

ZIP Code 411:

A Well-Kept SAS® Secret

Louise Hadden, Abt Associates Inc., Cambridge, MA

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

ABSTRACT

SAS provides many useful files in its SASHELP directory. One of the less well-known but incredibly useful files is the SASHELP.ZIPCODE file that ships with SAS V9 and is regularly updated.

This paper explores current and potential uses of the SASHELP.ZIPCODE file data. Data and programming resources provided by SAS and other sources will be highlighted, with a walk-through of several programming projects recently conducted by the authors.

WHAT, WHERE, AND HOW?

WHAT IS THE SASHELP.ZIPCODE FILE?

The SASHELP.ZIPCODE file is a file containing ZIPCODE level information for the United States including ZIPCODE centroids (x, y coordinates), Area Codes, city names, FIPS codes, and more. The file is indexed on ZIPCODE to facilitate processing, and is updated on a regular basis by SAS. It is provided in transport format so that installations with dissimilar releases of SAS and operating systems can make use of the file. A content listing of the 2005 Q2 file is provided below. SAS has since provided several updates to the file as of March 2006, but the contents remain the same.



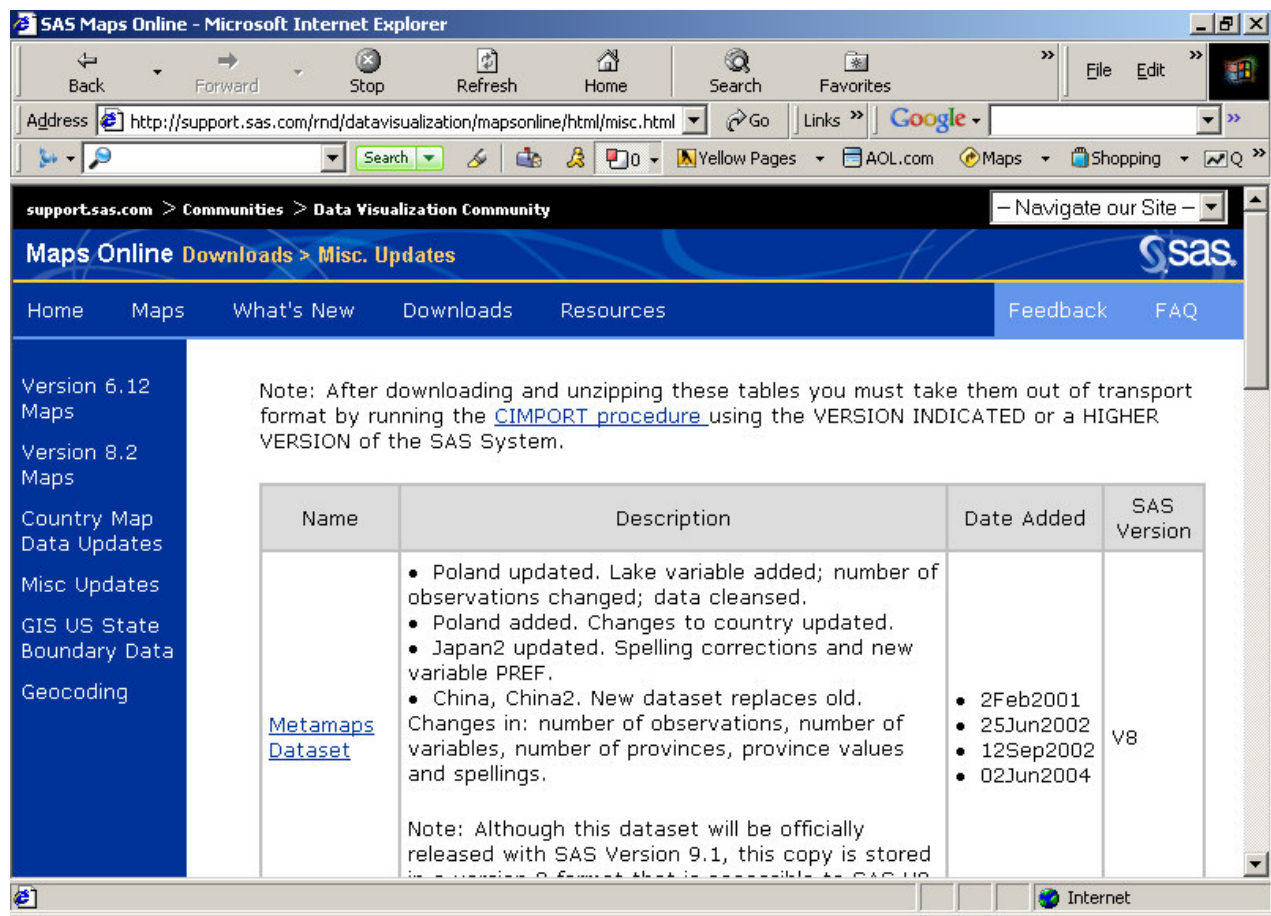
Microsoft Excel - sashelp_zipcode.xls					
File Edit View Insert Format Tools Data Window Help					
F8					
	B	C	D	E	F
29	Alphabetic List of Variables and Attributes				
30	Variable	Type	Len	Format	Label
31	AREACODE	Num	8		Area Code for ZIP Code. None for APO/FPO
32	CITY	Char	35		Name of city/org
33	COUNTY	Num	8		FIPS county code. Blank for APO/FPO
34	COUNTYNM	Char	25		Name of county/parish. No county for APO/FPO
35	DST	Char	1		ZIP Code obeys Daylight Savings: Y-Yes N-No
36	GMTOFFSET	Num	8		Diff (hrs) between GMT and time zone for ZIP Code
37	MSA	Num	8		Metro Service Area code by common pop; no MSA for rural
38	STATE	Num	8		Two-digit number (FIPS code) for state/territory
39	STATECODE	Char	2		Two-letter abbrev. for state name.
40	STATENAME	Char	25		Full name of state/territory
41	TIMEZONE	Char	9		Time Zone for ZIP Code. None for APO/FPO
42	X	Num	8	11.6	Longitude (degrees) of the center (centroid) of ZIP Code. 0.0 for APO/FPO
43	Y	Num	8	11.6	Latitude (degrees) of the center (centroid) of ZIP Code. 0.0 for APO/FPO
44	ZIP	Num	8	Z5.	The 5-digit ZIP Code
45	ZIP_CLASS	Char	1		ZIP Code Classification: M= APO/FPO P=PO Box U=Unique zip used for large orgs/businesses/bldgs Blank=Standard/non-unique
46	sashelp_zipcode				
Ready					

As you can see, the variables offer many potential uses.

WHERE CAN I FIND THE SASHELP.ZIPCODE FILE?

The SASHELP.ZIPCODE FILE is shipped with current installations of SAS and updates can also be located in SAS MAPS ONLINE downloads page. Updates, which occur regularly, must be obtained from SAS MAPS ONLINE.

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



Don't be intimidated by the fact that the data set is maintained by the SAS MAPS ONLINE staff! The file can be and is used by SAS for more than mapping. You should check the MAPS ONLINE site regularly for updates to the SASHELP.ZIPCODE file, and for much, much more.

HOW CAN THE SASHELP.ZIPCODE FILE BE USEFUL TO ME?

The SASHELP.ZIPCODE FILE can be used to enable SAS-written functions, to develop user-defined formats, calculate distances between zipcodes and to annotate SAS/GRAPH maps with information in the file. It is important to note if you do not have a current ZIPCODE file in your SASHELP subdirectory, these uses are not possible!

Current SAS-written functions utilizing the SASHELP.ZIPCODE file include ZIPCITY, ZIPSTATE, ZIPNAME, ZIPNAMEL and ZIPFIPS. For example, the ZIPCITY function takes ZIPCODE as its argument and returns a proper case city name and two character postal state abbreviation.

```

ZIPCITY('02138') returns 'Cambridge, MA'
ZIPSTATE('02138') returns 'MA'
ZIPNAME('02138') returns 'MASSACHUSETTS'
ZIPNAMEL('02138') returns 'Massachusetts'
ZIPFIPS('02138') returns 25 (FIPS state code for Massachusetts)

```

Potential user-written formats include: conversion of zipcode to MSA code (and by extension, urban/rural status); conversion of zipcode to FIPS county code; conversion of zipcode to county name; conversion of zipcode to latitude and longitude; conversion of zipcode to telephone area codes; conversion of zipcode to daylight savings time flag; and conversion of zipcode to time zones.

Example:

```

Data zip2msa (rename=(zipcode=start msacode=label));
  Set sashelp.zipcode end=last;
  Msacode=put(msa,z4.);
  Zipcode=put(zip,z5.);
  Retain fmtname '$zip2msa' type 'c' hlo=' ';
  If last then hlo='h';
Run;

Proc format library=library cntlin=zip2msa;
Run;

Data x;
  Length msacode $ 4;
  Zipcode='02138';
  Msacode=put(zipcode,$zip2msa.);
Run;

Proc print data=x;
Run;

```

Result:

Obs	msacode	Zipcode
1	1120	02138

The SASHELP.ZIPCODE file can be used to calculate distances between zipcodes. support.sas.com has provided an example of this use at <http://support.sas.com/techsup/unotes/SN/005/005325.html>. A macro incorporating the distance calculation and simple example using the macro follow below.

```

/* calculate the distance between zipcode centroids */
%macro geodist(lat1,long1,lat2,long2);
%let pi180=0.0174532925199433;
7921.6623*arsin(sqrt((sin((&pi180*&lat2-&pi180*&lat1)/2))**2+
cos(&pi180*&lat1)*cos(&pi180*&lat2)*(sin((&pi180*&long2-
&pi180*&long1)/2))**2));
%mend;

data temp1 (keep=long1 lat1 city1);
  set sashelp.zipcode (where=(zip in(02138)));
  rename x=long1 y=lat1 city=city1;
run;

```

```

data temp2 (keep=long2 lat2 city2);
  set sashelp.zipcode (where=(zip in(01776)));
  rename x=long2 y=lat2 city=city2;
run;

data temp;
  merge temp1 temp2;
  zipdist=%geodist(lat1,long1,lat2,long2);
run;

proc print data=temp;
run;

```

Result:

Obs	lat1	long1	city1	lat2	long2	city2	zipdist
1	42.377465	-71.131249	Cambridge	42.382955	-71.427696	Sudbury	15.1429

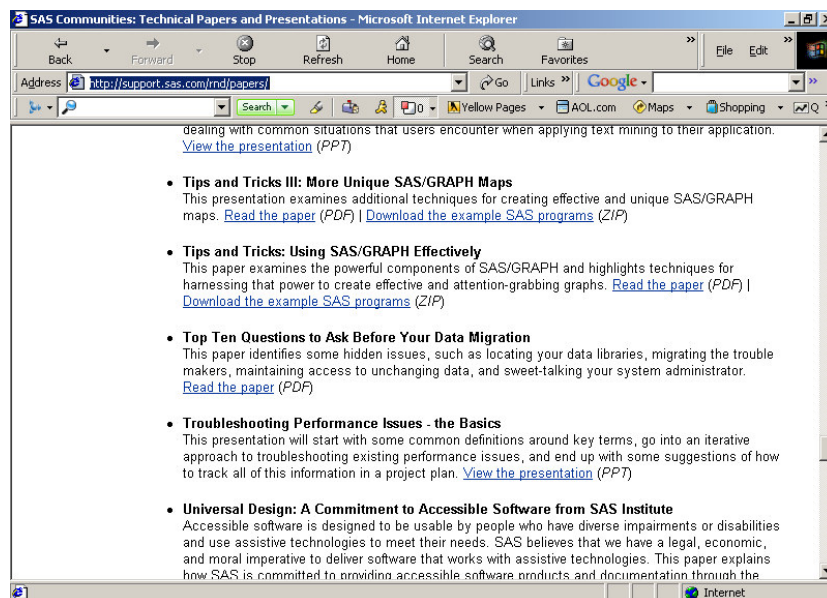
The distance between the zip code centroid of the 02138 zipcode in Cambridge, MA and the zip code centroid of Sudbury, MA is 15.1429 miles.

Thus far the uses of SASHELP.ZIPCODE presented have NOT involved SAS/GRAPH. However, the SASHELP.ZIPCODE file is eminently useful both off and on the map. Several different current uses of the SASHELP.ZIPCODE file in conjunction with SAS/GRAPH will be presented below in glorious, living color! In the process, many of the programming tools that SAS puts at our disposal will be explored, as well as some non-SAS data sources.

USES OF SASHELP.ZIPCODE

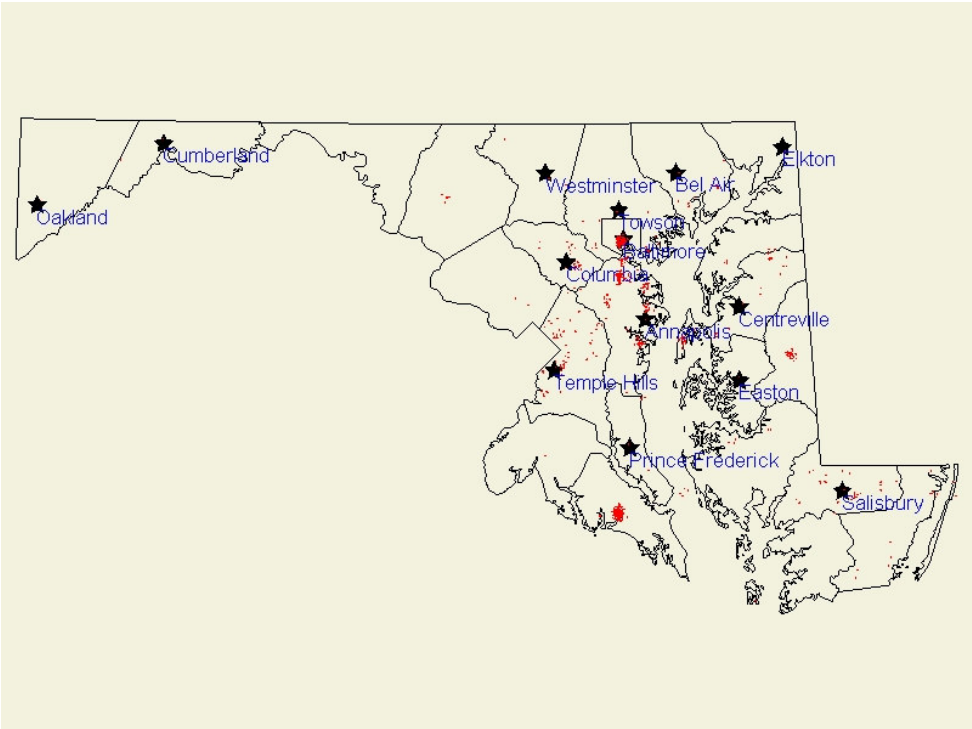
WHAT DOES SAS DO WITH SASHELP.ZIPCODE?

SAS updates the SASHELP.ZIPCODE regularly, and writes functions available to SAS users based on the file. SAS also writes papers and provides sample code utilizing the SASHELP.ZIPCODE file. Sample code is easy to find on the support.sas.com website. Support.sas.com\rnd\papers gives you a searchable list of papers, and support.sas.com\samples gives you a searchable index with menus. This is a great starting point if you are interested in doing ANYTHING with SAS, not just using the SASHELP.ZIPCODE file.



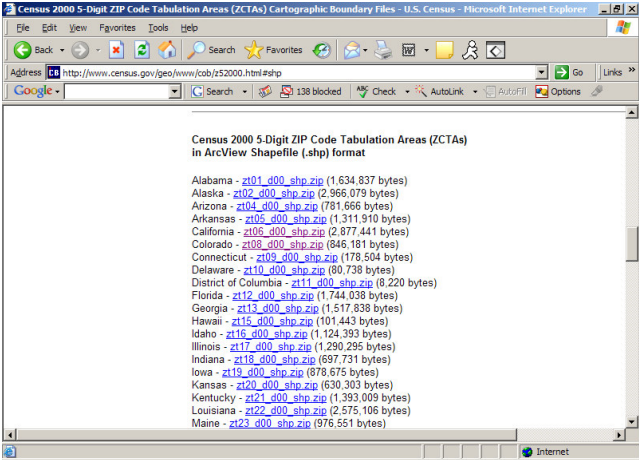
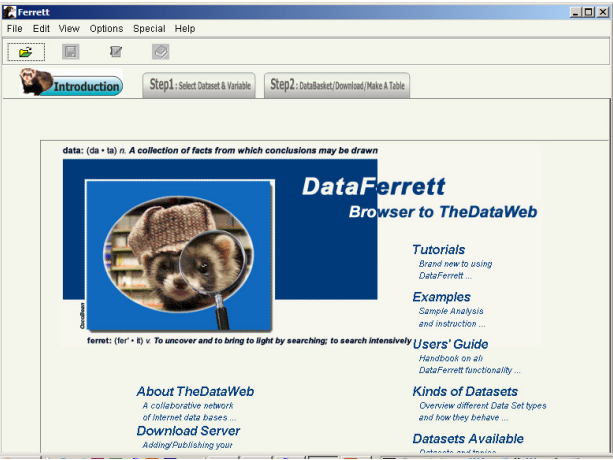


Counseling Locations in Maryland



SAMPLE 2

An author's company is conducting a study of beneficiaries in Medicare's Part D program. In order to sample participants for focus groups in selected communities, the project director wished to graphically present poverty rates on zip code level maps for given communities. The project involved obtaining FREE zip code level poverty data from Census 2000 data via Data Ferrett, another well-kept secret (<http://dataferrett.census.gov/>). FREE zip code ESRI shapefiles were obtained from the Bureau of the Census Cartographic Boundary Files web page (<http://www.census.gov/geo/www/cog/z52000.html>). SAS's PROC MAPIMPORT was used to import the zip code boundary maps for selected communities. Poverty data from Data Ferrett were collected for the same communities and combined with the SASHELP.ZIPCODE file to obtain zip code centroids. The poverty data were used to annotate zip codes in the various communities with poverty rates. The project team was then able to select zip codes with predominantly low incomes close to the focus group facility for sampling. Code snippets are presented below and the full code is available from Louise Hadden (louise_hadden@abtassoc.com).



PROC MAPIMPORT SAMPLE

```

proc mapimport out=vazips datafile='zt51_d00.shp';
run;

proc sort data=vazips;
    by zcta;
run;

data dd.vazips;
length labelzip $ 22;
merge vazips (in=a) site5 (in=b);
by zcta;
if b;
labelzip=cats(zcta, '-', put(pctpov, percent7.1));
if zcta='23462' then labelzip=cats(zcta, '-FG Fac-', put(pctpov, percent7.1));
run;

```

ANNOTATE DATA SET BUILDING SAMPLE

```

proc gproject data=combine out=vazips;
    id zip;
run;

data newport annozip2;
    set vazips (where=(zip not in('23310','23306','23417') and
        cityname eq 'Newport News'));
    if annoflag=1 then output annozip2;
    if annoflag=0 then output newport;
run;

data labels_black2;
length function color $ 8 text $ 22;
retain function 'label' xsys ysys '2' hsys '3' when 'a';
set annozip2;
text=' ||labelzip;
style="'Arial/bo"; cbox='white'; cborder='black';
color='black'; size=2.50; position='B';
if zip='23232' then cbox='pink'; /* highlight focus group facility */
run;

```

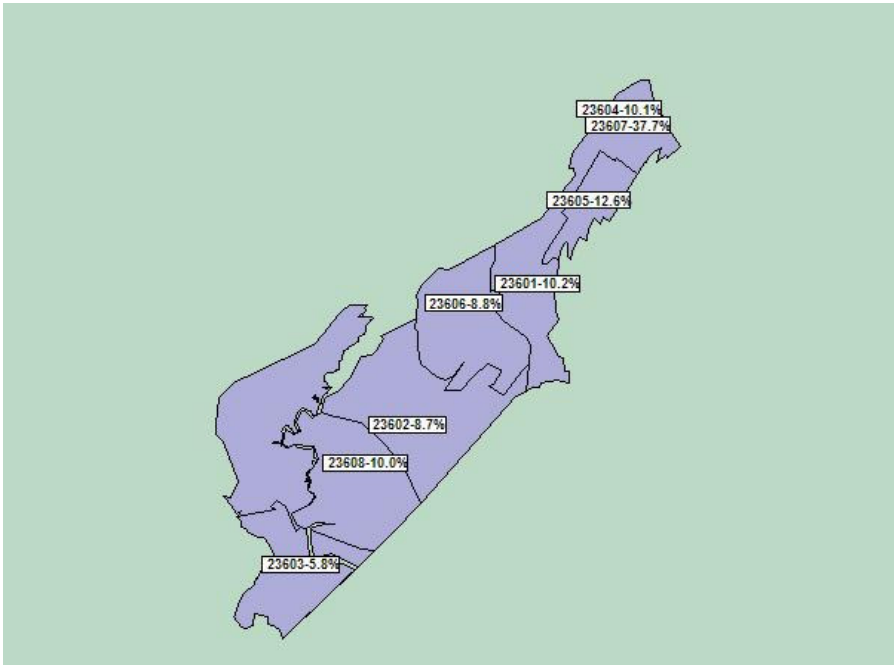
MAPPING SAMPLE

```

goptions xpixels=600 ypixels=450 device=jpeg ftext="Arial/bo" cback=vpag;
ods listing close;
ods html path=odsout body='newport.htm';
pattern v=s color=vpab r=100;
proc gmap data=newport map=newport;
    id zip;
    choro zip / discrete
        nolegend coutline=black
        annotate=labels_black2 name="newport";
run;
quit;
ods html close;
ods listing;

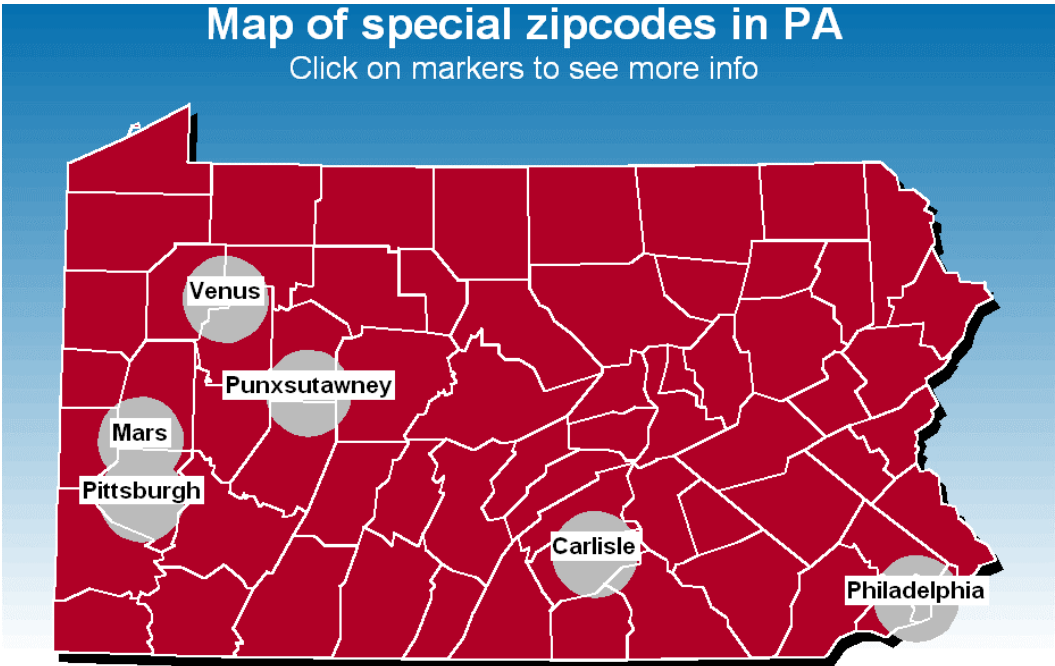
```

Focus Group Zip Codes in Newport News, VA



SAMPLE 3

Mike Zdeb and Robert Allison presented this sample at a Hands-On Workshop at SUGI 30. PROC GMAP is used again in combination with the Annotate facility to create a map of Pennsylvania with: fake transparent effects; 'shadows' behind the map; a gradient shaded background. The example starts with a fairly basic map, a county map, with Annotate markers and labels, based on longitude/latitude coordinates from SASHELP.ZIPCODE. Then, the all of the 'extras' listed above are added. Only SAS code fragments are shown in this section, but the SAS code used to produce the sample map (and more!) is available for download at <http://support.sas.com/usergroups/sugi/sugi30/handson/137-30.zip>.



The city markers are put in front of the map, but made to appear 'transparent' so that you can still see the county outlines behind the markers (we do not want the markers to obscure the county outlines). To accomplish this illusion, the map is drawn and markers are added using the Annotate facility. A second copy of the county outlines is added on top of the markers by first creating an Annotate data set from the county outlines and adding the outline to the map with the Annotate facility. Various portions of the annotation can be layered, whereas without using Annotate the map and the border outlines must appear at a single depth.

The county outlines are changed to an Annotate data set using a data step. The process is fairly simple. The data are processed by county and for each county the first observation is associated with an Annotate 'move' to the first X-Y location, while subsequent observations within a county are associated with an Annotate 'draw' to remaining X-Y points on the county border. Here is a summarized version of the code used to produce the Annotated county outlines shown in this example.

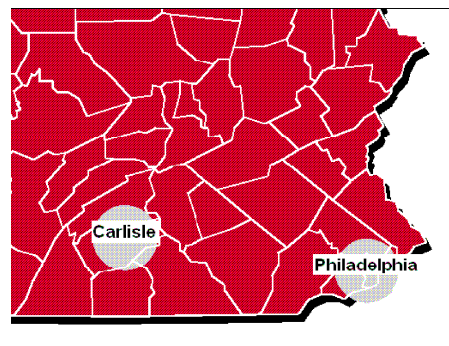
```
data border_anno;
  set my_map; by county segment;
  if first.Segment then do;
    FUNCTION = 'Move'; FX = X; FY = Y; end;
  else if FUNCTION ^= ' ' then do;
    if X = . then do;
      X = FX; Y = FY; output; FUNCTION = ' ' ; end;
    else FUNCTION = 'Draw';
  end;
  if FUNCTION ^= ' ' then do;
    output;
  end;
  if last.Segment then do;
    X = FX; Y = FY; output; end;
end;
run;
```



Projecting a shadow behind a map helps to give the illusion that the map 'leaps' off the page, helping to catch the attention of a person viewing the map. A shadow also helps to separate the borders of the map from the background. There is no option to add a shadow in PROC GMAP, but once again, the Annotate facility can be used. PROC GREMOVE is used to create a new map with all the internal boundaries removed. The new map contains only the X-Y coordinates for the state border and it is converted into an Annotate polygon using the 'poly' and 'polycont' Annotate functions. This is done since the entire area of this polygon will be filled with a color, which cannot be done with 'move' and 'draw'. A dark color such as black usually works well for the shadow. The Annotated polygon is drawn before (behind) the map by using the when='b' Annotate option. A slight offset is added to the X-Y coordinates to produce the shadow shown on the right.

Here is a summarized version of the code.

```
* remove internal map boundaries leaving only the state border;
proc gremove data=anno_shadow out=anno_shadow;
  by county;
  id county;
run;
* convert the map data set to an Annotate data set;
data anno_shadow;
  set anno_shadow;
  by county segment;
  color='black';
  when='b';
  style='msolid';
  if first.segment then do;
    Function = 'poly'; fx = x; fy = y;
  end;
  else if function ^= ' ' then do;
    if x = . then do;
      X = fx; y = fy; output; function = ' ' ;
    end;
  end;
```



```

else function = 'polycont';
end;
if function ^= ' ' then do;
    output;
    if last.segment then do;
        X = fx; y = fy; output;
    end;
end;
run;
* move the coordinates 'slightly' to create a shadow;
data anno_shadow;
    set anno_shadow;
    x=x+.0004; y=y-.0006;
run;

```

The last step is to add a gradient-shaded background. Just as there is no 'shadow=yes' option, there is no "gradient=yes" option in PROC GMAP. However, a gradient-shaded background can be produced using the Annotate facility. The gradient background is simulated by using a series of wide-and-short Annotate bars of slightly differing color. The xsys and ysys='3' coordinate system is used so the X-Y values within the Annotate data set can be specified as a percentage of the entire graphics output area, from zero to 100. The gradient starts at Y=100 and progresses to 0 in -2 increments. This technique draws fifty wide-and-short bars that start at the top of the screen and progress to the bottom of the screen. An Annotated bar is created by an Annotate 'move' to the bottom-left corner of the bar, and then an Annotate 'bar' using the coordinates of the top-right corner of the bar. The gradient colors are specified using RGB (red/green/blue) color codes at the beginning of the loop, incrementing the colors each time through the loop. As the colors increment, they become 'lighter' and eventually fade out to white. The RGB values are kept separate for the mathematical calculations, and at the end of each pass through the loop they are converted from decimal to hex and concatenated. Putting a 'cx' in front of the concatenated hex codes provides a color name that SAS/GRAPH understands (such as 'cx0570B0'). Here is a summarized version of the code.

```

data anno_gradient;
    retain xsys '3' ysys '3' when 'b' style 'solid';
    orig_red = input('05',hex2.);
    orig_green = input('70',hex2.);
    orig_blue = input('B0',hex2.);
    do i=100 to 0 by -2;
        count+1;
        function='move'; x=0; y=i; output;
        function='bar'; x=100; y=i-2;
        percent=(count-1)/(45);
        red=orig_red + ( (256-orig_red) * percent );
        if red > 255 then red=255;
        green=orig_green + ( (256-orig_green) * percent );
        if green > 255 then green=255;
        blue=orig_blue + ( (256-orig_blue) * percent );
        if blue > 255 then blue=255;
        rgb=put (red,hex2.) || put (green,hex2.) || put (blue,hex2.);
        color="cx" || rgb;
        output;
    end;
run;

```

While this sample truly "stretches the bounds" of SAS/GRAPH® software, it also serves to show some of the amazing things that can be done with the help of the SASHELP.ZIPCODE file.

CONCLUSION

Like other utility files in the SASHELP subdirectory supplied by SAS, the SASHELP.ZIPCODE file is well worth exploring. The SASHELP.ZIPCODE file has utility in many different SAS arenas such as the data step, reporting, distance calculations and of course, mapping. The use of the SASHELP.ZIPCODE file in conjunction with other little known resources such as Data Ferrett and the US Census Cartographic Maps greatly enhances the SAS programmer's toolbox.

REFERENCES

<http://support.sas.com/documentation/onlinedoc/index.html>

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

<http://support.sas.com/rnd/papers>

<http://support.sas.com/samples>

<http://www.census.gov/geo/www/cob/z52000.html>

<http://dataferrett.census.gov/>

<http://support.sas.com/usergroups/sugi/sugi30/handson/137-30.zip>

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

Cody, Ron. 2004. *SAS Functions by Example*. Cary, NC: SAS Institute Inc.

Zdeb, Mike. 2003. *Maps Made Easy Using SAS*. Cary, NC: SAS Institute Inc.

ACKNOWLEDGMENTS

Carolyn Robinson of Abt Associates Inc., who provided invaluable assistance with calculating distances between zip code centroids.

Darrell Massengill and Robert Allison of SAS who work tirelessly to improve and facilitate the use of SAS/GRAPH® and mapping with SAS.

Liz Simon of SAS who is the caretaker of the SASMAPS ONLINE website and patiently answers many questions.



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 registered trademarks or trademarks of their respective companies.

CONTACT INFORMATION

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

Louise Hadden

Email: louise_hadden@abtassoc.com

Mike Zdeb

Email: msz03@albany.edu



Sample code for the published examples is available on the Proceedings CD or from the authors.



KEYWORDS

SAS®; SASHELP.ZIPCODE; PROC MAPIMPORT; PROC GMAP; ANNOTATE; FUNCTIONS;
PROC FORMAT