

Where in the World is Bacon Level?

Keith J. Brown

University of North Carolina - General Administration
Chapel Hill, NC

Abstract

While SAS/Graph's mapping facilities provide a convenient way to marry data to geographical representation at the national, state, or county level, their IDNAME datasets provide locations of very few cities or towns. By using readily available electronic data, you can create a program to map any given county in the United States or any county containing a a locality with a given name, and provide basic information about the county in footnotes. If the data are stored in the appropriate form, you can even use the FTP options on the FILENAME statement to access the data from the remote site, thereby reducing the amount of disk space needed at your own site.

Finding the Data

The U.S. Census Bureau has a gazetteer of all counties in the United States available on the World-Wide Web at [ftp.census.gov/geo/www/gazetteer/places.html](ftp://ftp.census.gov/geo/www/gazetteer/places.html) or for anonymous FTP from www.census.gov/pub/tiger/tms/gazetteer. This file is available in zipped or text format; a codebook is available at the web address. In addition to FIPS codes and names, 'counties.txt' contains the population and number of housing units in 1990, land and water area, and latitude and longitude. This file provides the information for footnotes for our maps.

The primary database for this program comes from the U.S. Geological Survey's Geographic Name Information Service (GNIS). This organization has databases for each state and territory which include practically every named place in that area. One file, called 'populated-places.gz', contains over 160,000 places, their locations, and, for some places, their population in 1990. These files, available from the USGS server at www-nmd.usgs.gov/pub/gnis, will need to be gunzipped before using. The README file provides a description of the file layout. While

the USGS labels this file "populated places", a better label would be "named places", since only about one-sixth of the places in this file big enough to counted by the Census Bureau; most are little more than crossroads. The compilation of this file is an impressive feat.

Accessing the Data

In preliminary versions of this program, each dataset had to be moved via FTP from the Census Bureau or Geological Survey to the platform where the program was to be run, and then uncompressed. The total amount of online available disk space needed for the two files is about 25 megabytes. While this is not an enormous amount of space, it can be reduced somewhat by using an exciting new feature in SAS: the FTP options of the FILENAME statement. These options allow a program running on a platform with access to the Internet to dynamically read or write files on another Internet-accessible machine, by specifying the IP number or name, a userid and password, and a path and filename on the remote machine.

```
filename in1 ftp "counties.txt"
             cd="pub/tiger/tms/gazetteer/"
             host="www.census.gov"
             user="anonymous"
             pass="user@site.domain";
```

This feature works well with the unzipped county file from the Census Bureau. The U.S. Geological Survey normally does not store data in uncompressed form, but agreed to do so temporarily while the program for this paper was being tested. Unfortunately, the combination of the size of the dataset to be read and the slowness of the USGS's temporary data server made it necessary to move this file to a local platform.

Where Do You Want to Map?

This program accepts as input from a CARDS dataset three pieces of information -- a state abbreviation (two-letter U.S. Postal Service abbreviations), a place type ('C' for county, 'P' for place), and a place name (up to 66 characters long); the latter two pieces of information are required. If the state is not specified, the program will search for any county or locality in the United States with the appropriate name. For this particular invocation of the program, we want to map the county of our host city and any county in which we find a locality named either Bacon Level or named for one of those mythically perfect places, Utopia and Camelot.

From the CARDS dataset, we create three datasets that will be used to subset the data from the Census Bureau and Geological Survey -- one for county names, one for place names, and one for state abbreviations. We create temporary SAS datasets from the Census Bureau and USGS files, and merge each with the dataset containing state abbreviations, to restrict our search to only the appropriate states.

We next search the USGS places file for localities or counties with names matching those in our selection dataset, and keep only the state abbreviation and county name where we find a match. We then subset the counties datasets to keep only those counties, and subset the places dataset to keep all localities in the relevant counties.

At this point, we venture into the world of SAS macros. For many programmers, the idea of writing or even using a macro is enough to send them screaming from the room; indeed, this was the author's usual reaction up until a relatively short time ago. In this case, however, the 'macrotized' section of the code is fairly small and uncomplicated; we simply want to repeat a section of SAS code once for each county to be mapped.

The first part of our macro builds titles and footnotes from information in our counties dataset. In addition to mapping each county, our output will also include information on the 1990 population and number of households, number of square miles of land and water, and state and county FIPS codes. We print out basic information about the places within that county that will appear on our map, and then proceed to

build the annotate dataset we will need for our map.

Annotation is another feature of SAS that has often aroused feelings of panic in users who have newly ventured into the realm of graphics. By using the annotate macro '%label', we are able to produce painlessly the data needed to annotate our map.

The parameters needed for the %label macro are, in order:

x -- longitude in radians;
y -- latitude in radians;
text -- the text to be printed;
color -- color of the text;
angle -- orientation of the label;
rotate -- orientation of each letter;
size -- font size;
style -- font type; and
position -- position of the text around the point.

The code segment below illustrates the use of this macro. The first time the macro is called, it produces the output needed to write the name of the locality below its location, as determined by the x and y coordinates. The second invocation of the macro puts an asterisk at the locality's position. In some cases where there are a number of localities in a small area, it may be necessary to move some names by changing their positions to accommodate the others.

```
data anno;
set usgs(drop=state where=(obsno=&i));
length function style color $ 8
      position $ 1 text $ 40;
retain xsys ysys '2' hsys '1'
      state 100;
state+1; when='a';
%label(x,y,name,black,0,0,1,swiss,E);
state+1; when='a';
%label(x,y,'T',black,0,0,1,marker,5);
run;
```

After we have created the annodata dataset, we use proc summary and a data step to determine whether the map should be in portrait or landscape orientation, and create a macro variable to pass to the appropriated GOPTIONS statement. We obtain the data needed to draw the county outline from SAS's library of

map datasets and combine it with the annotate data. PROC GPROJECT is used to ensure that all points are represented by the proper map projection techniques. After splitting the projected data back into mapping and annotating datasets, we're ready to pass everything to PROC GMAP to produce our output. At this point, we end the code for our macro, and include a statement to execute it.

When the program is submitted, the Census Bureau's data is read from the remote site and combined with the USGS's data resident on the local platform. The macro is executed, and the resulting maps are written out to a postscript file to be printed as needed. The only part of the program that isn't fully automated is resolving the problem of too many names in too small an area; to produce a map with no overlapping names usually requires several runs of the program.

The results of this particular version of the program are rather interesting. Eight places in the United States lay claim to the name "Utopia", while twenty-seven localities vie for the title of that most genial spot, "Camelot", including five in Georgia, five in Tennessee, and four in North Carolina. Surprisingly, or perhaps not so surprisingly, there is only one locality that answers the question posed in the title of this paper. Randolph County, Alabama, my home county, lays claim to the one and only Bacon Level.

References

In addition to the Internet sites mentioned above, the following are invaluable resources for annotating maps in SAS

SAS/GRAPH® Software: Reference, Version 6, First Edition, Volume 1, Cary, NC: SAS Institute, Inc., 1990, pp 465-596.

SAS/GRAPH® Software: Reference, Version 6, First Edition, Volume 2, Cary, NC: SAS Institute, Inc., 1990, pp 1001-1062 & 1147-1168.

SAS® Technical Report P-196, *SAS/GRAPH® Software: Map Data Sets, Release 6.06*, Cary, NC: SAS Institute, Inc., 1990.

For Further Contact

The author can be reached for comments or questions at:

Keith J. Brown
P.O. Box 2688
UNC-General Administration
Chapel Hill, NC 27515-2688

(919) 962-1000 (Telephone)
(919) 962-4316 (Fax)
Keith_Brown@unc.edu (E-mail)

SAS and SAS/GRAPH are registered trademarks of SAS Institute Inc. in the USA and other countries. ® Indicates USA registration.

```
options nofmterr errors=1 nobyline nodate
nonumber;
```

```
filename in1 ftp "counties.txt"
    cd="pub/tiger/tms/gazetteer/"
    host="www.census.gov"
    user="anonymous"
    pass="user@site.domain";
```

```
filename in2
    "data/usgs.popplace.data";
```

```
goptions gunit=pct border
    ftext=swiss gaccess=gsasfile
    gsfmode=replace device=clj
    ctext=black cback=white graphrc
    interpol=join;
```

```
%annomac;
```

```
data cnty(rename=(name=cntyname))
    states(keep=stabbr)
    place;
input @1 stabbr $2.
    @5 type $1.
    @7 name $66.;
keep stabbr name;
name=trim(name);
if name='' or type='' then delete;
if stabbr > '' then do;
if type='C' then output cnty;
    else if type='P' then
        output place;
    output states;
end;
```

```
else if stabbr=' ' then do
    fips=1 to 100;
stabbr=fipstate(fips);
if type='C' and
    stabbr not in('**','') then
    output cnty;
else if type='P' and
    stabbr not in('**','') then
    output place;
if stabbr not in('**','') then
    output states;
end;
```

```
cards;
```

```
FL P Jacksonville
    P Bacon Level
    P Utopia
    P Camelot
```

```
;
run;
```

```
proc sort data=cnty nodupkey;
    by stabbr cntyname;
```

```
proc sort data=place nodupkey;
    by stabbr name;
```

```
proc sort data=states nodupkey;
    by stabbr;
```

```
data counties;
```

```
infile in1;
```

```
input @1 state 2.
    @6 county 3.
    @10 cntyname $66.
    @77 stabbr $2.
    @80 pop90 9.
    @90 house90 9.
    @100 landarea 10.3
    @111 watrarea 10.3
    @122 lat 9.6
    @132 long 10.6
```

```
;
* Convert West longitudes to postive *;
long=0-long;
* Convert areas to square miles *;
landarea=round(landarea*0.3861);
watrarea=round(watrarea*0.3861);
i=index(cntyname,'County');
if i then cntyname=trim(substr(cntyname,1,i-1));
```

```
keep state county cntyname stabbr pop90
    house90 landarea watrarea;
```

```
run;
```

```
proc sort data=counties;
    by stabbr cntyname;
```

```
run;
```

```
data counties;
```

```
merge counties(in=in1) states(in=in2);
    by stabbr;
```

```
if in1 and in2;
```

```
run;
```

```
data usgs;
```

```
infile in2 missover;
```

```
input @1 name $49.
    @50 feature $5.
    @55 stabbr $4.
    @59 cntyname $20.
    @79 state 4.
    @83 county 6.
    @89 coord $19.
    @108 fedstat $10.
    @118 elev 8.
    @126 topmap $23.
    @149 placepop 9.
```

```
;
```

```
if index(coord,'UNKNOWN') or
    index(coord,'???') then delete;
```

```
length type $ 30;
```

```
i=index(name,'(');
```

```
if index(name,'(') then do;
```

```
    type=substr(name,i);
```

```
    name=substr(name,1,i-1);
```

```
end;
```

```
lat1=input(substr(coord,1,2),2.);
```

```
lat2=input(substr(coord,3,2),2.);
```

```
lat3=input(substr(coord,5,2),2.);
```

```
long1=input(substr(coord,10,3),3.);
```

```
long2=input(substr(coord,13,2),2.);
```

```
long3=input(substr(coord,15,2),2.);
```

```
lat=sum(of lat1,lat2/60,lat3/6000);
```

```
long=sum(of long1,long2/60,long3/6000);
```

```

x=long*acos(-1)/180;
y=lat*acos(-1)/180;
if type ne '' then delete;
i=index(name,'Subdivision');
if i then name=substr(name,1,i-1);
keep stabbr cntyname name lat long
    elev x y state county placepop;
run;

proc sort data=usgs;
    by stabbr name;
run;

data usgs;
merge usgs(in=in1) states(in=in2);
    by stabbr;
if in1 and in2;
run;

data usgs1;
merge usgs(in=in1) place(in=in2);
    by stabbr name;
if in1 and in2;
keep stabbr cntyname;
run;

proc sort data=usgs;
    by stabbr cntyname;

data usgs2;
merge usgs(in=in1) cnty(in=in2);
    by stabbr cntyname;
if in1 and in2;
keep stabbr cntyname;
run;

data places;
set usgs1 usgs2;
run;

proc sort data=places nodupkey;
    by stabbr cntyname;
run;

data places;
set places end=last;
obsno=_n_;
if last then
call symput('max',obsno);
run;

data counties;
merge counties(in=in1) places(in=in2);
    by stabbr cntyname;
if in1 and in2;
run;

data usgs;
merge usgs(in=in1) places(in=in2);
    by stabbr cntyname;
if in1 and in2;
run;
proc sort data=usgs;
    by stabbr cntyname name;

```

```

%macro mapit;
%do i=1 %to &max;
data _null_;
set counties(where=(obsno=&i));
call symput("state",put(state,z2.));
call symput("county",put(county,z3.));
call symput("cntyname",trim(cntyname));
call symput("stabbr",compress(stabbr));
call
symput("pop90",put(pop90,comma8.));
call
symput("house90",put(house90,comma8.));
call
symput("land",put(landarea,comma8.));
call
symput("water",put(watrarea,comma8.));
run;

```

```

title1 h=3 f=swiss color=red j=c
"&cntyname County, &stabbr";
footnotel h=1 f=swissu color=green j=1
" 1990 Population: &pop90";
footnote2 h=1 f=swissu color=green j=1
"    Households: &house90";
footnote3 h=1 f=swissu color=brown j=1
"  Land Area:  &land square miles";
footnote4 h=1 f=swissu color=blue j=1
"  Water Area: &water square miles";
footnote5 h=1 f=swissu color=black j=1
"  FIPS Codes: State=&state
County=&county";
run;

```

```

proc print data=usgs(where=(obsno=&i));
var name placepop long lat obsno;
run;

```

```

filename gsafile
    "f&state.&county..ps";

```

```

data anno;
set usgs(drop=state where=(obsno=&i));
length function style color $ 8
    position $ 1 text $ 40;
retain xsys ysys '2' hsys '1'
    state 100;
state+1; when='a';
%label(x,y,name,*,0,0,1,swiss,E);
state+1; when='a';
%label(x,y,'T',*,0,0,1,marker,5);
run;

```

```

proc summary data=anno nway;
class obsno;
var lat long;
output out=size(drop=_type_ _freq)
    max=maxlat maxlong
    min=minlat minlong;
run;

```

```
data _null_;
set size;
longdist=69*(maxlong-minlong);
latdist=69*(maxlat-minlat);
if 1.1*latdist=>longdist then
rot='portrait';
    else rot='landscape';
call symput('rot',compress(rot));
run;

data county;
set maps.counties;
if state=input("&state",2.) and
    county=input("&county",3.);
run;

data map1;
set county anno;
run;

proc gproject data=map1
    out=map2;
id state;
run;

data annodata mapdata;
set map2;
if state>100 then output annodata;
    else output mapdata;
run;

data dummy;
state=input("&state",2.);
county=input("&county",3.);
count=1;
output;
run;

pattern value=mempty repeat=9999
    color=black;

goptions rotate=&rot;

proc gmap data=dummy map=mapdata all;
id county;
choro count
    /nolegend
    annotate=annodata;
run;
quit;
%end;
%mend;
run;
%mapit;
run;
```