Static and Interactive Mapping in R

Diksha Radhakrishnan

November 28, 2017

Introduction

Hello! My name is Diksha Radhakrishnan, and I'll be guiding you through this (hopefully interesting and not too complex) post!

As a fellow student in Stat 133, I have thoroughly enjoyed interacting with the course material thus far, finding the homeworks a necessary challenge and the labs incredibly informative. However, I think that we could have engaged more with different types of data visualization tools outside of graphing with base R, ggplot and ggvis. I find that **map visualization** is often a very useful tool, and today, I am going to guide you through a computationally reproducible post on static and interactive data mapping - using data from the *2016 U.S. elections*. Feel free to follow along!

Motivation

My motivation behind wanting to introduce whoever reads this post to mapping in R is:

1. To go beyond the scope of what we are learning in this class
2. My interest in mapping, given my exposure to these tools in other computing languages such as Python.

Background

We will be getting introduced to static mapping using packages **tmap** and **tmaptools**, and interactive mapping using **leaflet**. We will also be installing **sf**, which is the "Simple Features for R" package - we will explore its purpose soon. Let us install and load them right now:

```
# run following command if you do not have these packages:
install.packages(c("tmap", "tmaptools", "sf", "leaflet"))
```

```
library(tmap)
```

```
## Warning: package 'tmap' was built under R version 3.4.3
```

```
library(tmaptools)
```

```
## Warning: package 'tmaptools' was built under R version 3.4.3
```

```
library(sf)
```

```
## Warning: package 'sf' was built under R version 3.4.2
```

```
## Linking to GEOS 3.6.1, GDAL 2.1.3, proj.4 4.9.3
```

```
library(leaflet)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

I used the following tutorials by the **tmap** package creator Martijn Tennekes to learn about static mapping in order to teach you:

- Presentation on "tmap"
- "Tmap in a nutshell"

Example: NH Elections Data

The Data

I will be using a dataset of the Democratic primary results in the state of New Hampshire, which are originally available on the state secretary of state's office website as an Excel spreadsheet, but I obtained a cleaner version from this zip archive. The zip folder also contains the original unclean version. Let's read in the .csv file:

```
nh_data <- read.csv("NHD2016.csv")
nh_data
```

```
##          County Adams Burke Clinton De.La.Fuente Elbot French Greenstein
## 1       Belknap     4     4    3495            5     1       1          NA
## 2       Carroll     2     3    3230            4     2       2           2
## 3      Cheshire     3     7    5132            8     4       2           2
## 4          Coos     1     6    2013            4    NA      NA           1
## 5       Grafton     4     9    6918            4     1       3           1
## 6  Hillsborough    13    35   28147           36    12       6           7
## 7     Merrimack     4     7   12250            6     3       5           6
## 8    Rockingham    17    20   22829           19    12       7           3
## 9     Strafford     3     9    8813            7     1       2           6
## 10     Sullivan     2     7    2497            2    NA       1           1
##    Hewes Hutton Judd Kelso Lipscomb Locke Lovitt McGaughey..Jr. Moroz
## 1     NA     NA    2     4       NA     2      2             NA     1
## 2     NA     NA    1     1       NA     1      2              2    NA
## 3      1     NA    4     3        2     1      1              1    NA
## 4     NA     NA    1    NA        1    NA      1              1    NA
## 5      1     NA    2     2        2     1      1             NA     2
## 6     10      5   13    12        2     5      4              4    24
## 7      2      1    6     3        2     8      3              2    NA
## 8      4      3    9    14        4     9      4              8    NA
## 9     NA      4    4     3       NA     3      2              1    NA
## 10    NA      1    2     4        1     2      1             NA    NA
##    O.Donnell..Jr. O.Malley Sanders Schwass Sloan Sonnino Steinberg Supreme
## 1               2       35    6005       6     2      NA         NA      10
## 2               5       20    5638       4     1      NA         NA      11
## 3               1       42   12441       5     2       1          2      14
## 4               1       20    3639       4     1       2         NA       1
## 5              NA       41   14245       5    NA      NA          1       9
## 6              11      202   39245      32     4       6         11      82
## 7               3       78   18107      21     1       4          4      39
## 8               2      123   31065      41     2       3         NA      54
## 9               1       72   15881      20    NA       1          2      31
## 10             NA       27    5915       4     1      NA          1      14
##    Thistle Valentine Weil Wolfe
## 1       12         2   NA     3
## 2        6        NA   NA    NA
## 3        6         1    1     6
## 4       10        NA   NA    NA
## 5       14        NA   NA     5
## 6       91        13    4    17
## 7       24         1    3     4
## 8       40         3   NA    12
## 9       15         2   NA     6
## 10       5         2   NA     1
```

In order to focus on mapping and not really piecing through the data, let us only focus our attention on the two major candidates: Bernie Sanders and Hillary Clinton, and the county data!

```
nh_data <- nh_data[ ,c("County", "Clinton", "Sanders")]
nh_data
```

```
##          County Clinton Sanders
## 1       Belknap    3495    6005
## 2       Carroll    3230    5638
## 3      Cheshire    5132   12441
## 4          Coos    2013    3639
## 5       Grafton    6918   14245
## 6  Hillsborough   28147   39245
## 7     Merrimack   12250   18107
## 8    Rockingham   22829   31065
## 9     Strafford    8813   15881
## 10     Sullivan    2497    5915
```

Some Wrangling

Let's calculate the candidates' margins of victory and percentage of total votes (Bernie won in NH) and add some columns to this dataframe:

```
nh_data$MarginVotes_Sanders <- nh_data$Sanders - nh_data$Clinton
nh_data$Percent_Sanders <- round((nh_data$Sanders / (nh_data$Sanders + nh_data$Clinton))*100 , 2)
nh_data$Percent_Clinton <- round((nh_data$Clinton  / (nh_data$Sanders + nh_data$Clinton))*100 , 2)
nh_data$MarginPercent_Sanders <- nh_data$Percent_Sanders - nh_data$Percent_Clinton
nh_data
```

```
##          County Clinton Sanders MarginVotes_Sanders Percent_Sanders
## 1       Belknap    3495    6005                2510           63.21
## 2       Carroll    3230    5638                2408           63.58
## 3      Cheshire    5132   12441                7309           70.80
## 4          Coos    2013    3639                1626           64.38
## 5       Grafton    6918   14245                7327           67.31
## 6  Hillsborough   28147   39245               11098           58.23
## 7     Merrimack   12250   18107                5857           59.65
## 8    Rockingham   22829   31065                8236           57.64
## 9     Strafford    8813   15881                7068           64.31
## 10     Sullivan    2497    5915                3418           70.32
##    Percent_Clinton MarginPercent_Sanders
## 1            36.79                26.42
## 2            36.42                27.16
## 3            29.20                41.60
## 4            35.62                28.76
## 5            32.69                34.62
## 6            41.77                16.46
## 7            40.35                19.30
## 8            42.36                15.28
## 9            35.69                28.62
## 10           29.68                40.64
```

Geographic mapping data

We'll use shapefiles from the U.S. Census Bureau, from their page on cartographic boundaries, in order to map the election results for the various counties of the state. The specific zip file has all the national counties, by state (and can also be found in the zip file from earlier). We want the shape file and so we will use read_shape() from **tmaptools**:

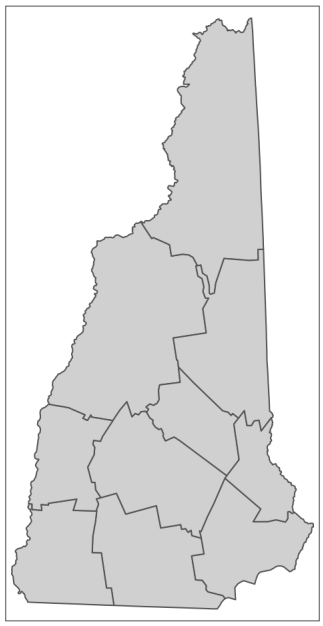
```
us_shape <- "cb_2014_us_county_5m/cb_2014_us_county_5m.shp"
us_geog <- read_shape(file=us_shape, as.sf = TRUE)
```

```
## Warning in readOGR(dir, base, verbose = FALSE, ...): Z-dimension discarded
```

where as.sf = TRUE implies that we wnat usgeog to be a *simple features* object, which gives it a more simple (surpise surprise!) structure. You can see what the map command qtm(us_geog) outputs, as well as inspect the structure of it with str(us_geog).

Now we need to extract New Hampshire data, using its FIPS I.D., which is 33.

```
nh_geog <- filter(us_geog, STATEFP == "33")
qtm(nh_geog)
```



And voila!

Merging election results and mapping/spatial data

Do these two databases share a column, or other similarities that can help us with merging?

```
str(nh_geog$NAME)
```

```
##  Factor w/ 1921 levels "A\xf1asco","Abbeville",..: 684 791 416 138 1470 334 1653 1131 282 1657
```

```
str(nh_data$County)
```

```
##  Factor w/ 10 levels "Belknap","Carroll",..: 1 2 3 4 5 6 7 8 9 10
```

I would like to convert the factors to vectors of strings:

```
nh_geog$NAME <- as.character(nh_geog$NAME)
nh_data$County <- as.character(nh_data$County)
```

Now we can compare the two sorted datasets:

```
nh_geog <- nh_geog[order(nh_geog$NAME), ]
nh_data <- nh_data[order(nh_data$County), ]
nh_data$County == nh_geog$NAME
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Voila! We can now join these datasets.

```
nh_map <- append_data(nh_geog, nh_data, key.shp = "NAME", key.data="County")
```

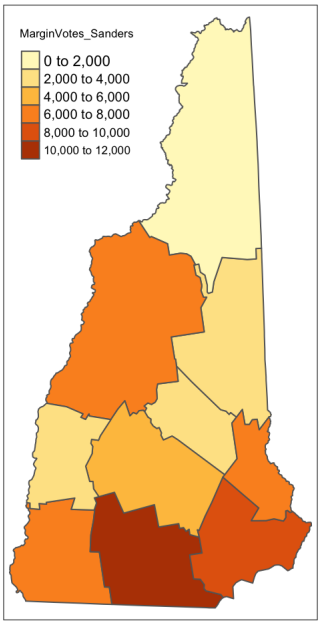
```
## Keys match perfectly.
```

And we can now go ahead with the mapping!

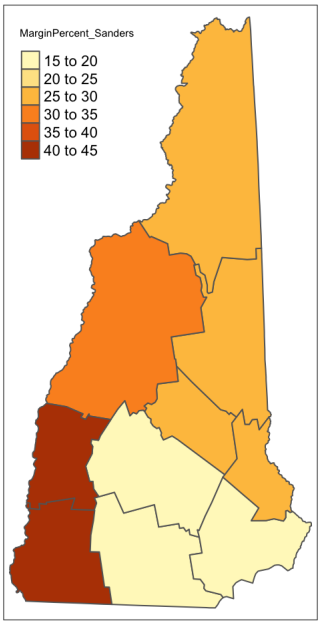
Static Mapping

We can finally create a map of Bernie's margins (in votes and in percentages):

```
qtm(nh_map, "MarginVotes_Sanders")
```

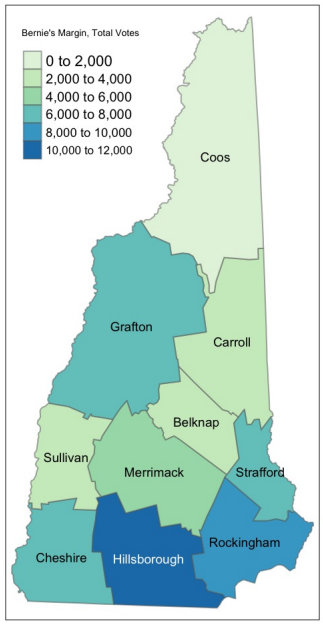



```
qtm(nh_map, "MarginPercent_Sanders")
```



In order to edit the colors, borders, etc., we can use tm_shape, which has a syntax similar to ggplot2:

```
nh_staticmap <- tm_shape(nh_map) +
tm_fill("MarginVotes_Sanders", title="Bernie's Margin, Total Votes", palette = "GnBu") +
tm_borders(alpha=.5) +
tm_text("NAME", size=0.6)
nh_staticmap
```



I used the GnBu color palette from this website.

Now let's save this map, using the save_tmap() function!

```
save_tmap(nh_staticmap, filename = "NHDemocraticPrimary2016.jpg")
```

```
## Map saved to /Users/diksharadhakrishnan/Desktop/stat133/stat133-hws-fall17/post02/NHDemocraticPrimary2016.jpg
```

```
## Resolution: 2100 by 1500 pixels
```

```
## Size: 7 by 5 inches (300 dpi)
```

Making a color palette and a pop-up window for Interactive Mapping

This next map will hopefully enable us to have an idea of the underlying data behind it, using the **leaflet** package.

Create a Leaflet palette like so:

```
mypalette <- colorFunction(palette = "colors I want", domain = mydataframe$dataColumnToMap)
```

In this case, let's map the opposite of before, showing the counties in which Hillary was the **strongest**:

```
clinton_palette <- colorNumeric(palette = "Blues", domain=nh_map$Percent_Clinton)
```

Pop-up to show the percentages of votes cast for Bernie and Hillary in each county:

```
nh_popup <- paste0("County: ", nh_map$NAME, ", Sanders ", nh_map$Percent_Sanders,"%", "  - Clinton ", nh_map$Perce
nt_Clinton, "%")
```

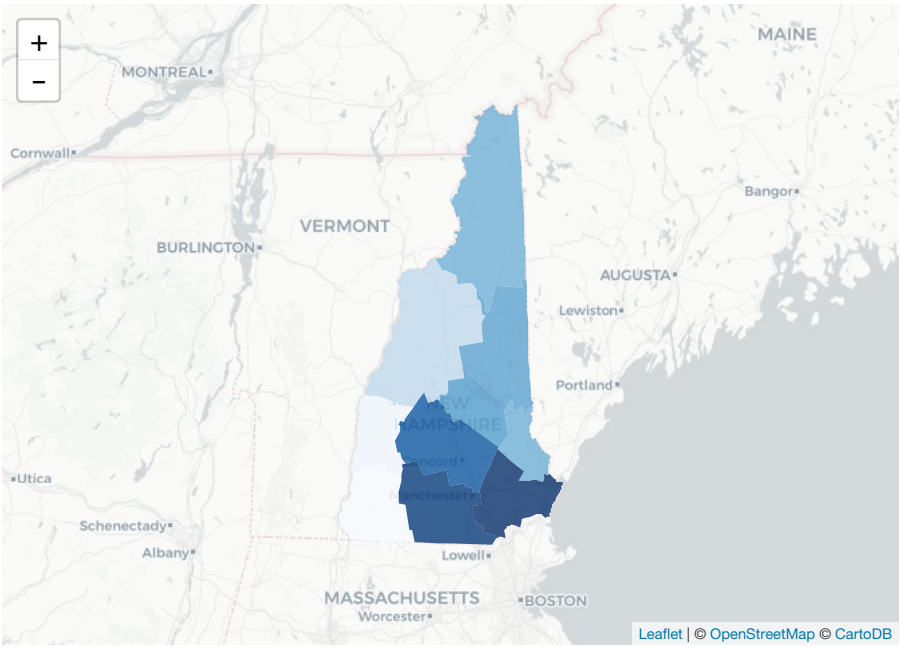
Interactive Mapping

Finally! Input this code:

```
# "leaflet(nh_map)" creates a map object using the dataset nh_map. "addProviderTiles" adds design, and the addPoly
gons() call does everything else: putting the shapes of the counties and adding the color to the map.

leaflet(nh_map) %>%
  addProviderTiles("CartoDB.Positron") %>%
  addPolygons(stroke=FALSE,
              smoothFactor = 0.2,
```


```
,0,0 +no_defs).
```



And we're done! How cool was that?!

Conclusion

We see that we are able to use static and interactive mapping to draw meaningful conclusions from datasets. For example, we were able to extract that Bernie Sanders earned the highest percentage margin of votes in the Sullivan and Cheshire counties of New Hampshire, whereas Hillary Clinton displayed her strongest performance in counties such as Rockingham and Hillsborough. Sanders won all the major cities in NH, swept the working-class population away, and his win was propelled by younger voter, according to this NYTimes article. Thus, these map visualization packages have helped us draw probing conclusions from Elections data in 2016, in New Hampshire. We can generalize these tools in order to assess every other state.

I hope you enjoyed accompanying me in this journey to discover more about mapping as a data visualization tool. I sincerely look forward to hearing you embark on similar ventures!

References

1. Martijn Tennekes' presentation
2. "Tmap in a nutshell" by Martijn Tennekes
3. New Hampshire's Secretary of State Office website
4. Zip archive of the cleaned and original New Hampshire election data, a shapefile of U.S. counties and states from the U.S. Census Bureau
5. US Census Bureau - page on cartographic boundaries
6. Shapefile of national counties zip
7. U.S. Census Bureau FIPS Codes for States
8. ColorBrewer
9. RStudio "leaflet" package
10. "Donald Trump and Bernie Sanders win in New Hampshire Primary"