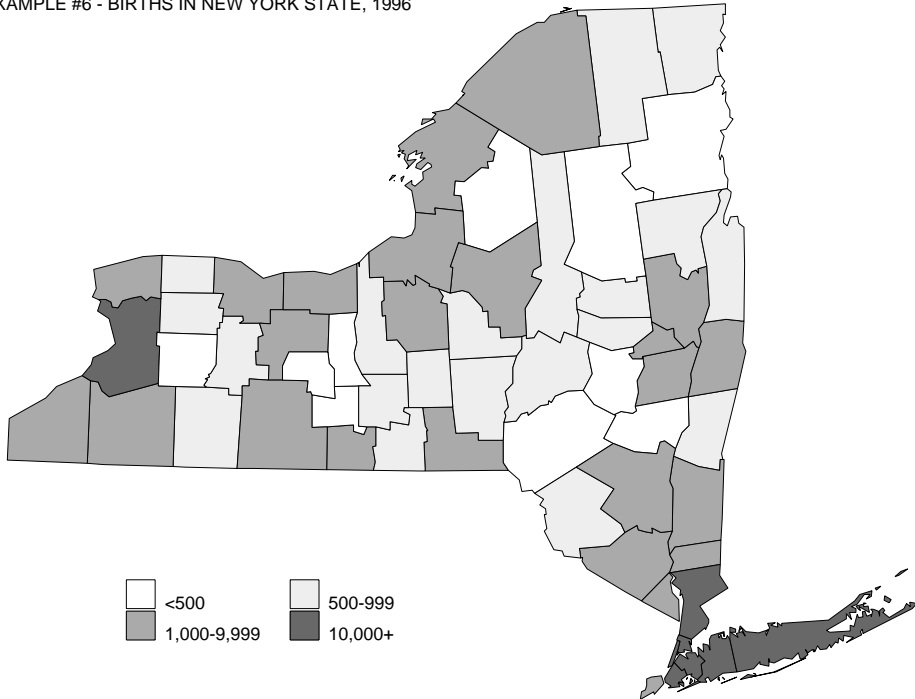


BIRTHS <500 500-999 1,000-9,999 10,000+

EXAMPLE #7 - HEALTH SERVICE AREAS
AREAS FORMED FROM COUNTIES VIA GREMOVE



EXAMPLE #6 - BIRTHS IN NEW YORK STATE, 1996



<500 500-999
1,000-9,999 10,000+

EXAMPLE #8 - HEALTH SERVICE AREAS
LABELS ADDED TO AREAS VIA ANNOTATION

