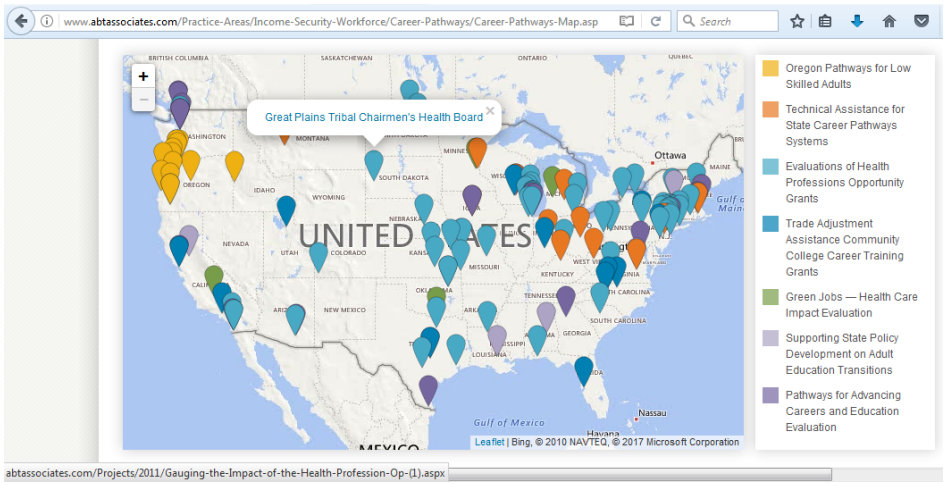
Using Stata to Create Interactive Maps

Ali Lauer and
Austin Nichols

July 27, 2017

Stata Conference



Contents

- Standard maps in Stata
  - **twoway** command
  - **shp2dta** command
  - Geographic analysis
- Interactive Maps
  - **file** command
  - Javascript and JSON objects for the web

twoway

- Maps can be constructed in Stata using **twoway** commands
  - Built-in command
  - Available in secure environments
- Use polygon “shape file” dataset
  - **twoway area**
  - Options: **nodropbase cmiss(no) yscale(off) xscale(off)**
- Separated by || or set off by (), as in any graph

Map Types:

- Choropleth: use multiple calls to **area** each with **if** qualifiers to build a choropleth
- Point Data: use **scatter** to superimpose point data

shp2dta

- **shp2dta** command converts various shapefile formats into Stata data
  - User-written command by Kevin Crow, freely available through SSC Archive
  - Shape file (coordinates file) contains _X (longitude), _Y (latitude), and _ID (shape identifier) variables
  - Data file has info on each shape

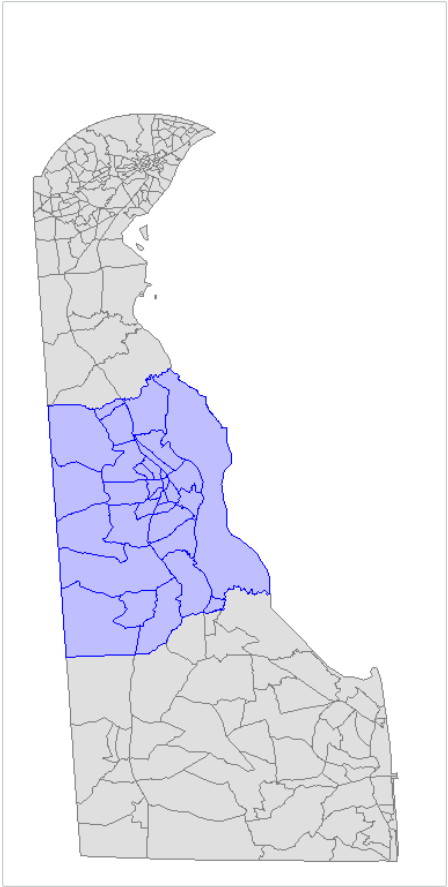
- **.shp**
- **.dbf**

shp2dta →

2 Stata datasets:
- Shape file
- Data file

Example – Mapping Census Tracts

- Map of Census tracts in Delaware, with one county highlighted
- Download the shape file from Census Bureau
  - Publically available data
- Convert to a Stata dataset using **shp2dta** command
- Map using **twoway** command



Delaware example

- Get shapefile, unzip and convert
  - zipped tract shapefiles www.census.gov/cgi-bin/geo/shapefiles/index.php?year=2016&layergroup=Census+Tracts) or via FTP:

copy ftp://ftp2.census.gov/geo/tiger/TIGER2016/TRACT/tl_2016_10_tract.zip tl_2016_10_tract.zip

unzipfile tl_2016_10_tract.zip

shp2dta using tl_2016_10_tract, data(d10) coord(c10)

- Draw map

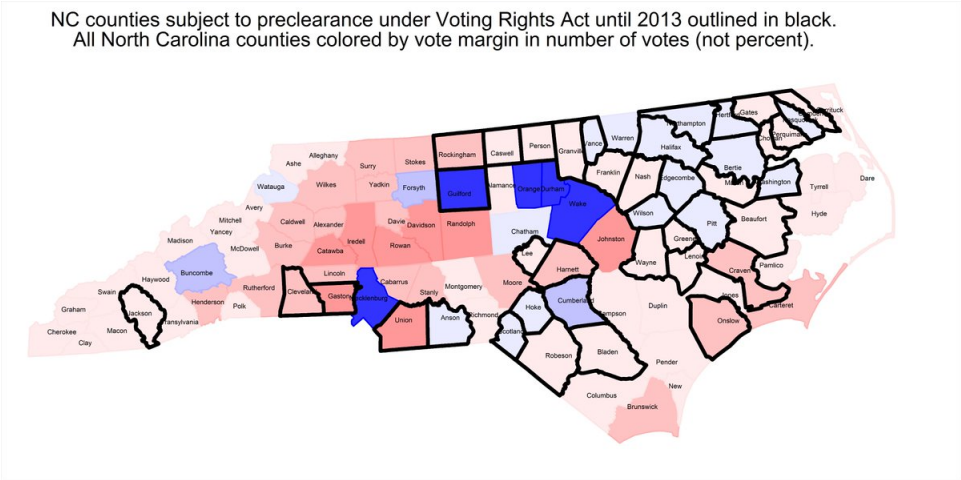
use c10, clear

merge m:1 _ID using d10

twoway area _Y _X if ALAND>0, nodropbase cmiss(n) fi(25) col(gray)||area _Y _X if COUNTYFP=="001" & ALAND>0, nodropbase cmiss(n) fi(25) col(blue) leg(off) ysc(off) yla(,nogrid) xsc(off) graphr(fc(white)) xsize(4) ysize(8)

Why use twoway?

- This approach can also be automated by user-written commands **spmap** or **tmap**, but these commands do not admit the full complement of **twoway** options and possible overlays.
- **twoway** allows user to add arbitrary graphs on top of your map, reposition shapes, draw curves or arrows or text boxes**.** Change coordinates, rotate, customize legends.



Career Pathways example

- Easy to add a scatter of point data on top of the area "shape" map.
- Suppose we have lat/lon for a bunch of sites where we are doing work (on Career Pathways here).
- Dataset cp.dta looks like:

```
. u cp, clear
. li in 1/5
     +----------------------+
     |        v1          v2 |
     |----------------------|
  1. | -95.99277   36.15398 |
  2. |   -72.146     41.609 |
  3. | -79.99589   40.44062 |
  4. | -73.86543   40.83705 |
  5. | -97.08195   32.83707 |
     +----------------------+
```

Career Pathways example

- Use a map, add the scatter:

```
. u http://pped.org/usa
. keep if !inlist(_ID,56,1,13)
(4,772 observations deleted)
. merge using cp
(note: you are using old merge syntax; see [D] merge
for new syntax)

. line _Y _X, cmiss(n)||sc v2 v1, leg(off) xla(-124/-
71) yla(,nogrid) ysc(off) xsc(off) graphr(fc(white))
```

Why use Stata for your maps?

- Advantages
  - Avoid using multiple programs
  - Allows for complicated spatial analysis before or after making your map
  - Easy to customize calculations
- Implications of new features in Stata 15
  - Transparency in graphs
  - Spatial autoregressive models (SAR)
  - Improvement for Java plugins

Further advantages of Stata

- Manipulation of data that is not possible in other mapping-specific applications
- Change the projection
  - convert lat/lon into different y and x coordinate system e.g. using **geo2xy** by Robert Picard (on SSC).
- Smooth geographic data as a pre-processing step,
  - Using built-in commands, such as methods from **twoway contour**
- Use predictions from SAR models **spregress, spivregress,** etc. newly available in Stata 15 and described in the [SP] manual. Geographic regression output can be added to maps.

Further advantages of Stata

- There are many other handy downloadable packages, e.g. **gpsbound**, **geocode**, **traveltime** (findit will find these and others online):
- T. S. L. Brophy, R. C. Daniels, and S. Musundwa. 2015. Importing & verifying geographical info. from a shapefile. SJ 15(2):523--536
- Includes **gpsbound**, to check whether the GPS coordinates lie within the bounds of a polygon demarcated in the shapefile.
- A. Ozimek and D. Miles. 2011. Geocoding and generating travel time and distance info. SJ11(1):106--119.
- Has programs to assign latitude and longitude (geocode) as well as driving distances and travel times between points using Google Maps (with an API key).
- Approaches using Google Maps need occasional updating as Google changes their API and whole architecture on a whim, frequently.

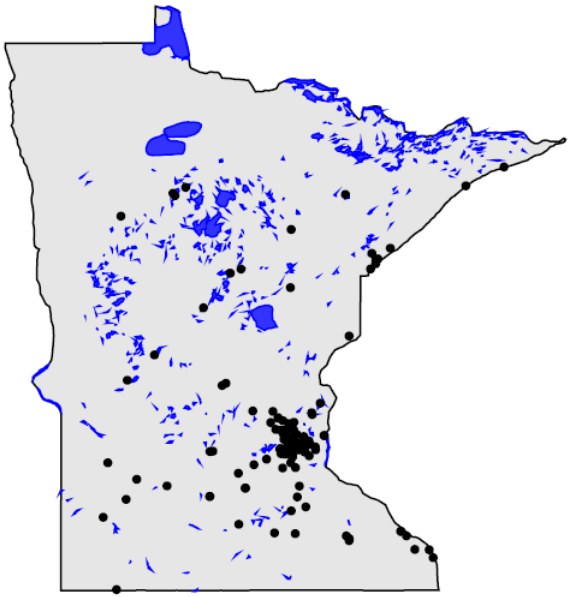
Another kind of advantage

- Create custom maps by computing distances between datasets
  - Second dataset is a shapefile of polygon vertices used in mapping
  - Want to find the distance of points in the first datasets, to shapes in the second

Minnesota example

- For example, suppose we have a dataset **mnch** with locations of charter schools and a dataset **mnlakes** of lake boundaries, and we want to find the nearest lake to each school. Distance calculation is done here by **vincenty** (on SSC), but could also be calculated by hand using any formula one prefers:

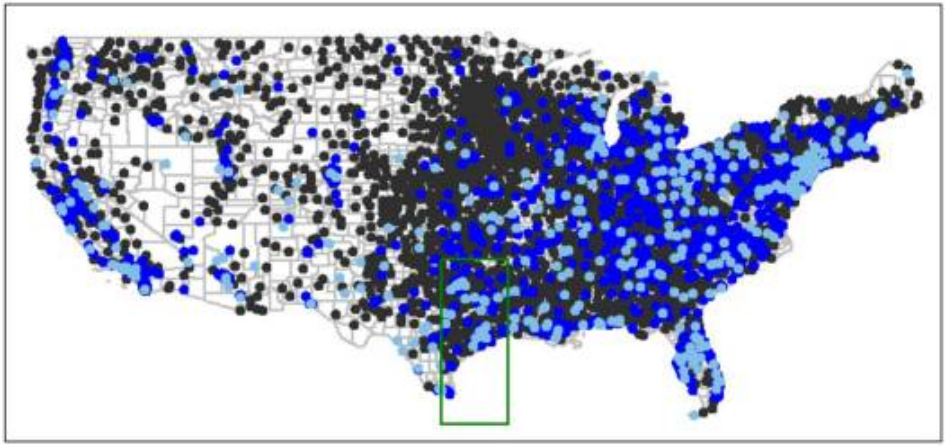
```
use http://pped.org/mnch, clear
local n=_N
g double m=.
merge using http://pped.org/mnlakes
qui forv i=1/‘n’ {
        vincenty lat lon y[‘i’] x[‘i’], v(d)
        su d, meanonly
        replace m=r(min) in ‘i’
        drop d
        }
```

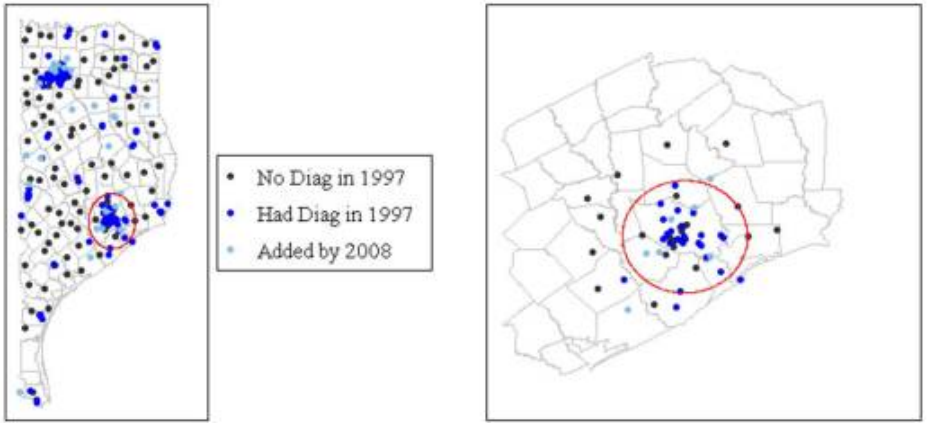


Hospitals example

- Or suppose we want to know if new cardiac capacity is added far from existing capacity (improving access) or near existing capacity (duplicative provision that can lower average quality):

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3938331/figure/F1/





Interactive Maps

You can make any map in Stata as a twoway graph, except interactive maps for the web

- Use **file** command to write **JSON files:**
  - short for JavaScript Object Notation
  - JavaScript object (collection of data) in text form
  - Data can be sent to or received from browsers, and saved for serving on browsers
- Integrate JSON file with open-source map tools such as Leaflet to create interactive maps for the web

Interactive Maps

- A minimal object is open curly brace, name:value pair in double quotes, end curly brace, like

  { "name":"value" }

- But you can write arrays [{},{}] in a value to encode a whole dataset:

```
{"authors":[
   { "firstName":"Ali", "lastName":"Lauer" },
   { "firstName":"Austin", "lastName":"Nichols" }
        ]}
```

- Or put lat/lon pairs, with data corresponding to those points, in the JSON file:

```
markers = [
{
"name": "Peru",
"url": '<li><a href=http://www.politicasensalud.org/blogpolsalud/ target=_blank>Partners for Health Reform plus;
Health Policy Reform</a>',
"lat": -9.099697113037109,
"lng": -74.39474487304688
},
```

Many off-the-shelf .js maps

- Once you have your map data in JSON form, can program the interactive map in Javascript, or use one of many off-the-shelf options e.g. Leaflet/OpenStreetMaps.
- Can also write out all the code from Stata, with **file** again. Or just use a text editor.

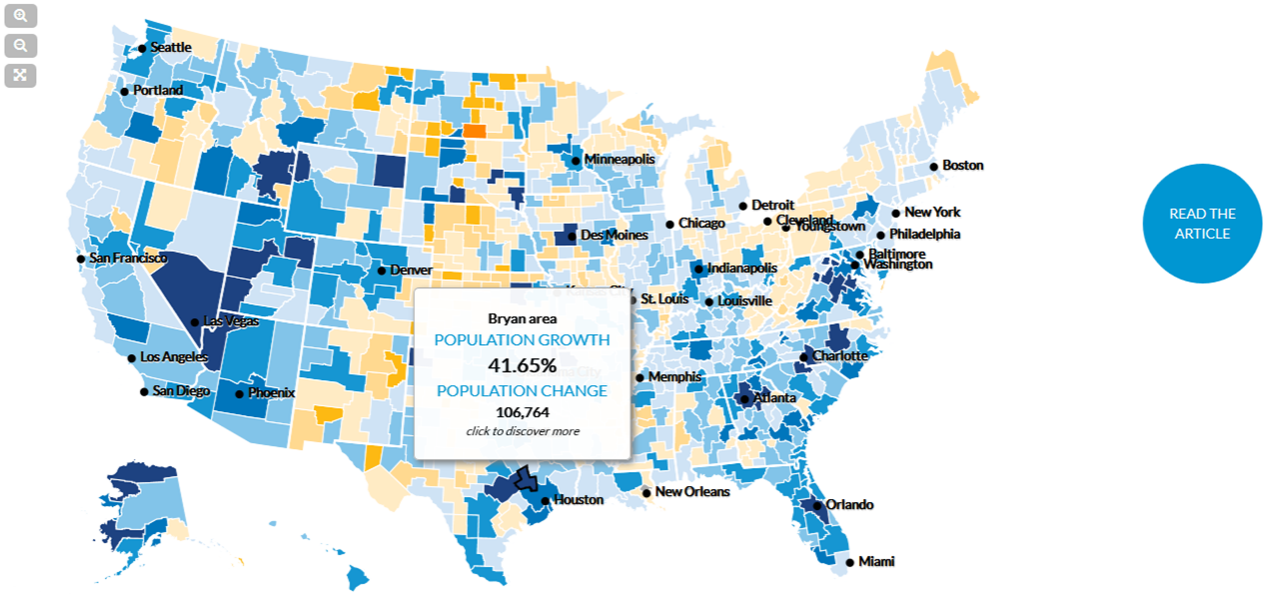
```
var map = L.map( 'map', {   center: [20.0, 5.0],  minZoom: 2,   zoom: 2 })
L.tileLayer( 'http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
  attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a>',
  subdomains: ['a', 'b', 'c']
}).addTo( map )
var myURL = jQuery( 'script[src$="leaf-demo.js"]' ).attr( 'src' ).replace( 'leaf-demo.js', '' )
var myIcon = L.icon({
  iconUrl: 'pin24.png',  iconRetinaUrl: 'pin48.png',   iconSize: [29, 24],   iconAnchor: [9, 21],   popupAnchor: [0, -14]
})
for ( var i=0; i < markers.length; ++i ) {
 L.marker( [markers[i].lat, markers[i].lng] )
  .bindPopup( '<b>' + markers[i].name + markers[i].url )
  .addTo( map ); }
```

GHS map example

- That code produces this map:

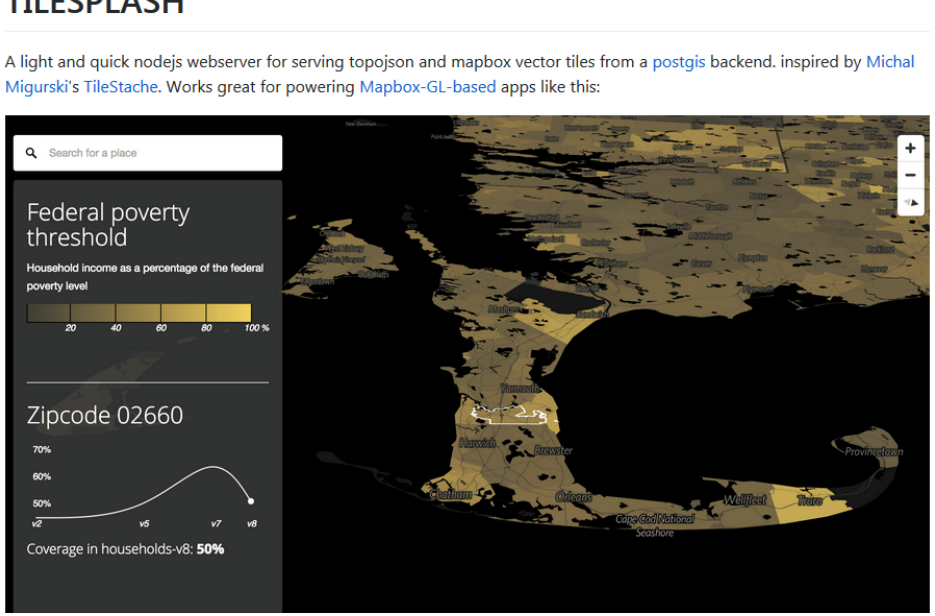
Mapping America's Futures

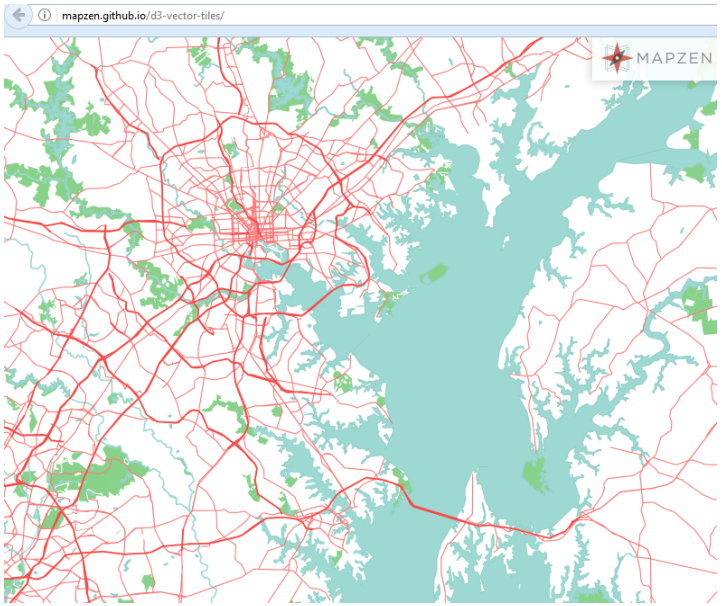
- Custom map with code at https://github.com/UrbanInstitute/mapping-americas-futures also uses the same idea, really, writing out JSON files with data that is manipulated with a relatively short Javascript program.
- But more complicated examples will have a lot more data in the JSON, and probably map polygons saved as data as well, not just markers.



Options

- If you don't want to build your own cool shapefile in Javascript written out from Stata via **file**, there are literally hundreds of free map utilities on github that can make wonderful maps using a JSON file.

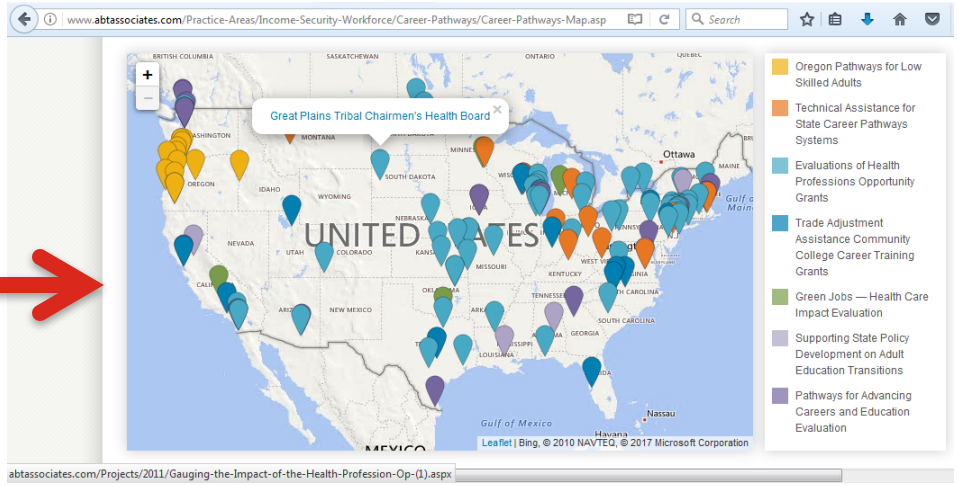




Career Pathways example redux

- Remember the Career pathways map from earlier? Easy to turn it into an interactive web map.
- Just need JSON with info like:

var CareerPathwaysCities = { features: [{ geometry: { coordinates: [-95.992775, 36.153982], }, properties: {GranteeProgram: [{ link: "http://abtassociates.com/Projects/2011/Gauging-the-Impact-of-the-Health-Profession-Op-(1).aspx", name: "Community Action Project of Tulsa County Inc.", type: "hpog" }], },



Thank you!