Lecture 5:
Maps 1 of 2

February 24, 2020

Overview

Course Administration

1. Comments in 2 weeks on charts
2. Beginning of a 3-lecture deviation from charts
    - maps 1
    - functions and stories
    - maps 2
3. Sign up for consultations!
    - sign up for slots April 7, 9 or 10
    - no class meeting April 13
4. Next class: come prepared to work on your policy brief storyline

Next Week's Assignment

Find a descriptive or choropleth map. Post link to google sheet by Wednesday noon.

Finder	Commenter
Janice W.	Reeve J.
Emily H.	Kaila C.
Tereese S.	Connor D.

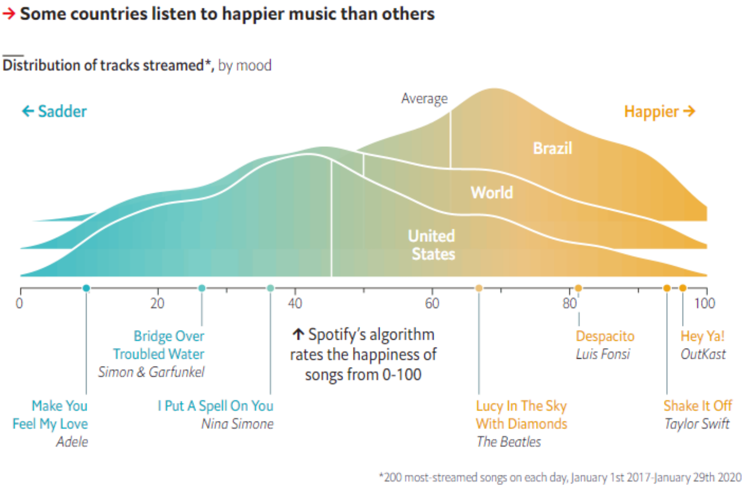
This Week's Good Bad and Ugly

Finder	Commenter
Boyd G.	Janice W.
Didem B.	Betsy K.
Dallas C.	Lindsay R.

Boyds's Example from *The Economist*

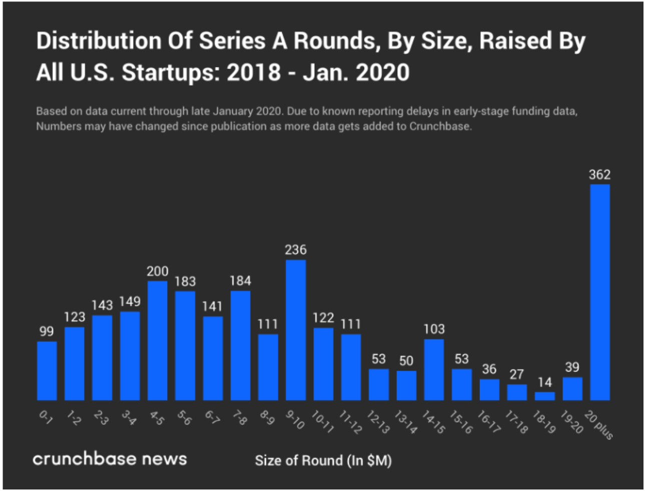
Data from Spotify suggest that listeners are gloomiest in February

Around the world, the most popular tunes this month will be depressing ones

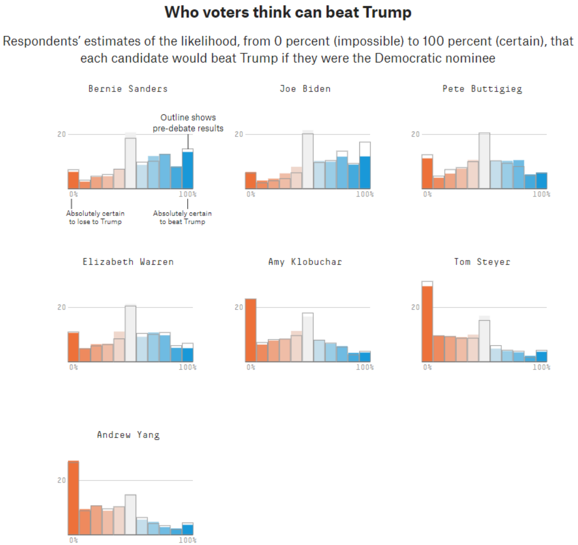


Didem's Example from *Cruchbase*

"The Distribution of Series A Deal Size in the US"



Basia's Example from *FiveThirtyEight*



What and Why of Maps

Today

1. What is a map?
2. Why maps?
3. When do maps deceive?
4. Save for next time: Choropleth maps and dot density maps

1. What is a Map?

- "scale model of reality" (Monmonier)
- "almost always smaller" than reality

1. What is a Map?

- "scale model of reality" (Monmonier)
- "almost always smaller" than reality
- in distilling reality, there are three key choices

1. What is a Map?

- “scale model of reality” (Monmonier)
- “almost always smaller” than reality
- in distilling reality, there are three key choices
    1. scale
    2. projection
    3. symbolization

Projection

- We want to show both
  - equivalence: size proportional to physical size
  - conformality: shape proportional to true shape

Projection

- We want to show both
  - equivalence: size proportional to physical size
  - conformality: shape proportional to true shape
- But you cannot do both!
- When does this matter?

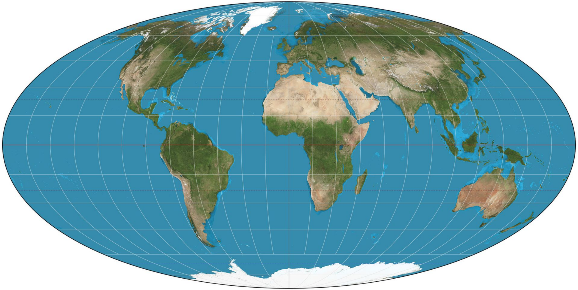
Projection

- We want to show both
  - equivalence: size proportional to physical size
  - conformality: shape proportional to true shape
- But you cannot do both!
- When does this matter?
  - This matters for maps of the world
  - It is practically irrelevant for a map of DC
  - For small areas, we care about precision of distance
  - Frequently use a UTM (Universal Transverse Meractor) projection: units in meters

Rules of Thumb for Projections for Medium Areas

- Monmonier (p. 45) suggests for US either
  - Albers equal-area conic
  - Lambert conformal conic
- However, most maps you use should come with a projection defined

An Equal-Area Projection

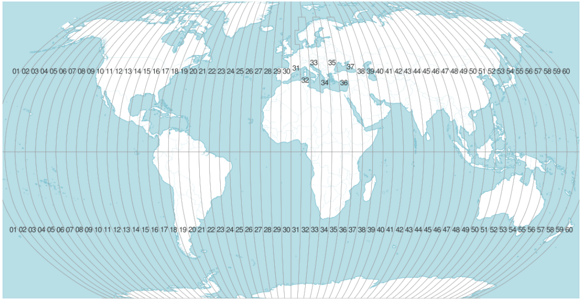


Thanks, Wikipedia.

The USA Four Ways

Thanks to Michael Corey.

UTM Zones



For small areas, use UTM projection if you need to calculate distances. Each number is a zone.

Thanks to Michael Corey.

2. Why Maps?

- Use a map when you want to show a **spatial** relationship
- Don't use a map if you want to compare geographic units

When is Space Important?

1. To show relationship between two geographic things. Examples?

When is Space Important?

1. To show relationship between two geographic things. Examples?
    - metro stops relative to average home prices
    - population density relative to the equator
2. To show a geographic pattern in an outcome. Examples?

When is Space Important?

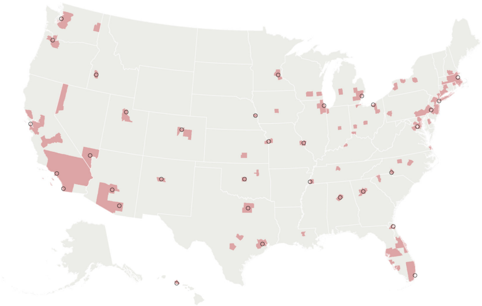
1. To show relationship between two geographic things. Examples?
   - metro stops relative to average home prices
   - population density relative to the equator
2. To show a geographic pattern in an outcome. Examples?
   - voting outcomes correlated over space
   - geographic features that change smoothly and sharply over space

Don't use a map if you can do something simpler!

3. Why Avoid Maps?

- They add complexity
- Geographic unit size infrequently related to importance
    - but remember that size indicates value
    - problematic!
- Examples?

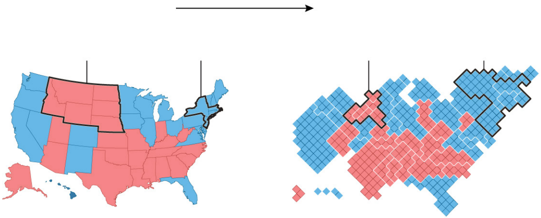
Red and Grey Areas Have About the Same Number of Votes Cast in 2012



With many thanks to the Washington Post

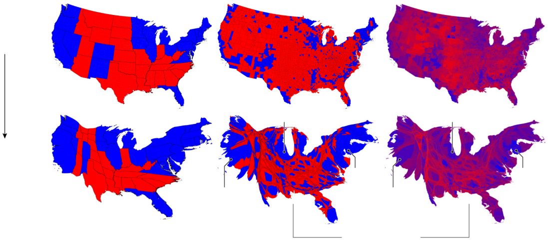
One Possible Solution

- A "cartogram" sizes locations by something: votes or people or electoral votes
- Five red midwestern states correspond to red block
- Mid-Atlantic corresponds to blue block

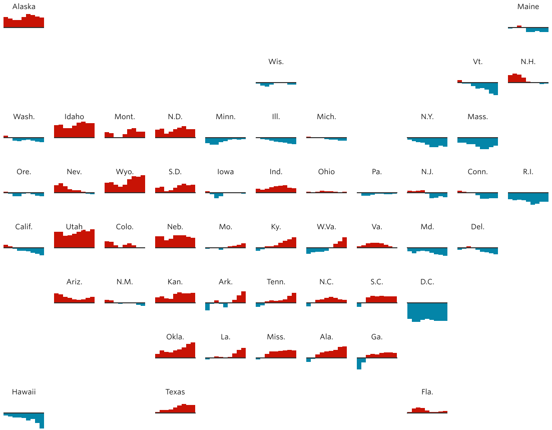


Another Possible Solution

- Thanks to U of Michigan physicist Newman
- Columns are state winner, county winner, county shaded by popular vote share
- Top is real map, bottom is cartogram
- Leftmost sized by electoral votes, others by votes cast



And a Quasi Map



Thanks to the Wall Street Journal, here.

How Do Computers Make Maps?

Maps Have

- Units defined by coordinates in space
- Data for each unit

Examples of a map unit of observation, please!

Digital Maps

- A map is a representation of space
- A digital map is a file that tells a computer how to do this
- There are many formats, but we'll focus on shapefiles
- Shapefiles are a ArcInfo format, but can be read in R

Three Major Types of Shapes for Maps

1. points
2. lines
3. polygons

Points in Space

- location 1: (x, y)
- location 2: (x, y)
- location 3: (x, y)

What would you represent with points?

A Points Dataframe Example

LibID	X	Y	Name	Books
Ana	38.866	-76.980	Anacostia	500
CV	38.889	-76.932	Capitol View	501
Gtn	38.913	-77.068	Georgetown	499

Lines in Space

- location 1: $(x_1, y_1), (x_2, y_2)$
- location 2: $(x_1, y_1), (x_2, y_2)$
- location 3: $(x_1, y_1), (x_2, y_2)$

What would you represent with lines?

A Lines Dataframe Example

Int	X1	Y1	X2	Y2	Name	Condition
495	45	-62	26	-62	I495W	good
695	23	-50	25	-50	I695S	poor
10	15	-23	18	-24	I10	excellent

Polygons in Space

- location 1: $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_1, y_1)$
- location 2: $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_1, y_1)$
- location 3: $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_1, y_1)$

Note that last point is the same as the first point.[1]
What would you represent with polygons?

[1] Polygons can have holes; we can talk about this.

A Polygon Dataframe Example

Triangle	X1	Y1	X2	Y2	X3	Y3	X4	Y4
a	1	1	1	2	2	1	1	1
b	1	1	1	3	3	1	1	1

But Where Do the Points Go?

- A map file needs some instructions on what the points mean
- We are not drawing on a globe, so we need some way of taking true coordinates and making them flat: projection
- Map makers define coordinate systems so that everyone agrees on what $(x_1, y_1), (x_2, y_2)$ means
- Coordinate systems have a defined unit of measurement: meters, feet, decimal degrees
- There are two major types of systems
    1. geographic/global/spherical system: in latitude/longitude
    2. projected coordinate system: in terms of meters/feet/miles

Implications for Mapping

- You can't put maps with two different coordinate systems on top of each other
- Easier to calculate distances and areas with projected coordinate systems
- You can go from one projection to another, but **use the right command**
- Digital maps usually come with a projection defined

R

Today

A. `sf` package
B. Reading
C. Plotting
D. Projections
E. Spatially combining

A. sf Package

- a new package as of 2018
- works with tidyverse and ggplot
- use all the other commands you've used to date
- ok for all map data except rasters

```
library(ggplot2)
library(sf)
```

B.1. Reading a Shapefile

- there are many types of digitial maps
- the most common is a “shapefile”
- a proprietary format from ESRI
- most downloads come in this format

B.2. What is a Shapefile?

- shapefiles have 4 to 7 parts
- all have the same name and these extensions
  - .shp
  - .shx
  - .dbf
  - .prj
  - .xml
  - .cpg
- the first 3 are mandatory
- it's odd if you don't have a projection, but you can still draw a map

B.3. Read the shapefile

The key command is `st_read("FILENAME.MAP_EXTENSION")`

```
shp.df <- st_read("c:/stuff/map.shp")
```

B.3. Read the shapefile

The key command is `st_read("FILENAME.MAP_EXTENSION")`

```
shp.df <- st_read("c:/stuff/map.shp")
```

This new file

- works like a dataframe
- plus it has spatial information
- is called a "simple feature"

C.1 Plotting

Two main commands for plotting simple features in R

1. `plot()`
2. `ggplot()` using `geom_sf()`

Happily, `geom_sf()` works a lot like the other `geom_XXX()` commands you already know.

C.2. Example

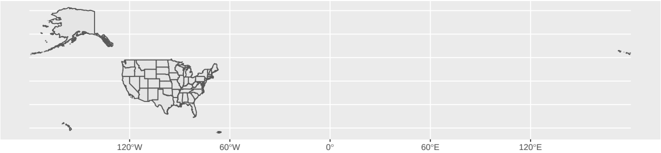
```
usmap <- st_read("H:/maps/united_states/census2010/states/gz_2010_us_040_0
```

```
## Reading layer `gz_2010_us_040_00_20m' from data source `H:\maps\united_
## Simple feature collection with 52 features and 5 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -179.1473 ymin: 17.88481 xmax: 179.7785 ymax: 71.
## epsg (SRID):    4269
## proj4string:    +proj=longlat +datum=NAD83 +no_defs
```

```
states <- ggplot() +
  geom_sf(data = usmap)
```

C.3. Example plot

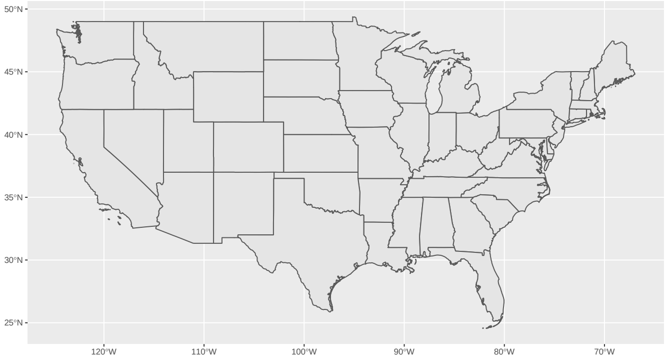
```
states
```



C.4. Just the Continental US

```r
# omit AK, HI, PR
usmap.cont <- usmap[which(!(usmap$STATE %in% c("02","15","72"))),]
cont.us <-
  ggplot() + geom_sf(data = usmap.cont)
```

C.4. Just the Continental US



D. Projections

- maps should have a projection
- to tell R where to put points in space
- these are viewable

D. Projections

- maps should have a projection
- to tell R where to put points in space
- these are viewable

```
st_crs(usmap)
```

```
## Coordinate Reference System:
##   EPSG: 4269
##   proj4string: "+proj=longlat +datum=NAD83 +no_defs"
```

E. Spatially combining

Questions you can answer with `st_intersection()`

- Which states are cities in?
  - points and polygons: should return points
- What share of national park land area (polygons) is in cities (polygons)?
  - polygons and polygons: should return polygons
- How many miles of roads (lines) are in the 3 western coastal states (polygons)?
  - lines and polygons: should return lines, then sum to state level

E.1 Example: Which states are cities in?

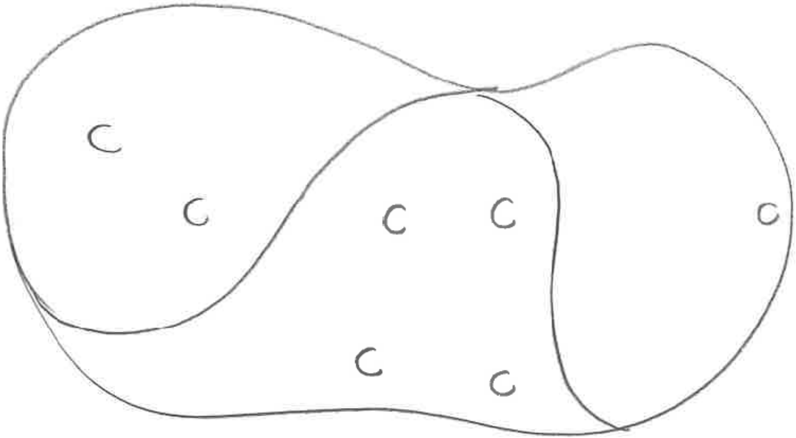


Figure 1:

E.1 Example: What share of national park land area is in cities?

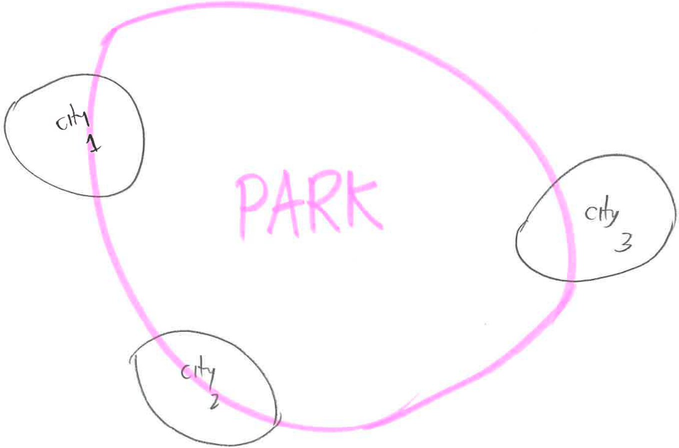


Figure 2:

E.1 Example: How many miles of roads in each state?

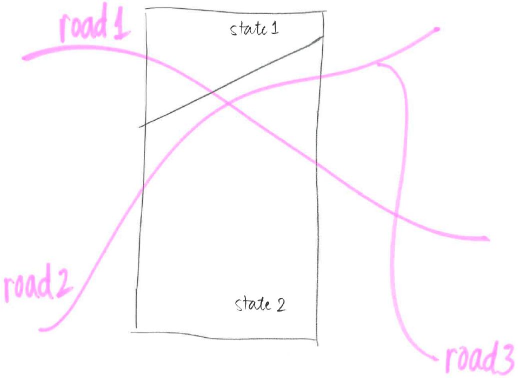


Figure 3:

E.2. How to do it

Use `st_intersection()`

commands

Don't confuse with `st_intersects()` which does the same thing but returns a matrix, not a simple feature.

E.3. Example

```
b0 = st_polygon(list(rbind(c(-1,-1),
                            c(1,-1),
                            c(1,1),
                            c(-1,1),
                            c(-1,-1))))
b1 = b0 + 2
b2 = b0 + c(-0.2, 2)
x = st_sfc(b0, b1, b2)
a0 = b0 * 0.8
a1 = a0 * 0.5 + c(2, 0.7)
a2 = a0 + 1
a3 = b0 * 0.5 + c(2, -0.5)
y = st_sfc(a0,a1,a2,a3)
```

Taken directly from sf vignette here.

E.3. Simple Feature `x`

```
x
```

```
## Geometry set for 3 features
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: -1.2 ymin: -1 xmax: 3 ymax: 3
## epsg (SRID):    NA
## proj4string:    NA
```

```
## POLYGON ((-1 -1, 1 -1, 1 1, -1 1, -1 -1))
```

```
## POLYGON ((1 1, 3 1, 3 3, 1 3, 1 1))
```

```
## POLYGON ((-1.2 1, 0.8 1, 0.8 3, -1.2 3, -1.2 1))
```

E.3. Simple Feature Y

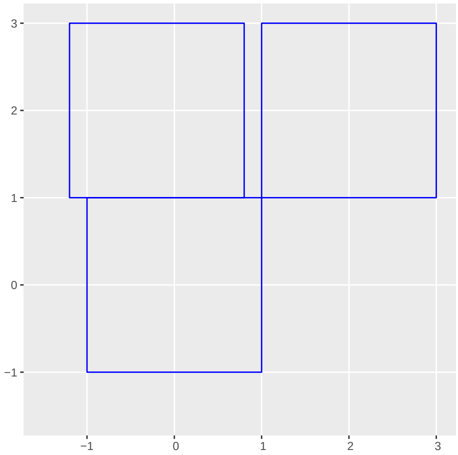
```
y
```

```
## Geometry set for 4 features
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: -0.8 ymin: -1 xmax: 2.5 ymax: 1.8
## epsg (SRID):    NA
## proj4string:    NA

## POLYGON ((-0.8 -0.8, 0.8 -0.8, 0.8 0.8, -0.8 0....

## POLYGON ((1.6 0.3, 2.4 0.3, 2.4 1.1, 1.6 1.1, 1...

## POLYGON ((0.2 0.2, 1.8 0.2, 1.8 1.8, 0.2 1.8, 0...

## POLYGON ((1.5 -1, 2.5 -1, 2.5 0, 1.5 0, 1.5 -1))
```

E.3. Plot X

```r
xplot <- ggplot() +
  geom_sf(data = x, color = "blue", fill = NA) +
  scale_x_continuous(limits = c(-1.5,3)) +
  scale_y_continuous(limits = c(-1.5,3))
```

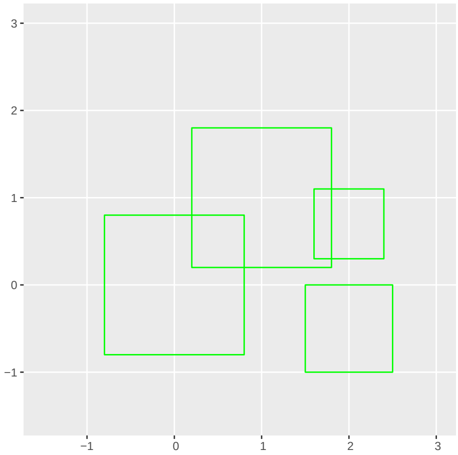
E.3. Plot $\mathtt{x}$



E.3. Plot Y

```r
yplot <- ggplot() +
  geom_sf(data = y, color = "green", fill = NA) +
  scale_x_continuous(limits = c(-1.5,3)) +
  scale_y_continuous(limits = c(-1.5,3))
```

E.3. Plot Y



E.4. Intersection

```r
xy <- st_intersection(x,y)
```

E.5. How the New Simple Feature Looks

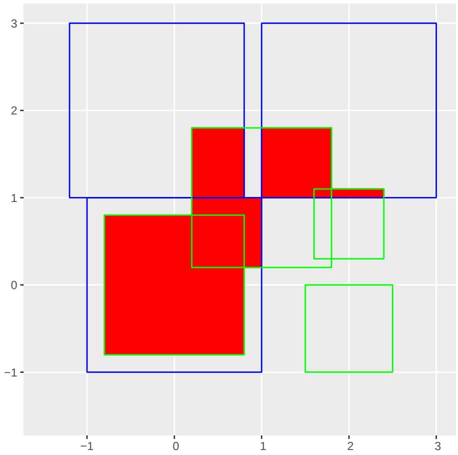
```
xy
```

```
## Geometry set for 5 features
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: -0.8 ymin: -0.8 xmax: 2.4 ymax: 1.8
## epsg (SRID):    NA
## proj4string:    NA

## POLYGON ((-0.8 -0.8, -0.8 0.8, 0.8 0.8, 0.8 -0....

## POLYGON ((2.4 1, 1.6 1, 1.6 1.1, 2.4 1.1, 2.4 1))

## POLYGON ((0.2 1, 1 1, 1 0.2, 0.2 0.2, 0.2 1))

## POLYGON ((1.8 1, 1 1, 1 1.8, 1.8 1.8, 1.8 1))

## POLYGON ((0.8 1.8, 0.8 1, 0.2 1, 0.2 1.8, 0.8 1
```

E.5 What the Picture Looks Like

```
xyplot <- ggplot() +
  geom_sf(data = xy, color = "red", fill = "red") +
  geom_sf(data = x, color = "blue", fill = NA) +
  geom_sf(data = y, color = "green", fill = NA) +
  scale_x_continuous(limits = c(-1.5,3)) +
  scale_y_continuous(limits = c(-1.5,3))
```

E.5 What the Picture Looks Like



Next Lecture

- Next class: come prepared to work on your policy brief storyline
- Read Knaflic, Chapters 7 and 8