Mapping: Methods & Tips

Choropleth Google Maps

Michael Peterson
Department of Geography/Geology
University of Nebraska at Omaha

Introduced in 2005, Google Maps offers 18 maps of the world at different scales, varying from approximately 1:85 million to 1:4,800 at the equator at a screen resolution of 100 dpi. Each map has been tiled into individual raster squares that are downloaded separately, often from different servers. A typical Google Map might download map tiles from seven or eight different IP addresses, each associated with a different server that could be located in different Google data centers. Subdividing the map into tiles improves the perceived map download time and allows the map to be easily panned. Google Maps also makes use of the Asynchronous JavaScript and XML (AJAX) server/client technology that maintains a constant connection to the map server, a major improvement in server/client performance.

Maps and imagery in Google Maps have been projected with the Mercator projection. The limitations of this projection have been well-documented, and its distorted depiction of the world has been a major cause for concern. For example, Greenland is represented as being larger than Africa when, in fact, Africa is 14 times larger than Greenland. Scale varies continuously from the equator to the polar areas. Changes in scale in the Google Maps display can be observed by examining the scale bar when moving north or south from the equator. The change in map scale is particularly noticeable at the extreme latitudes. The distortion caused by the Mercator projection is not noticeable with larger scale maps.

In 2006, Google introduced an Application Programming Interface (API) that includes a series of functions that may be invoked by the user. These functions control the appearance of the map, including the scale, position, and any added information in the form of points, lines, or areas. The API makes it possible to incorporate Google Maps on Web sites, and to overlay information from other sources – a process referred to as a "map mashup."

One application of the Google Maps API is the construction of choropleth maps by super-imposing shadings. Current examples include maps of London by the UCL Centre for Advanced Spatial Analysis (CASA) and election result maps by county or state (see Web Resources). The UCL CASA provides Google Map Creator, a freeware application for thematic mapping with Google Maps (see Web Resources). One advantage of choropleth mapping with Google is that the underlying map can remain visible, providing some geographic context to the representation of the data. Normally, thematic maps lack the necessary background map to properly interpret the locational component. While it can be argued that stripping background information may result in the better formation of spatial patterns by the map user, providing more locational information may be viewed as a necessary component for all thematic maps. The purpose here is to demonstrate how choropleth maps can be made with Google Maps.

JavaScript and the Google Map API

API functions may be used with a variety of programming languages. The examples presented by Google use JavaScript. Originally developed for the Netscape browser, JavaScript is a compact, object-based language for developing client-side applications. It is not a computer language that makes executable code, like C++ or Java. Rather, the browser interprets JavaScript statements that are embedded in, or referenced from, an HTML page. The JavaScript program is executed when the browser page is opened. This was initially viewed as a problem because it slowed down the execution of the program. With today's faster computers, there is no longer a major advantage to pre-compiling computer code. JavaScript can reside within an HTML file or a reference can be made to an external file. The external file that contains JavaScript functions can be on the same computer as the HTML file, or it can be on another computer or server. This is how Application Programming Interface (API) code is distributed. One reference to a library of API code makes it possible for a Web page designer to access thousands of mapping functions.

To use the Google Map API, a free numeric key must be requested from Google. This is a unique identifier that is matched to the website. Google has a number of terms of use, including that there be no more than 500,000 page views per day without prior warning, a limit of 15,000 geocode requests (finding street addresses), no advertising, free accessibility to end users, no altering or obscuring the logos in the map, and no illegal activity. The key gives Google some control in how their API is used and provides data on the amount of usage by website. Google Map functions begin with the letter "G." All functions re-

volve around GMap2, a central class in the API that is used to initialize the map.

Polygon Conversion

Encoding polygon outlines is a very time-consuming task. Fortunately, there are many polygon files available that can be converted for use by Google Maps. Once the polygons are defined, shadings can be assigned to indicate the value of the area.

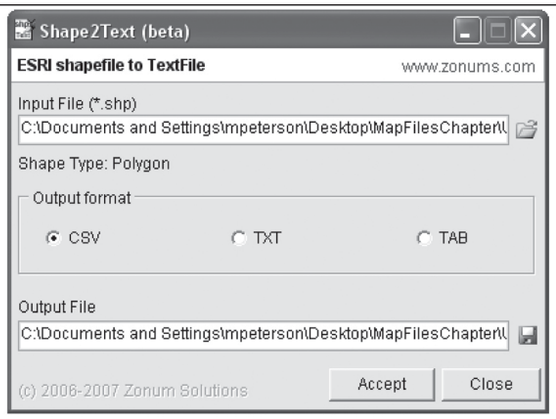
The most common format for map files is the so-called shapefile (.shp), a format developed by ESRI–the major distributor of GIS software. Developed in the early 1990s, many files in this format are available through the Internet from libraries and other online portals. Shapefiles can be very large with many points. For example, a shapefile with the boundaries for the counties of Nebraska, a state with many rectilinear county outlines, has over 25,000 x,y coordinates. JavaScript would require a considerable amount of time to read this many coordinates from a file.

One option to speed the display of the map is to reduce the number of coordinates in the file. MapShaper.org is an online resource for the generalization of lines within shapefiles. Using a simple interface implemented with Adobe Flash, the website uploads a shapefile and then asks the user to specify the amount of generalization. Following this, the shapefile can be downloaded to the user's computer.

The conversion of the shapefile to a text file is necessary for use by Google Maps. A program called Shape2Text extracts the coordinates and places them into an Excel file. The points then must be put into the proper XML format as shown in the bottom of Figure 1. The map in Figure 2 shows the 93 polygons for Nebraska plotted with Google Maps.

Choropleth Map

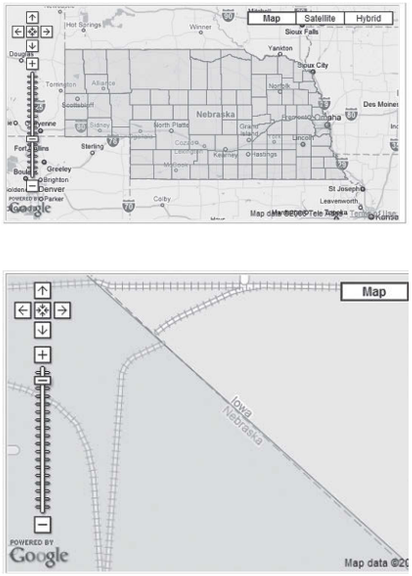
Altering the shading of each polygon based on a data value would result in a choropleth map. In this case, we map out population data for the state of Nebraska. The population values could be input through another XML file. To simplify matters, we simply put the 93 population data values into an array (see Figure 2). These population values are listed in alphabetical




```
12      Nodes
-122.9679783  48.44379451
-123.0952329  48.47942282
-123.1597199  48.52184222
-123.1698993  48.56256471
-123.1410538  48.62364712
-123.1037214  48.60837712
-123.0120949  48.55747774
-123.0086988  48.53371932
-122.9679800  48.52693332
-123.0222711  48.51335968
-123.0188829  48.48960517
-122.9679783  48.44379451
```

```
<poly linecolor="#008800" linewidth="4" lineopacity = "1.0" fillcolor=
"#FFCC00" fillopacity = "0.5" html="State">
  <point lat="-122.9679783421" lng="48.4437945085"/>
  <point lat="-123.0952328681" lng="48.4794228153"/>
  <point lat="-123.1597199251" lng="48.5218422237"/>
  <point lat="-123.1698993372" lng="48.5625647146"/>
  <point lat="-123.1410538081" lng="48.6236471212"/>
  <point lat="-123.1037213929" lng="48.6083771192"/>
  <point lat="-123.0120949153" lng="48.5574777421"/>
  <point lat="-123.0086987596" lng="48.5337193216"/>
  <point lat="-122.9679800006" lng="48.5269333223"/>
  <point lat="-123.0222711218" lng="48.5133596826"/>
  <point lat="-123.0188828947" lng="48.4896051705"/>
  <point lat="-122.9679783421" lng="48.4437945085"/>
</poly>
```

*Figure 1. The Shape2Text conversion process leading to the creation of an XML file. The program asks for the location of the *.shp file, the output format, and the output location. A single polygon with 12 points (nodes) is shown in the upper-right. These points are then converted into the proper XML poly format using the Excel concatenate function. (see page 95 for color version)*




```
// Put the population data for the counties into the popdata array
      popdata = new Array
(33185,6931,372,783,492,5668,11132,2185,3354,43954,7341,8595,25963,8819,3811,5934,9865,
50,892,3710,8812,267135,35865,749,656,497,35279,7954,5171,3705,7247,4650,15747,2804,299

// Find the min and max population values for the 93 counties after doing a non-linear
      var min=100000000;
      var max=-100000000;
      for (var i = 0; i < 93; i++) {
        popdata[i] = Math.log(popdata[i])
        if (popdata[i] < min) { min=popdata[i] }
        if (popdata[i] > max) { max=popdata[i] }
       }
// Find the range in the data values
         var range = max-min
// Compute the opacity for each county, a value between 0 and 1
         opacities = new Array ()
         for (var i = 0; i < 93; i++) {
           opacities[i] = 1-((max - popdata[i]) / range)
       }
```

Figure 2. A shapefile map of Nebraska by county mapped with Google Maps after line coordinate thinning with MapShaper and conversion to a text file by Shape2Text. The state border between the shapefile and the Google Map matches nearly perfectly, although the underlying map from Google may have errors along the border as with the discontinuity in the railroad line that is visible in the enlarged map. (see page 95 for color version)

order by county, the same as for the polygons in the map file.

After the data have been assigned to the pop data array, they are converted to their natural log value using the "Math.log" function. The log conversion compensates for the extreme population values for the counties that contain the cities of Omaha and Lincoln. The minimum and maximum values are determined in the same loop. A second loop computes the opacities for each county based on the maximum data value and the range of the data. As such, the map represents an unclassed choropleth map because the data values have not been put into categories. The opacity of each county is directly proportional to the log of the population. The zoomed-in map in Figure 3 shows how the background map is visible in the less populated counties.

The online version of the Nebraska population map using variations in opacity (see Web Resources). The source code for the program can be viewed by selecting View or Page Source. The XML file that contains the map of Nebraska is also available (see Web Resources).

Summary

The Google Maps API represents a powerful mapping tool. By providing base maps and imagery at multiple scales as a backdrop, all manner of information can be added to the foreground of the map. While we have little control over the base map, Google provides a great deal of flexibility in what may be added on top of the map. With competition from sites like Microsoft Live and Yahoo! Maps, there will be a great deal of development in this area. It is unfortunate, however, that all of these online mapping sites are providing APIs that are incompatible with each other.

There are a number of ways to speed the display of the map. The major approach, and the one favored by Google, is to pre-compile the map and convert it to a tiled representation at 18 different scales – just as is done for the base Google Map. This tiled map can be as quickly displayed as the base map.

Web Resources

CASA's Google Map Creator:
http://www.casa.ucl.ac.uk/software/googlemapcreator.asp

Iowa Presidential Caucus Results:
http://maps.google.com/maps/mpl?moduleurl=http://www.google.com/mapfiles/mapplets/iowacaucus/iowacaucus.xml&utm_campaign=en

Nebraska Population Map Example:
http://maps.unomaha.edu/GoogleMapGallery/Nebraska/Population.html

XML Source Code File:
http://maps.unomaha.edu/GoogleMapGallery/Nebraska/Nebraska.xml

* From an upcoming book by the author on online mapping to be published by Springer Verlag.

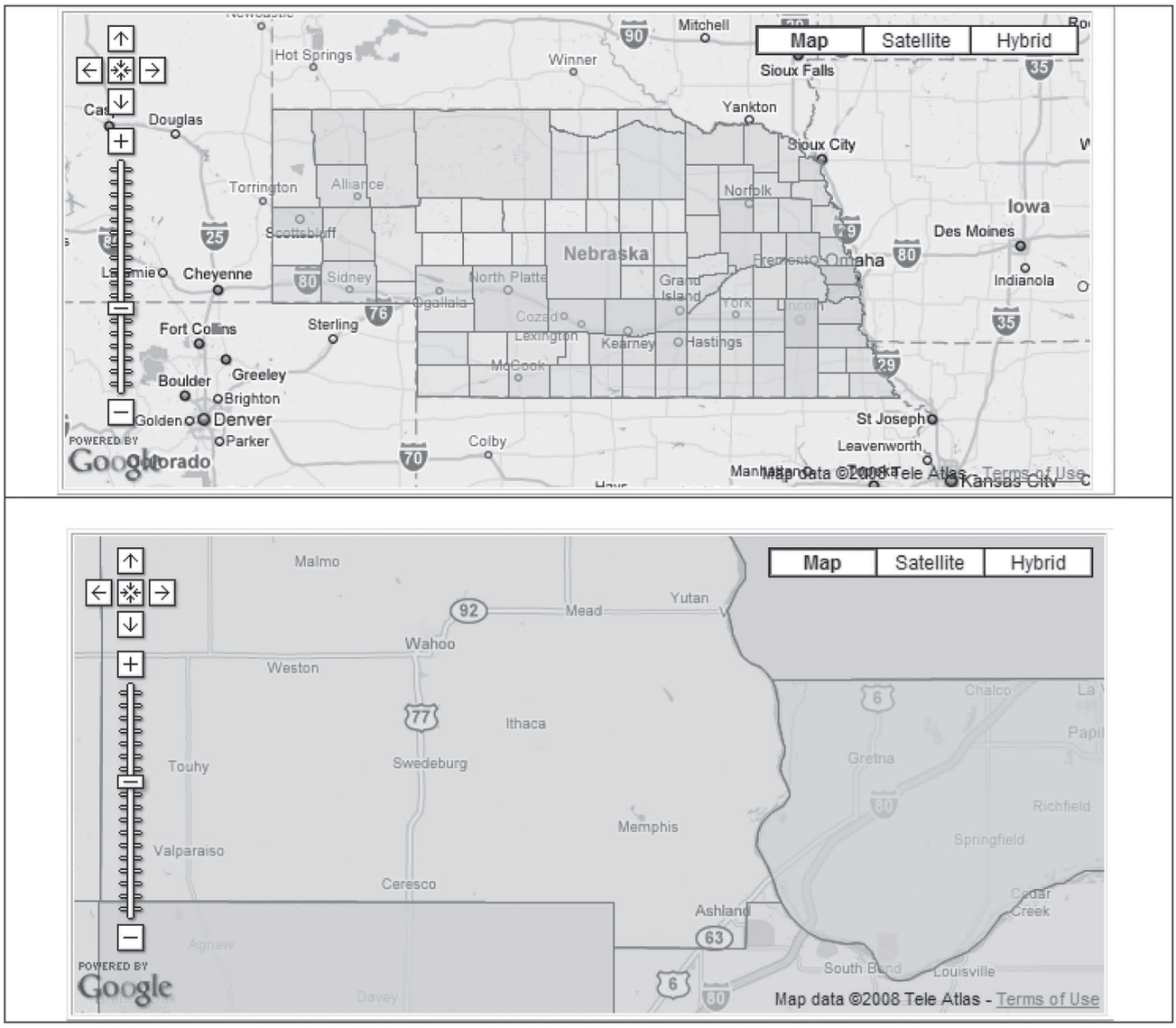


Figure 3. A population map of Nebraska. The opacity of the color that is assigned to each county is proportional to its population. The data have been converted to a log value to compensate for the skewed population distribution caused by the two largest cities, Omaha and Lincoln. The zoomed-in map on the bottom shows that place locations are visible in the less populated counties that have been assigned a lower opacity value. (see page 96 for color version)