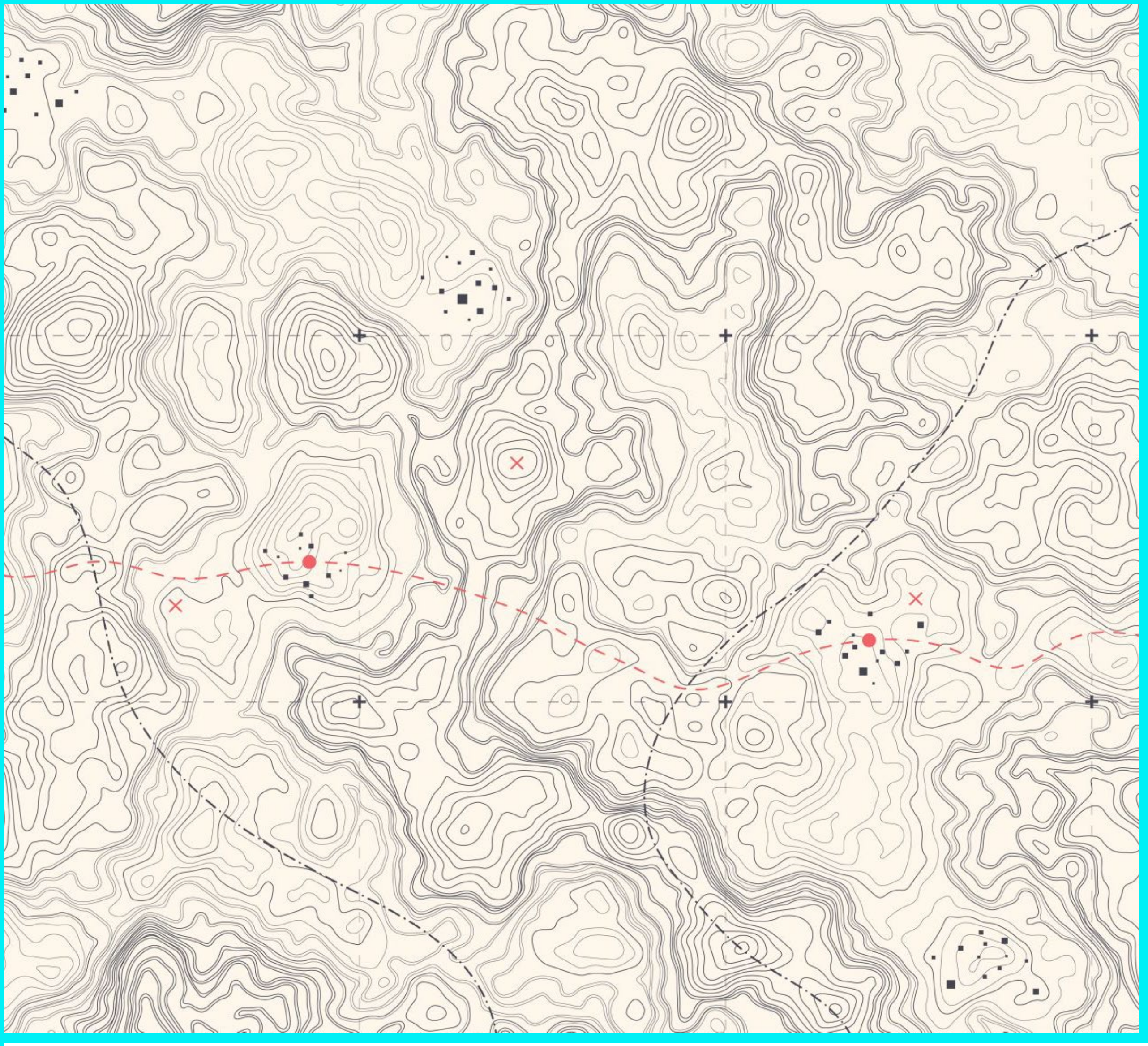
Working With Spatial Data in R!

A Brief Introduction

Michael France Nelson
Apr 13 - University of Massachusetts, Amherst

Outline



1. What the heck is R and why is it awesome?
2. Crash course in ggplot
3. Vector spatial data types in R
4. Plotting spatial data with ggplot
5. US Census data in R
6. Let's R!

Workshop Homepage

- You can find the R code that I used for all the figures in this presentation at:

https://tinyurl.com/m5p4yv2c

R and RStudio

What The Heck Is R???

"R is a language and environment for statistical computing and graphics."

"One of R's strengths is the ease with which well-designed publication-quality plots can be produced…"

R is free!!!

What The Heck Is RStudio???

*"RStudio is an integrated development environment (**IDE**) for R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management." – rstudio.com*

RStudio helps you organize your R projects and code.

What’s So Great About R???

It’s free, extensible, and open-source

- Free is great, right?
- R has millions of active users and tons of packages that can do just about anything you can think of.
- If the analysis you need isn’t covered in a package…write your own!

It’s reproducible

- Remembering (and communicating) a series of point-and-click commands is difficult.
- R scripting ensures that anyone (including you) can reproduce your work!

What’s The Downside?

So…what’s the catch?

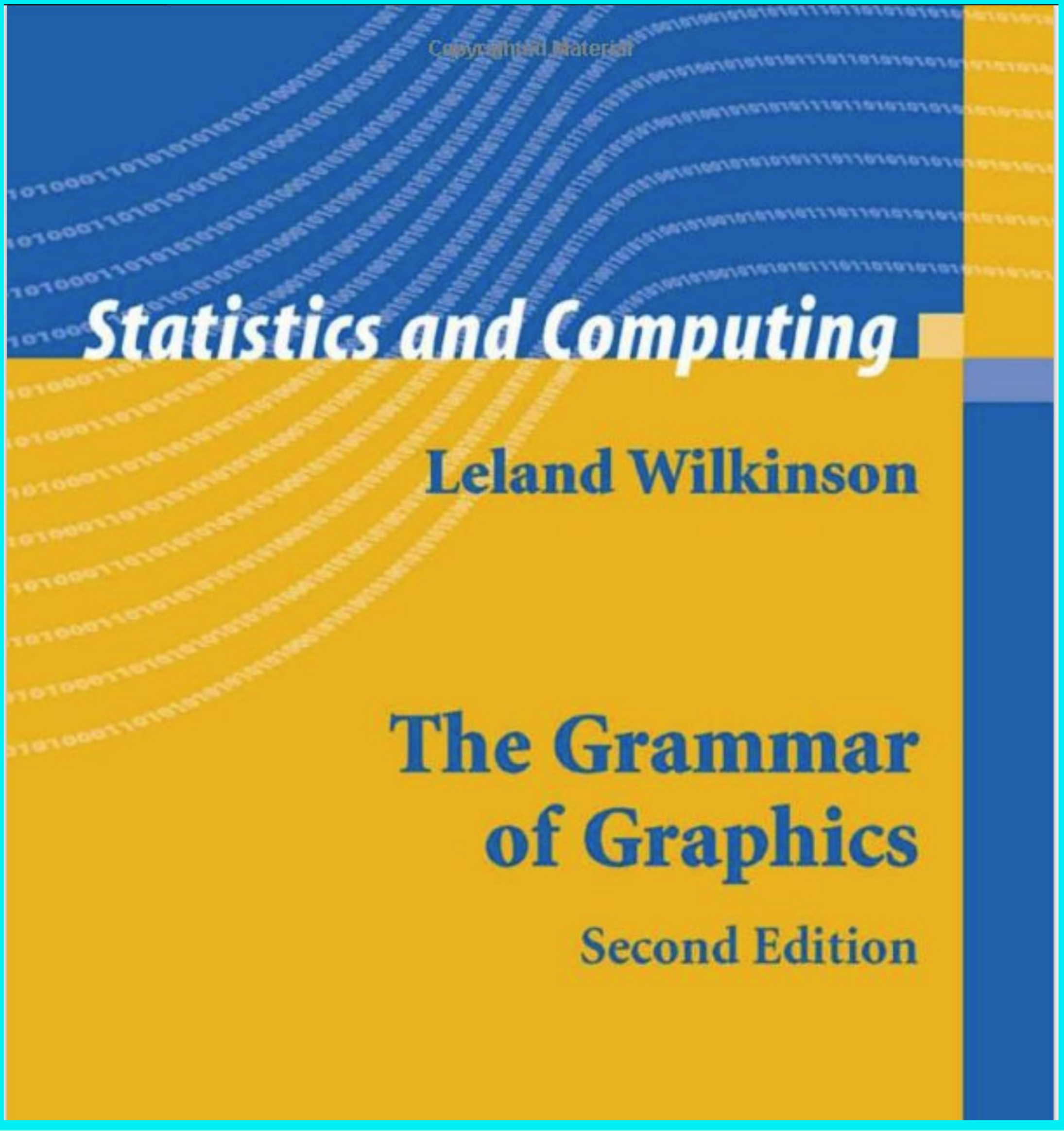
- R is a programming language, so it can seem intimidating if you’ve never coded before.
  - But…there are tons of great resources for learning
- Some tasks are more laborious in R than in Arc or QGIS
  - True, but many tasks are simpler

Crash Course in ggplot

Just the basics

The Grammar of Graphics

- A logical, layered approach to creating graphics.
- The Grammar of Graphics is a philosophy.
- Being intentional about graphic elements.
- Grammar of graphics is a great fit for spatial data.



Grammar of Graphics in R

- The Grammar of Graphics is a philosophy.
- It is *implemented* in R with the **ggplot2** package.

Grammar of Graphics in R

1. Data
2. Aesthetics
3. Geometry

ggplot: Data (Frames) In Row Format

Data in row-format is key to success with ggplot!

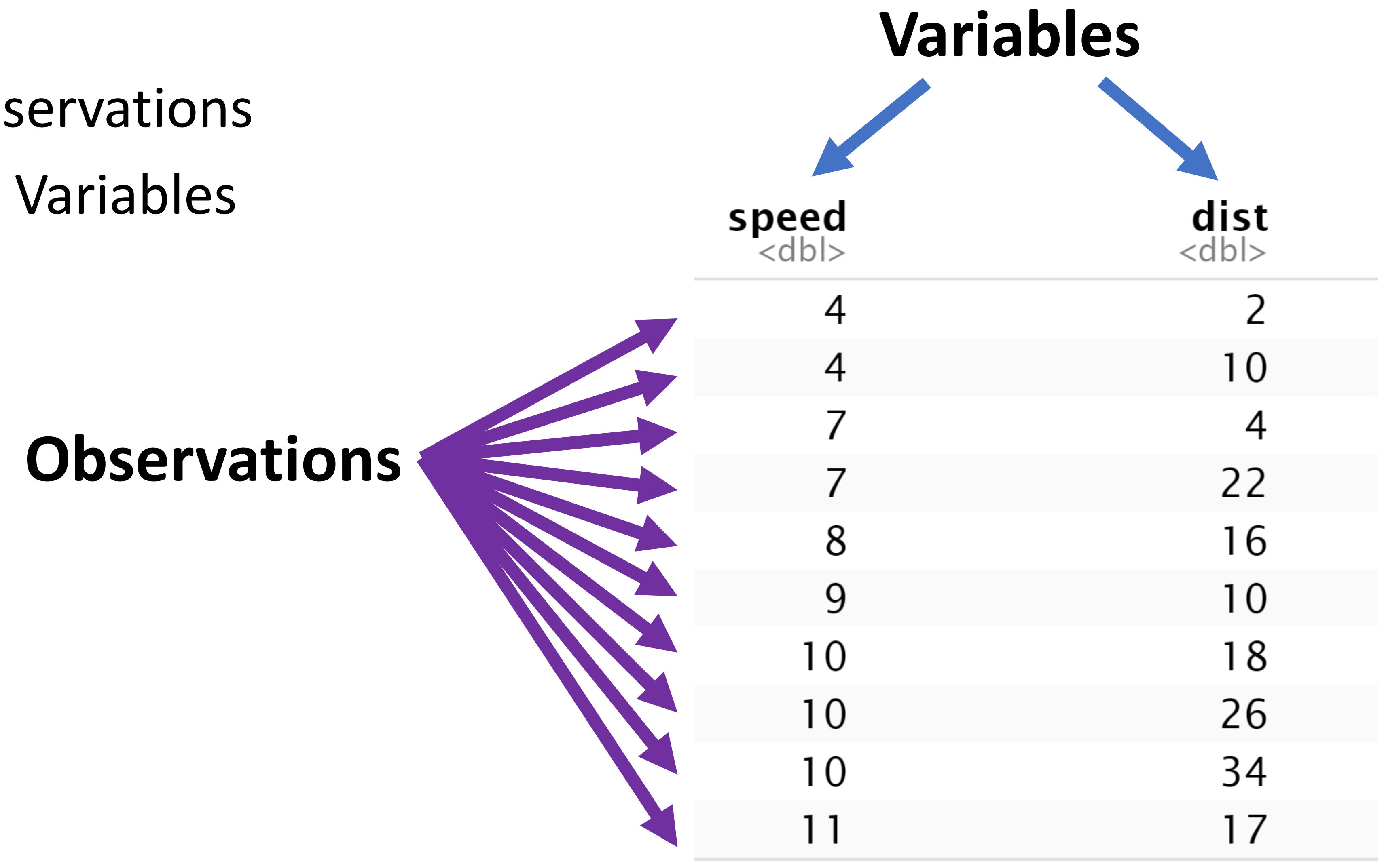
But… what is the row data paradigm?

The row data paradigm:

- Rows are observations, or features.
- Columns are attributes, i.e. variables.
- This sounds a lot like an attribute table in Arc GIS…

Simple Data Example: Cars

- Rows = Observations
- Columns = Variables



speed <dbl>	dist <dbl>
4	2
4	10
7	4
7	22
8	16
9	10
10	18
10	26
10	34
11	17

ggplot: Aesthetics

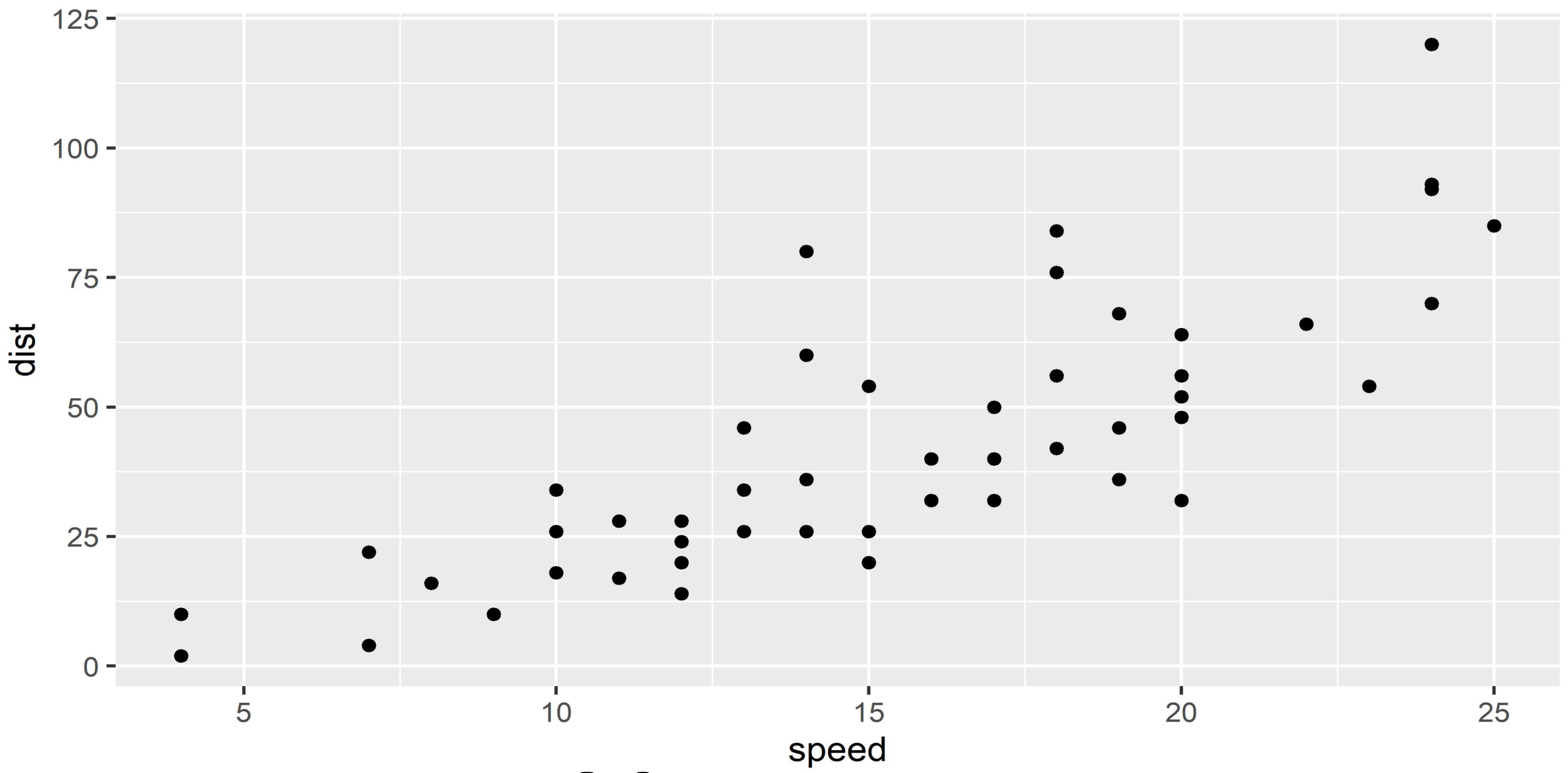
- In ggplot, aesthetics specify how variables are mapped to components of a plot. For example:
  - X- and Y- values
  - Groups
  - Color and fill
- Aesthetics are specified with the aes() function.

```
ggplot(cars, aes(x = speed, y = dist))
```

X speed <dbl>	Y dist <dbl>
4	2
4	10
7	4
7	22
8	16
9	10
10	18
10	26
10	34
11	17

Cars Scatterplot: X- and Y- Aesthetics

Y

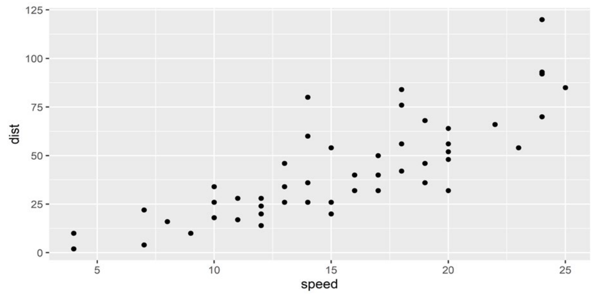


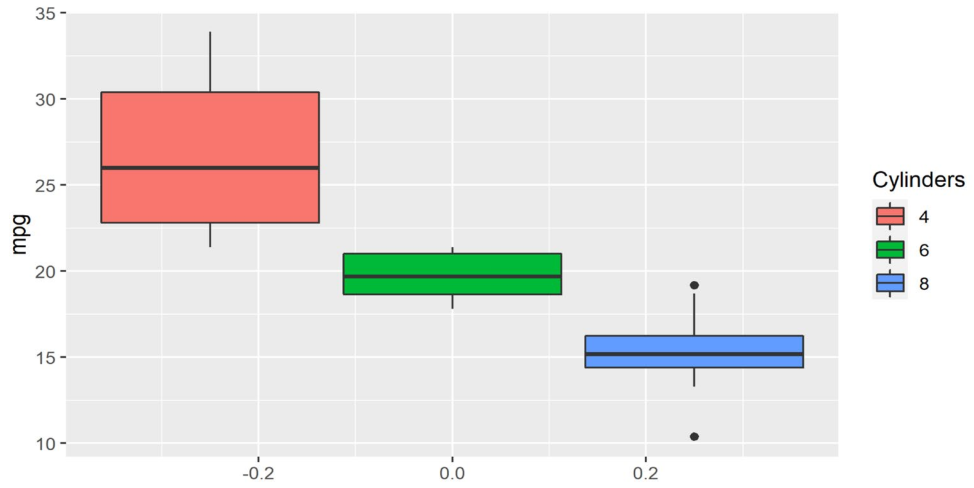
X

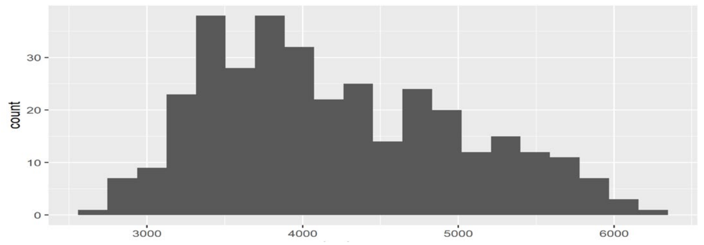
ggplot: Geometries

Geometries specify the type of plot. For example:

- Scatterplots
- Boxplots
- Histograms





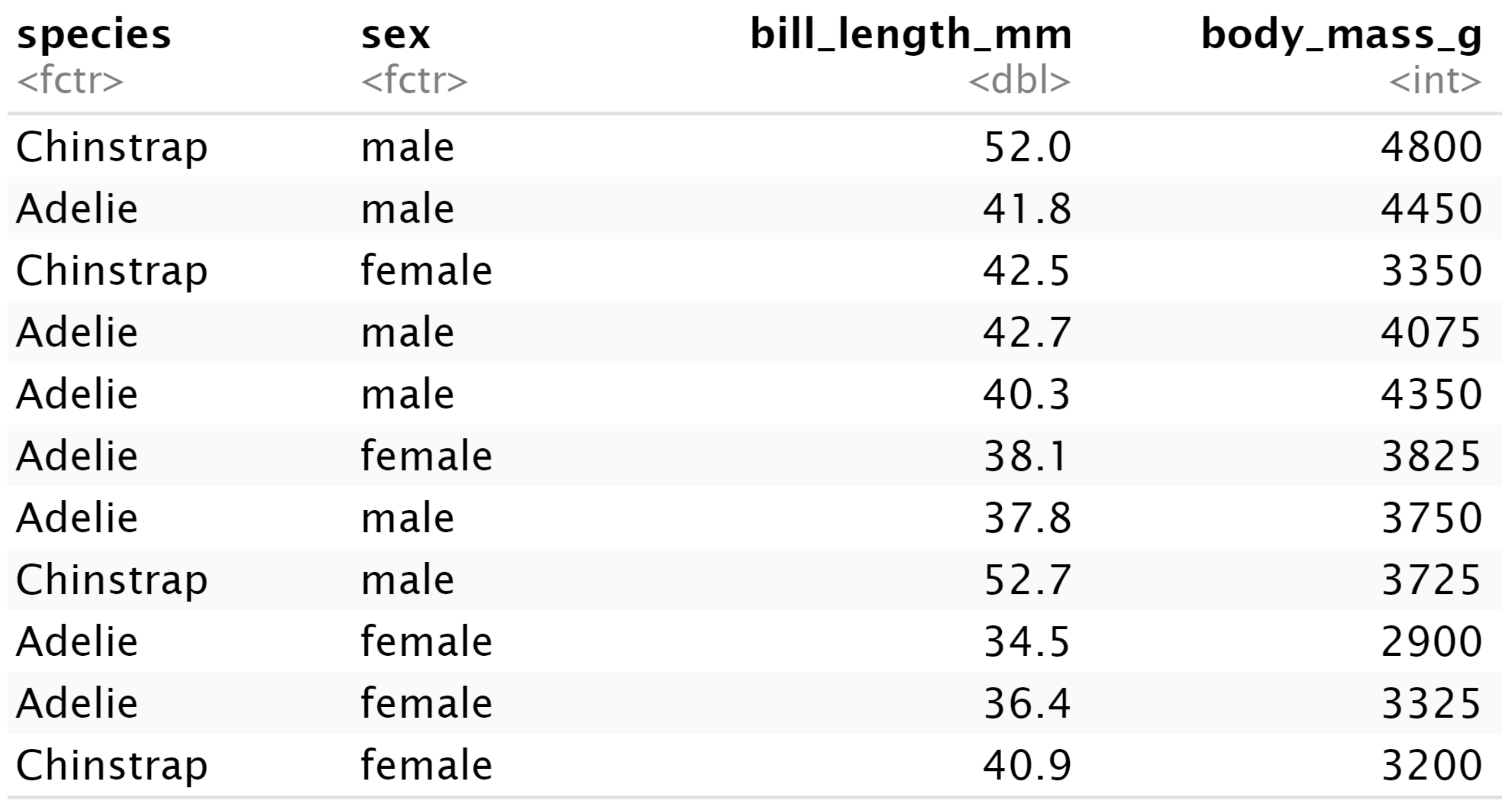


There are tons of different geometries, each one recognizes a different set of one or more aesthetics.

Example Scatterplot: Penguins Data

The Palmer penguins dataset contains lots of variables, we'll concentrate on:

- Species
- Sex
- Bill Length
- Body Mass



species <fctr>	sex <fctr>	bill_length_mm <dbl>	body_mass_g <int>
Chinstrap	male	52.0	4800
Adelie	male	41.8	4450
Chinstrap	female	42.5	3350
Adelie	male	42.7	4075
Adelie	male	40.3	4350
Adelie	female	38.1	3825
Adelie	male	37.8	3750
Chinstrap	male	52.7	3725
Adelie	female	34.5	2900
Adelie	female	36.4	3325
Chinstrap	female	40.9	3200

Scatterplot 1: Body Mass and Bill Length

1: Data

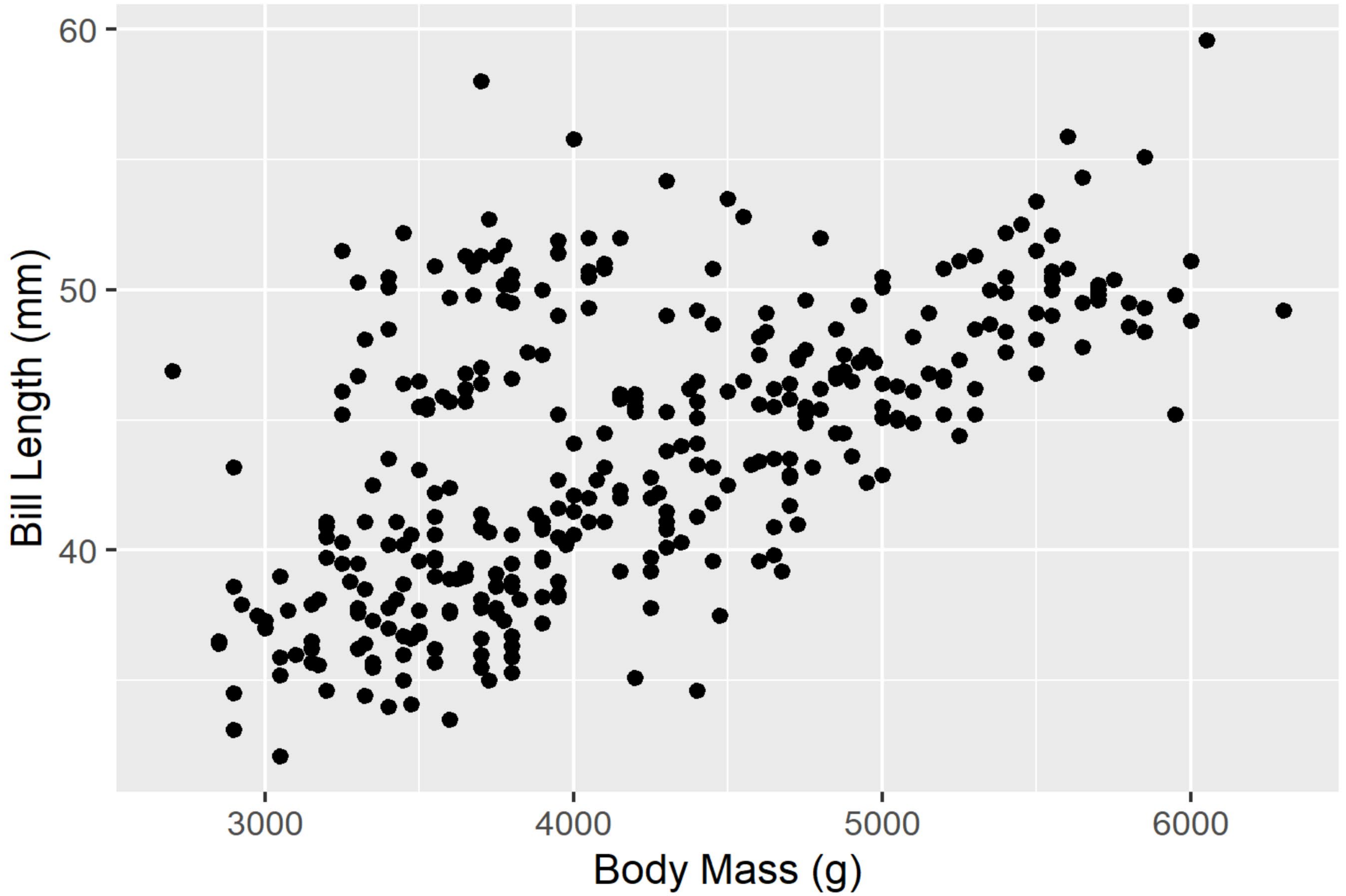
- Penguins

2: Aesthetics

- X: Body Mass
- Y: Bill Length

3: Geometry

- Points (scatterplot)



Scatterplot 1: Body Mass and Bill Length

1: Data

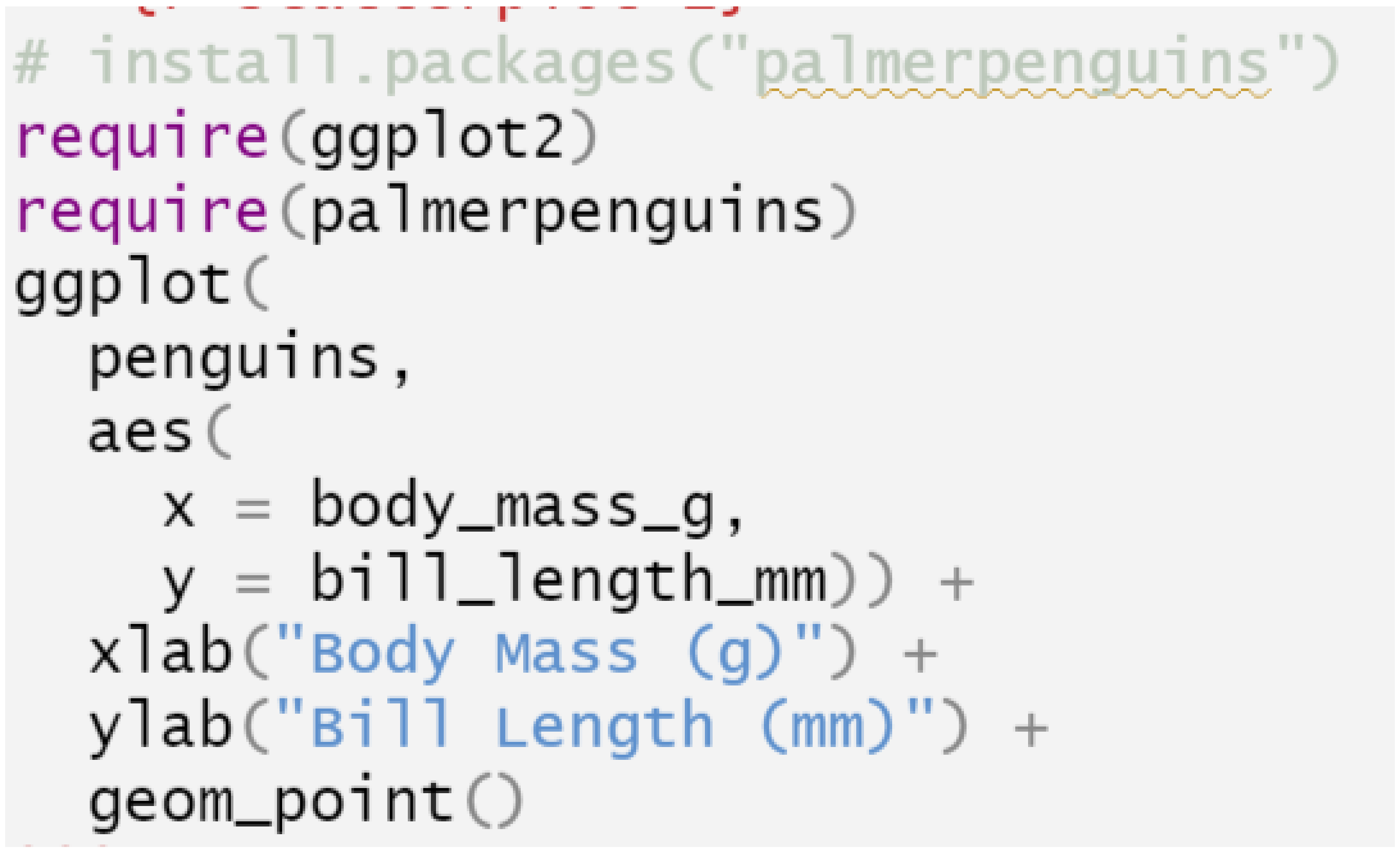
- Penguins

2: Aesthetics

- X: Body Mass
- Y: Bill Length

3: Geometry

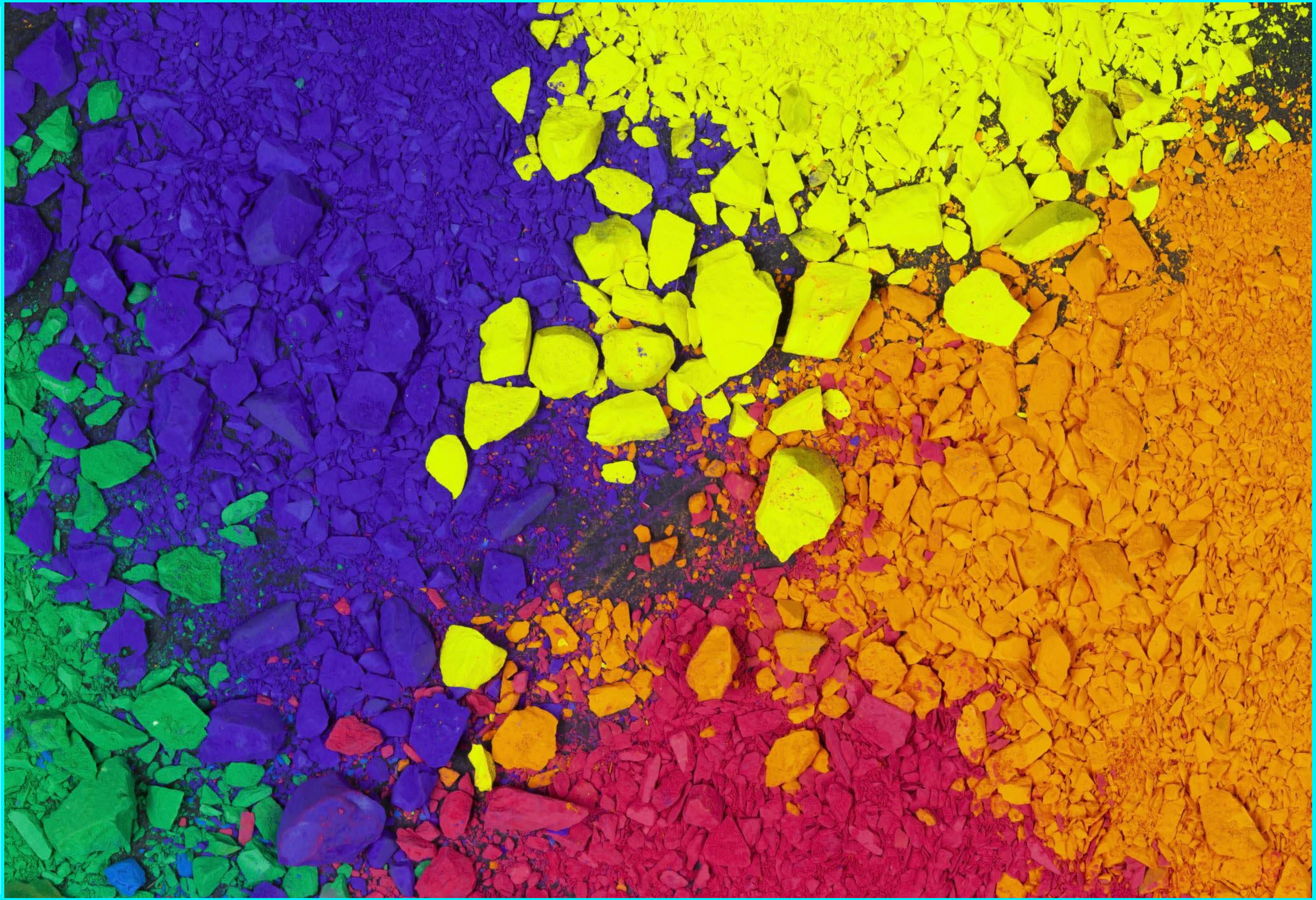
- Points (scatterplot)



```r
# install.packages("palmerpenguins")
require(ggplot2)
require(palmerpenguins)
ggplot(
  penguins,
  aes(
    x = body_mass_g,
    y = bill_length_mm)) +
  xlab("Body Mass (g)") +
  ylab("Bill Length (mm)") +
  geom_point()
```

Now Let’s Add A Colour Aesthetic!

- That was cool!
- Let’s add some color



Scatterplot 2: Body Mass, Bill Length, Species

1: Data

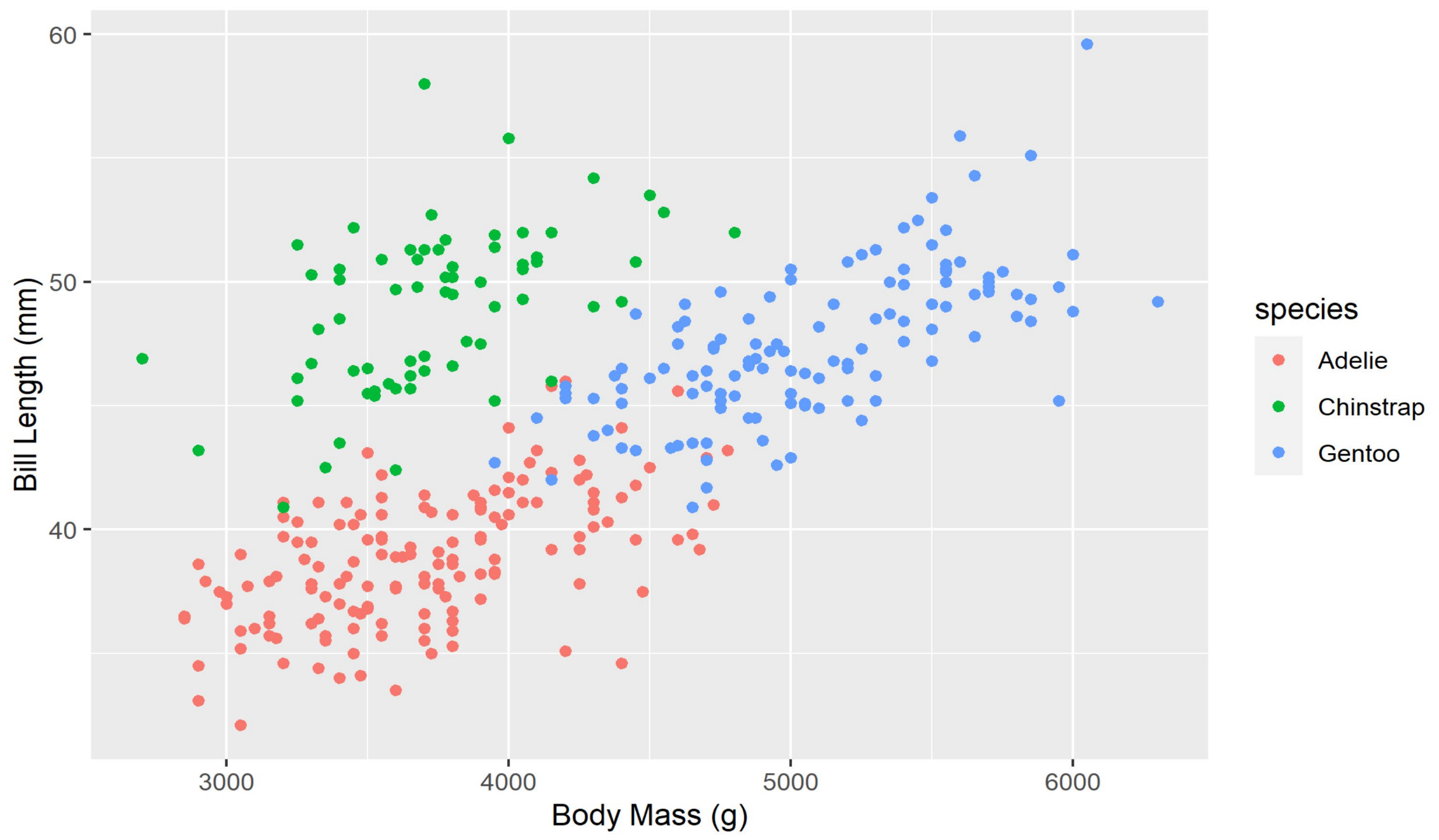
- Penguins

2: Aesthetics

- X: Body Mass
- Y: Bill Length
- Color: Species

3: Geometry

- Points (scatterplot)



Scatterplot 2: Body Mass, Bill Length, Species

1: Data

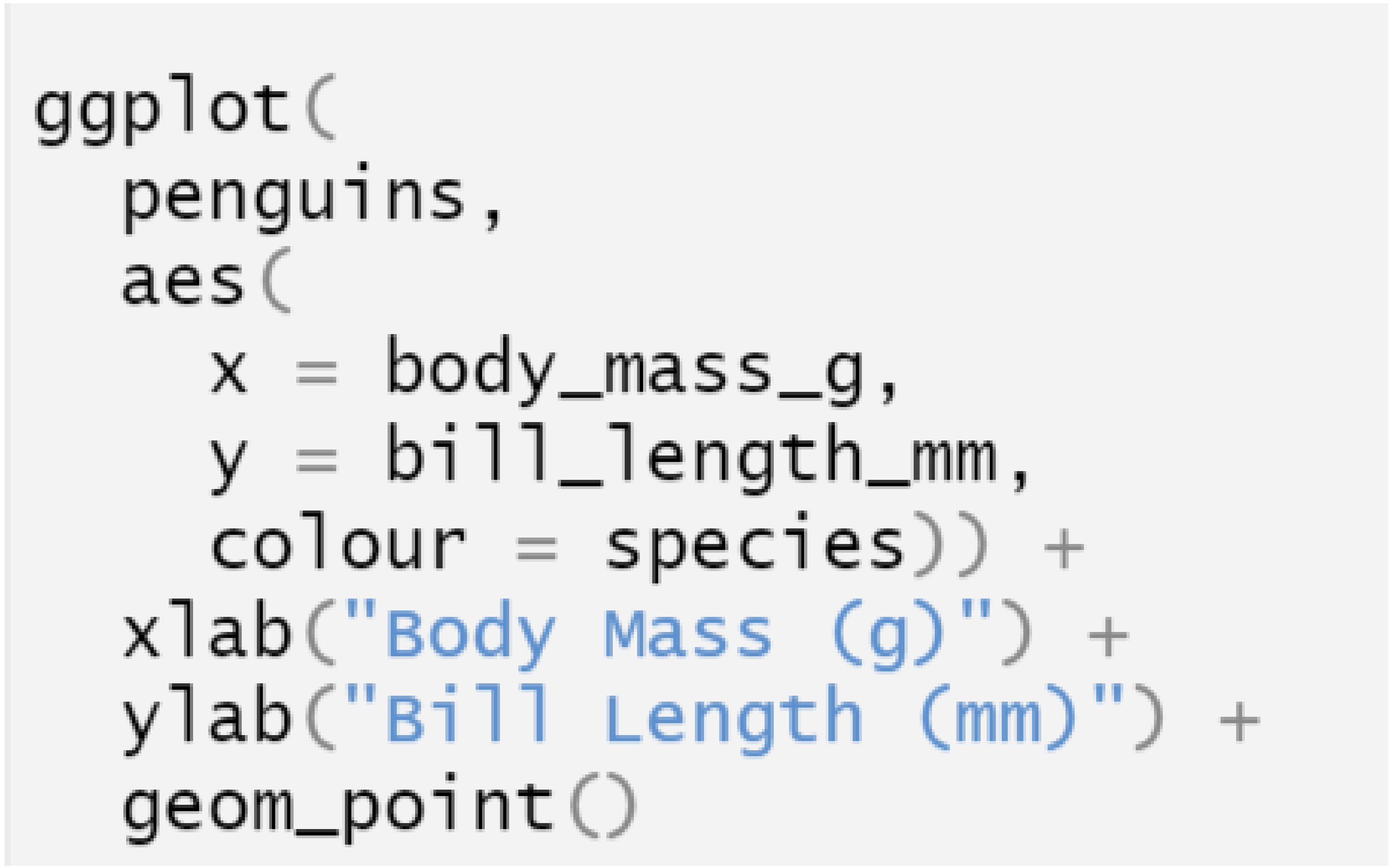
- Penguins

2: Aesthetics

- X: Body Mass
- Y: Bill Length
- Color: Species

3: Geometry

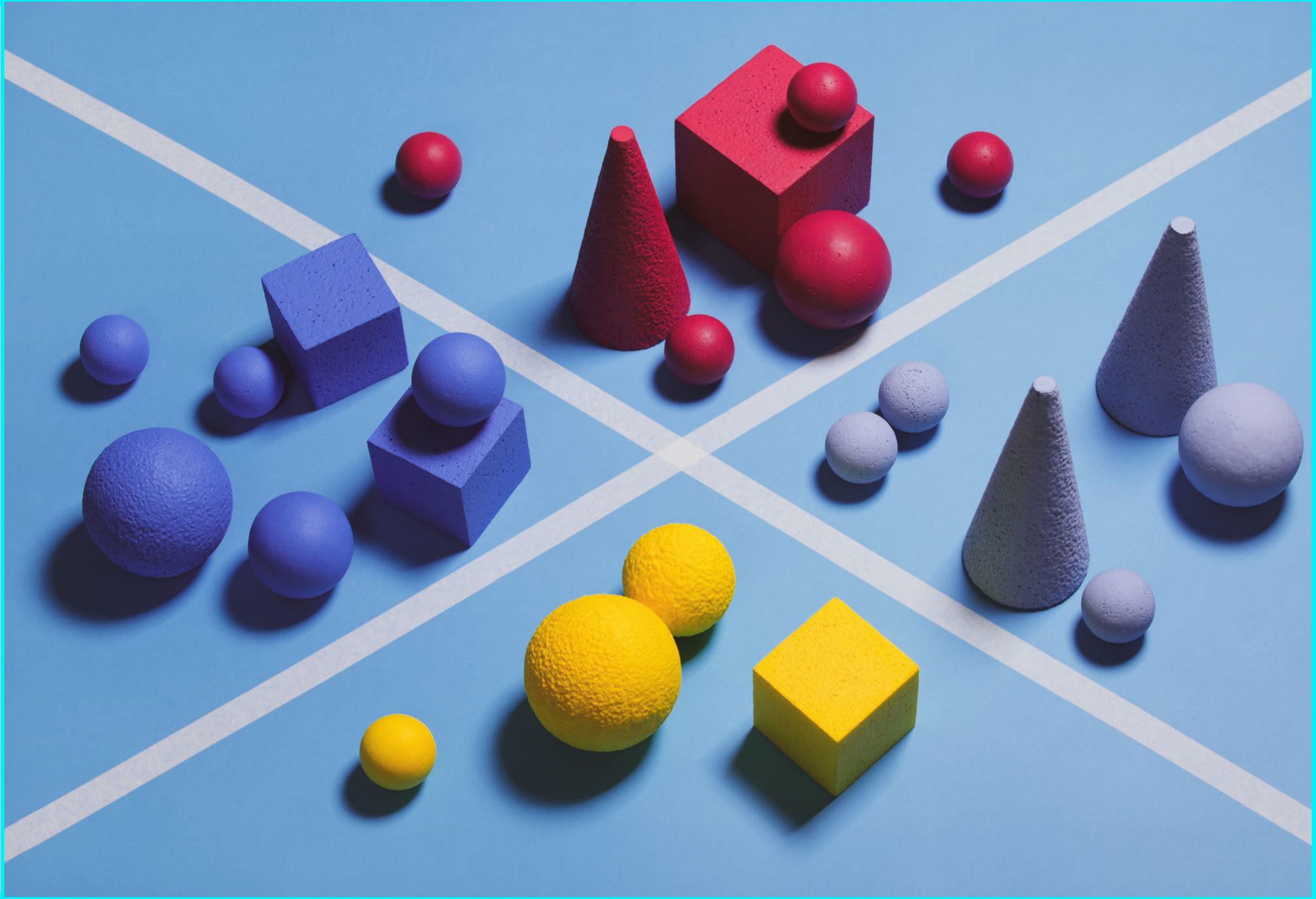
- Points (scatterplot)



```
ggplot(
  penguins,
  aes(
    x = body_mass_g,
    y = bill_length_mm,
    colour = species)) +
  xlab("Body Mass (g)") +
  ylab("Bill Length (mm)") +
  geom_point()
```

Shape Aesthetic

- Great!
- Now let’s try some different shapes



Scatterplot 3: Body Mass, Bill Length, Species

1: Data

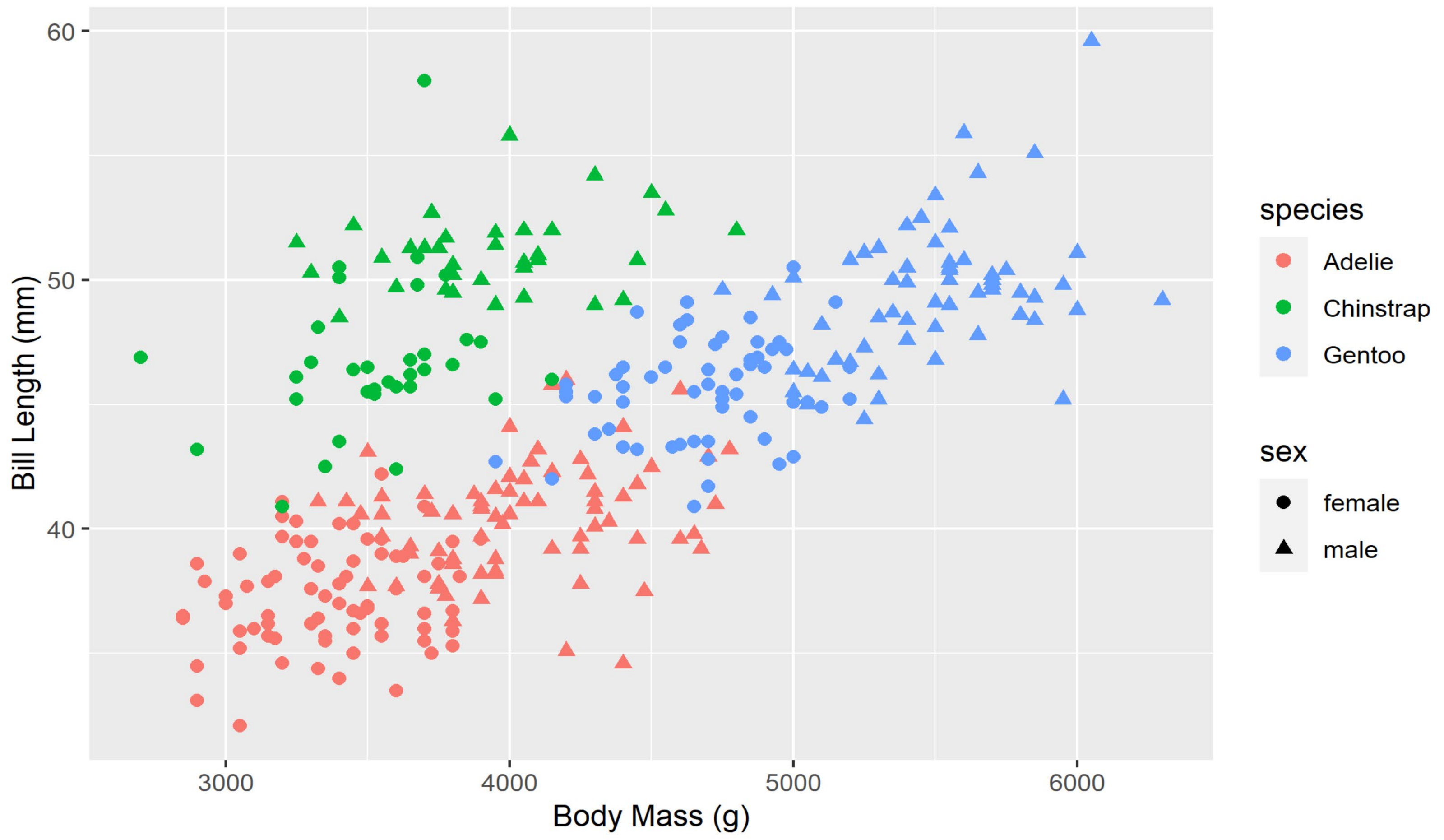
- Penguins

2: Aesthetics

- X: Body Mass
- Y: Bill Length
- Color: Species
- Shape: Sex

3: Geometry

- Points (scatterplot)



Scatterplot 3: Body Mass, Bill Length, Species

1: Data

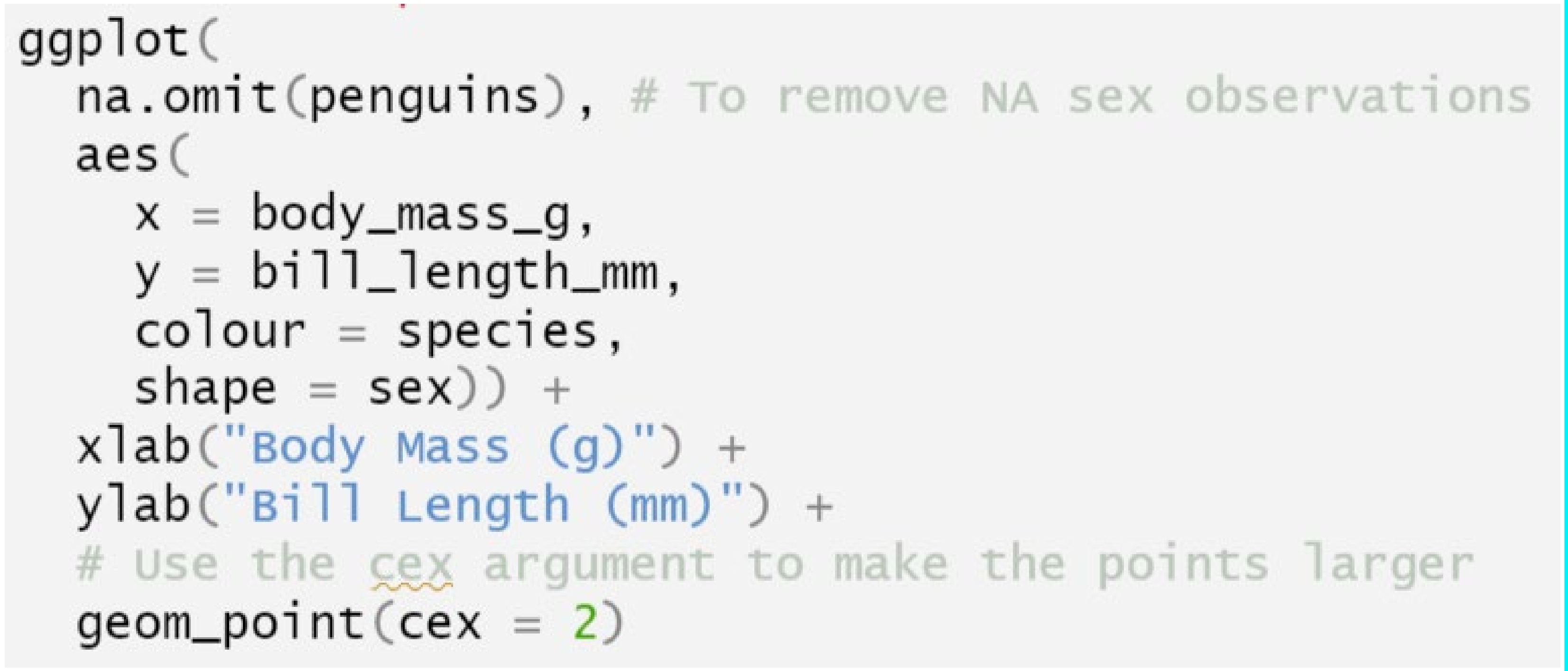
- Penguins

2: Aesthetics

- X: Body Mass
- Y: Bill Length
- Color: Species
- Shape: Sex

3: Geometry

- Points (scatterplot)



```
ggplot(
  na.omit(penguins), # To remove NA sex observations
  aes(
    x = body_mass_g,
    y = bill_length_mm,
    colour = species,
    shape = sex)) +
  xlab("Body Mass (g)") +
  ylab("Bill Length (mm)") +
  # Use the cex argument to make the points larger
  geom_point(cex = 2)
```

Recap: Basic Grammar of Graphics in R

1. Data
2. Aesthetics
3. Geometry

Vector Data in R

The adventure begins…

History Lesson: A Tale Of Two Packages

The spatial world in R is in the midst of a transition from Spatial* objects in package ‘sp’ to sf objects in the package ‘sf’

‘sf’ = simple features: a standard for GIS vector data format:

- Open Geospatial Consortium (OGC)
- International Organization for Standardization (ISO)

Simple Features

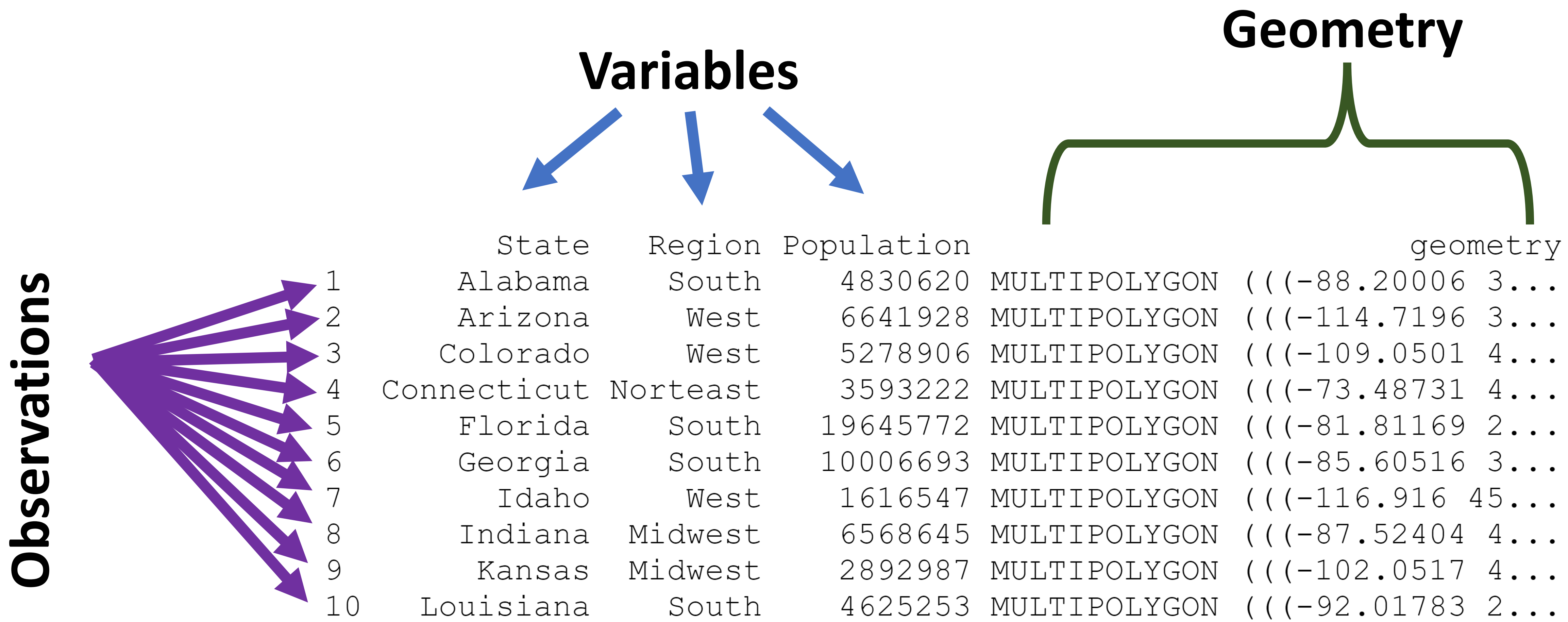
Simple Features objects have 2 main parts:

1. Table of data in row-format
   - Sounds familiar, right?
   - Variables are called ‘fields’

2. Set of geometries (one per each row)

```
Simple feature collection with 49 features and 3 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -124.7042 ymin: 24.55868 xmax: -66.9824 ymax: 
49.38436
Geodetic CRS:  NAD83
First 10 features:
         State    Region Population                       geometry
1      Alabama     South    4830620 MULTIPOLYGON (((-88.20006 3...
2      Arizona      West    6641928 MULTIPOLYGON (((-114.7196 3...
3     Colorado      West    5278906 MULTIPOLYGON (((-109.0501 4...
4  Connecticut Norteast    3593222 MULTIPOLYGON (((-73.48731 4...
5      Florida     South   19645772 MULTIPOLYGON (((-81.81169 2...
6      Georgia     South   10006693 MULTIPOLYGON (((-85.60516 3...
7        Idaho      West    1616547 MULTIPOLYGON (((-116.916 45...
8      Indiana   Midwest    6568645 MULTIPOLYGON (((-87.52404 4...
9       Kansas   Midwest    2892987 MULTIPOLYGON (((-102.0517 4...
10   Louisiana     South    4625253 MULTIPOLYGON (((-92.01783 2...
```

Simple Features Are gg-Friendly



```
          State    Region Population                       geometry
1       Alabama     South    4830620 MULTIPOLYGON (((-88.20006 3...
2       Arizona      West    6641928 MULTIPOLYGON (((-114.7196 3...
3      Colorado      West    5278906 MULTIPOLYGON (((-109.0501 4...
4   Connecticut Norteast    3593222 MULTIPOLYGON (((-73.48731 4...
5       Florida     South   19645772 MULTIPOLYGON (((-81.81169 2...
6       Georgia     South   10006693 MULTIPOLYGON (((-85.60516 3...
7         Idaho      West    1616547 MULTIPOLYGON (((-116.916 45...
8       Indiana   Midwest    6568645 MULTIPOLYGON (((-87.52404 4...
9        Kansas   Midwest    2892987 MULTIPOLYGON (((-102.0517 4...
10    Louisiana     South    4625253 MULTIPOLYGON (((-92.01783 2...
```

Data Management With Simple Features

You can leverage all the data manipulation tools of R when working with sf objects:

- Logical subsetting, extract/insert columns, merging, data transformations, etc.

You can also perform all the standard GIS operations!

sf objects are just **data.frame** objects with associated spatial geometries

Plotting Simple Features With ggplot

Let’s make some maps!

Grammar of Graphics With Spatial Data

We can follow the familiar ggplot procedure with an additional step:

- Specify a coordinate system

1 • Data

2 • Aesthetics

3 • Geometry

4 • Coordinate sys.

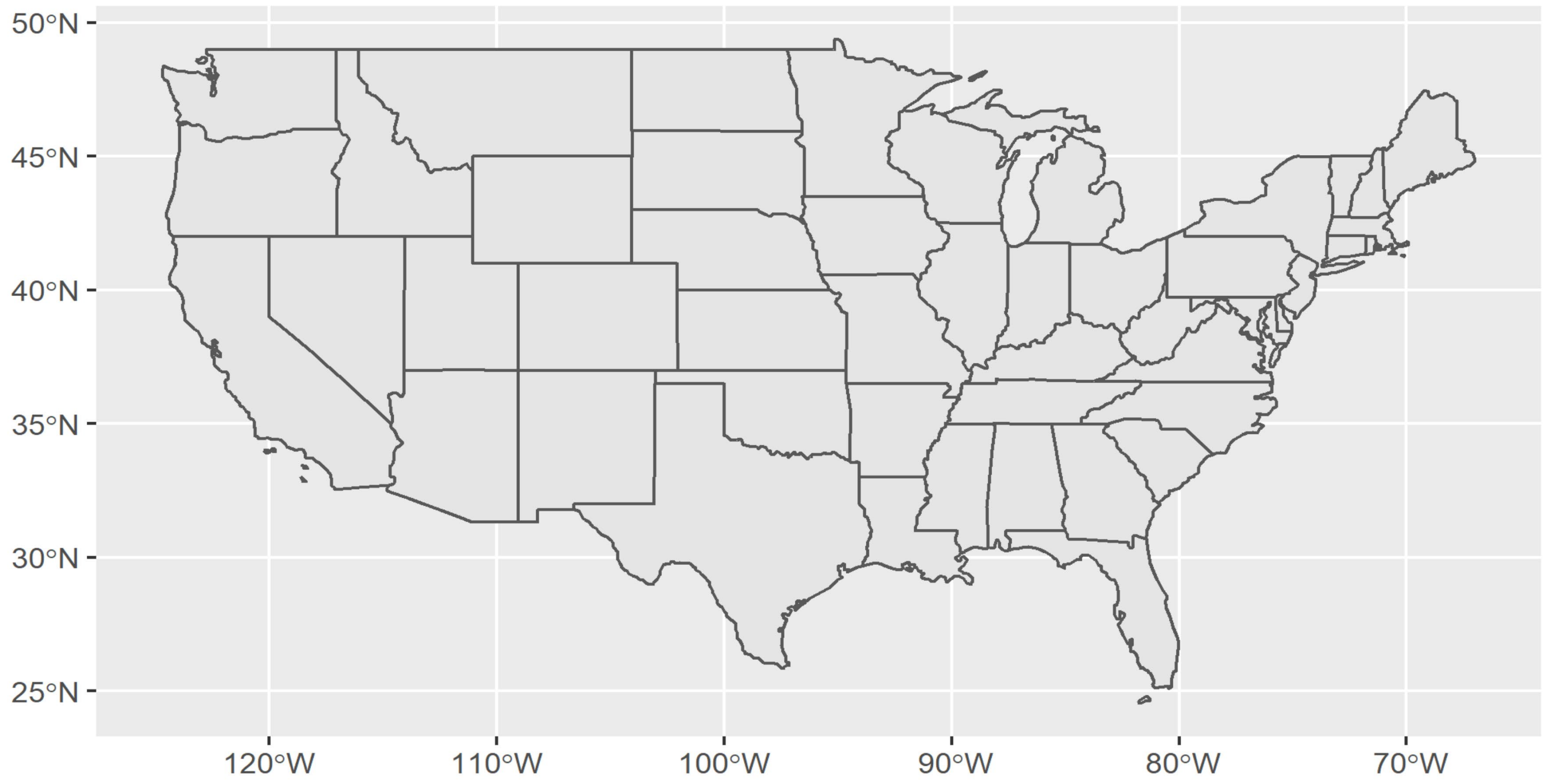
Map 1: State Borders

1: Data
- CONUS

2: Aesthetics
- None

3: Geometry
- Simple Feature: geom_sf

4: Coordinate System
- GCS: NAD83



Map 1: State Borders

1: Data

- CONUS

2: Aesthetics

- None

3: Geometry

- Simple Feature: geom_sf

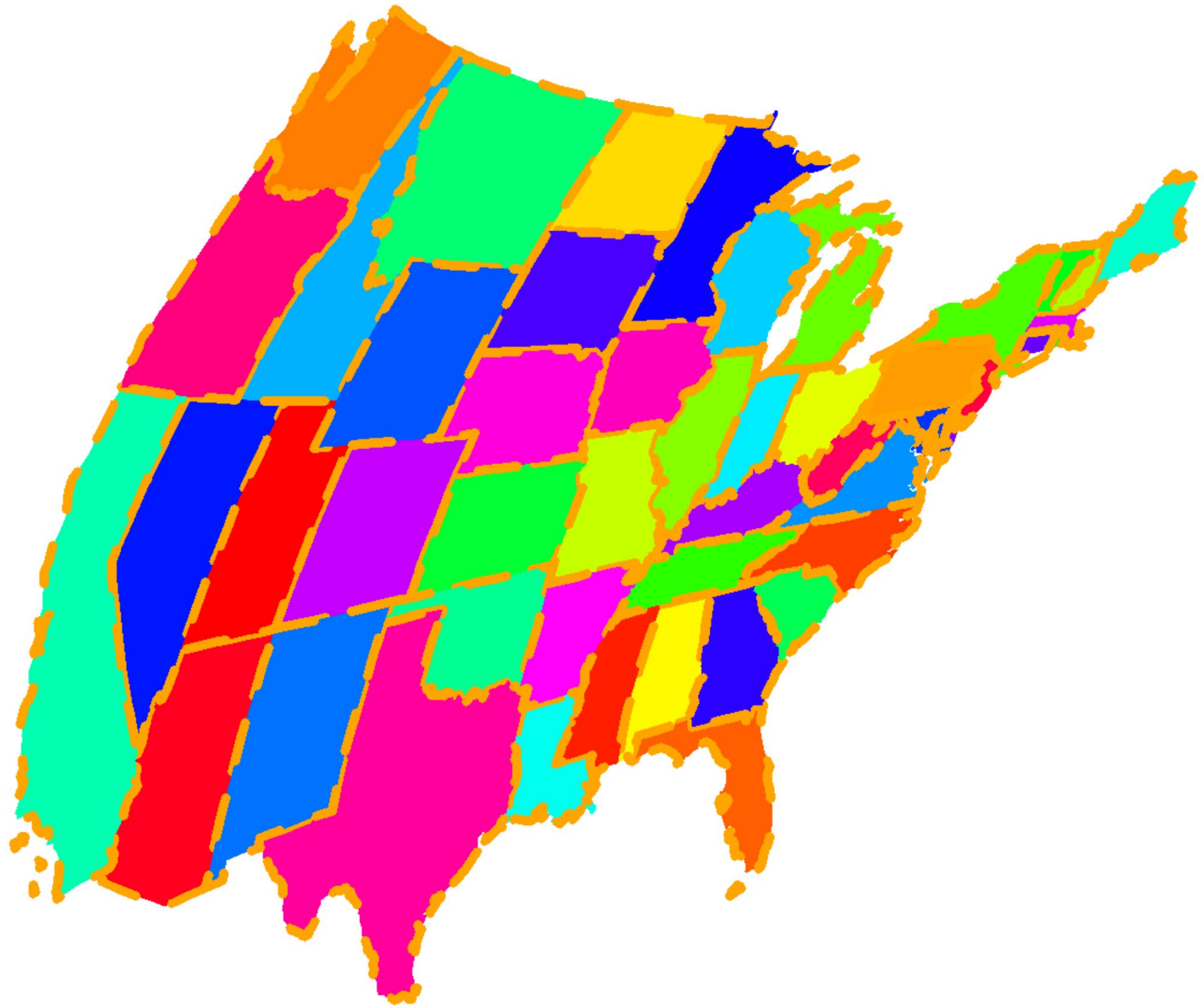
4: Coordinate System

- GCS: NAD83

```
require(spData)
conus = subset(us_states, !(NAME %in% c("Alaska", "Hawaii")))
conus_2 = conus[, c("NAME", "REGION", "total_pop_15")]
names(conus_2) = c("State", "Region", "Population", "geometry")
ggplot(conus_2) + geom_sf()
```

Map 1: State Borders

- Not terrible, but pretty basic… and in an ugly CRS.
- Let's elaborate by changing coordinate systems. We can use an Albers Equal Area Conic centered on CONUS: EPSG 5070



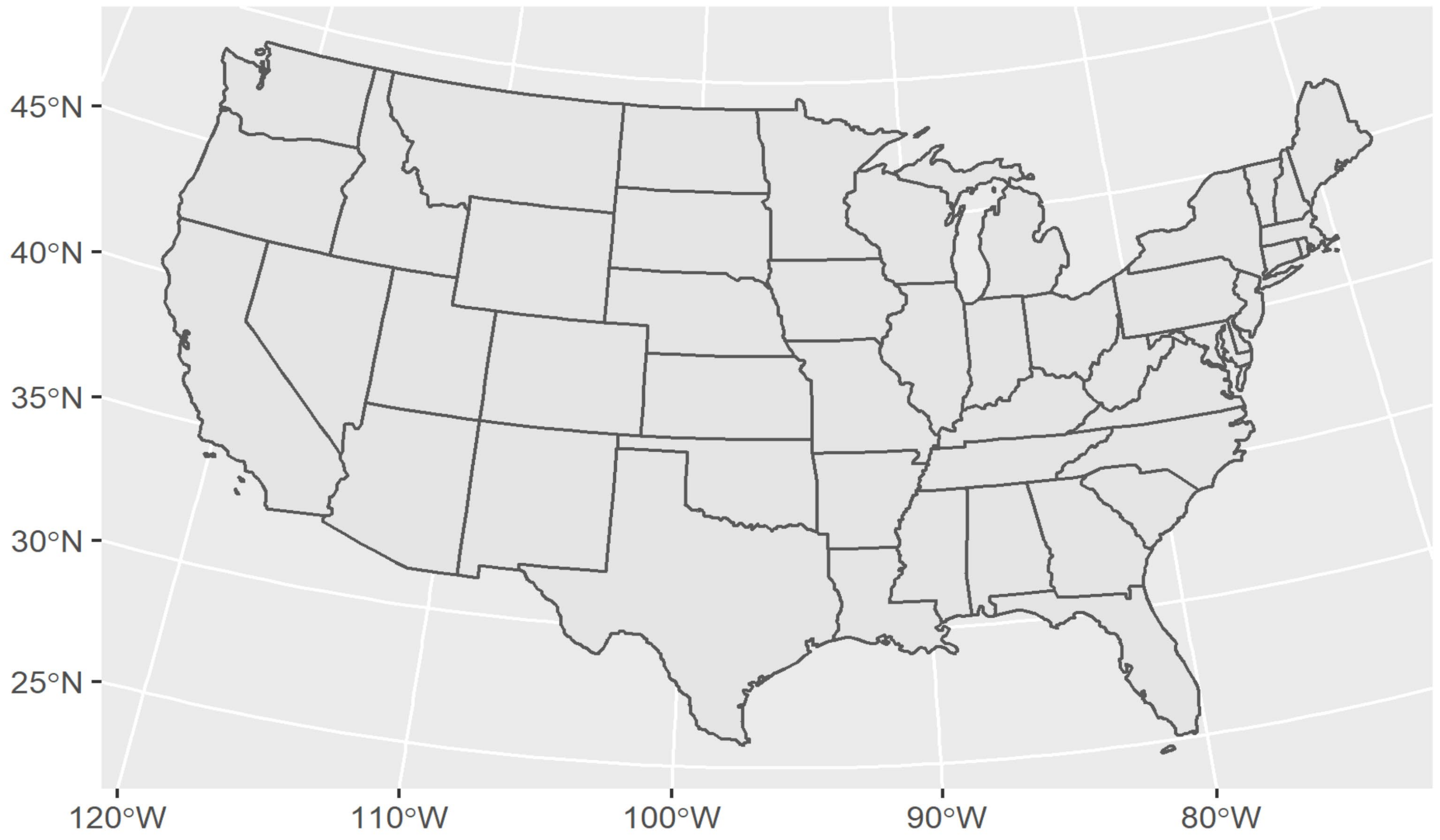
Map 2: State Borders

1: Data
- CONUS

2: Aesthetics
- None

3: Geometry
- Simple Feature: geom_sf

4: Coordinate System
- PCS: Equal Area



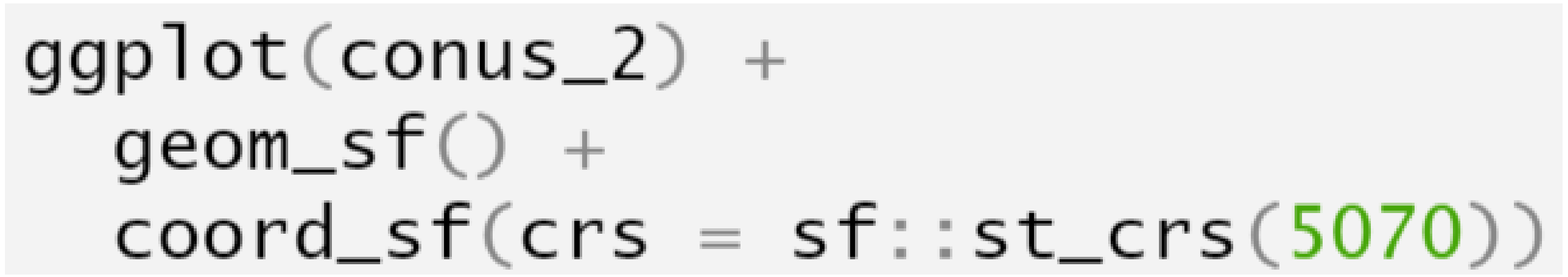
Map 2: State Borders

1: Data
- CONUS

2: Aesthetics
- None

3: Geometry
- Simple Feature: geom_sf

4: Coordinate System
- PCS: Equal Area



```r
ggplot(conus_2) +
  geom_sf() +
  coord_sf(crs = sf::st_crs(5070))
```

Map 2: State Borders

What should we do next?

Recall the data attributes:

- State Name
- Region
- Population

Let’s color the borders by region

```
          State    Region Population
1       Alabama     South    4830620
2       Arizona      West    6641928
3      Colorado      West    5278906
4   Connecticut Norteast    3593222
5       Florida     South   19645772
6       Georgia     South   10006693
7         Idaho      West    1616547
8       Indiana   Midwest    6568645
9        Kansas   Midwest    2892987
10    Louisiana     South    4625253
```

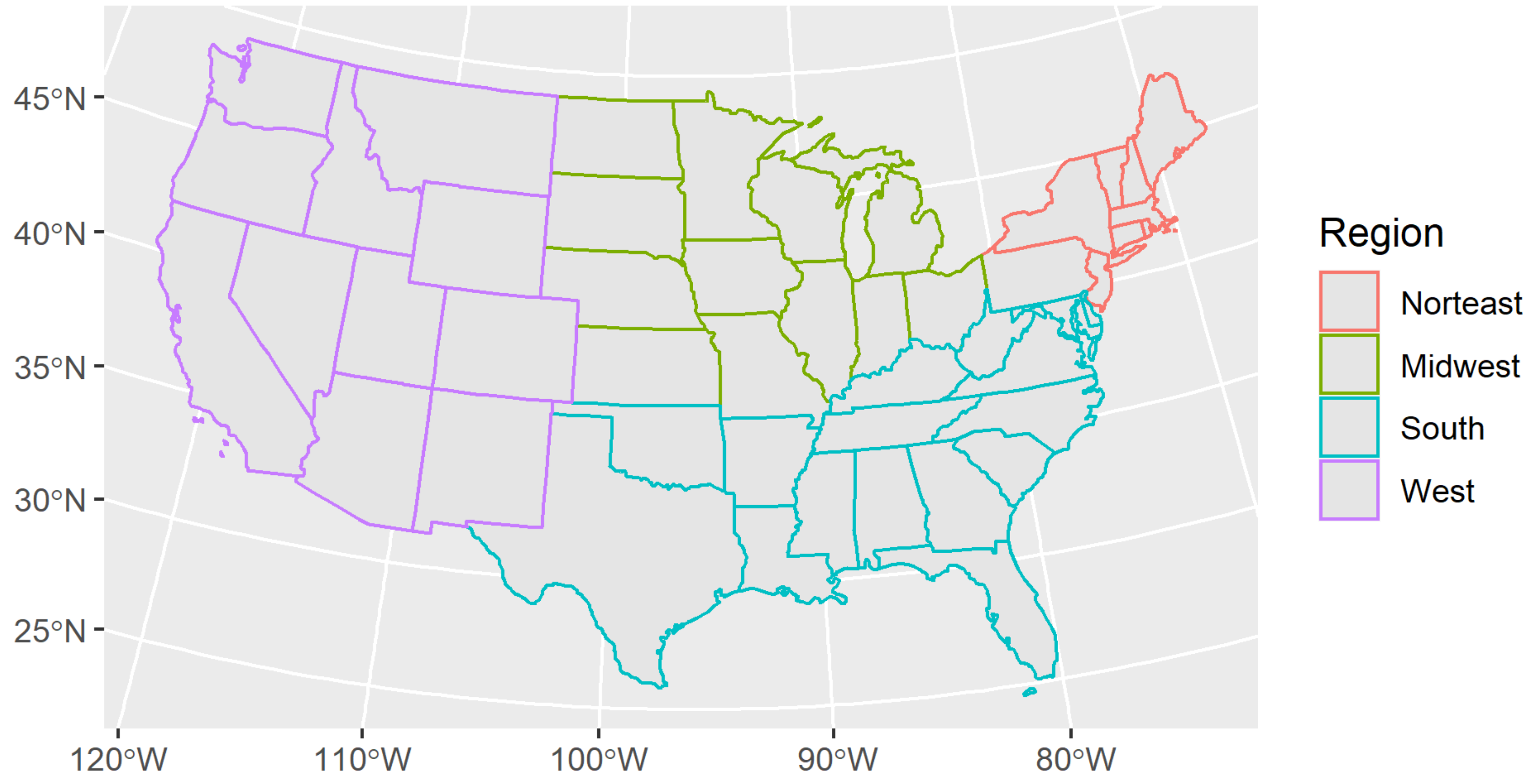
Map 3: Regions 1

1: Data
- CONUS

2: Aesthetics
- Color: Region

3: Geometry
- Simple Feature: geom_sf

4: Coordinate System
- PCS: Equal Area



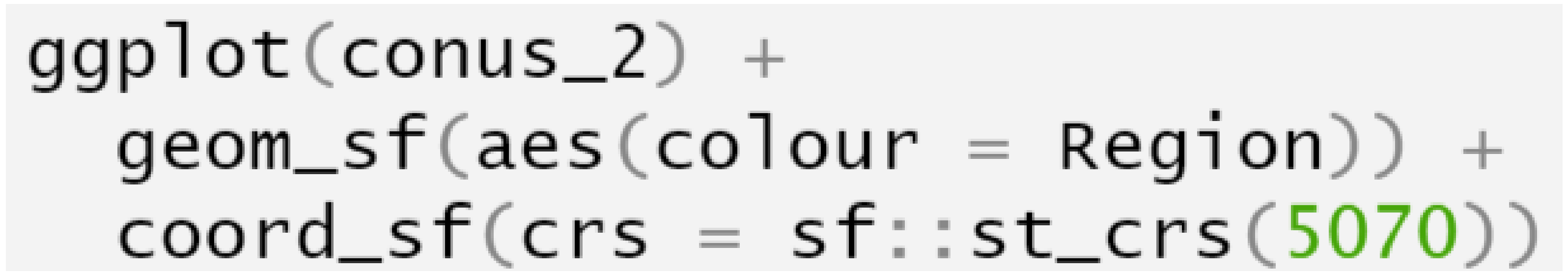
Map 3: Regions 1

1: Data
- CONUS

2: Aesthetics
- Color: Region

3: Geometry
- Simple Feature: geom_sf

4: Coordinate System
- PCS: Equal Area



```r
ggplot(conus_2) +
  geom_sf(aes(colour = Region)) +
  coord_sf(crs = sf::st_crs(5070))
```

Color and Fill

- Let’s be a fancy and change the fill color by region too.
- We can make the fill semitransparent by adjusting the alpha parameter

$$\alpha$$

Map 4: Regions 2

1: Data

- CONUS

2: Aesthetics

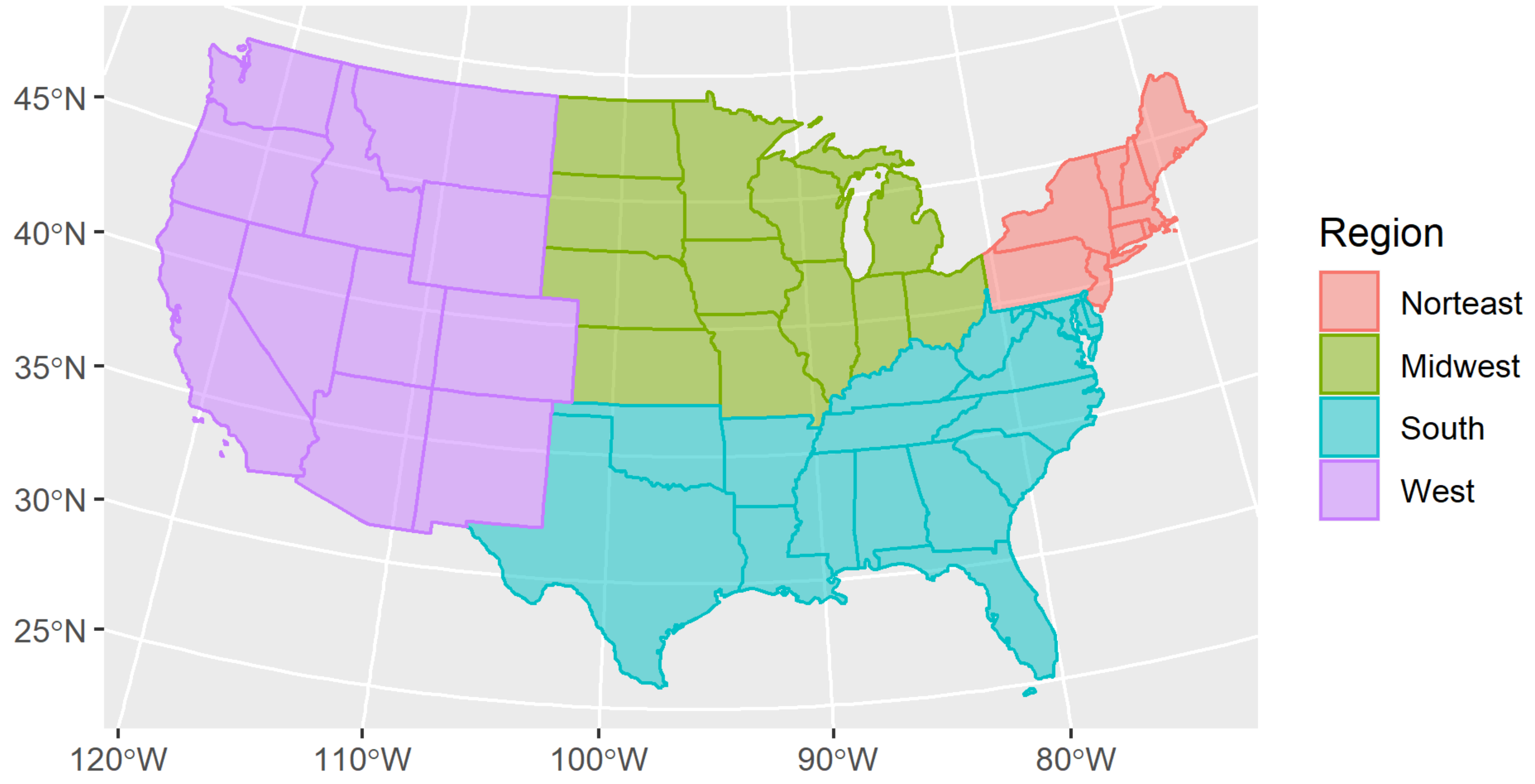
- Color: Region
- Fill: Region

3: Geometry

- Simple Feature: geom_sf

4: Coordinate System

- PCS: Equal Area



Map 4: Regions 2

1: Data

- CONUS

2: Aesthetics

- Color: Region
- Fill: Region

3: Geometry

- Simple Feature: geom_sf

4: Coordinate System

- PCS: Equal Area

```
ggplot(conus_2) +
  geom_sf(
    aes(
      colour = Region,
      fill = Region),
    # The alpha argument sets the
    # transparency for the fill color
    alpha = 0.5) +
  coord_sf(crs = sf::st_crs(5070))
```

What About Data?

- That’s cool, but how can we use R’s data manipulation potential?

- Let’s make a choropleth from population.
- Which aesthetic do we need to use?

Map 5a: Choropleth 1

1: Data

- CONUS

2: Aesthetics

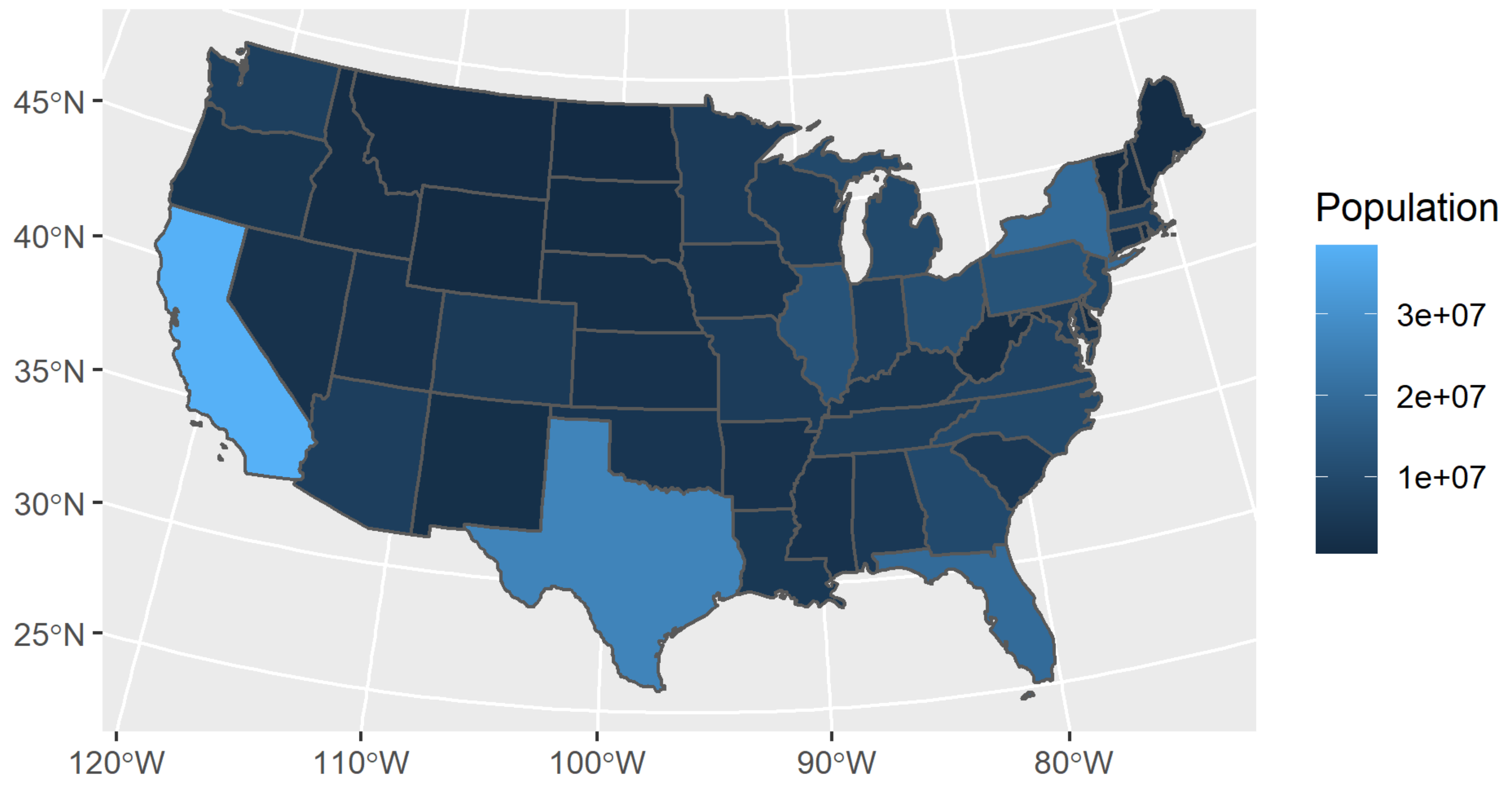
- Fill: Population (blue color scale)

3: Geometry

- Simple Feature: geom_sf

4: Coordinate System

- PCS: Equal Area



Map 5a: Choropleth 1

1: Data

- CONUS

2: Aesthetics

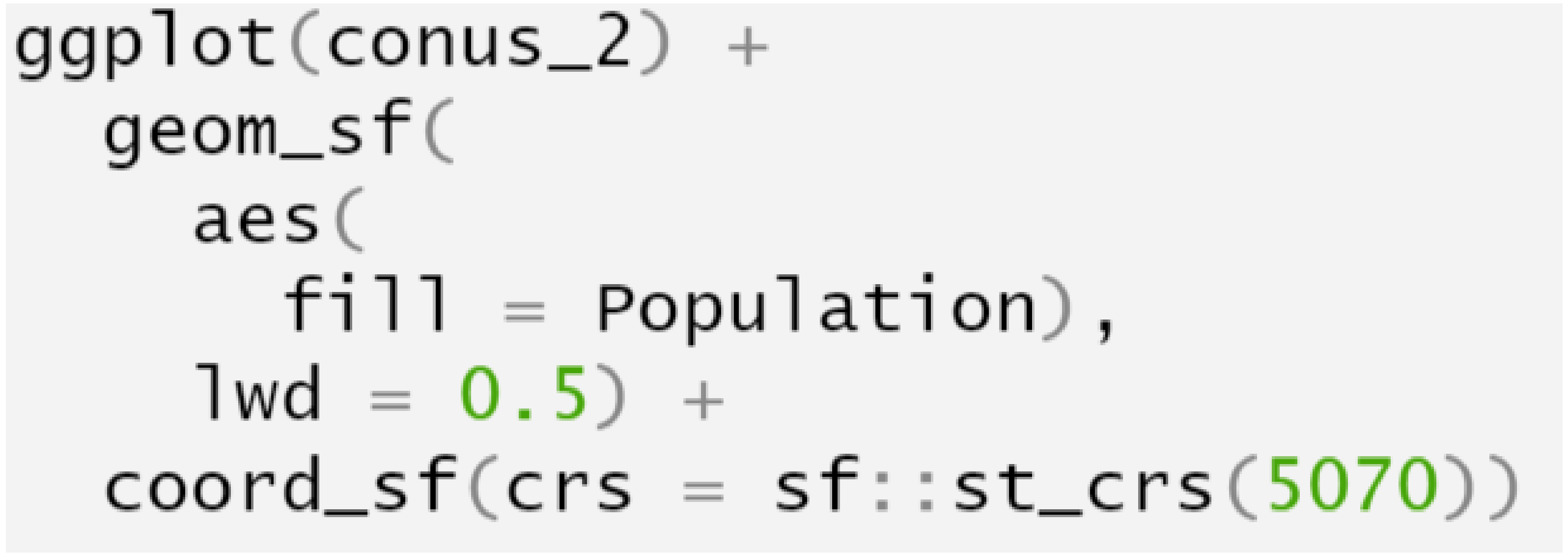
- Fill: Population (blue color scale)

3: Geometry

- Simple Feature: geom_sf

4: Coordinate System

- PCS: Equal Area



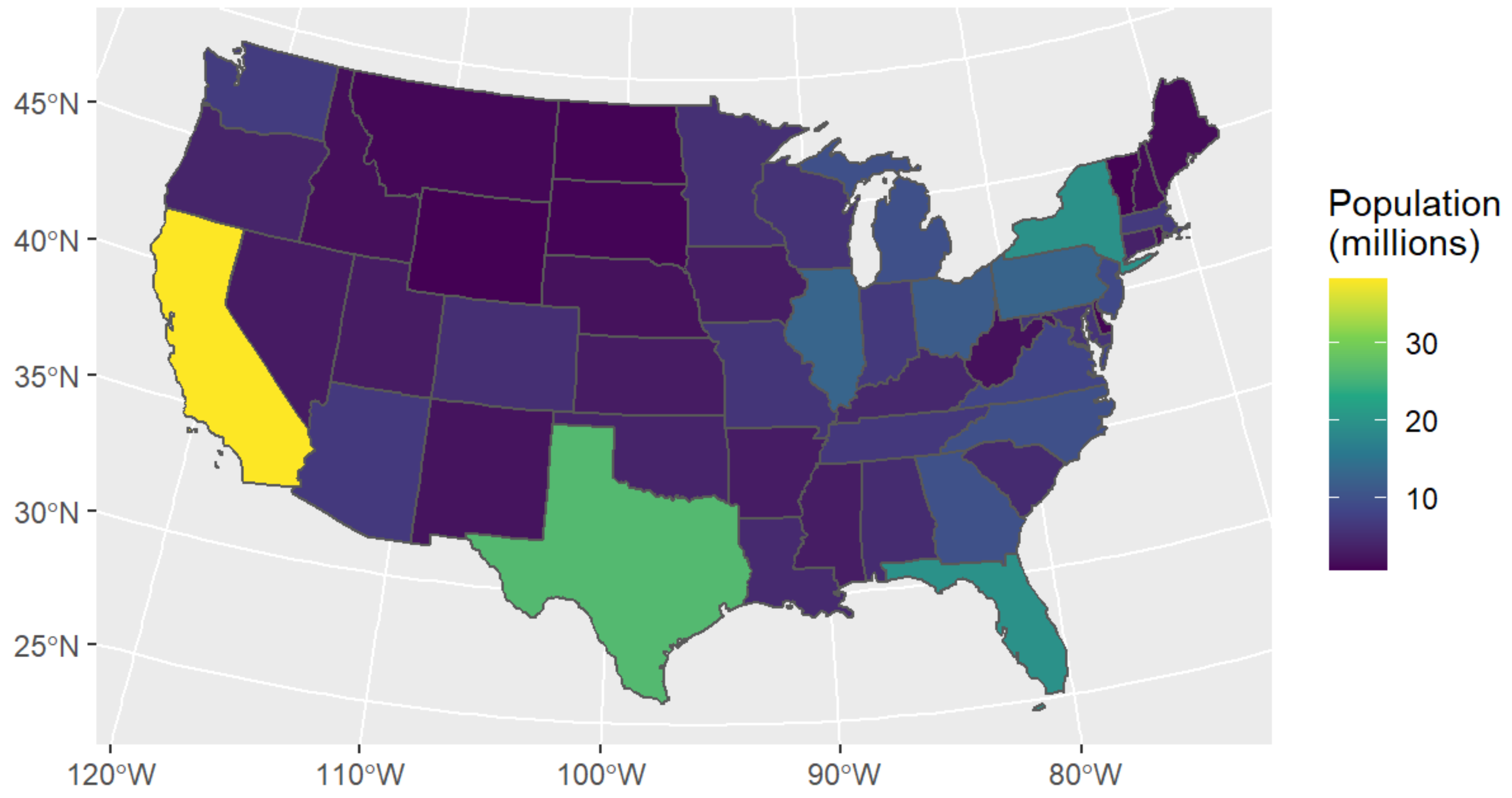
```
ggplot(conus_2) +
  geom_sf(
    aes(
      fill = Population),
    lwd = 0.5) +
  coord_sf(crs = sf::st_crs(5070))
```

Map 5b: Choropleth 1b

That’s OK, but the color scale isn’t the best.

We can use one of the viridis scales, which are optimized for colorblindness and grayscale.

Now, let’s symbolize the region using the state border color…



Map 5b: Choropleth 1b

That's OK, but the color scale isn't the best.

We can use one of the viridis scales, which are optimized for colorblindness and grayscale.

Now, let's symbolize the region using the state border color…

```r
ggplot(conus_2) +
  geom_sf(
    aes(
      fill = Population * 1e-6),
    lwd = 0.5) +
  coord_sf(crs = sf::st_crs(5070)) +
  scale_fill_viridis_c(name = "Population\n(millions)")
```

Map 6: Choropleth 2

1: Data

- CONUS

2: Aesthetics

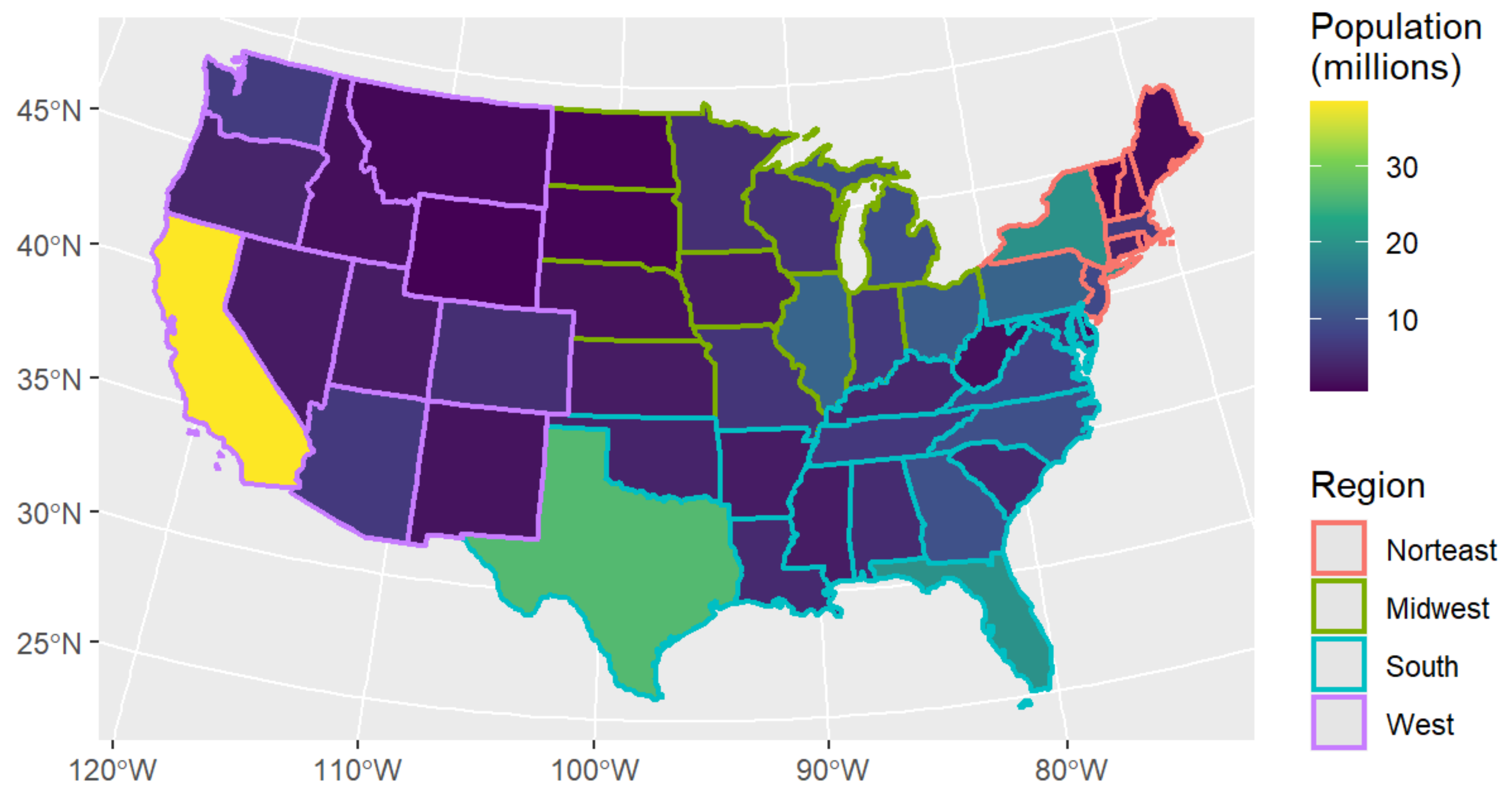
- Fill: Population
  - Viridis scale 'c'
- Color: Region

3: Geometry

- Simple Feature: geom_sf

4: Coordinate System

- PCS: Equal Area



Map 6: Choropleth 2

1: Data

- CONUS

2: Aesthetics

- Fill: Population
    - Viridis scale ‘c’
- Color: Region

3: Geometry

- Simple Feature: geom_sf

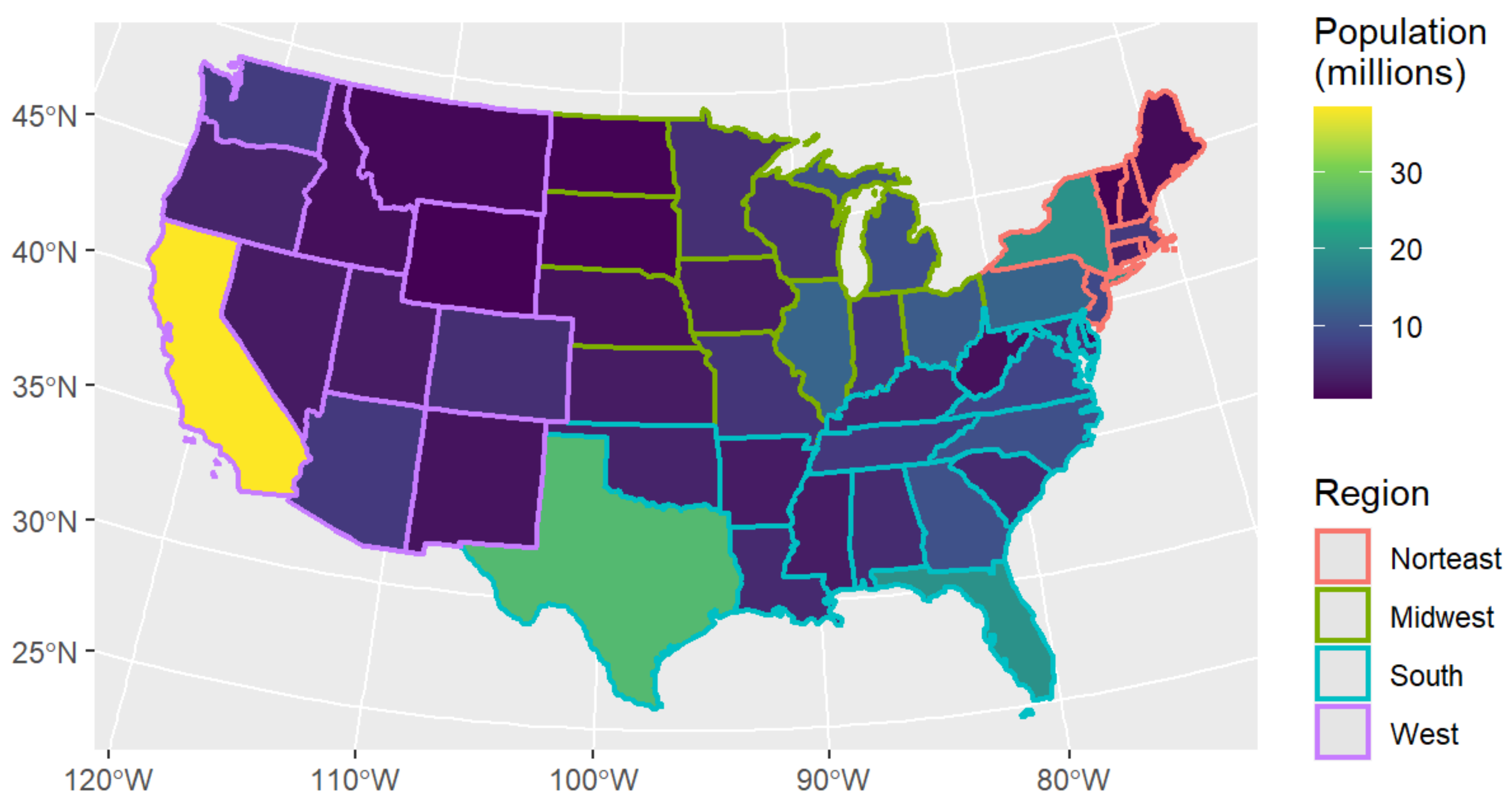
4: Coordinate System

- PCS: Equal Area

```
ggplot(conus_2) + geom_sf() +
  geom_sf(
    aes(
      colour = Region,
      fill = Population * 1e-6),
    lwd = 0.8) +
  coord_sf(crs = sf::st_crs(5070)) +
  scale_fill_viridis_c(name = "Population\n(millions)")
```

Map 6: Choropleth 2

What's the viridict on this map?



US Census Data in R

Using tidycensus

US Census - ACS

- US Census Bureau is an embarrassment of data riches.
- The American Community Survey (ACS) contains a wealth of demographic and socioeconomic data.
- Census data is inherently spatial – the US Census Bureau produces lots of spatial products for many geometries including
  - States
  - Counties
  - Census tracts
- Wouldn't it be nice if we could all of these datasets in R?

US Census Data - tidycensus

There's a great package called tidycensus that allows you to directly import ACS data into R.

It even lets you download spatial versions of the data as sf objects!

tidycensus - API

To download census data with tidycensus, you need to register with the Census Bureau to receive an API.

What’s an API, you might ask?

- It stands for Application Programming Interface, which sounds intimidating…
- In this case, it’s just a set of protocols that tidycensus uses to communicate with the Census Bureau.

API

Median Household Income

- Let’s examine the Median Household Income in Hampshire County, MA.
- ACS Variable B19013_001

Hampshire County: Median Income

1: Data

- Hampshire County – census tracts

2: Aesthetics

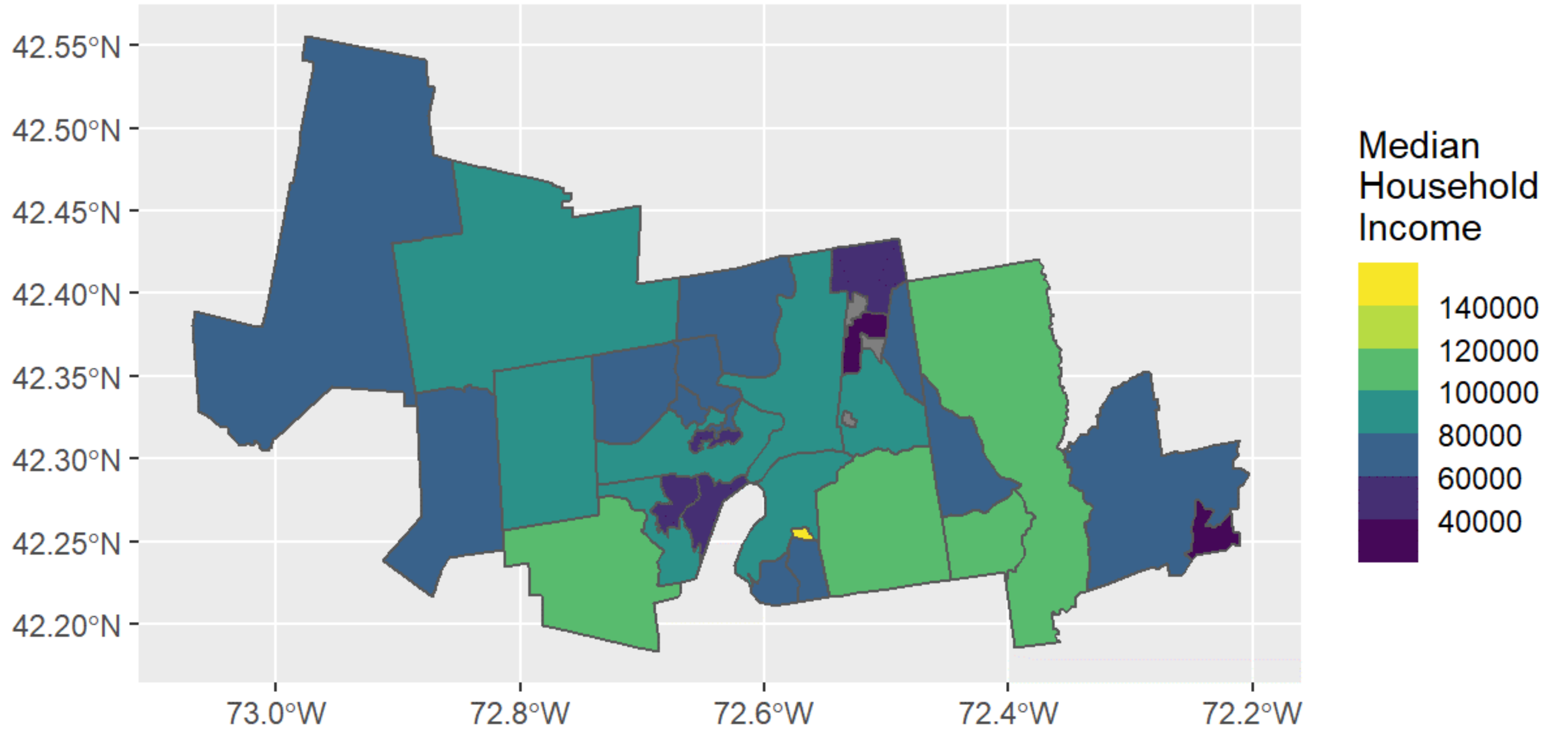
- Fill: Income
  - Viridis scale ‘c’

3: Geometry

- Simple Feature: geom_sf

4: Coordinate System

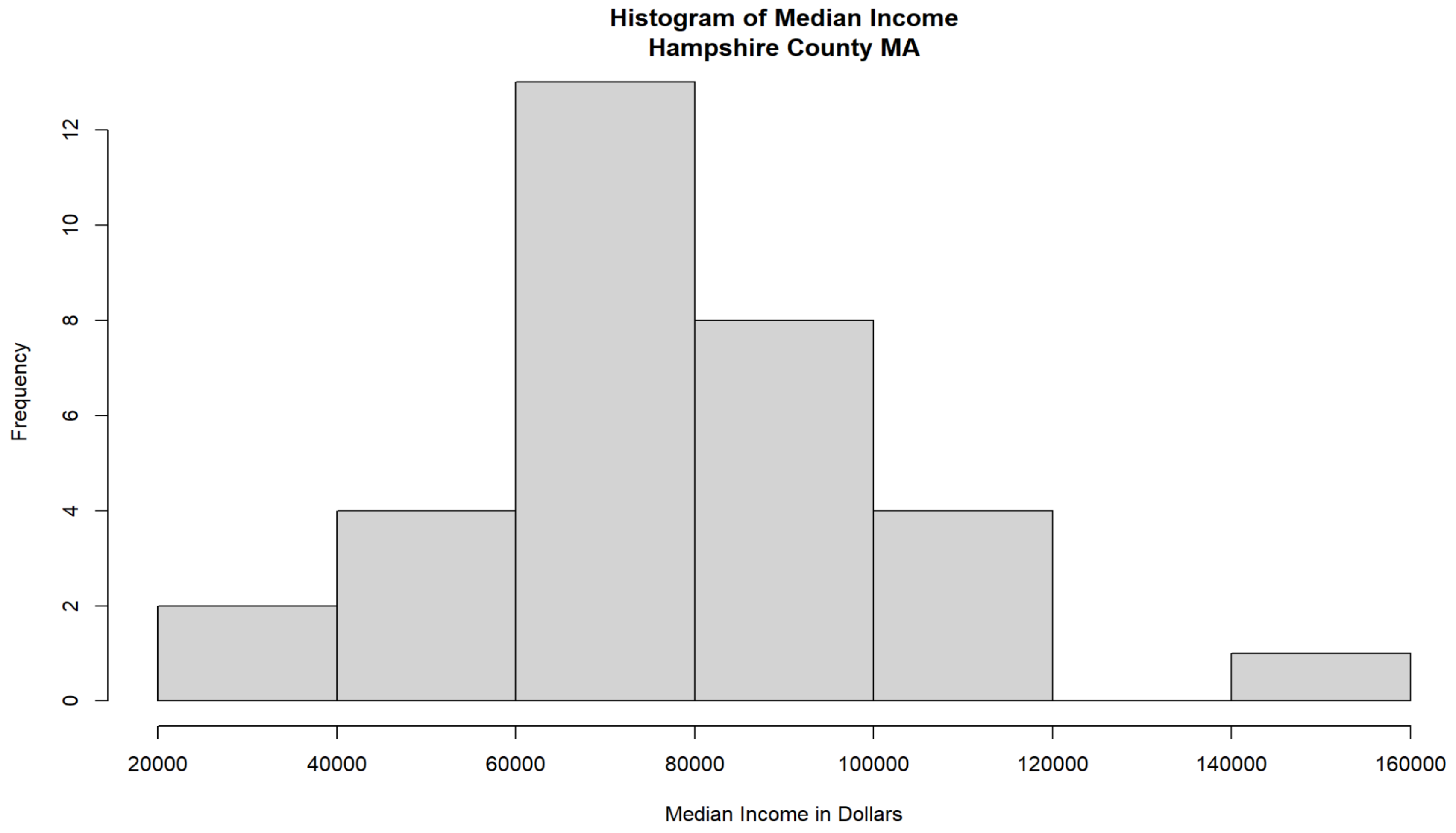
- GCS: NAD83



Median Household Income

- What’s up with that one census tract?
  - It looks very wealthy!
- Let’s look at a histogram.

Median Income Histogram



Recap

What Have We Learned?

?

- R is awesome.
- ggplot is the way to go.
- Working with spatial data in R is not scary.

- Thank so much for your attention!