

Paper 251-29

The Basics of Map Creation with SAS/GRAPH®

Mike Zdeb, University@Albany School of Public Health, Rensselaer, NY

ABSTRACT

Maps can be created with SAS® by using either SAS/GIS® or the GMAP procedure (PROC), one of the procedures available within SAS/GRAPH. This paper shows you how to use the GMAP procedure, not SAS/GIS. 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 after using a series of examples, you will also learn how to customize the output of GMAP using an annotate data set and also be introduced to some methods of producing web-based maps.

INTRODUCTION

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. population, birth rates, death rates, 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 data sets, 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 data sets 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 data sets using one or more by-variables. PROC GMAP matches observations from a response data set to those in a map data set using one or more ID variables. A data step match-merge results in a new data set, while PROC GMAP results in a geographical display of your response data.

There may also be other variables in a map data set. All of the SAS-supplied map data sets 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 data sets 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 data set 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 map data set;
data poly_map;
input id x y @@;
datalines;
1 0 0 1 1 0 1 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;
```

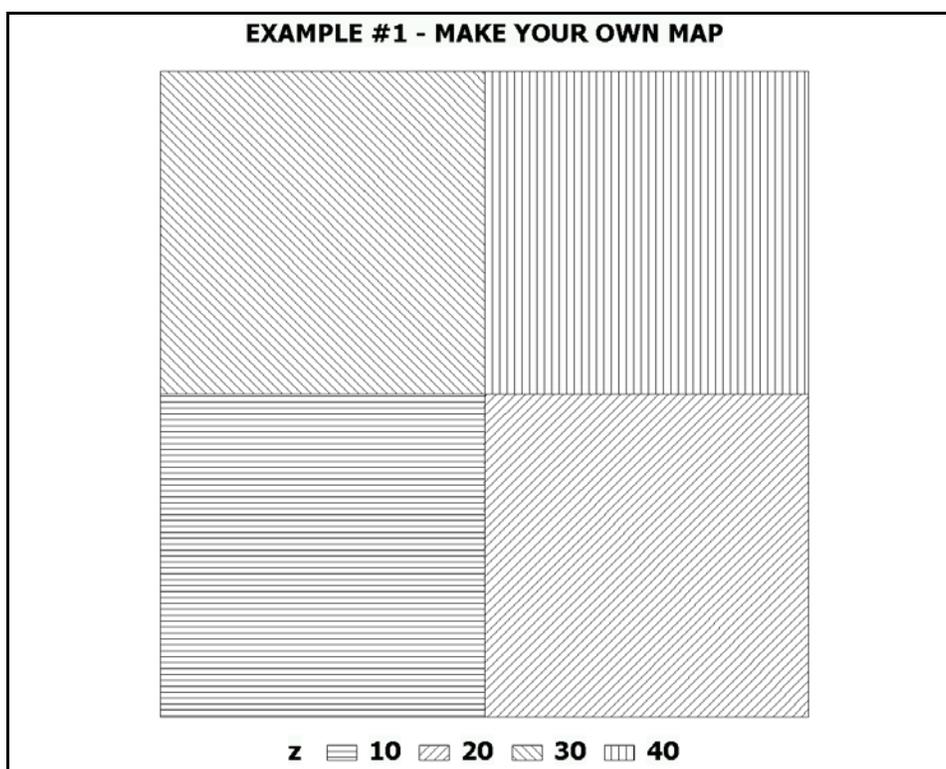
```

* the response data set;
data poly_dat;
input id z @@;
datalines;
1 10 2 20 3 30 4 40
;
run;

* fill patterns for map areas and a title to appear above the map;
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 id;
choro z / discrete;
run;
quit;

```



The map data set 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=) data sets 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

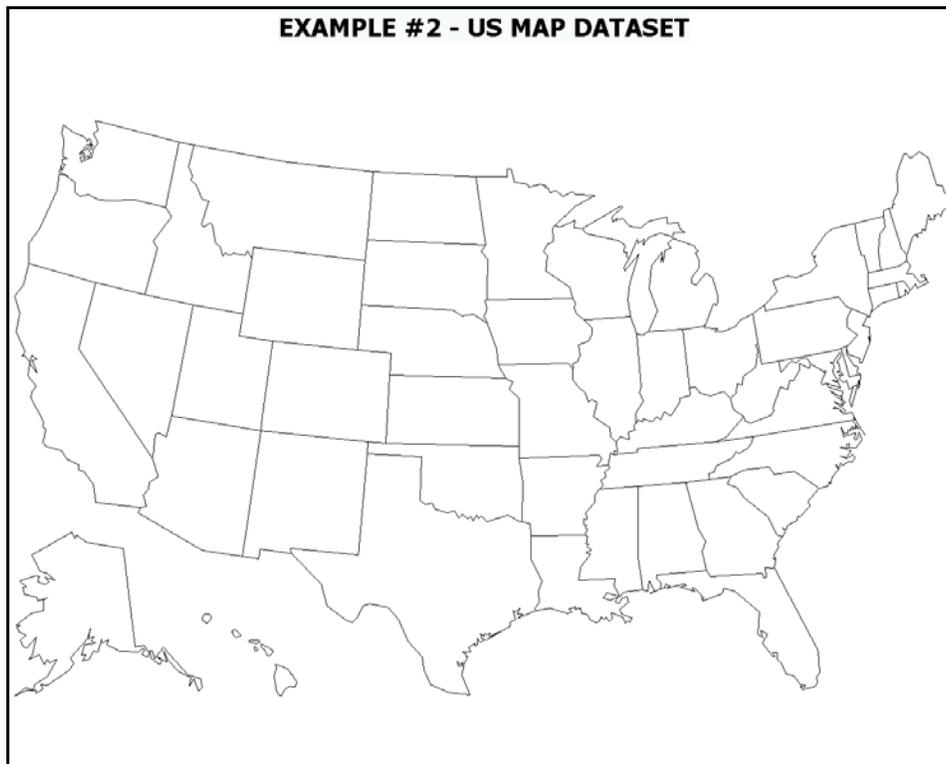
to treat the numeric variable Z as discrete rather than continuous. Each of the four values of the response variable (10, 20, 30, and 40) is displayed with a different pattern, and a legend is produced at the bottom of the page. Without the DISCRETE option, PROC GMAP uses its own set of rules to determine 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 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 data set MAPS.US as both the map and response data set.

```
* example 2;
* select an empty pattern (v=e) and a title;
pattern v=e;
title 'EXAMPLE #2 - US MAP Data set';

* use a SAS-supplied map data set (US) as both the map and response data sets;
proc gmap
map=maps.us
data=maps.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 data set MAPS.US is specified. We do not need a LIBNAME statement to specify the location of the of the SAS-supplied map dat sets. When you install the SAS-supplied map data sets, a default LIBREF with the name MAPS is created and it is available each time you use SAS. We also need a response data set with an ID variable named STATE. We know the data set MAPS.US contains the variable STATE, so it is also used as the response data set. A data set option is used to limit the response data set to one observation. PROC GMAP only displays those areas that are found in both the map and response data sets 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 MAPS.US data

set. The ALL option tells GMAP to draw all of the geographic areas in the map data set, even when there is no data for an area in the response data set. 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.

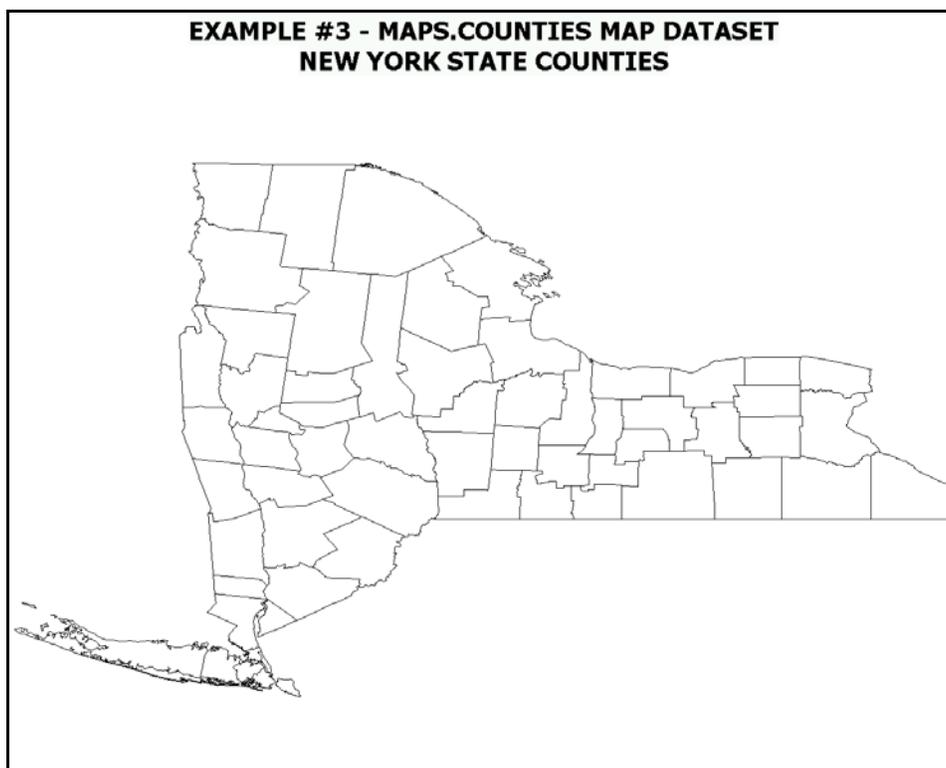
USING A SUBSET OF A SAS-SUPPLIED MAP

Several of the SAS-supplied map data sets allow you to produce county maps of any state. The next example uses a portion of one of the data sets to produce an outline map showing New York State counties (notice the author's affiliation).

```
* example3;
* create a 'no fill' pattern (repeated 62 times - one for each NYS county) and titles;
pattern c=black v=e r=62;

title1 'EXAMPLE #3 - MAPS.COUNTIES MAP Data set';
title2 'NEW YORK STATE COUNTIES';

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



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

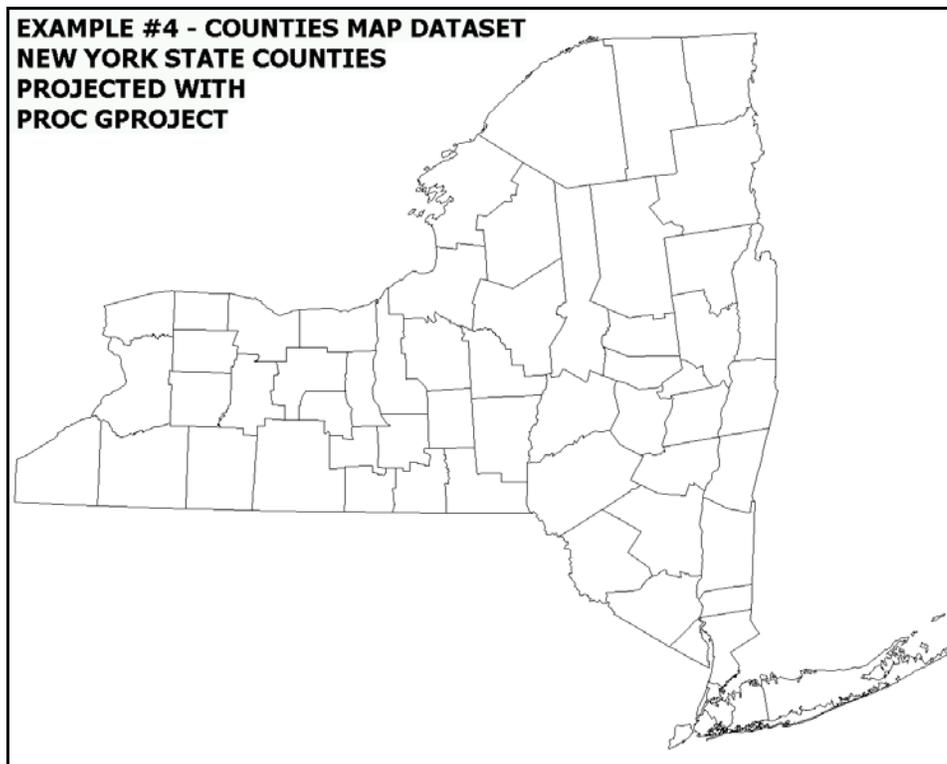
You may have noticed (then again, if you are not familiar with New York, you may not have) that the map produced by example 3 is backward and distorted (then again, if you are familiar with New Yorkers...). Why did that occur if it did not in example 2? The answer lies in the type of x-y coordinates present in the MAPS.US versus the MAPS.COUNTIES data sets. Both data sets define areas in terms of longitude and latitude. In the MAPS.US data set, the values have already been projected, while the COUNTIES data set 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 (our Y-variable) increases (in the northern hemisphere) as you move up from the equator, but longitude (our X-variable) 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 project the map data set, correcting both the direction and distortion

```
* example 4;
* create a map data set containing projected coordinates - limit the map to New York
state;
proc gproject
data=maps.counties
out=nys;
where state eq 36;
id state;
run;

* select an empty pattern (v=e);
pattern v=e;

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



A WHERE statement is used in the GPROJECT procedure to select New York State counties. A map data set is produced containing projected longitude and latitude. We can revert to the method used in example 2 (no repeated patterns, and an OBS=1 data set option on the response data set) 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.

Rather than use TITLE statements, a NOTE is used to label the map. In examples 1 through 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. Each time a text justification is specified (J=L), the NOTE statement goes to a new line. A map of New York state offers a lot of room for text in the upper left side of the display area. One caution with NOTES, since they share the map area, it is possible that text will over write a map. This will not occur with TITLES since they do not share the map area.

USING REAL DATA

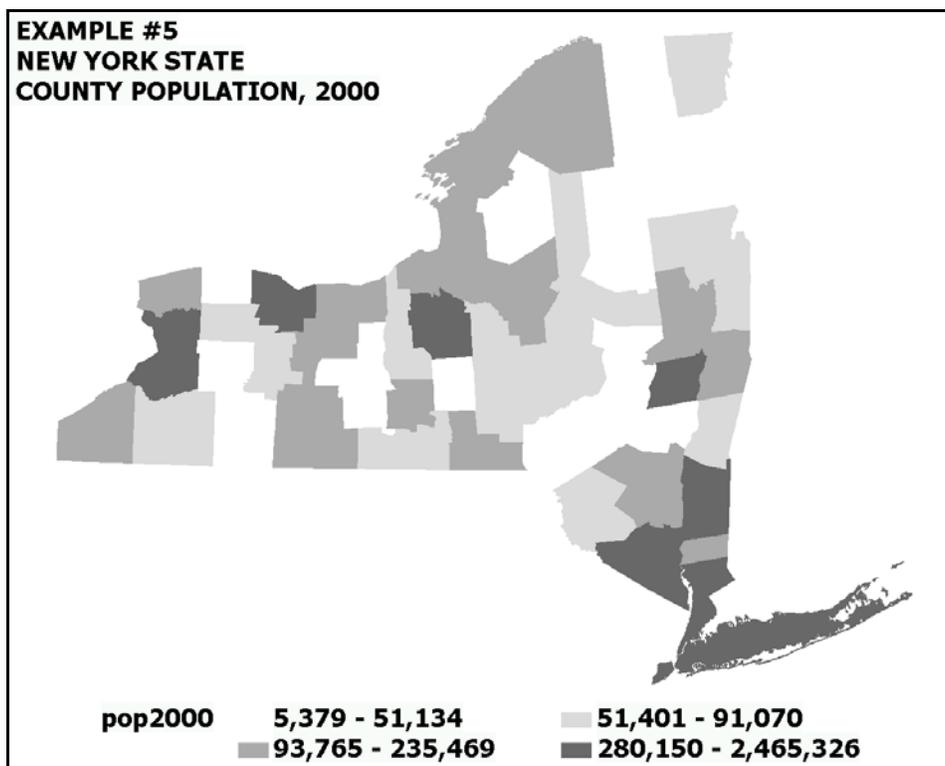
Thus far, we have produced maps that displayed geographic areas, but no real data. The next example will use the NYS data set produced using PROC GPROJECT in example 4 and display some 'real' data. The appendix shows the data step that produces data set NEWBORNS containing one observation for each New 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 (counties) based on the population;
proc format;
value pop2000
5379 - 51134 = '5,379 - 51,134'
51401 - 91070 = '51,401 - 91,070'
93765 - 235469 = '93,765 - 235,469'
280150 - 2465326 = '280,150 - 2,465,326'
;
run;

* 2000 census population, New York State counties;
data nys2000;
input county pop2000 @@;
datalines;
001      294565      003      49927      005      1332650      007      200536
009      83955      011      81963      013      139750      015      91070
017      51401      019      79894      021      63094      023      48599
025      48055      027      280150      029      950265      031      38851
033      51134      035      55073      037      60370      039      48195
041      5379      043      64427      045      111738      047      2465326
049      26944      051      64328      053      69441      055      735343
057      49708      059      1334544      061      1537195      063      219846
065      235469      067      458336      069      100224      071      341367
073      44171      075      122377      077      61676      079      95745
081      2229379      083      152538      085      443728      087      286753
089      111931      091      200635      093      146555      095      31582
097      19224      099      33342      101      98726      103      1419369
105      73966      107      51784      109      96501      111      177749
113      63303      115      61042      117      93765      119      923459
121      43424      123      24621
;
run;

* fill patterns for the map areas (gray-scale fills);
pattern1 v=s c=grayff;
pattern2 v=s c=grayda;
pattern3 v=s c=grayaa;
pattern4 v=s c=gray68;

proc gmap
data=nys2000
map=nys;
id county;
choro pop2000 / discrete;
format pop2000 pop2000_.;
note ' EXAMPLE #5'
j=1 ' NEW YORK STATE'
j=1 ' COUNTY POPULATION, 2000';
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.

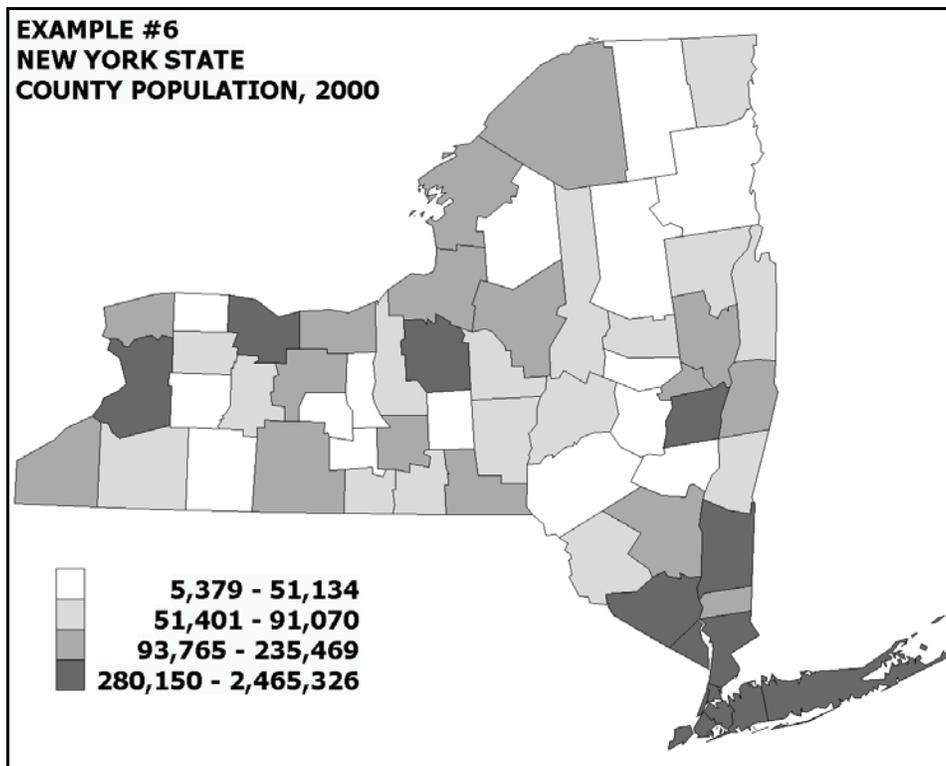
One problem with this map is that impossible to distinguish adjacent counties that are shaded with the same gray fill. When a solid pattern is specified, it is outlined in the same color used to fill the area. The white box outlined in white in the legend blends in with the background and adjacent counties 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;
* fill patterns for the map areas (gray-scale fills);
pattern1 v=s c=grayff;
pattern2 v=s c=grayda;
pattern3 v=s c=grayaa;
pattern4 v=s c=gray68;

* create instructions for legend location and appearance;
legend1
origin=(5,10) pct
mode=share
across=1
label=none
value=(j=r)
shape=bar(3,4) pct;

proc gmap
data=nys2000
map=nys (where=(density lt 6));
id county;
choro pop2000 /
discrete
legend=legend1
coutline=black;
```

```
format pop2000 pop2000_.;
note ' EXAMPLE #6'
j=1 ' NEW YORK STATE'
j=1 ' COUNTY POPULATION, 2000';
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, in this case 5% in the X-direction, 10% in the Y-direction). The MODE option tells the GMAP procedure 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 with an option (LEGEND=1). 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 data sets. It is present in the COUNTIES data set that was used to produce the data set 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 data set, 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 data set into a health service area map. First, a new geographic identifier (HSA) must be added to each observation in the data set. 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;
* assign new geographic designation (HSA) to each New York State counties;
data hsatemp;
set maps.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;

* sort map data set in order by new geographic designation;
proc sort data=hsatemp;
by hsa;
run;

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

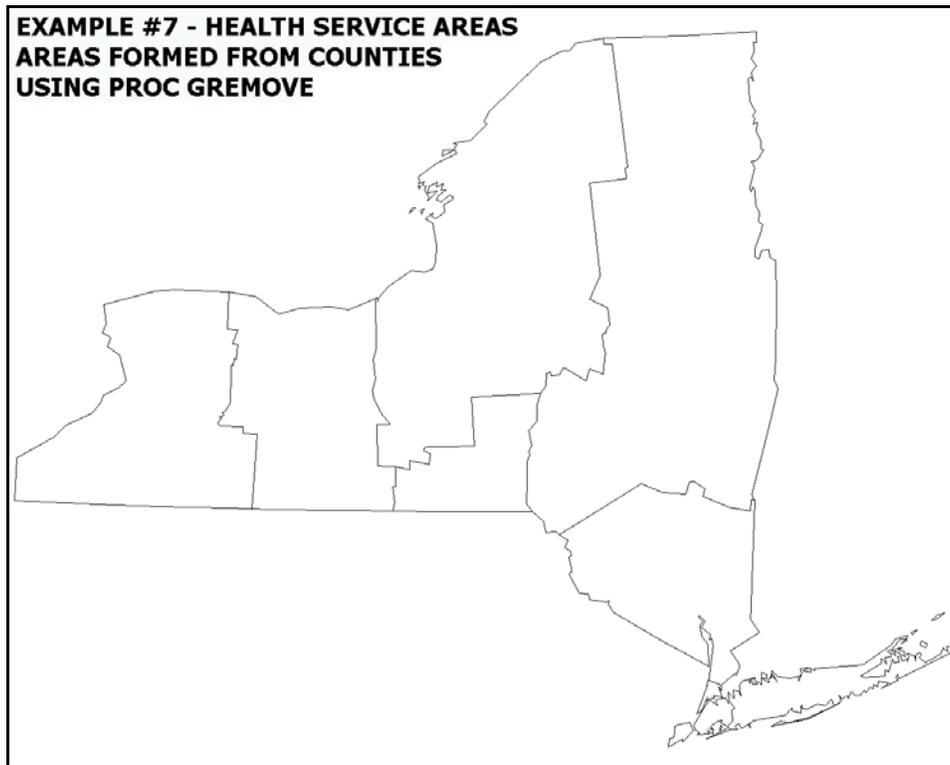
proc gproject
data=hsamap
out=hsaproj;
id hsa;
run;

pattern v=e;

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

```

Once the new variable HSA has been added to each observation in the map data set, PROC GREMOVE is used to create a new map data set with HSA rather than county boundaries. Since HSA is used as a by-variable within GREMOVE, the data set 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 USING 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 using an annotate data set. 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 data set can be accomplished in either display manager or batch mode. The annotate data set 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 data sets.

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 data set 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;
* add labels to map areas;
data labels;
length text $5;
retain
xsys ysys '2'
hsys      '3'
position  '5'
function  'label'
size      3
style     "'Tahoma/bo'"
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 data set with map labels;
data map_labl;
set hsamap labels;
run;

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

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

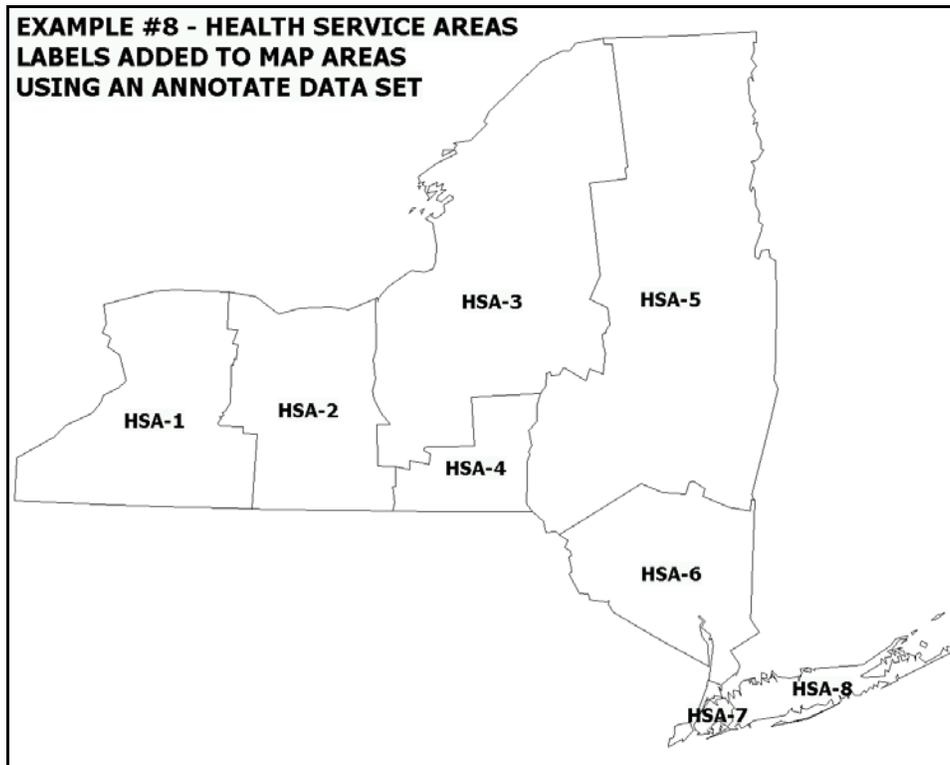
pattern v=e;

* 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 #8 - HEALTH SERVICE AREAS '
j=1 ' LABELS ADDED TO MAP AREAS '
j=1 ' USING AN ANNOTATE DATA SET ' ;
run;
quit;

```

The first data step creates the annotate data set. In the data set, 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 data set are in the same metric as the map data set. In this example, both the map and annotate data sets 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 data set LABELS is created, it is combined with the map data set HSAMAP to form MAP_LABEL. The combined data set 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 data set from the projected annotate data set. This is done in a data step. Since only observations from the original data set LABELS have a value for the variable FUNCTION, the if-then-else statement will correctly separate the data sets. PROC GMAP looks much the same as it did in example 8, except for the additional option ANNOTATE= that uses the data set LABELS to label the map areas.

GRAPHICS OUTPUT

When running SAS/GRAPH in display manager mode, the output from PROC GMAP is directed to the graphics 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 possibly one or more ODS (Output Delivery System) statements. SAS/GRAPH provides drivers for a large number of output devices. All the graphics shown in this paper were produced using the SAS-supplied GIF driver. The GOPTIONS statement used to produce all the output was...

```
goptions
reset=all
dev=gif
xpixels=1280
ypixels=1024
gsfname=gout
ftext='Tahoma/bo'
htext=4
gunit=pct
;
```

If you are new to SAS/GRAPH, the GOPTIONS statement works in much the same manner as an OPTIONS statement. Selections made in a GOPTIONS statement govern both the destination and the appearance of graphical output produced using SAS/GRAPH PROCs. The above GOPTIONS statement instructs SAS/GRAPH to produce a GIF file (the destination) with the FILEREF GOUT. The XPIXELS and YPIXELS options are used to override the default resolution of 800 (XPIXELS) by 600 (YPIXELS), giving more detail to the GIF output files.

You'll also need a FILENAME statement to associate the FILEREF GOUT with a real file name, for example...

```
filename gout 'c:\usamap.gif';
```

The font Tahoma Bold (a TrueType font) is selected for all text with a height of 4. The units of height are percent of the graphics output area, specified with GUNIT=PCT.

If you would like to create PDF output, you can use ODS. The following statements placed prior to and after PROC GMAP will produce a PDF file in landscape mode (named as shown below):

placed prior to PROC GMAP...

```
options orientation=landscape;
options
reset=all
ftext='Helvetica/bo'
htext=4
gunit=pct
rotate=landscape
;

ods listing close;
ods pdf file='c:\usamap.pdf' notoc;
```

placed after PROC GMAP...

```
ods pdf close;
ods listing;
```

The HTEXT and GUNIT options are the same as those used when GIF output was created. PDF output can use postscript fonts, so Helvetica Bold is selected. The NOTOC (no table of contents) option on the ODS PDF statement prevents the addition of a bookmark to the PDF output.

In addition to creating static output, you can create maps that have drill-down capabilities, i.e. maps where map areas and/or the map legend are linked to additional information. The next example uses 2000 census data for the state of New Jersey and the MAPS.COUNTIES map data set to create a map where both New Jersey counties and the map legend are linked to HTML pages.

```
* example 9;
* create a map with drill-down areas;
proc format;
value pop2000
  64285 - 144166 = '64,285-144,166'
  146438 - 350761 = '146,438-350,761'
  423394 - 522541 = '423,394-522,541'
  608975 - 884118 = '608,975-884,118'
;
run;

data njpop;
input county_name & : $supcase20. (pop1990 pop2000) (: comma8.);
* create a link for each county;
county_link = 'ALT="' || trim(county_name) || '" href=county'
              || put(county,z3.) || '.htm';
* create a link for the legend;
if pop2000 le 144166 then legend_link = 'ALT="1ST QUARTILE" href=nj2000_q1.htm';
else
if pop2000 le 350761 then legend_link = 'ALT="2ND QUARTILE" href=nj2000_q2.htm';
else
if pop2000 le 522541 then legend_link = 'ALT="3RD QUARTILE" href=nj2000_q3.htm';
else
              legend_link = 'ALT="4TH QUARTILE" href=nj2000_q4.htm';
datalines;
001 Atlantic County          224,327          252,552
003 Bergen County           825,380          884,118
005 Burlington County       395,066          423,394
007 Camden County           502,824          508,932
009 Cape May County          95,089           102,326
011 Cumberland County        138,053          146,438
013 Essex County             778,206          793,633
015 Gloucester County        230,082          254,673
017 Hudson County            553,099          608,975
```

```

019 Hunterdon County      107,776      121,989
021 Mercer County        325,824      350,761
023 Middlesex County     671,780      750,162
025 Monmouth County      553,124      615,301
027 Morris County        421,353      470,212
029 Ocean County         433,203      510,916
031 Passaic County       453,060      489,049
033 Salem County        65,294        64,285
035 Somerset County     240,279      297,490
037 Sussex County        130,943      144,166
039 Union County         493,819      522,541
041 Warren County        91,607       102,437
;
run;

goptions
reset=all
dev=gif
xpixels=1024
ypixels=768
ftext='Tahoma/bo'
htext=4
gunit=pct
;

pattern1 v=ms c=grayfa; pattern2 v=ms c=grayca;
pattern3 v=ms c=grayaa; pattern4 v=ms c=gray5a;

legend1
mode=share
origin=(5,50)
label=none
value=(h=3 j=r)
shape=bar(3,4)
across=1
;

* select New Jersey (state number 34) from the MAPS.COUNTIES map data set;
proc gproject data=maps.counties out=nj asis;
where state eq 34;
id county;
run;

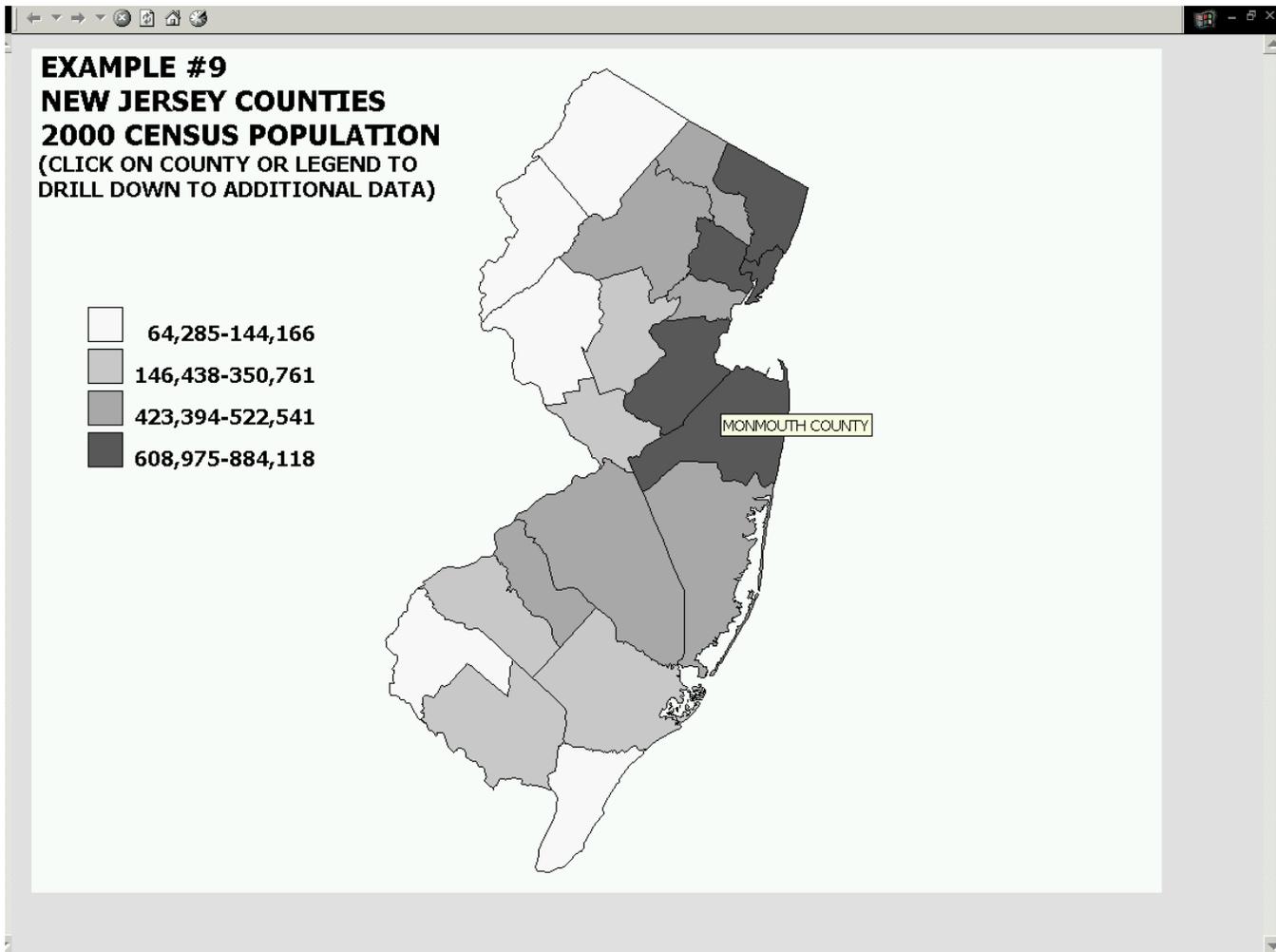
* use ODS to create an HTML file;
* it will link to the GIF file with the actual map;
ods listing close;
ods html path='z:\' (url=none) file='nj2000.htm';

proc gmap
map=nj
data=njpop
;
id county;
choro pop2000 / discrete outline=black legend=legend1
               html=county_link html_legend=legend_link;
format pop2000 pop2000_.;
note
j=1 ' EXAMPLE #9'
j=1 ' NEW JERSEY COUNTIES'
j=1 ' 2000 CENSUS POPULATION'
h=3
j=1 ' (CLICK ON COUNTY OR LEGEND TO'
j=1 ' DRILL DOWN TO ADDITIONAL DATA)'
;
run;
quit;

ods html close;
ods listing;

```

A format is created to group the counties into four groups (quartiles) based on 2000 population. In the data step where the population data are turned into a SAS data set, two new variables are added, COUNTY_LINK and LEGEND_LINK. The value of the variable COUNTY_LINK is formed using the county name as an ALT tag (causing the county name to appear on the screen when the mouse is dragged over a county area) and the county number as a link (used to create the name of the HTML



file to which the county is linked). The GIF device driver is specified in a GOPTIONS statement, but no GSFNAME option is used (as was used in creating the figures shown for examples 1 through 8).

An ODS statement is used to create an HTML file that will have a link (an IMG HTML tag) to the GIF file that contains the map. The URL=NONE option places no path information on the GIF file name in the IMG tag (requiring that both the HTML and GIF files be placed in the same location on the web site). This explanation leaves out a lot of details, but the SAS code shown above for example 9 will create a map with drill-down links. The output shown for example 9 is a screen capture of a map displayed on a web page and it illustrates what happens when the mouse cursor is placed over a county, a county name appears.

It is also possible to create drill-down links from features added to a map using annotation. The next example uses the SAS code from example 8, adding a drill-down link at the area (HSA) labels that are added with annotation. The HSA labels are change to white and black boxes are added behind the labels. Note, as with example 8, you must use the SAS code in example 7 first to create the map data set HSAMAP, used in both examples 8 and 10.

```
* example 10;
* create a map with drill-down annotation;
data labels;
length text $5;
retain
xsys ysys '2'
hsys '3'
position '5'
function 'label'
size 3
style "'Tahoma/bo'"
cbox 'black'
color 'white'
```

```

when      'a'
;
input text x y;
* add a variable named HTML;
html = 'HREF="' || text || '.htm"';
* 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 data set 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 data set;
data hsaproj labels;
set hsaproj;
if function eq 'label' then output labels;
else                          output hsaproj;
run;

goptions
reset=all
dev=gif
xpixels=1024
ypixels=768
ftext='Tahoma/bo'
htext=4
gunit=pct
;

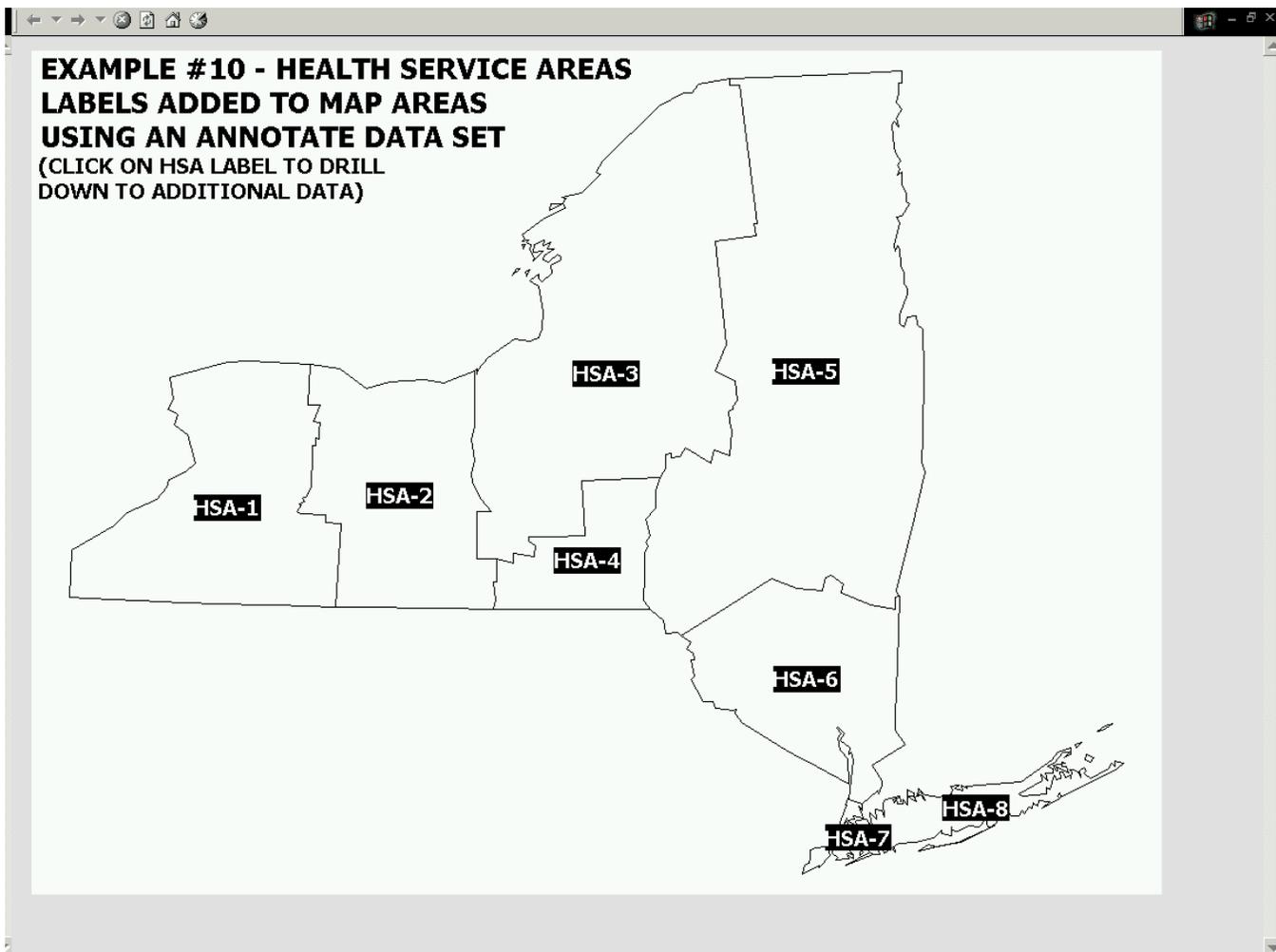
pattern v=e;

* use ODS to create an HTML file;
* it will link to the GIF file with the actual map;
ods listing close;
ods html path='z:\' (url=none) file='nys_hsa.htm';

* 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 #10 - HEALTH SERVICE AREAS'
j=1 ' LABELS ADDED TO MAP AREAS'
j=1 ' USING AN ANNOTATE DATA SET'
h=3
j=1 ' (CLICK ON HSA LABEL TO DRILL'
j=1 ' DOWN TO ADDITIONAL DATA)'
;
run;
quit;

ods html close;
ods listing;

```



Most of the code in example 10 is the same as that in example 8. However, in the data step that create the data set LABELS, the CBOX variable is used to place a black box behind the text added with annotation, and the text color is changed to white. As in example 9, the GIF device driver is specified in a GOPTIONS statement, and ODS is used to create HTML output. The figure produced by example 10 is a screen capture of the web displayed on web page, and the HSA labels are all drill-down links.

The last type of output to be shown is a map with pop-up (not drill-down) information associated with map areas. In example 9, a county-based map of New Jersey was created and each county was linked to another HTML file. The SAS code from example 9 will be modified to create a map that has a pop-up box associated with each county. The pop-up box will contain the county name, population in both 1990 and 2000, and the %-change in population from 1990 to 2000. The GIF device driver and ODS were use din example 9. In example 11, the JAVAMETA device driver and ODS are used.

```
* example 11;
* create a map with pop-up information for map areas;
proc format;
value pop
  64285 - 144166 = '64,285-144,166'
  146438 - 350761 = '146,438-350,761'
  423394 - 522541 = '423,394-522,541'
  608975 - 884118 = '608,975-884,118'
;
run;

data njpop;
input county couname & : $upcase20. (pop1990 pop2000) (: comma8.);
datalines;
001 Atlantic County          224,327          252,552
003 Bergen County           825,380          884,118
005 Burlington County       395,066          423,394
```

```

007 Camden County          502,824          508,932
009 Cape May County        95,089           102,326
011 Cumberland County     138,053          146,438
013 Essex County          778,206          793,633
015 Gloucester County     230,082          254,673
017 Hudson County         553,099          608,975
019 Hunterdon County      107,776          121,989
021 Mercer County         325,824          350,761
023 Middlesex County      671,780          750,162
025 Monmouth County       553,124          615,301
027 Morris County         421,353          470,212
029 Ocean County          433,203          510,916
031 Passaic County        453,060          489,049
033 Salem County         65,294           64,285
035 Somerset County      240,279          297,490
037 Sussex County         130,943          144,166
039 Union County          493,819          522,541
041 Warren County         91,607           102,437
;
run;

* text to put in floating text box (after line with TIP);
data popup;
length popvar $200;
set njpop;
pct = 100*(pop2000 - pop1990) / pop1990;
popvar = 'tip=[| |
        quote(couname) || ' ' ||
        quote('POPULATION') || ' ' ||
        quote('1990 : ' || put(pop1990,comma10.)) || ' ' ||
        quote('2000 : ' || put(pop2000,comma10.)) || ' ' ||
        quote('% CHANGE : ' || put(pct,10.1))
        || '|']';
run;

* use the JAVAMETA device driver;
goptions
reset=all
device=javameta
ftext='HelveticaBold'
xpixels=1024
ypixels=768
htext=5
gunit=pct
;

pattern1 v=ms c=grayfa; pattern2 v=ms c=grayca;
pattern3 v=ms c=grayaa; pattern4 v=ms c=gray5a;

legend1
mode=share
origin=(5,50)
label=none
shape=bar(3,4)
across=1
value=(h=3.5)
;

proc gproject data=maps.counties out=nj asis;
where state eq 34;
id county;
run;

* create an HTML file;
ods listing close;
ods html body="z:\nj_pop.htm"
        codebase="z:\\"
        parameters=("BackgroundColor"="0xFFFFFFFF"
                    "DataTipStyle"="Stick Fixed"
                    "ZoomControlEnabled"="False");

proc gmap
map=nj
data=popup
;
id county;

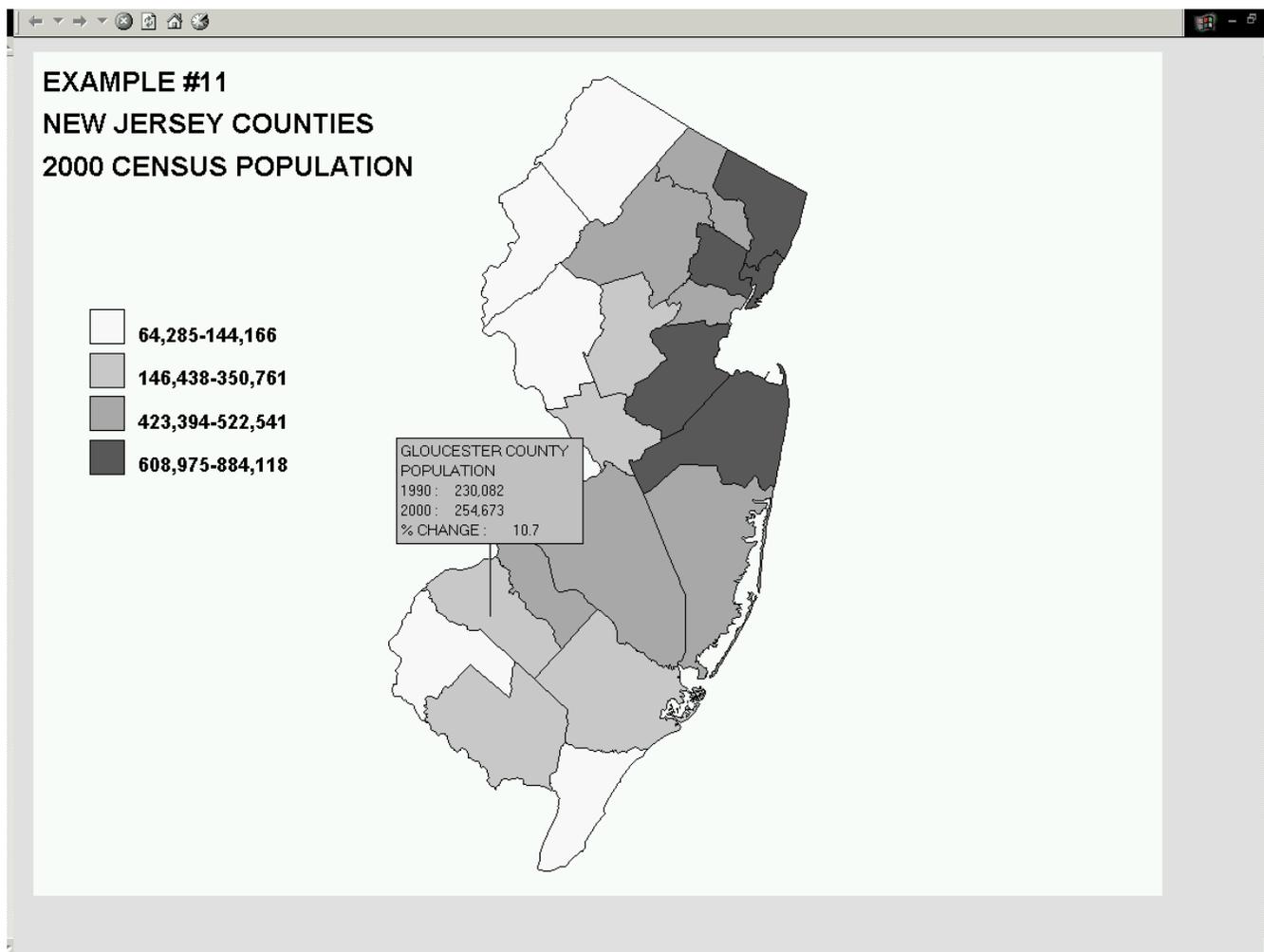
```

```

choro pop2000 / discrete outline=black legend=legend1 html=popvar;
format pop2000 pop.;
note j=1 ' EXAMPLE #11'
j=1 ' NEW JERSEY COUNTIES'
j=1 ' 2000 CENSUS POPULATION';
run;
quit;

ods pdf close;
ods listing;

```



After the census data are read into a SAS data set, another data step is used to create data set POPUP. This new data set contains the census data to be mapped (the 2000 population), plus a new variable named POPVAR. The value of this variable contains the information that is to appear in the pop-box associated with each county. The exact syntax needed to construct this variable is not explained here (it is available on the SAS web site). The JAVAMETA device driver is specified in a GOPTIONS statement. Then ODS is used to create an HTML file. Neither the codebase nor the parameter options will be explained here either. The purpose of this example is to both show you another way of displaying map-based data and to get you interested in using SAS/GRAPH to create maps. The figure produced by example 11 is a screen capture of a web page, showing the pop-up box that appears when you place the mouse cursor on a county. Note, you should also know that the display of the web page shown in example 11 requires the use of the METAVIEW applet that is distributed with SAS/GRAPH software.

RESOURCES

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. Information on SAS-supplied maps can be found at...

<http://support.sas.com/rnd/datavisualization/mapsonline/html>

Examples of PROC GMAP output and the sample code can be found at...

<http://ftp.sas.com/techsup/download/sample/graph/gmap-examples-list.html>

For more information about the METAVIEW applet and the SAS code used in example 11, look at...

<http://support.sas.com/rnd/datavisualization/webgraphs/v8v81Usage/metaviewapplet.htm>

If you want to create maps of other areas, e.g. census tracts or zip-based maps, you will have to create your own map data sets or purchase them. Some vendors sell files already in SAS map data set format. If you want more information on vendors, go to the SAS Institute web site...

<http://support.sas.com/faq/004/FAQ00400.html>

If you are using Version 9 of SAS, a new procedure, PROC MAPIMPORT, allows you to convert files in ESRI shapefile format into SAS/GRAPH map data sets. Information on this procedure is available in V9 help, and also at...

<http://support.sas.com/rnd/datavisualization/papers/SASMapping.pdf>

Many ESRI shapefiles are posted for free download at the US Census Bureau web site...

http://www.census.gov/geo/www/cob/bdy_files.html

When creating maps, it is sometimes difficult to select appropriate colors for map areas. The web site...

<http://www.colorbrewer.com>

is designed to assist you in selecting colors for maps.

Finally, there is a book, *Maps Made Easy Using SAS...*

<http://www.sas.com/apps/pubscat/bookdetails.jsp?catid=1&pc=57495>

that expands on the material covered in this paper and also includes a chapter on creating maps for posting on the web.

CONCLUSION

PROC GMAP either by itself or in combination with several other SAS/GRAPH procedures allows you to create a number of different types of maps. Though it does not have all the features of a true geographic information system, there are enough features in PROC GMAP, especially when used in conjunction with annotate data sets, to produce both publication quality choropleth maps and maps with special features (drill-downs and pop-ups) for web posting.

REFERENCES

SAS Institute Inc., *SAS/GRAPH Software Usage, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1991

CONTACT INFORMATION

The author can be contacted using [e-mail...msz03@albany.edu](mailto:msz03@albany.edu)

by phone...518-402-6479

or by regular mail...

Mike Zdeb
U@Albany School of Public Health
1 University Place
Rensselaer, NY 12144-3456

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.