Introduction to Spatial Data Analysis and Mapping in R

🌎🗺️

with sf and tmap

Angela Li
@CivicAngela
Center for Spatial Data Science, UChicago
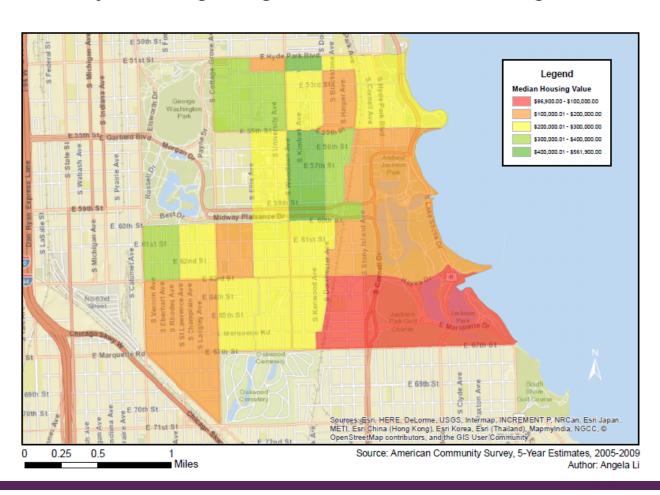Slides available at http://bit.ly/rladies-spatial

2018-04-28

Three parts to this talk:

1. Why spatial data in R?

2. A quick tutorial

3. Some thoughts on the future of #rspatial
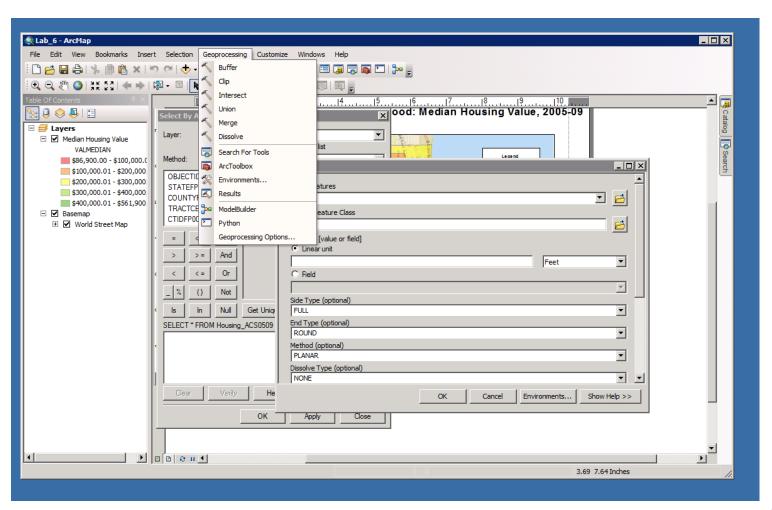
Why spatial data in R?

I had to make a map for a class

The University of Chicago Neighborhood: Median Housing Value, 2005-09

Legend
Median Housing Value
$86,900.00 - $100,000.00
$100,000.01 - $200,000.00
$200,000.01 - $300,000.00
$300,000.01 - $400,000.00
$400,000.01 - $561,900.00

Source: American Community Survey, 5-Year Estimates, 2005-2009
Author: Angela Li

Here's how you do it with a traditional GIS

This is fine, until

This is fine, until

👎 You want to remake your map with a slightly different set of data and have to redo
everything

This is fine, until

👎 You want to remake your map with a slightly different set of data and have to redo everything

👎 You want to make a bunch of maps quickly

This is fine, until

👎 You want to remake your map with a slightly different set of data and have to redo everything

👎 You want to make a bunch of maps quickly

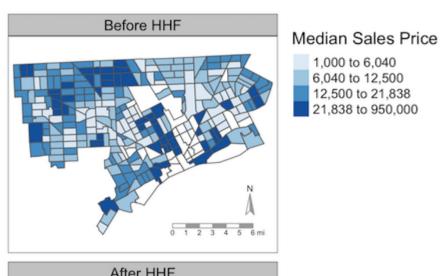👎 You forgot what buttons you even clicked to make the map

This is fine, until

👎 You want to remake your map with a slightly different set of data and have to redo everything

👎 You want to make a bunch of maps quickly

👎 You forgot what buttons you even clicked to make the map
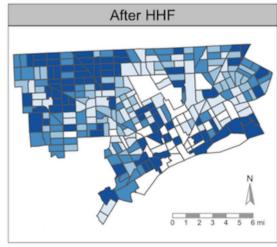
👎👎👎 **The GIS software crashes!**

Enter R

(and geographic data science)

A map from my thesis

And the code used to produce it

```r
library(tidyverse)
library(sf)
library(tmap)

sales <- read_csv("output/sales-tidy.csv")
tracts <- st_read("data/orig/shapefiles/detroit_tracts.shp")
tracts <- rename(tracts, tract = GEOID)

sales <- sales %>%
  right_join(tracts, ., by = "tract")

med_sales_map <- tm_shape(sales, unit = "mi") +
  tm_fill("med_price", palette = "Blues", breaks = quantile(a$med_price), title =
  tm_facets("after_hhf") +
  tm_shape(tracts) +
  tm_borders() +
  tm_compass(fontsize = 0.6, color.dark = "dark grey") +
  tm_scale_bar(color.dark = "dark grey")

save_tmap(med_sales_map, "doc/figs/med_sales_map.png")
```

Not much code = pretty good results

Thanks, sf and tmap!

A quick tutorial

Getting started

Install the `sf` and `tmap` packages.

```
install.packages("sf")
install.packages("tmap")
```

- `sf` stores spatial data as (tidyverse-friendly!) dataframes
  - `sp` is the original way to store spatial data in R, but it doesn't use dataframes
  - Many spatial statistics and mapping packages still rely on `sp`, so you'll probably encounter a `SpatialPolygonsDataFrame` at some point.
  - No worries, you can convert from `sf` to `sp` and vice versa pretty easily
- `tmap` provides a quick way to make useful thematic maps and works directly with spatial objects
  - There are a bunch of other packages you can use to make interactive maps (`mapview`, `leaflet`, `ggplot2`, `shiny`), which I won't go into today
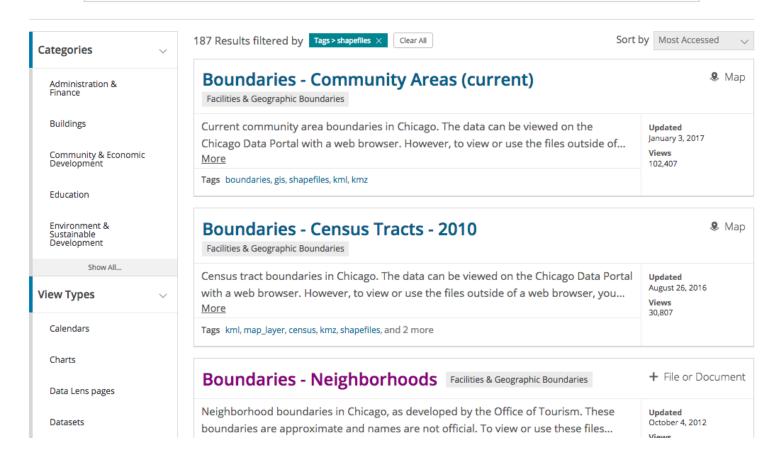
Get some data

- You're looking for "shapefiles" but data with XY coordinates works too
- Many packages have been developed to acquire spatial data:
  - `spData`
  - `tidycensus`
  - `usaboundaries`
  - `osmdata`
  - etc.
- If you have address data, you can geocode (translate addresses to latitude and longitude) with the `opencage` package, which I won't discuss today
- Check out all of these spatial packages later!

Let's download some data

Support your local open data portal

You've downloaded the data

What the heck are all of these files??

⬆ ..

☐ 🗋 Neighborhoods_2012b.dbf
☐ 🗋 Neighborhoods_2012b.prj
☐ 🗋 Neighborhoods_2012b.sbn
☐ 🗋 Neighborhoods_2012b.sbx
☐ 🗋 Neighborhoods_2012b.shp
☐ 🗋 Neighborhoods_2012b.shx

In general:

- .shp is the actual shape ("feature geometry") of the data
- .dbf represents the attributes associated with each shape
- .prj tells you how 3-D coordinates are "projected" into a 2-D map
- .sbn, .sbx, .shx are indexes that make it easier to work quickly with spatial data

FYI: spatial data tends to be BIG (because you have to store all the info about how to make the shapes!)

Make your first map (1)

```r
# Load package
library(sf)
```

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

```r
# Read in shapefile
chi <- st_read("data/Neighborhoods_2012b.shp")
```

```
## Reading layer `Neighborhoods_2012b' from data source `/Users/angela/Desktop/R-Projects/Te
## Simple feature collection with 98 features and 4 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 1091131 ymin: 1813892 xmax: 1205199 ymax: 1951669
## epsg (SRID):    NA
## proj4string:    +proj=tmerc +lat_0=36.66666666666666 +lon_0=-88.33333333333333 +k=0.99997
```

Make your first map (2)

```
# Map it using base R: just shape outlines
plot(st_geometry(chi))
```
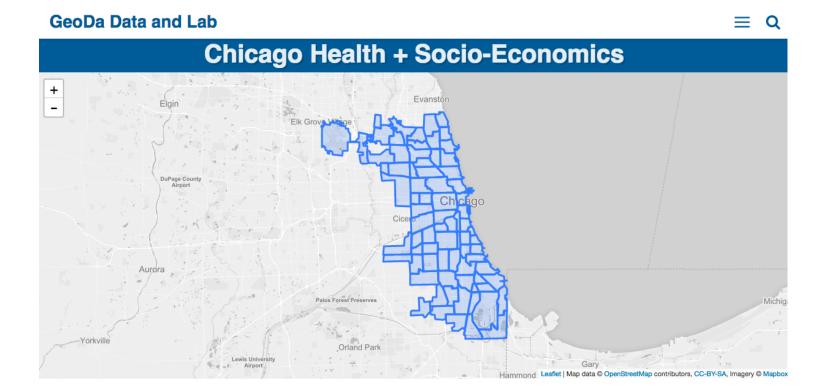
Make your first map (3)

```
# This maps all the attributes
plot(chi)
```

Get more interesting data

Lots of great cleaned datasets at my research center's website to play with.

https://geodacenter.github.io/data-and-lab/

Chicago Health + Socio-Economics

DOWNLOAD DATA

Public health and socio-economic indicators for the 77 community areas of Chicago, IL, 2014.

- Variables = 86

- Observations = 77

- Years = 2012-2014

Variable	Description	Source
Pop2012	Population in 2012	American Community Survey 2014
Pop2014	Population in 2014	American Community Survey 2014
PopM	Male Population in 2014	American Community Survey 2014
PopF	Female population in 2014	American Community Survey 2014
Under5	Population age 0-5 in 2014	American Community Survey 2014
Under18	Population age 0-18 in 2014	American Community Survey 2014
Over21	Population over age 21 in 2014	American Community Survey 2014
Over65	Population over age 65 in 2014	American Community Survey 2014
Wht14	White population in 2014	American Community Survey 2014
Blk14	Black population in 2014	American Community Survey 2014
AI14	American Indian population in 2014	American Community Survey 2014
AS14	Asian population in 2014	American Community Survey 2014
NHP14	Native Hawaiian population in 2014	American Community Survey 2014
Oth14	Population of other races in 2014	American Community Survey 2014
Hisp14	Hispanic population in 2014	American Community Survey 2014

Make a second map (1)

```
chi2 <- st_read("data/ComArea_ACS14_f.shp")
```

```
## Reading layer `ComArea_ACS14_f' from data source `/Users/angela/Desktop/R-Projects/Teachi
## Simple feature collection with 77 features and 86 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -87.94011 ymin: 41.64454 xmax: -87.52414 ymax: 42.02304
## epsg (SRID):    4326
## proj4string:    +proj=longlat +ellps=WGS84 +no_defs
```
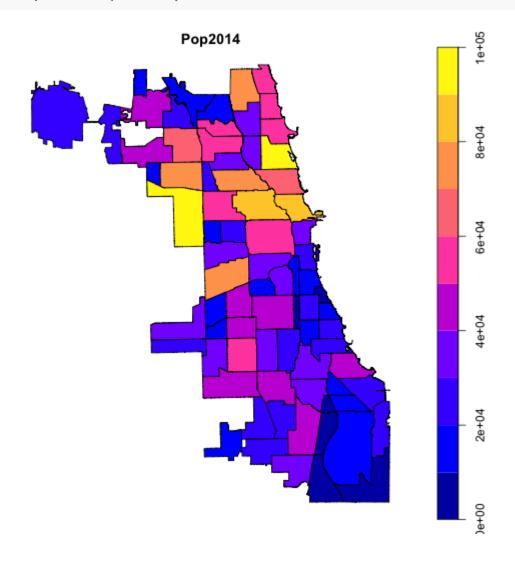
For reference:

- `geometry type` describes the basic structure of the spatial data. You could have points, polygons, lines, and more.
- `bbox` gives the bounding box for the data, and can be used to crop other layers when you make a map.
- `epsg (SRID)` is a special code that indicates what projection is being used. When in doubt, `4326` is a good one.
- `proj4string` refers to the same thing as the EPSG code. If the string starts with `+proj=longlat`, that means your data is **unprojected**.
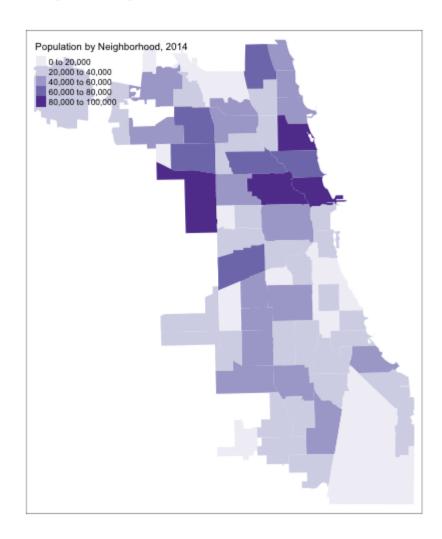
Make a second map (2)

Let's make a choropleth map of population by neighborhood!

```
# Check what variables we have
names(chi2)
```

```
##  [1] "ComAreaID"  "community"  "TRACTCnt"   "shape_area" "shape_len"
##  [6] "Pop2012"    "Pop2014"    "PopChng"    "PopM"       "PopMP"
## [11] "PopF"       "PopFP"      "Under5"     "Under5P"    "Under18"
## [16] "Under18P"   "Over18"     "Over18P"    "Over21"     "Over21P"
## [21] "Over65"     "Over65P"    "Wht14"      "Wht14P"     "Blk14"
## [26] "Blk14P"     "AI14"       "AI14P"      "AS14"       "AS14P"
## [31] "NHP14"      "NHP14P"     "Oth14"      "Oth14P"     "Hisp14"
## [36] "Hisp14P"    "Property_C" "PropCrRt"   "Violent_C"  "VlntCrRt"
## [41] "PerCInc14"  "PPop14"     "Pov14"      "field_37"   "ChldPov14"
## [46] "NoHS14"     "HSGrad14"   "SmClg14"    "ClgGrad14"  "LaborFrc"
## [51] "Unemp14"    "Pov50"      "Pov50P"     "Pov125"     "Pov125P"
## [56] "Pov150"     "Pov150P"    "Pov185"     "Pov185P"    "Pov200"
## [61] "Pov200P"    "COIave"     "HISave"     "SESave"     "Hlitave"
## [66] "BirthRate"  "FertRate"   "LoBirthR"   "PrenScrn"   "PretBrth"
## [71] "TeenBirth"  "Assault"    "BrstCancr"  "CancerAll"  "Colorect"
## [76] "DiabetM"    "FirearmM"   "InfntMR"    "LungCancer" "ProstateC"
## [81] "Stroke"     "ChlBLLS"    "ChlLeadP"   "GonorrF"    "GonorrM"
## [86] "Tuberc"     "geometry"
```

```
# Map population by neighborhood
plot(chi2["Pop2014"])
```


Pop2014

Use tmap to make a prettier map

```
library(tmap)
tm_shape(chi2) +
  tm_fill("Pop2014", palette = "Purples",
          title = "Population by Neighborhood, 2014")
```

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


Population by Neighborhood, 2014

0 to 20,000
20,000 to 40,000
40,000 to 60,000
60,000 to 80,000
80,000 to 100,000

Let's do some spatial analysis!

How ar

e grocery stores and population related?

Time to add a point layer with locations of grocery stores.

```
groceries <- st_read("data/groceries.shp")
```

```
## Reading layer `groceries' from data source `/Users/angela/Desktop/R-Projects/Teaching/rla
## Simple feature collection with 149 features and 14 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 1124188 ymin: 1826196 xmax: 1201803 ymax: 1950151
## epsg (SRID):    NA
## proj4string:    +proj=tmerc +lat_0=36.66666666666666 +lon_0=-88.33333333333333 +k=0.99997
```

Note that this is a `POINT` object, and that it has a projection: `+proj=tmerc` (Transverse Mercator). If we want to plot this in the same map as the neighborhood boundaries, we will need to make sure both files have the **same projection**.

⭐ This is a key source of frustration when working with spatial data. If some layers aren't showing up when you make a map, check that they all have the same projection! ⭐
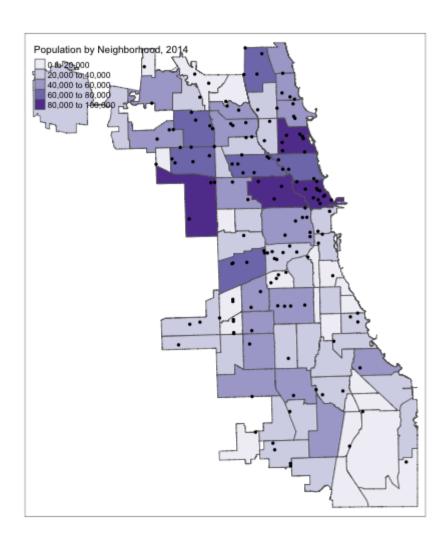
Project the neighborhood data

You generally project the data that has the `+proj=longlat` string, because it is initially **unprojected**.

```r
# Get the CRS (coordinate reference system) of the groceries point data
groceries_crs <- st_crs(groceries)

# Project the neighborhood boundaries
chi2 <- st_transform(chi2, groceries_crs)
```

Plot population and grocery stores

```
# Plot both
tm_shape(chi2) +
  tm_borders() +
  tm_fill("Pop2014", palette = "Purples",
          title = "Population by Neighborhood, 2014") +
  tm_shape(groceries) +
  tm_dots(title = "Groceries", size = 0.1, col = "black")
```

Population by Neighborhood, 2014

0 to 20,000
20,000 to 40,000
40,000 to 60,000
60,000 to 80,000
80,000 to 100,000

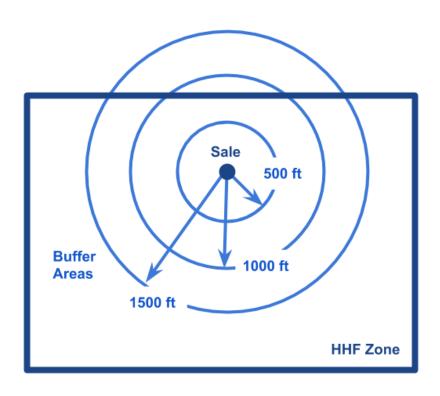You can also use dplyr to perform analysis!

Which neighborhoods in Chicago have the most grocery stores?

```
library(dplyr)

chi2 %>%
  st_join(groceries, .) %>%
  group_by(community) %>%
  tally() %>%
  arrange(desc(n))
```

```
## Simple feature collection with 55 features and 2 fields
## geometry type:  GEOMETRY
## dimension:      XY
## bbox:           xmin: 1124188 ymin: 1826196 xmax: 1201803 ymax: 1950151
## epsg (SRID):    NA
## proj4string:    +proj=tmerc +lat_0=36.66666666666666 +lon_0=-88.33333333333333 +k=0.99997
## # A tibble: 55 x 3
##    community            n                   geometry
##    <fct>            <int> <sf_geometry [US_survey_foot]>
##  1 NEAR NORTH SIDE     12 MULTIPOINT (1169391 1910436...
##  2 LOGAN SQUARE         8 MULTIPOINT (1150226 1915696...
##  3 LAKE VIEW            7 MULTIPOINT (1165054 1922112...
##  4 LOWER WEST SIDE      7 MULTIPOINT (1160692 1889319...
##  5 NEAR WEST SIDE       7 MULTIPOINT (1165889 1894599...
##  6 BELMONT CRAGIN       6 MULTIPOINT (1132379 1917230...
##  7 PORTAGE PARK         5 MULTIPOINT (1138161 1931033...
```

More advanced spatial analysis involves buffers, distance, intersections, etc.
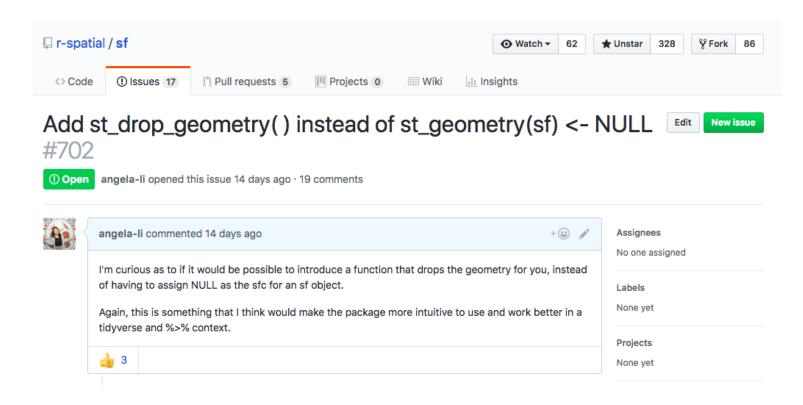
Code from my thesis:

```
get_point_counts_in_buffer <- function(points_to_buffer,
                                       points_to_intersect,
                                       buffer_size = 500) {
  number_points_within_buffer <- points_to_buffer %>%
    st_buffer(buffer_size) %>%
    st_contains(points_to_intersect) %>%
    map_dbl(length) %>%
    tibble(pts_in_buffer = .)

  return(number_points_within_buffer)
}
```

And what it did:

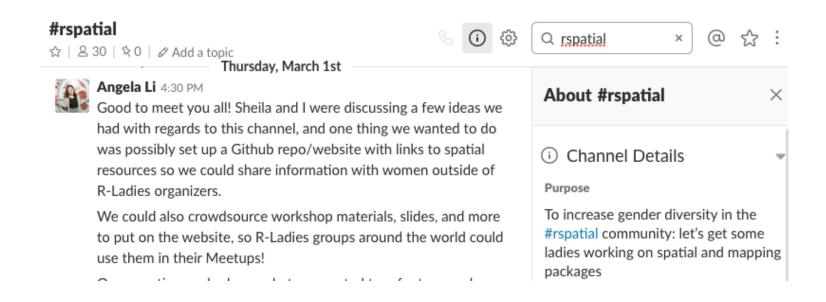Now a few words on the future of #rspatial

IT NEEDS MORE WOMEN

1. Submitting issues

2. Building communities

3. Getting our voices out there!

Stay tuned for more...

Thanks!

Slides created via the R package xaringan.