Paper 104

DELIVERING GEOGRAPHIC INFORMATION, PART II
JUST CLICK MY MAP OBJECT

Jeanne Spicer
The Pennsylvania State University

Problem: What's Happening Where?

Data collection for a face-to-face social survey is being conducted concurrently in five states. Each state has a team of interviewers assigned to cover the addresses of subjects in a particular county. Interviewers report the results of the contacts made each day and the disposition of those contacts is entered into a SAS dataset. The data collection manager needs a daily report summarizing the progress of the survey in terms of the following rates:

- contact rate -- number of subjects contacted / total sample
- response rate -- number of completed interviews/ eligible sample members
- refusal rate -- number of refusals / eligible sample members
- cooperation rate -- number of completed interviews / (completed + partial + refused).

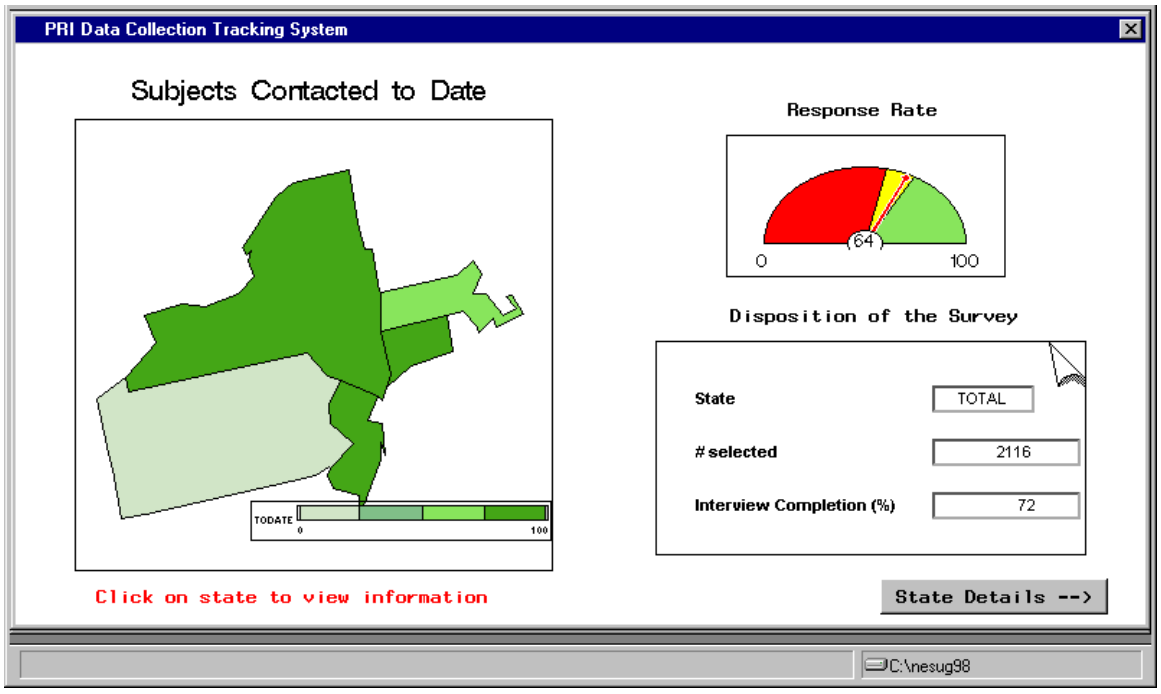
It is necessary to view these results from various perspectives -- the entire survey area, each state team and the interviewers in each county.

Solution: Just Click My Map Object

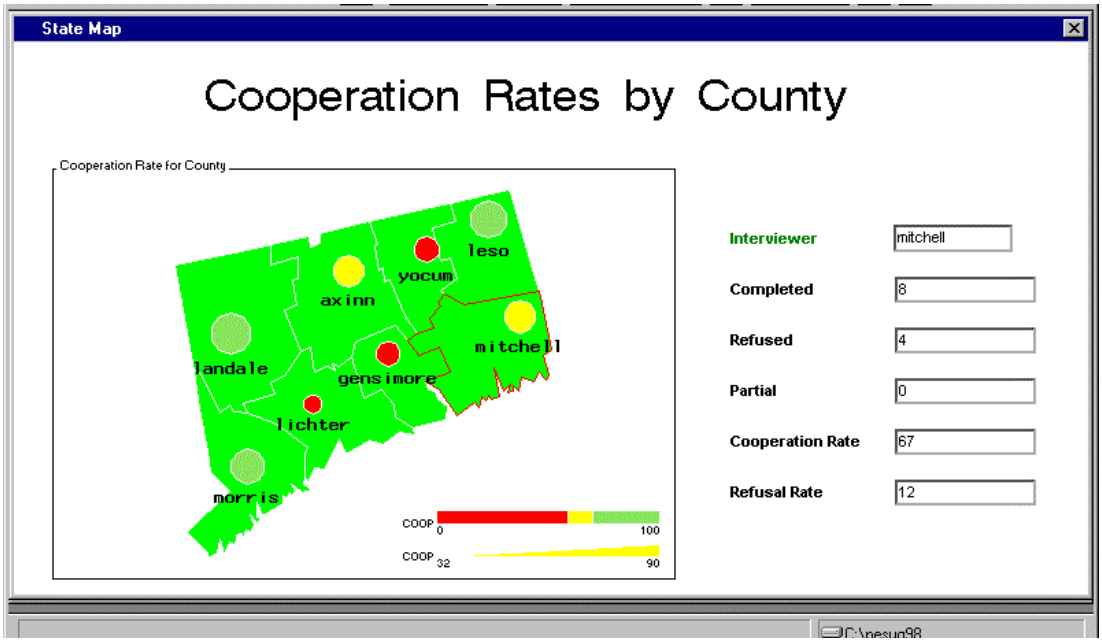
This application uses the SAS/AF Map Object to display the rates on choropleth & marker maps. The user can click on a geographic area of interest for more detailed information.

The Map Class in Frame entries uses a SAS/GRAPH map data set to produce the geographic boundaries and any SAS data set to provide the response (or thematic) variables displayed on the map. The value of the response variable for an area can be distinguished either by its color or the size of its marker.

The map object automatically creates a built-in "hotspot" for each area in the map having a unique geographic identifier in the map data set. These "hotspots" allow the user to "click" on an area to trigger an action based on the value of the ID of the area selected.



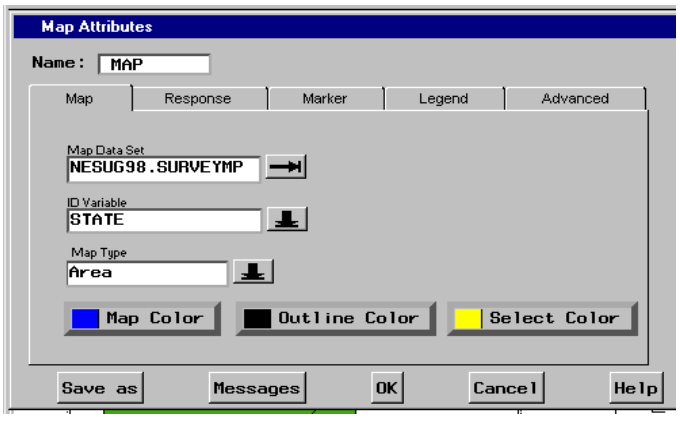
Survey Area Map



State Map

MAKE → MAP

The map class is new to SAS v 6.12. If you are familiar with SAS/GRAPH maps, it is surprisingly easy to build a map object in a Frame entry. In the Build window for a frame entry, select Actions → Make → Map. All the defaults for the object can be set within the tabbed Map Attribute build window that appears. The entries required to produce the map objects for this application are described below.



Tabbed Map Attribute Window -- Map Tab

Map Tab

Map Data Set ➢ Enter the name of a SAS/GRAPH map data set. This application used a subset of the MAPS.US state boundary map. The subset reduces the map to the five states covered by the survey.

ID variable ➢ Enter the name of the variable from the map data set which identifies the appropriate geographic level of detail for your map. The map object can only focus on one level of detail. The "Survey Area" map looks at the entire five state region and the ID variable STATE identifies each state by its FIPS code. The "State" map depicts only one state at a time but can identify each county in the area by its ID variable, COUNTY.

Map type ➢ Choose one: a) Area (as in the "Survey Area" map) or b) Marker (as in the "State" map) to determine how the response variable will be portrayed, i.e. by color or marker size.

Response Tab

Response Data Set ➢ Enter the name of the SAS data set containing the response variables which provide the thematic information for the map. This application uses an output data set summarizing the survey tracking data by state and county. The summary is requested whenever the application is launched, via a submit block.

ID variable ➢ Enter the name of the variable in the response data set that you want to associate with the geographic ID in the map data set. Unlike SAS/GRAPH maps, this variable does **not** have to match the map data set ID in terms of name, type or even value. However, if the values of the ID variables do not match, you will need to build n data set to act as a crosswalk between the two variables. The Map Object build window facilitates this task by allowing you to query the map itself for value of the ID for each area (see below).

Query ID ➢ Check this box to indicate that you want to interactively build a data set to link values for the ID

variable in the Map data set with values of the ID variable in the Response data set when they are not the same. The process queries the ID of the map data set and stores it in a variable called ID, then you are prompted to enter the value from the response data set which will be stored in the variable IDMATCH. This IDMATCH Info Dialog process will not take place until you close the Map Attributes window. Close the window and return to the frame's Build window. Select the Map object then right-mouse click to get a pop-up menu. Select MAP OBJECT →, choose ADD and click on each area of the map that is to receive data. If the map is an area map, the place you click in each area is not important, but if it is a marker map, the coordinates of the spot you click define the position of the marker by clicking on an area of the map. With each click, the value of the ID variable in the map data set is stored in the IDMATCH data set along with the x, y coordinates of the point you clicked. You are then given an opportunity to type in a corresponding value for the ID variable of the response data set. This value is stored in the variable IDMATCH for the observation. The response data set for the "State" map did not include the FIPS code for the county, only for the state. However, since each county is associated with the name of an interviewer I was able to use the IDMATCH Info Dialog to build a crosswalk table associating an interviewer name (from the response data set) with a county FIPS (from the map data set). The resulting data set looks like this:

X	Y	ID	IDMATCH
0.29187311	0.10591766	1	Morris
0.29708422	0.11682270	3	Axinn
0.30689823	0.11950569	15	Leso
Etc…			

In order to make the connection at run time, the user must be prompted to click on the *marker*, rather than the *area* on the map.

Size variable ➢ Valid only when you have selected a marker map. The greater the value of this response data set variable, the larger the marker. There are a variety of marker shapes to choose from. The "State" map employs a circle to display the value of the interviewer's cooperation rate.

Label variable ➢ Annotate each area or marker with the value of a character or numeric value in the response data set. The "State" map uses the interviewer's name.

Color variable ➢ Vary the color of an area in an area map or the color of a marker in a marker map according to the value of this response data set variable. A marker can have its size represent the value of one variable and its color represent the value of another. Or both can be the same. If you do not choose a color variable, SAS will generate a color that contrasts with the map background. In the "Survey Area" map, the color of each state indicates the value of the contact rate.

Color range entry ➢ In order to control the colors displayed for the Color Variable, you must build an additional AF entry. A "Range Entry" is similar to a format, however it allows you to associate a *color* to a value or range of values for display in various AF objects. I had never used one before and was frustrated at first. The build window for the entry is not very forgiving and the colors you select are not displayed accurately on the Color Scale bar for the window. Many times it appeared that I couldn't get enough contrast between colors in the range entry window, especially with gray scales. When I looked at the map object in the build window, the colors were quite different (thankfully they were much more distinct). But this necessitates going back & forth to come up with a visible color scheme.

Marker Tab

Marker data set ➢ Specify an existing file containing appropriate x, y coordinates for your map, and the map & response data set IDs. If the response data set in this application contained a county FIPS code, the "State" map could employ the SAS/GRAPH data set MAPS.USCENTER that contains the coordinates of the visual centers of each state as a marker data set. Since it did not, I created the Marker data set interactively by checking "Query ID" in the Response tab window.

Legend Tab

Legend ➢ Check the box to request the default legend.

Advanced Tab

No advanced techniques were employed for this application.

Map Object Methods (and Madness!)

All of the default attributes for the Map objects in this application were assigned in the Build windows. At runtime, the object is initialized and loaded as soon as the frame is opened. In order to change the appearance of the object on condition, or to capture the value of the ID of the area selected, you will want to become familiar with the methods associated with the map object in order to write a little SCL code.

When the user clicks on a state in the "Survey Area" map, the value of the ID variable for the state is captured and stored in an SCL list of the object's instance variables -- INFOID. The value is then assigned to an SCL variable which can be passed to another object or used in a "where" statement. The where clause is used to subset the summarized survey tracking data for the state selected. Subsequently the Data Form object is loaded with the

subset and the Critical Success Factor object is set to the value of the response rate for the selected state. In the "State" map, the value of the IDMATCH variable (Interviewer name) is captured from the Map object even though the ID variable for the map is the county FIPS code.

```
init:
* SCL STRING VAR TO HOLD WHERE CLAUSE;
length wclause $ 25;
* SET INITIAL APPEARANCE;
  call notify ('button','_hide_');
  _msg_= 'Click in a state to view its data';
* DECLARE SCL LIST TO STORE MAP INSTANCE VARS;
  infoid=makelist();
* OPEN DATASET, DEFAULT IS TO DISPLAY FIRST OBS;
  call notify ('report', '_set_dataset_',
'nesug98.sample');
  call notify
('.','_get_widget_','report',objid);
return;

map:
* LOAD CURRENT VALUES OF MAP INSTANCE VARIABLES
IN SCL LIST 'INFOID';
  call notify('map','_get_value_',infoid);
* DECLARE ONE ELEMENT LIST FOR _SET_WHERE_
METHOD;
  wherelst=makelist(1);
  state=getnitemn(infoid,'id');
* SUBSET DATASET BY STATE & DISPLAY IN DATA
FORM;
  wclause = 'state='||state;
  rc = setitemc(wherelst,wclause,1);
  call notify ('report',
'_set_where_',wherelst);
  refresh;
* GET VALUE OF RESPONSE RATE FROM CURRENT OBS
AND SET CSF;
  call send (objid, '_get_column_value_',
'respr', respr);
  call notify('csf', '_SET_VALUE_', respr);
  call notify ('button','_unhide_');
  refresh;
return;

term:
  infoid=dellist(infoid);
return;
```

Now you try it…

OK, now that you have seen my little frame, try building a frame with a Map Object yourself. If you have the SAS/GRAPH map library and the SASUSER sample data sets, you can experiment with a state level map depicting the incidence of auto theft.

1. To display this information in a Map Object, go into the build window for a SAS/AF frame entry.
2. Select Actions → Make → Map object. A tabbed window will appear.
3. Click the *Map* tab and supply the SASUSER.US map data set. Select the "Type of Map: Area".
4. Click the *Response* tab and supply the name SASUSERS.CRIME. Select AUTO as the "Color variable".
5. Click the *Legend* tab and request "Legend".
6. Click "OK" and view the Map Object in your frame. SAS has a default algorithm for displaying the data, but you can create a RANGE entry in the AF catalog to define the range of values to be displayed by each color.

Parting Thoughts

The Map object is basically a graph with hotspots; not a Geographic Information System. It allows the user to visualize the distribution of their data spatially and to make selections by expressing their choices not in terms of 'who' or 'what', but 'where'.

References:

Czaja, Ronald and Johnny Blair, *Designing Surveys: A Guide to Decisions and Procedures*, Pine Forge Press, 1996, pp269.

Horowitz, Lisa Ann, "Harnessing the Power of SCL Lists", *Proceedings of the Twenty-third Annual SAS Users Group International Conference* (1998), p.48-56.

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

SAS Institute Inc., *SAS/AF Software: Frame Entry Usage and Reference, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1993, 597 pp. 597.

Author:

Jeanne Spicer (spicer@pop.psu.edu) is Manager of Programming Services at the Population Research Institute at Penn State University. She made the transition from COBOL to SAS in 1986. In addition to providing programming support to researchers in sociology & demography, she teaches workshops on various SAS/UNIX topics and is the liaison for the Penn State SAS Users Group. Her team is currently building an environment to facilitate demographic research, including a macro library of utilities and a SAS accessible archive of data related to population studies. This paper is a sequel to her SUGI 23 paper "*Delivering Geographic Information: For Those Who Can't Read A GMAP and Won't Stop to Ask For Directions*". She plans to complete her "Geo-Trilogy" with a paper on SAS/GIS next year.