

Visualization for Data Science in R

Angela Zoss

Data Matters 2019

<https://github.com/amzoss/RVis-DM2019>

Objectives/Outline

Day 1: Static visualizations

- Visualization and data science
- Basic ggplot2 syntax
- Basics of geoms and aes
- Manipulating data
- Categorical variables
- Advanced topics: mapping, saving charts out

Day 2: Interactivity

- Simple interactive plots
- Arranging charts into dashboards
- Incorporating Shiny elements into documents, dashboards
- Advanced topics: full Shiny apps

Set up environment

- R
- RStudio
- packages

Packages:

- tidyverse
- maps
- mapproj
- sf
- knitr
- plotly
- flexdashboard
- shiny

Get workshop files

URL: <https://github.com/amzoss/RVis-DM2019>

With Git installed

In RStudio:

- Project → New project
- Version Control
- Git
 - Paste in GitHub URL
 - Project directory name:
RVis-DM2019
 - Subdirectory: you choose
- Create Project

Without Git installed

• Click green button to download ZIP

• Unzip files on your laptop

In RStudio:

- Project → New project...
- Existing directory
- Select unzipped folder
- Create Project

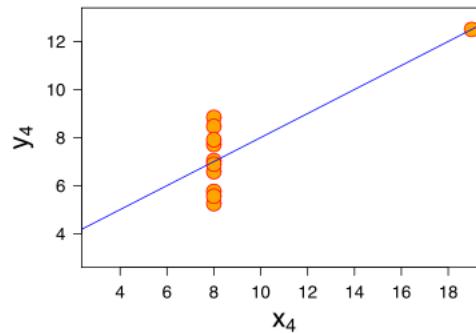
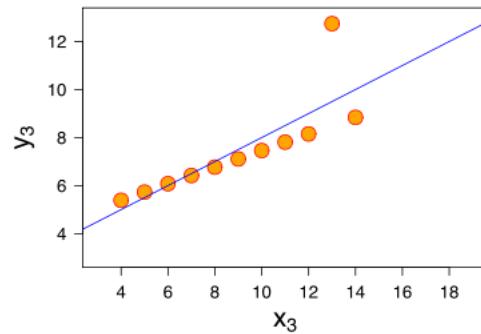
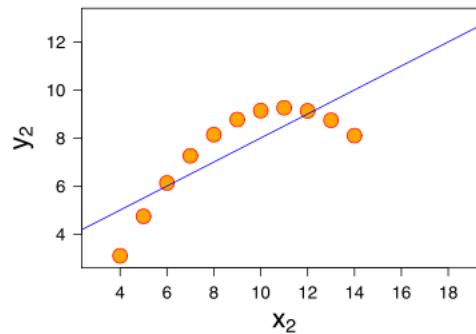
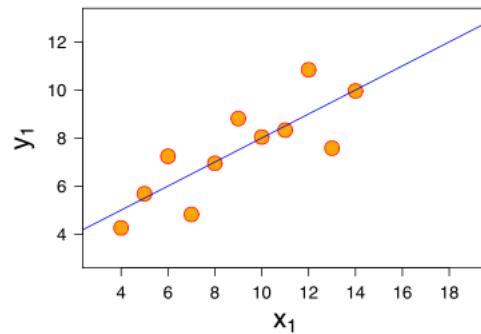
What does it mean to represent data visually?
Why do it?

Math is hard

| 1 | | 2 | | 3 | | 4 | |
|------|-------|------|------|------|-------|------|-------|
| x | y | x | y | x | y | x | y |
| 10.0 | 8.04 | 10.0 | 9.14 | 10.0 | 7.46 | 8.0 | 6.58 |
| 8.0 | 6.95 | 8.0 | 8.14 | 8.0 | 6.77 | 8.0 | 5.76 |
| 13.0 | 7.58 | 13.0 | 8.74 | 13.0 | 12.74 | 8.0 | 7.71 |
| 9.0 | 8.81 | 9.0 | 8.77 | 9.0 | 7.11 | 8.0 | 8.84 |
| 11.0 | 8.33 | 11.0 | 9.26 | 11.0 | 7.81 | 8.0 | 8.47 |
| 14.0 | 9.96 | 14.0 | 8.10 | 14.0 | 8.84 | 8.0 | 7.04 |
| 6.0 | 7.24 | 6.0 | 6.13 | 6.0 | 6.08 | 8.0 | 5.25 |
| 4.0 | 4.26 | 4.0 | 3.10 | 4.0 | 5.39 | 19.0 | 12.50 |
| 12.0 | 10.84 | 12.0 | 9.13 | 12.0 | 8.15 | 8.0 | 5.56 |
| 7.0 | 4.82 | 7.0 | 7.26 | 7.0 | 6.42 | 8.0 | 7.91 |
| 5.0 | 5.68 | 5.0 | 4.74 | 5.0 | 5.73 | 8.0 | 6.89 |

Almost identical summary statistics:
x & y mean
x & y variance
x-y correlation
x-y linear regression

Shapes are much easier



Anscombe's Quartet

http://en.wikipedia.org/wiki/Anscombe%27s_quartet

Why visualize in R?

- Quickly explore data
- Save time switching to another tool
- Use charts to inspire new analyses and vice versa
- Reproducibility

Why care about reproducibility?

- Open science makes review easier
- Increasingly a requirement
- Saves you a lot of time trying to figure out what you did last time!

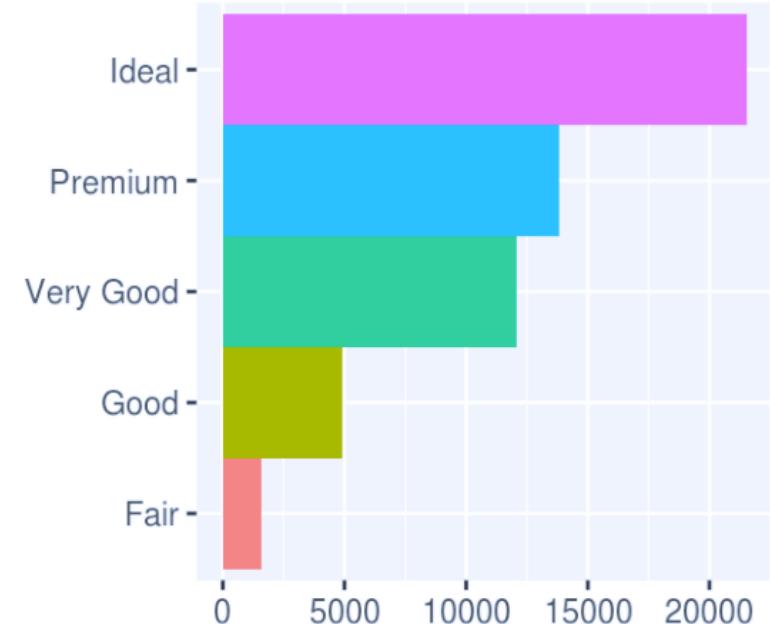
*“Your closest collaborator is **you** six months ago,
but you don’t reply to emails.”*

- *Mark Holder*

ggplot2

What is ggplot2?

an R package designed to create plots based on a theory of the grammar of graphics.



Grammar of graphics

1. DATA: a set of data operations that create variables from datasets
2. TRANS: variable transformations (e.g., rank)
3. SCALE: scale transformations (e.g., log)
4. COORD: a coordinate system (e.g., polar)
5. ELEMENT: graphs (e.g., points) and their aesthetic attributes (e.g., color)
6. GUIDE: one or more guides (axes, legends, etc.).

Wilkinson, Leland. (2005). *The grammar of graphics (2nd ed)*. New York: Springer.

Why ggplot2 instead of base R?

- nice defaults
- easy faceting
- (arguably) more natural syntax
- can switch chart types more easily

“Why I use ggplot2”, David Robinson

<http://varianceexplained.org/r/why-i-use-ggplot2/>

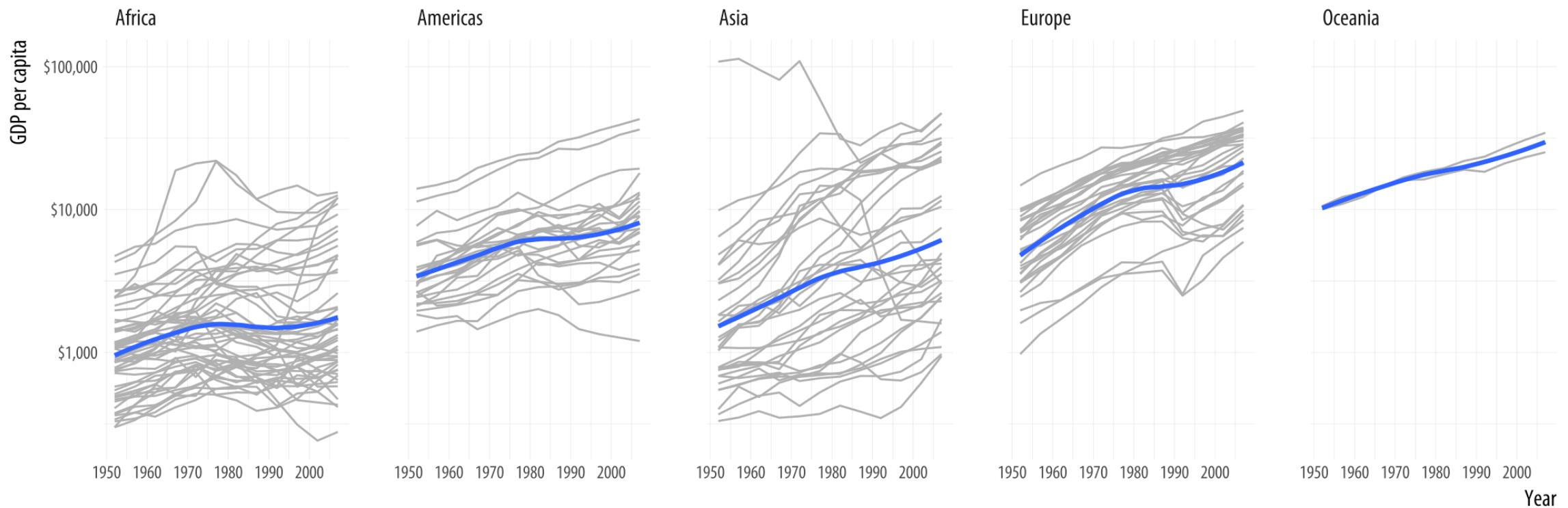
R vs. Excel, Tableau, etc.

Questions to ask:

- Are you already using R? Why switch?
- Are you going to have to share this process or reproduce it? Try R!
- Is it a quick project, or will others work on it? Maybe Excel is fine.
- Do you need to try a bunch of charts quickly, build interactive components, etc? Tableau might be more powerful and faster.

ggplot2 examples

GDP per capita on Five Continents

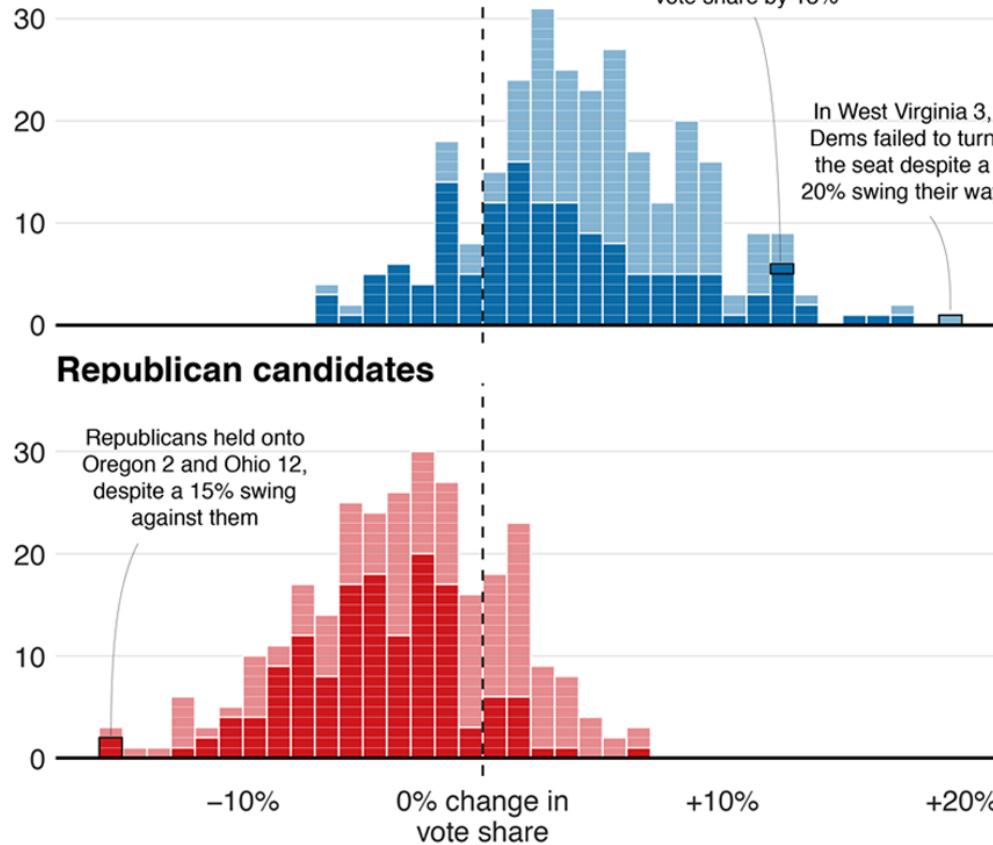


<http://socviz.co/groupfacettx.html>

Blue wave

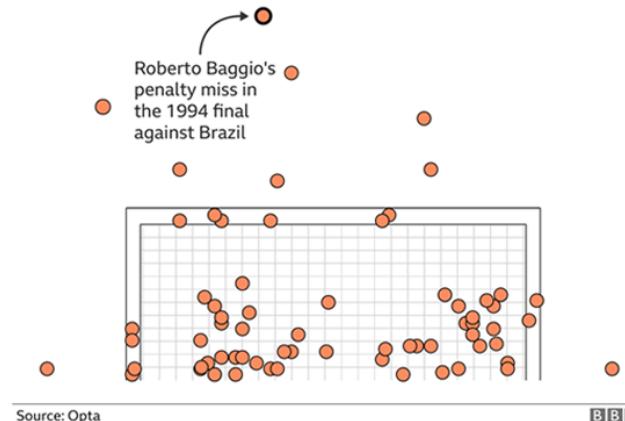
■ Won seat ■ Didn't win

Democrat candidates



Where penalties are saved

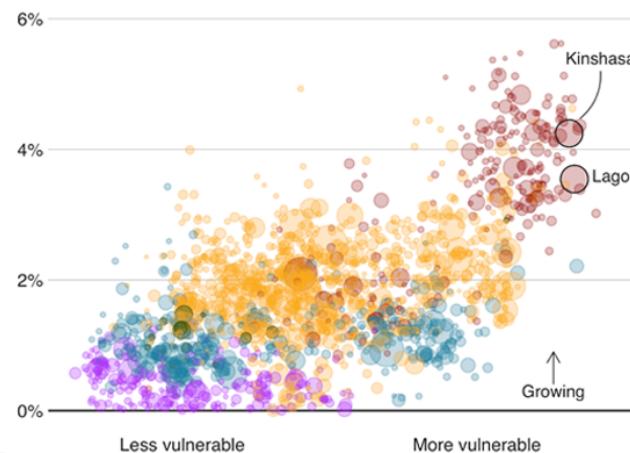
World Cup shootout misses and saves, 1982-2014



Fast-growing cities face worse climate risks

Population growth 2018-2035 over climate change vulnerability

■ Africa ■ Asia ■ Americas ■ Europe ■ Oceania

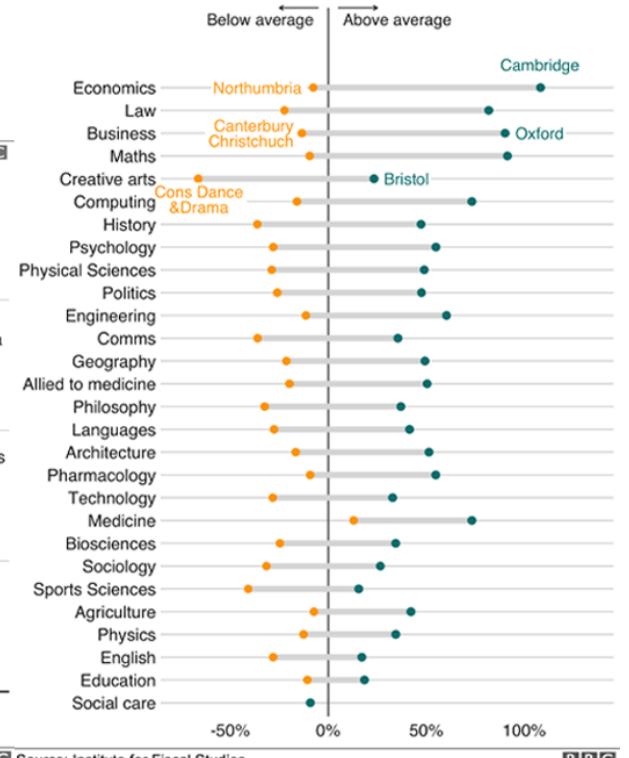


MPs rejected Theresa May's deal by 230 votes



Source: Commons Votes Services. Excludes 'tellers', the Speaker and deputies

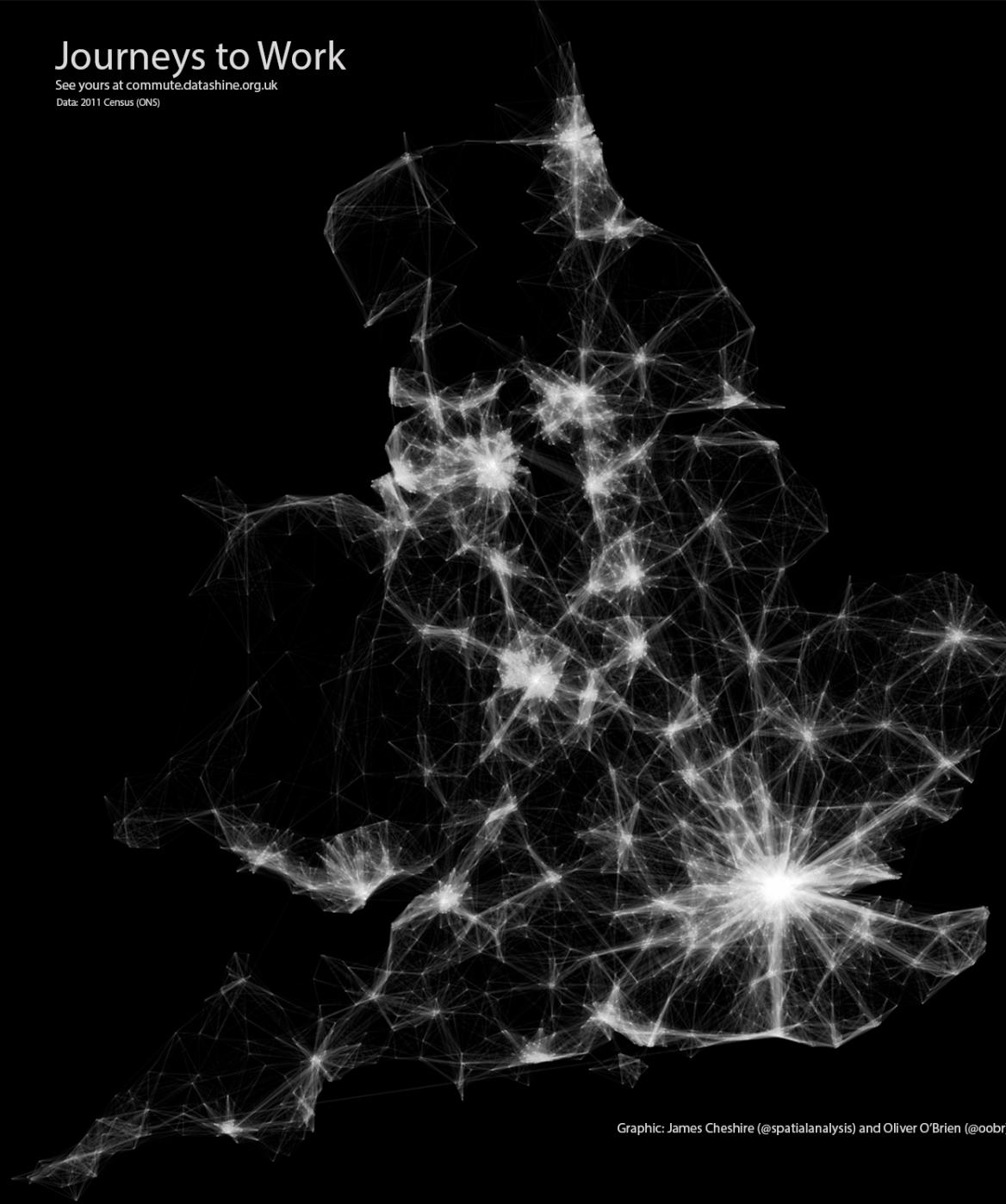
Earnings vary across unis even within subjects
Impact on men's earnings relative to the average degree



Journeys to Work

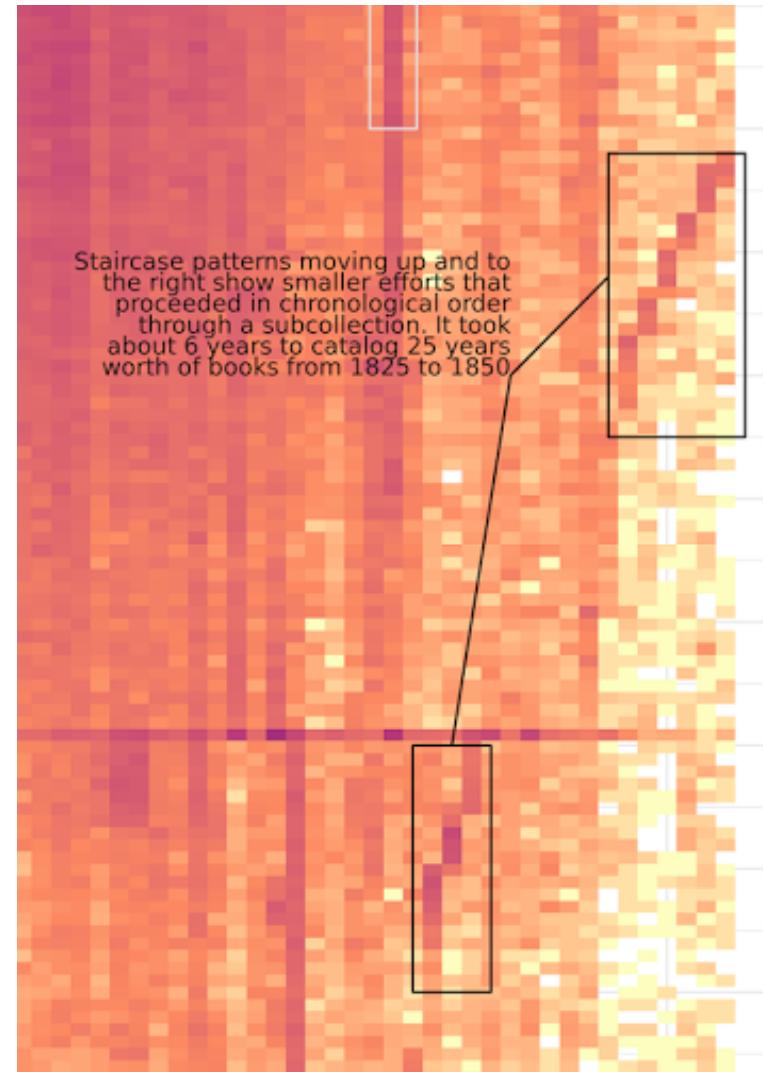
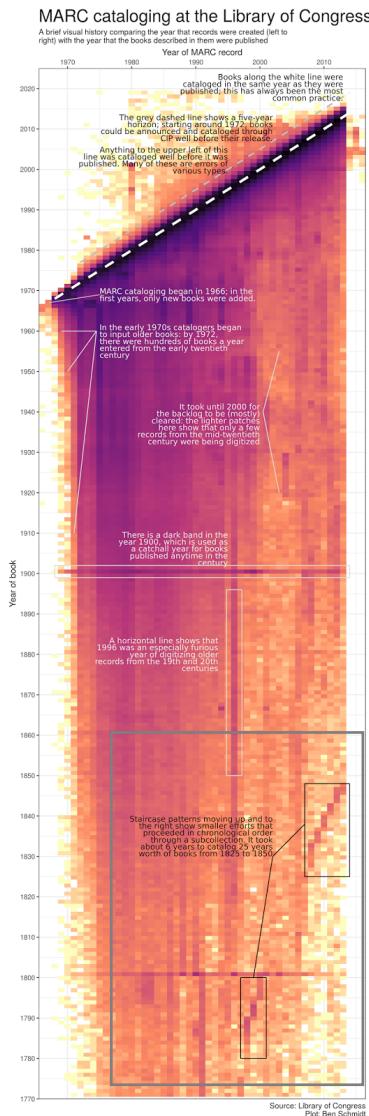
See yours at commute.datashine.org.uk

Data: 2011 Census (ONS)



Graphic: James Cheshire (@spatialanalysis) and Oliver O'Brien (@ooibr)

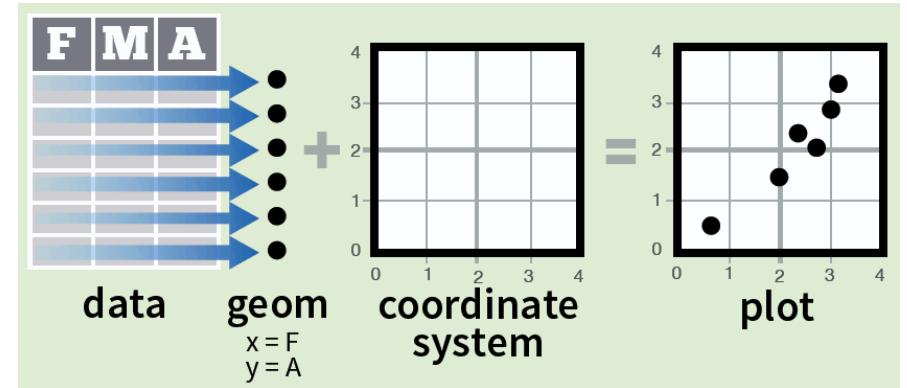
<http://spatial.ly/2015/03/mapping-flows/>



ggplot2: Elements

Basic elements in any ggplot2 visualization

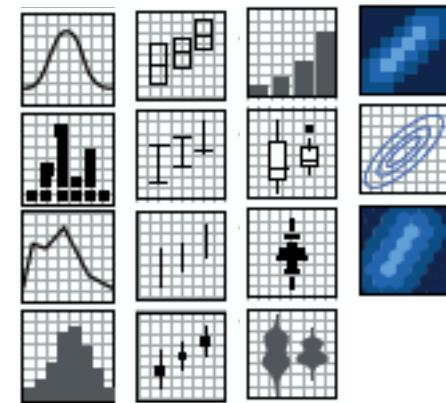
- **data**
- **aesthetics**
(variable mappings)
- **geom**
(chart type or shape)
- coordinate system
(the arrangement of the marks;
most geoms use default, cartesian)



<http://bit.ly/ggplot2-cheatsheet>

Types of geoms

- geom_bar()
- geom_point()
- geom_histogram()
- geom_map()
- etc.



<http://bit.ly/ggplot2-cheatsheet>

Note: some geoms also include data summary functions.
e.g., the “bar” geom will count data points in each category.

ggplot2: Basic syntax

Required functions:

Main function
`ggplot()`

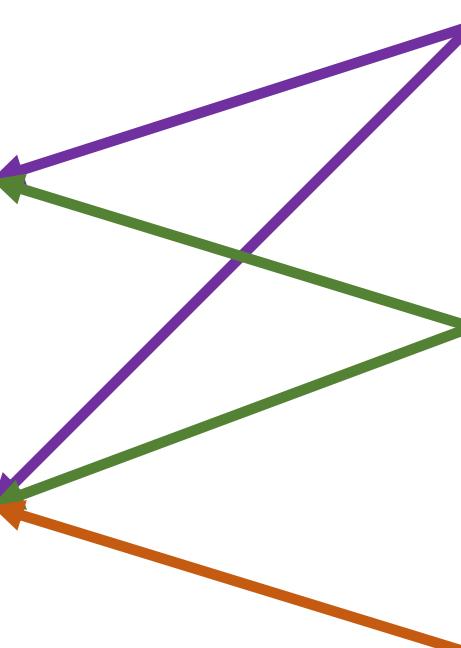
Shape layer
e.g., `geom_bar()`,
`geom_point()`

Required elements:

Data frame
name of df

Aesthetics mapping
`aes()`

Extra settings
anything not
attached to data



Template for a simple plot

**Main
function**

```
ggplot( data = data frame ) +
```

**Shape
layer**

```
geom_... ( mapping = aes(variable mappings) ,  
          non-variable adjustments )
```

Step-by-step

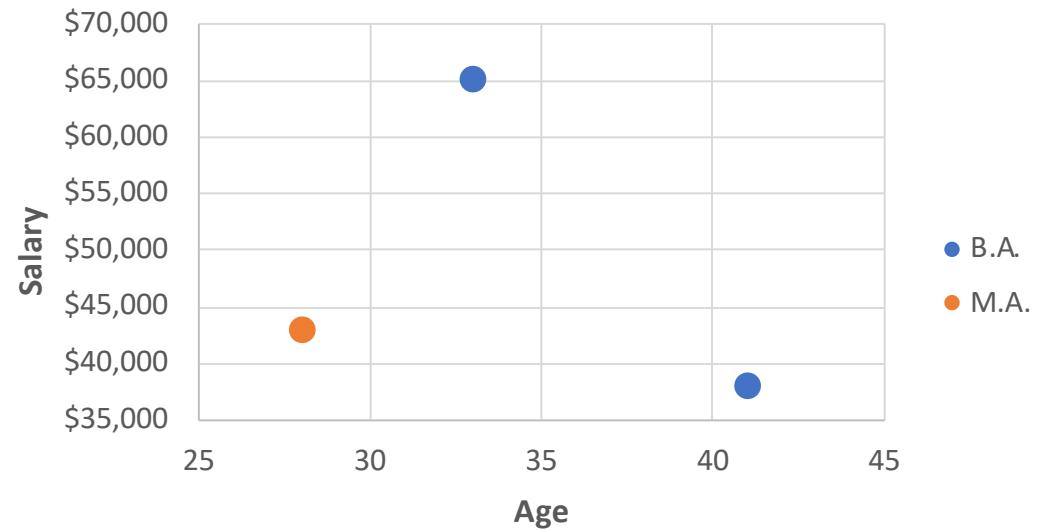
1. Set the data
2. Choose a shape layer
3. Map variables to aesthetics
4. Add non-variable adjustments

1. Set the data

“employees”

| Name | Age | Salary | Highest Degree |
|----------------|-----|----------|----------------|
| Jane Smith | 33 | \$65,000 | B.A. |
| Abby Jones | 28 | \$43,000 | M.A. |
| Bridget Carden | 41 | \$38,000 | B.A. |

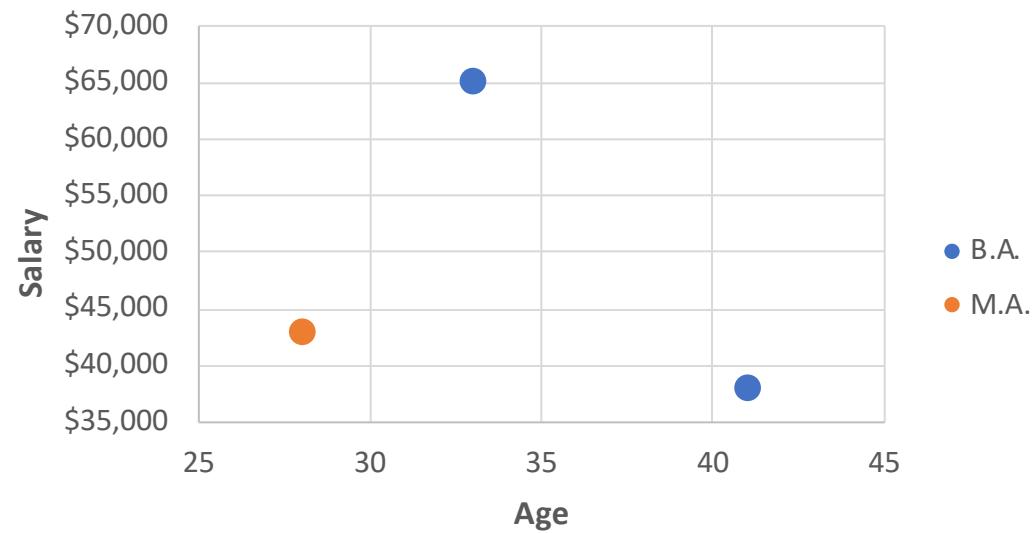
```
ggplot(employees)
```



2. Choose a shape layer

| Name | Age | Salary | Highest Degree |
|----------------|-----|----------|----------------|
| Jane Smith | 33 | \$65,000 | B.A. |
| Abby Jones | 28 | \$43,000 | M.A. |
| Bridget Carden | 41 | \$38,000 | B.A. |

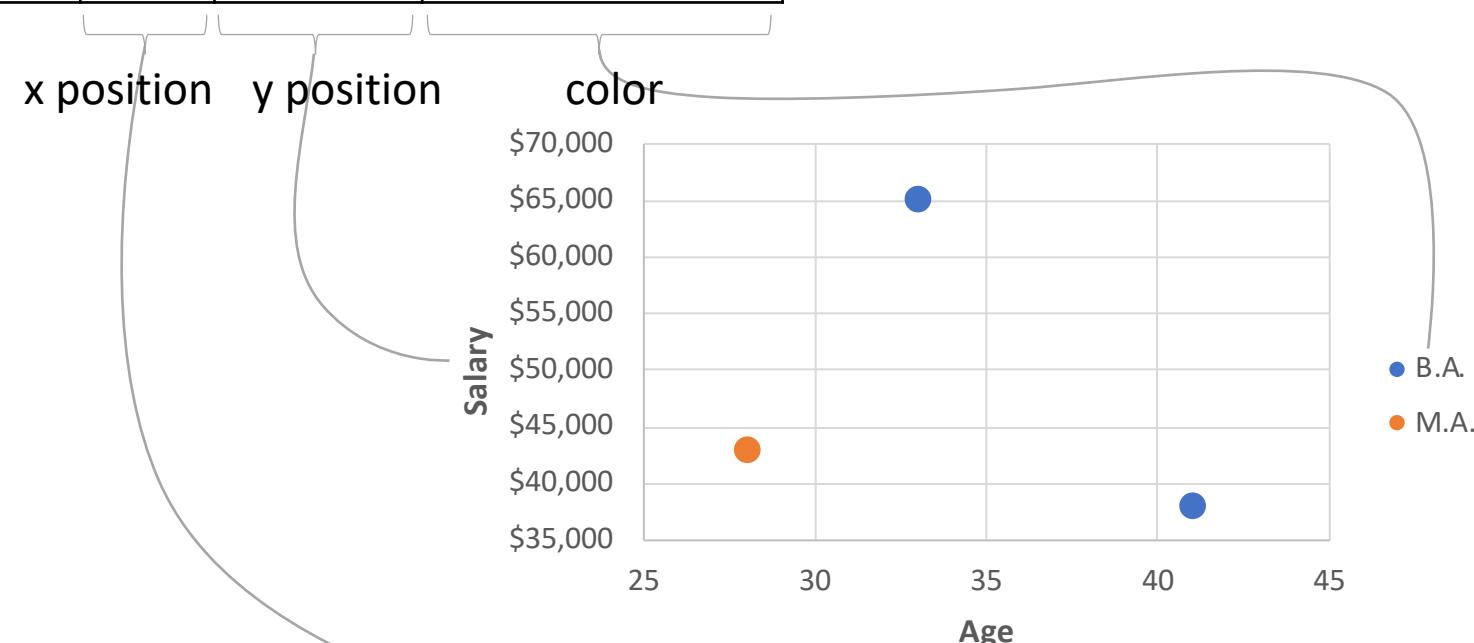
```
ggplot(employees) +  
  geom_point()
```



3. Map variables to aesthetics

| Name | Age | Salary | Highest Degree |
|----------------|-----|----------|----------------|
| Jane Smith | 33 | \$65,000 | B.A. |
| Abby Jones | 28 | \$43,000 | M.A. |
| Bridget Carden | 41 | \$38,000 | B.A. |

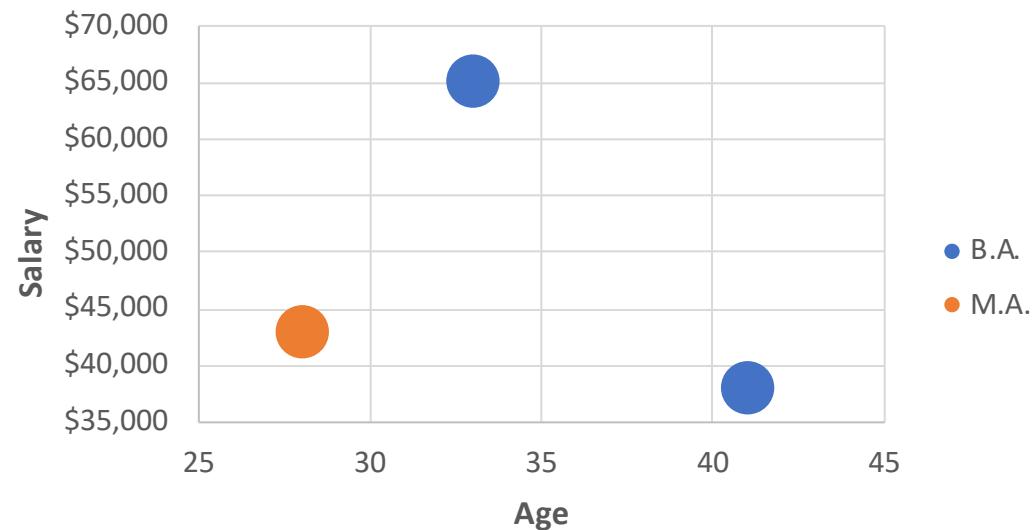
```
ggplot(employees) +  
  geom_point(  
    aes(x=age,  
        y=salary,  
        color=degree))
```



4. Add non-variable adjustments

| Name | Age | Salary | Highest Degree |
|----------------|-----|----------|----------------|
| Jane Smith | 33 | \$65,000 | B.A. |
| Abby Jones | 28 | \$43,000 | M.A. |
| Bridget Carden | 41 | \$38,000 | B.A. |

```
ggplot(employees) +  
  geom_point(  
    aes(x=age,  
        y=salary,  
        color=degree),  
    size=10)
```



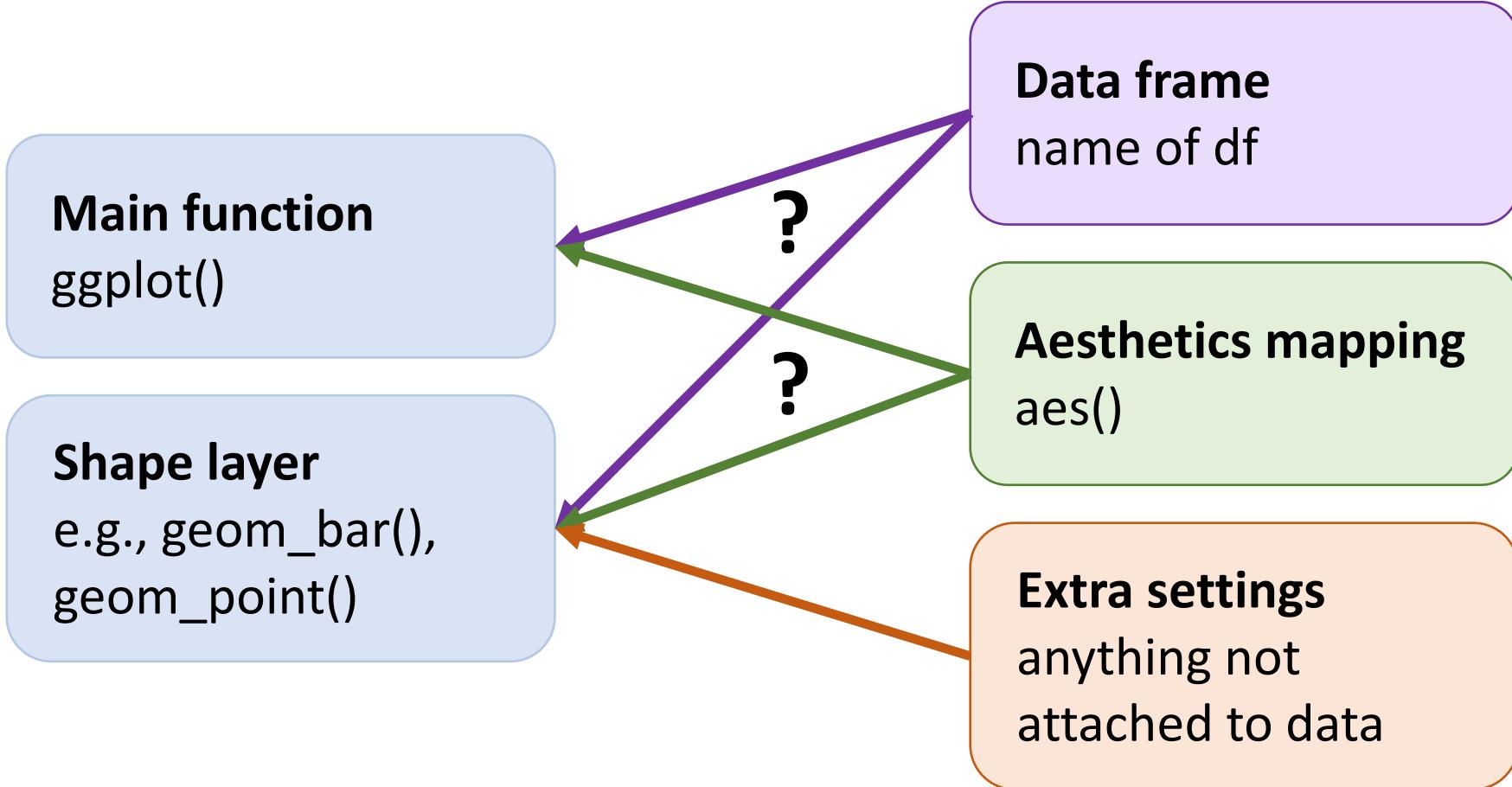
Template for a simple plot

**Main
function**

```
ggplot( data = data frame ) +
```

**Shape
layer**

```
geom_... ( mapping = aes(variable mappings) ,  
          non-variable adjustments )
```



Inheritance

data and aesthetics will carry through from main function to shape layers

**Main
function**

```
ggplot( data = data frame,  
        mapping = aes(variable mappings) )
```

+

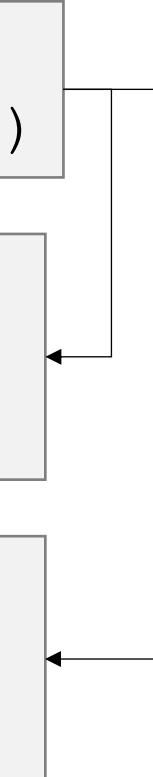
**Shape
layer**

```
geom_... ( aes(add'l variable mappings),  
           non-variable adjustments )
```

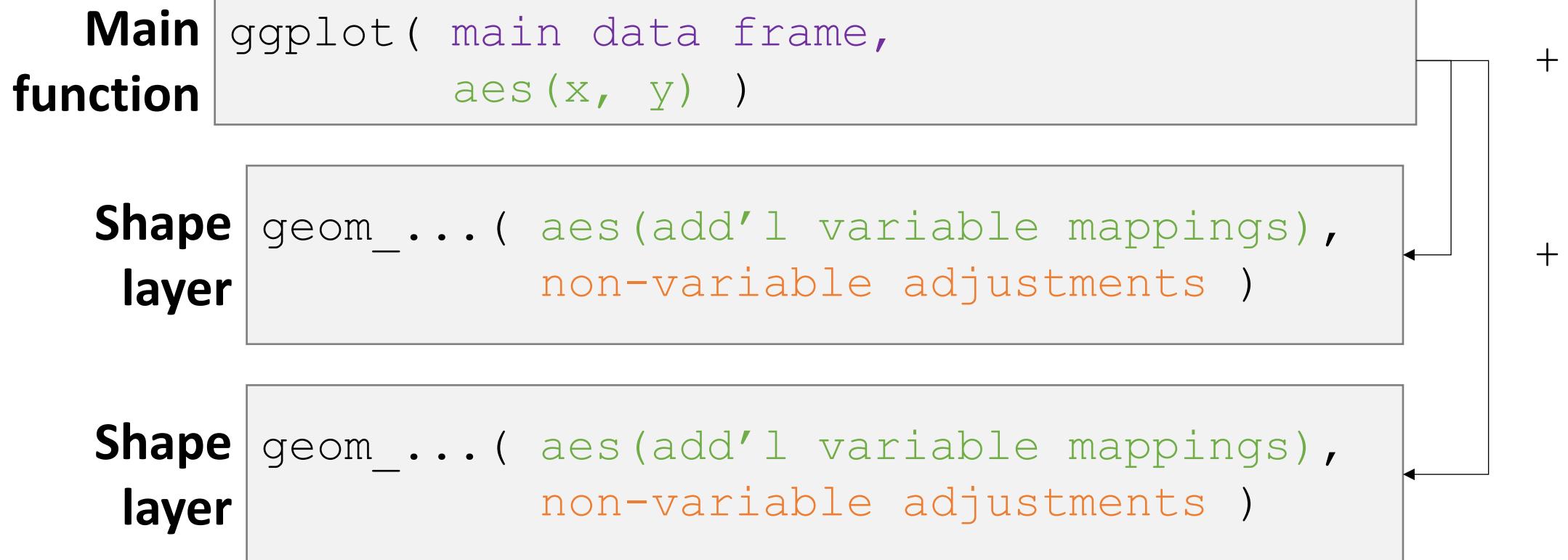
+

**Shape
layer**

```
geom_... ( aes(add'l variable mappings),  
           non-variable adjustments )
```



General inheritance strategy



Additional decisions

Adding something that will appear
inside the **chart coordinate space**?

You will (almost always) be adding a **geom**!

Changing the way a **variable is displayed**?
(e.g., different axis breaks, different color mapping)

You will be adding a **scale**!

Changing the **look and feel** of the chart?

You will be adding or making changes to a **theme**!

Working in RStudio

Get workshop files

URL: <https://github.com/amzoss/RVis-DM2019>

With Git installed

In RStudio:

- Project → New project
- Version Control
- Git
 - Paste in GitHub URL
 - Project directory name:
RVis-DM2019
 - Subdirectory: you choose
- Create Project

Without Git installed

• Click green button to download ZIP

• Unzip files on your laptop

In RStudio:

- Project → New project...
- Existing directory
- Select unzipped folder
- Create Project

Problems?

Try RStudio Cloud:

<http://bit.ly/RVisCloud>

Using RStudio

- Projects
- Rmarkdown
- Cheat sheets

Try right now:
Open RStudio
Try running “library(tidyverse)”
Tell me about any errors

<https://www.rstudio.com/resources/cheatsheets/#rmarkdown>

ggplot2 Cheat Sheet

Data Visualization with ggplot2 :: CHEAT SHEET



Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data set**, a **coordinate system**, and **geoms**—visual marks that represent data points.

To display values, map variables in the data to visual properties of the geom's **aesthetics** like **size**, **color**, and **x** and **y** locations.

Complete the template below to build a graph.

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(<mapping> = aes(<MAPPINGS>),  
  stat = <STAT>, position = <POSITION>) +  
  <COORDINATE_FUNCTION> +  
  <FACET_FUNCTIONS> +  
  <SCALE_FUNCTIONS> +  
  <THEME_FUNCTIONS>
```

ggplot(data = mpg, aes(x = cyl, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

- aesthetic mappings**: `data` | `geom`
- `qplot(x = cyl, y = hwy, data = mpg, geom = "point")` Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.
- `last_plot()` Returns the last plot
- `ggsave("plot.png", width = 5, height = 5)` Saves last plot as 5'x5' file named "plot.png" in working directory. Matches file type to file extension.

Geoms Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

```
a <- ggplot(economics, aes(date, unemployed))  
b <- ggplot(seals, aes(x = long, y = lat))  
  
a + geom_blank()  
(Useful for expanding limits)  
b + geom_curve(aes(yend = lat + 1,  
  nudge_y = -1, check_overlap = TRUE), x, y, label,  
  alpha, angle, color, curvature, linetype, size,  
  linemetre=1)  
a + geom_path(linend = "butt", linejoin = "round",  
  x, y, alpha, color, group, linetype, size)  
a + geom_polygon(aes(group = group))  
x, y, alpha, color, fill, group, linetype, size  
b + geom_rect(aes(xmin = long, ymin = lat, xmax =  
  long + 1, ymax = lat + 1), x, y, alpha, color, fill,  
  group, linetype, size)  
a + geom_ribbon(aes(ymin = unemployed - 900,  
  ymax = unemployed + 900), x, y, alpha, color, fill,  
  group, linetype, size)  
a + geom_smooth(method = lm), x, y, alpha,  
  color, fill, group, linetype, size, weight  
a + geom_text(aes(label = cyl), nudge_x = 1,  
  nudge_y = -1, check_overlap = TRUE), x, y, label,  
  alpha, angle, color, fontface, hjust,  
  linheight, size, vjust
```

TWO VARIABLES

```
c <- ggplot(mpg, aes(class, hwy))  
d <- data.frame(grp = c("A", "B"), fit = 4.5, se = 1.2)  
e <- ggplot(mpg, aes(class, hwy))  
f <- geom_col(), x, y, alpha, color, fill, group, linetype, size  
f + geom_boxplot(), x, y, lower, middle, upper,  
  ymax, ymin, alpha, color, fill, group, linetype, size  
f + geom_dotplot(binaxis = "y", stackdir = "center"), x, y, alpha, color, fill,  
  group, linetype, size  
f + geom_violin(scale = "area"), x, y, alpha, color,  
  fill, group, linetype, size, weight
```

continuous x, continuous y

```
f + geom_crossbar(fatten = 2)  
j + geom_errorbar(fatten = 2)  
j + geom_errorbar(l), x, y, max, min, alpha, color,  
  group, linetype, size, width (also  
  geom_errorbarl())  
j + geom_linerange()  
j + geom_pointrange()  
j + geom_pointrange()
```

discrete x, continuous y

```
j + geom_bar(), x, y, alpha, color, fill, group, linetype, size  
k + ggplot(data, aes(fill = murder))  
k + geom_map(aes(map_id = state), map = map)  
  + expand_limits(x = mapLong, y = mapLat),  
  map_id, alpha, color, fill, group, linetype, size
```

ONE VARIABLE

```
c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)  
c + geom_area(stat = "bin")  
x, y, alpha, color, fill, linetype, size  
c + geom_density(kern = "gaussian")  
x, y, alpha, color, fill, group, linetype, size, weight  
c + geom_dotplot()  
x, y, alpha, color, fill  
c + geom_freqpoly()  
x, y, alpha, color, group, linetype, size  
c + geom_histogram(binwidth = 5), x, y, alpha,  
  color, fill, linetype, size, weight  
c2 + geom_qq(aes(sample = hwy))  
x, y, alpha, color, fill, shape,  
  size, stroke
```

discrete x, discrete y

```
g + geom_count(), x, y, alpha, color, fill, shape,  
  size, stroke
```

THREE VARIABLES

```
seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))  
l <- ggplot(seals, aes(long, lat))  
l + geom_raster(aes(fill = z), hijust = -0.5, vjust = 0.5,  
  interpolate = FALSE)  
x, y, z, alpha, colour, group, linetype, size, weight  
l + geom_tile(aes(fill = z)), x, y, alpha, color, fill,  
  linetype, size, width
```

RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more at <http://ggplot2.tidyverse.org> • ggplot2 2.1.0 • Updated: 2016-11



<https://www.rstudio.com/resources/cheatsheets/#ggplot2>

Notes about RMarkdown

- Comments work like HTML
`<!--HTML Comment style -->`
- Practice a few non-code markdown elements (see cheat sheet)

Debugging code

- Start simple
 - If you see an error:
 - read error message for hints
 - check for problems with spelling/punctuation marks
 - Get code to run without errors
 - Check result to see if it makes sense
- 
- Add a small change
 - Get code to run without errors
 - Check result to see if it makes sense
 - etc.

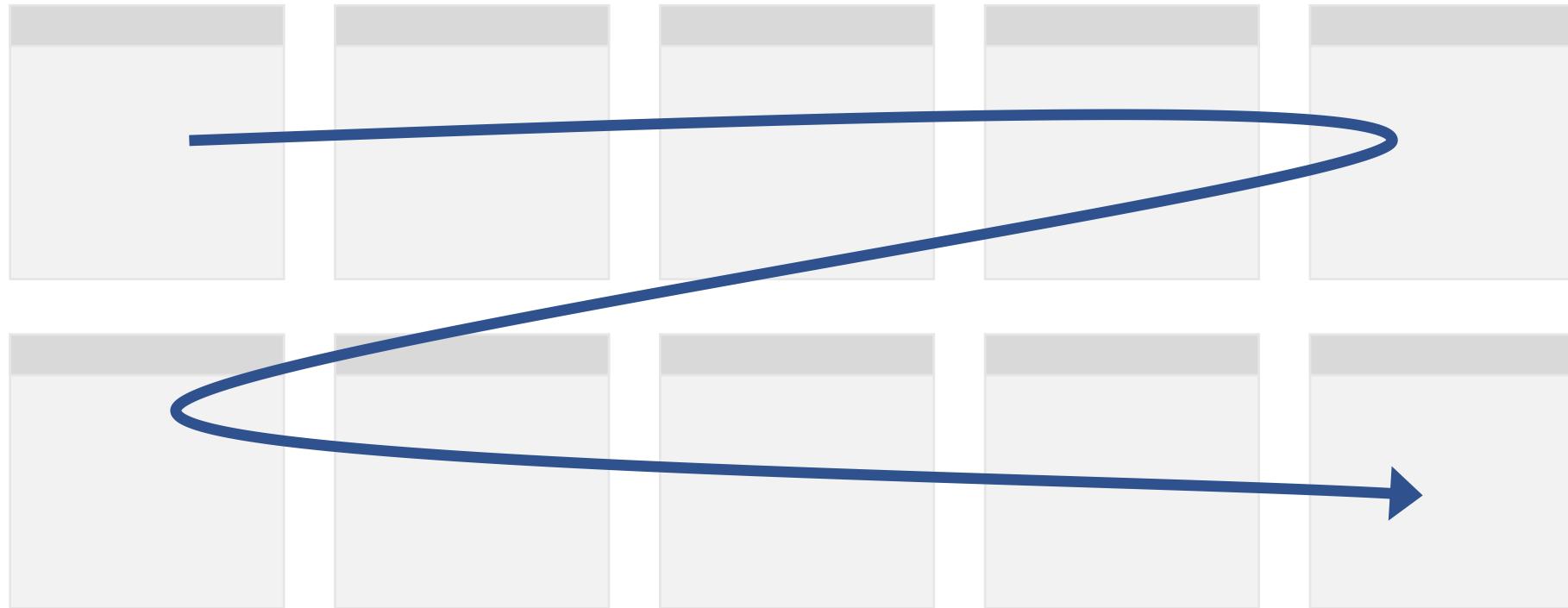
Dataset 1: Game of Thrones character ratings

[https://www.nytimes.com/interactive/2017/08/09/upshot/game-of-thrones-
chart.html](https://www.nytimes.com/interactive/2017/08/09/upshot/game-of-thrones-chart.html)

Creating repeated charts

facet_wrap()

```
+ facet_wrap(vars(variable))
```

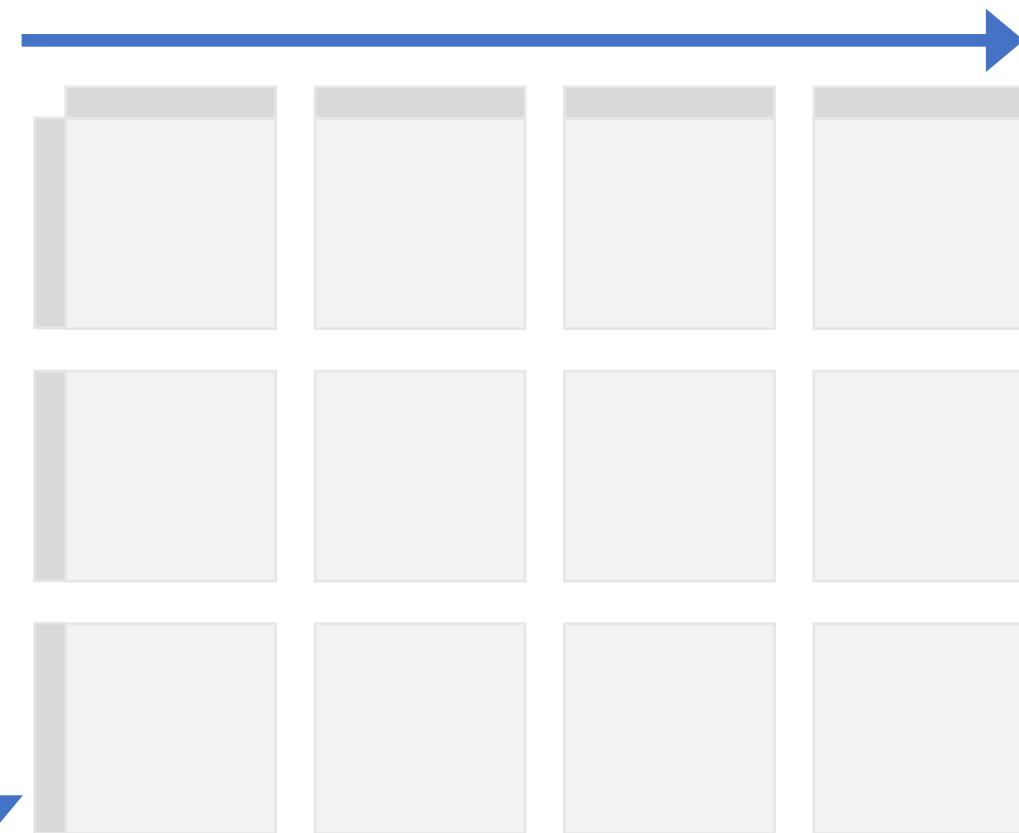


facet_grid()

```
+ facet_grid(rows=vars(yvar,  
cols=vars(xvar))
```

Another categorical variable

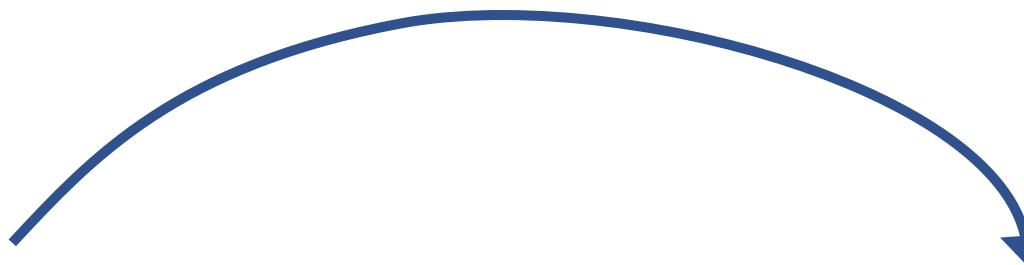
One categorical variable



Helpful data manipulation

About %>%

- Loads automatically with tidyverse
- Used throughout tidyverse (except for ggplot2)
- Pushes data from the left into the function on the right



data_frame %>% function(args)

A blue curved arrow originates from the left side of the word "data_frame" and points to the right side of the word "function".

filter

Select a subset of rows

```
data %>% dplyr::filter(name == "John")
```

same as

```
dplyr::filter(data, name == "John")
```

<https://www.rstudio.com/resources/cheatsheets/#dplyr>

select

Select a subset of columns (many options!)

```
data %>% dplyr::select(id, name, age)
```

```
data %>% dplyr::select(-count)
```

<https://www.rstudio.com/resources/cheatsheets/#dplyr>

drop_na

Remove rows with NA values, either in any column or in specified columns

```
data %>% drop_na()
```

```
data %>% drop_na(age)
```

<https://www.rstudio.com/resources/cheatsheets/#import>

Dataset 2: Star Wars character data

<https://dplyr.tidyverse.org/reference/starwars.html>

Working with text variables

Text variables

In R, “character” variables

| Gender | Age | Household Income | Education |
|----------|----------|-----------------------|----------------------------------|
| Response | Response | Response | Response |
| Male | 18-29 | | High school degree |
| Male | 18-29 | \$0 - \$24,999 | Bachelor degree |
| Male | 18-29 | \$0 - \$24,999 | High school degree |
| Male | 18-29 | \$100,000 - \$149,999 | Some college or Associate degree |
| Male | 18-29 | \$100,000 - \$149,999 | Some college or Associate degree |
| Male | 18-29 | \$25,000 - \$49,999 | Bachelor degree |
| Male | 18-29 | | High school degree |
| Male | 18-29 | | High school degree |
| Male | 18-29 | \$0 - \$24,999 | Some college or Associate degree |
| Male | 18-29 | \$25,000 - \$49,999 | Some college or Associate degree |
| Male | 18-29 | \$25,000 - \$49,999 | Bachelor degree |
| Male | 30-44 | \$50,000 - \$99,999 | Graduate degree |
| Male | 18-29 | | High school degree |
| Male | 18-29 | \$0 - \$24,999 | Some college or Associate degree |
| Male | 18-29 | \$50,000 - \$99,999 | Bachelor degree |

Factors

- Default ordering for categories: alphabetical
- Converting to factor allows you to:
 - Specify “levels” for a categorical variable
 - Specify the order of those levels
 - Specify whether the factor is “ordered”

<https://r4ds.had.co.nz/factors.html>

```
> x1 <- c("Dec", "Apr", "Jan",  
"Mar")
```

```
> factor(x1)  
[1] Dec Apr Jan Mar  
Levels: Apr Dec Jan Mar
```

```
> month_levels <- c( "Jan", "Feb",  
"Mar", "Apr", "May", "Jun", "Jul",  
"Aug", "Sep", "Oct", "Nov", "Dec" )
```

```
> y1 <- factor(x1,  
                levels = month_levels)  
> y1  
[1] Dec Apr Jan Mar
```

```
Levels: Jan Feb Mar Apr May Jun Jul  
Aug Sep Oct Nov Dec
```

forcats package: helpful functions

- `as_factor(char_var)`:
convert a character variable to a factor
- `fct_infreq(factor)`:
take factor levels and set the order according to
(inverse) category frequency
- `fct_reorder(factor, num_var)`:
sort factor levels by a second, numerical variable
(like a pre-calculated count or average)

<https://www.rstudio.com/resources/cheatsheets/#forcats>

Note about `read.csv` (base R)

- Converts string variables to factors by default
- Can either:
 - Include `stringsAsFactors=FALSE`
 - Use `read_csv()` instead

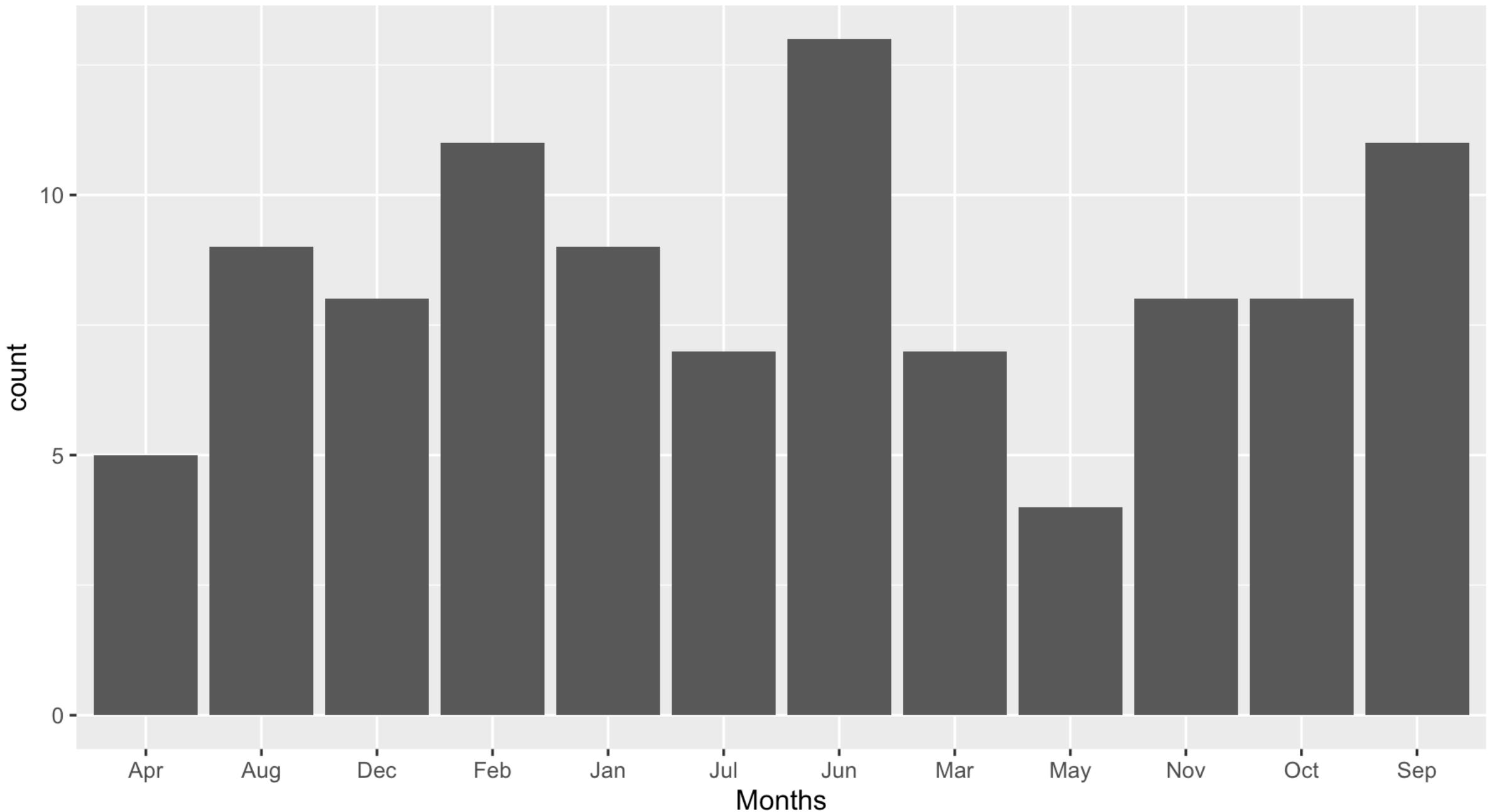
Factoring resources

From Amelia McNamara:

- RStudioConf 2019 slides:
[Working with Categorical Data in R Without Losing Your Mind](#)
- [Wrangling Categorical Data in R article](#)
- [Wrangling Categorical Data in R repository](#)

Design Principles for Text Variables

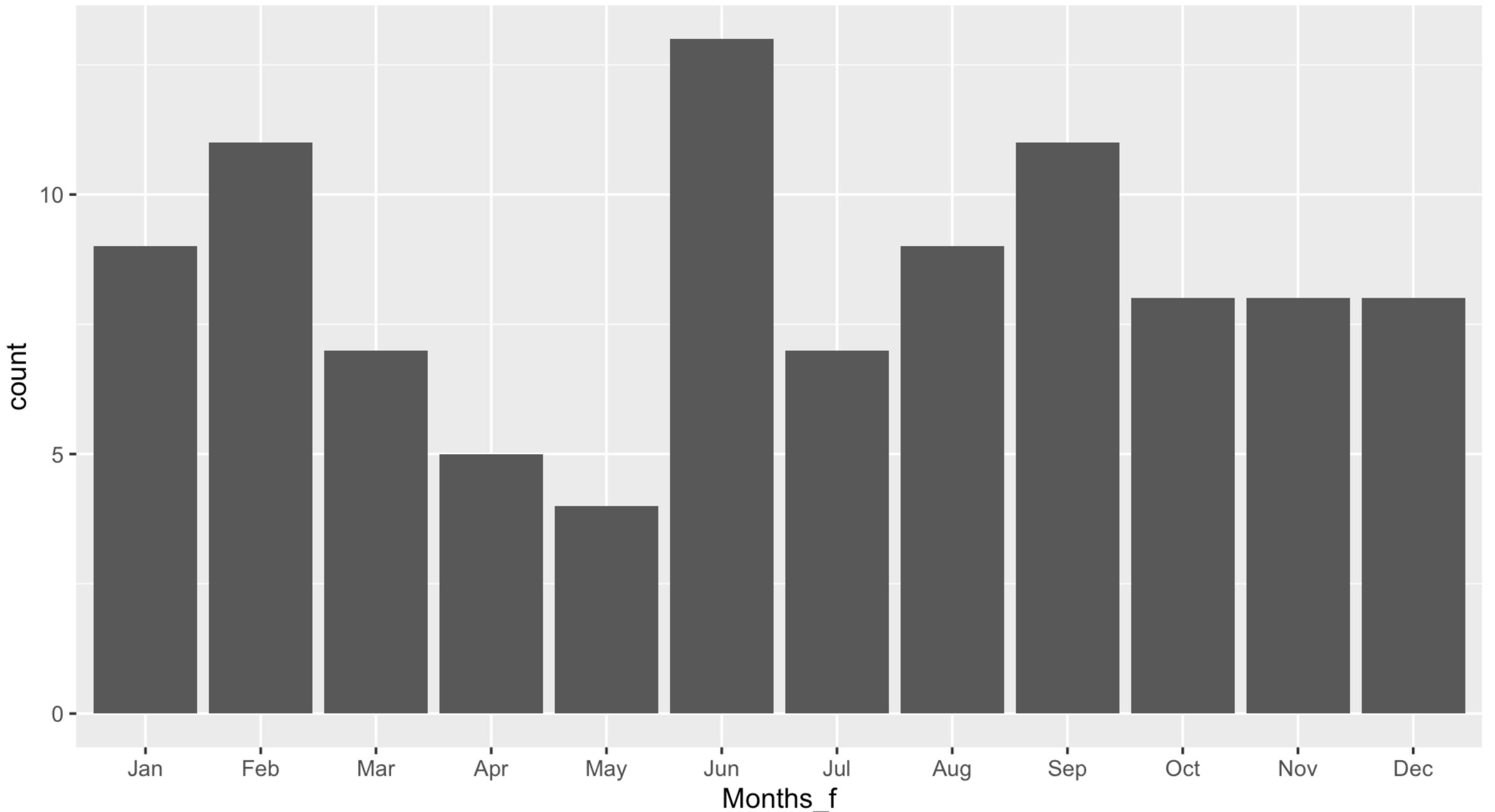
Principle 1: Order matters

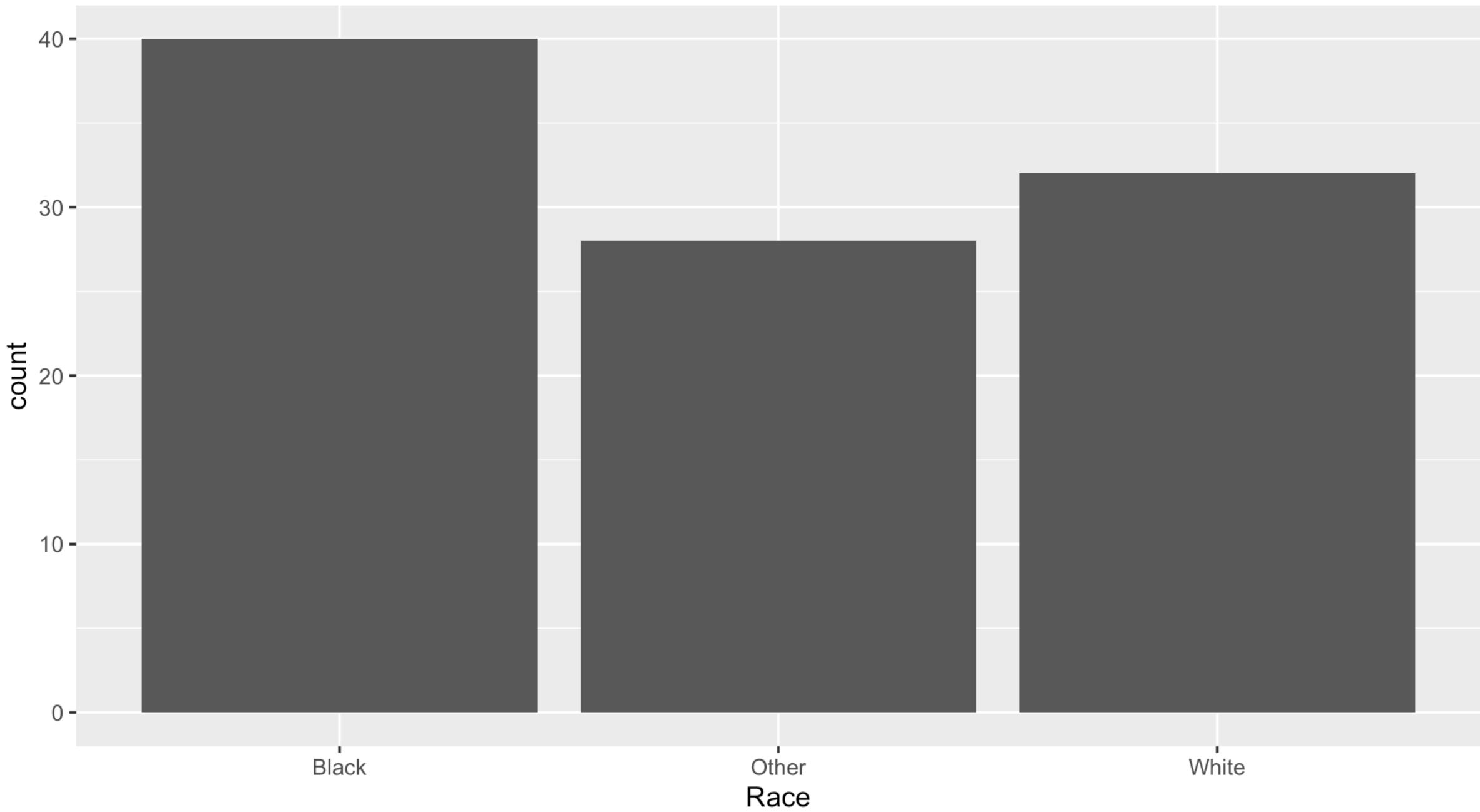


Order by meaning

```
month_levels <- c( "Jan", "Feb", "Mar", "Apr",
"May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov",
"Dec" )

data <- data %>%
  mutate(Months_f = Months %>%
    as_factor() %>%
    fct_relevel(month_levels) )
```

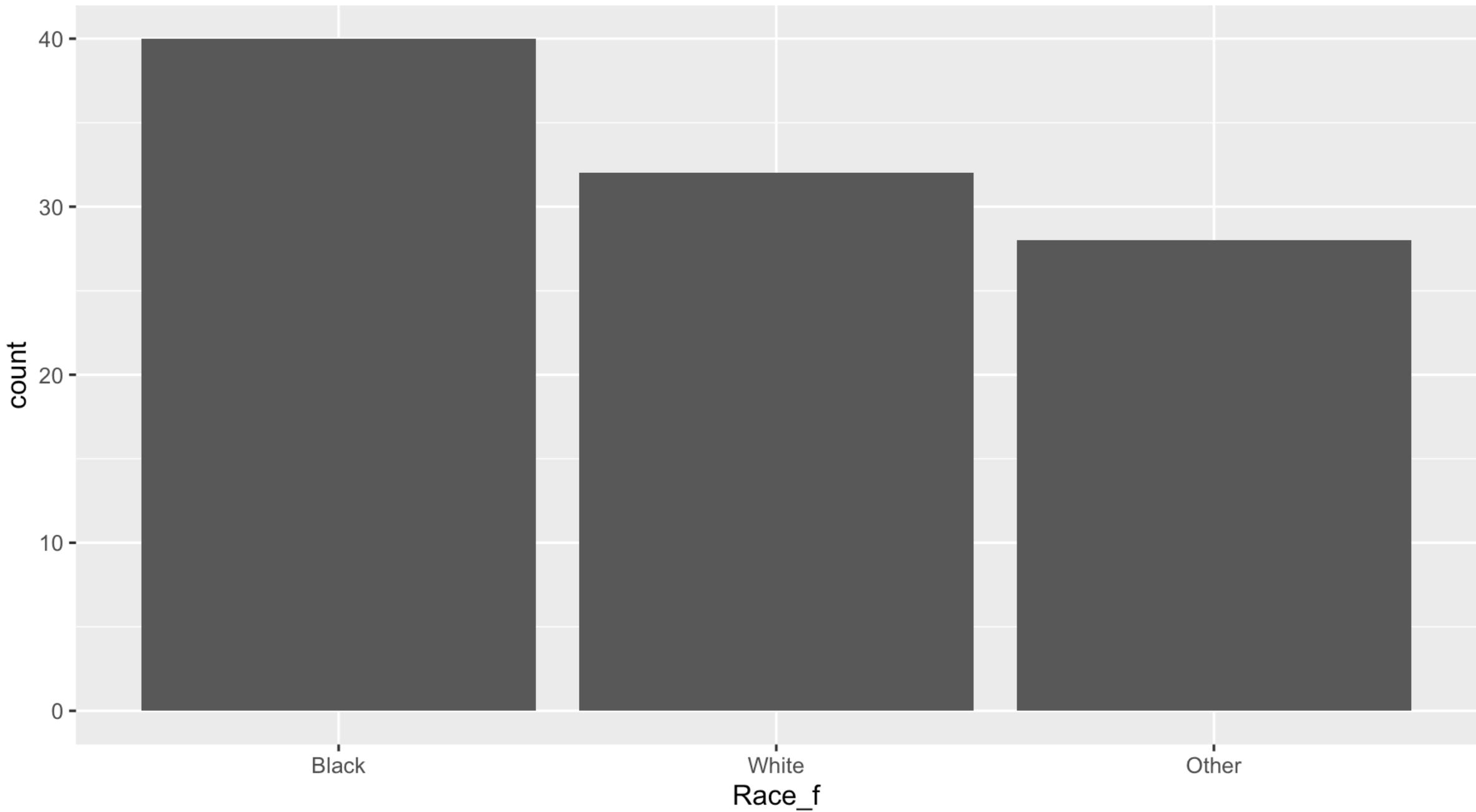




Order by value (usingforcats)

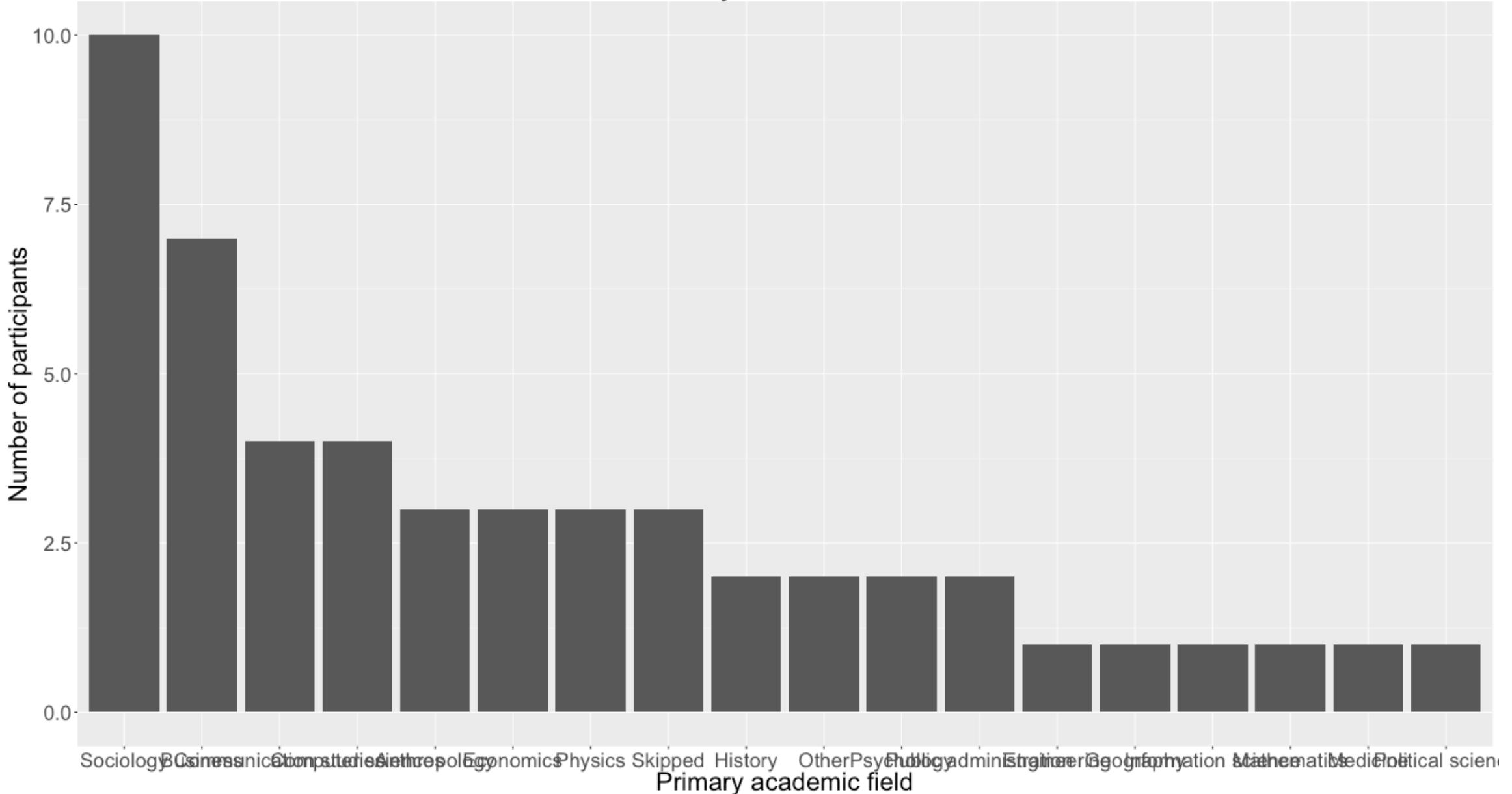
```
demo <- data %>%
  mutate(Race_f = Race %>%
          as_factor() %>%
          fct_infreq()))

ggplot(data,
       aes(Race %>%
             as_factor() %>%
             fct_infreq()) ) +
  geom_bar()
```



Principle 2:
Put long categories on y-axis

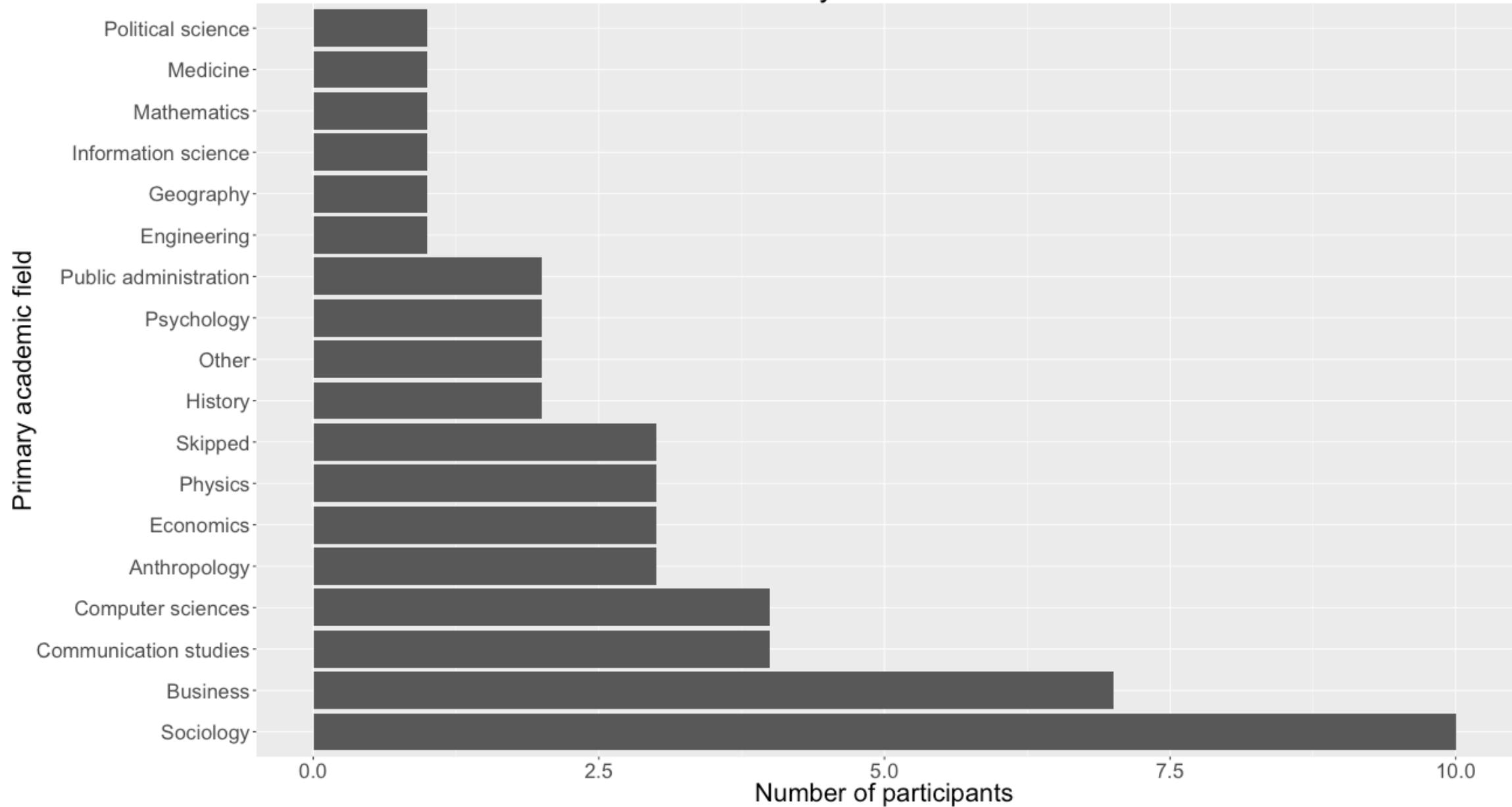
Primary academic field



Flip the axes

```
+ coord_flip()
```

Primary academic field



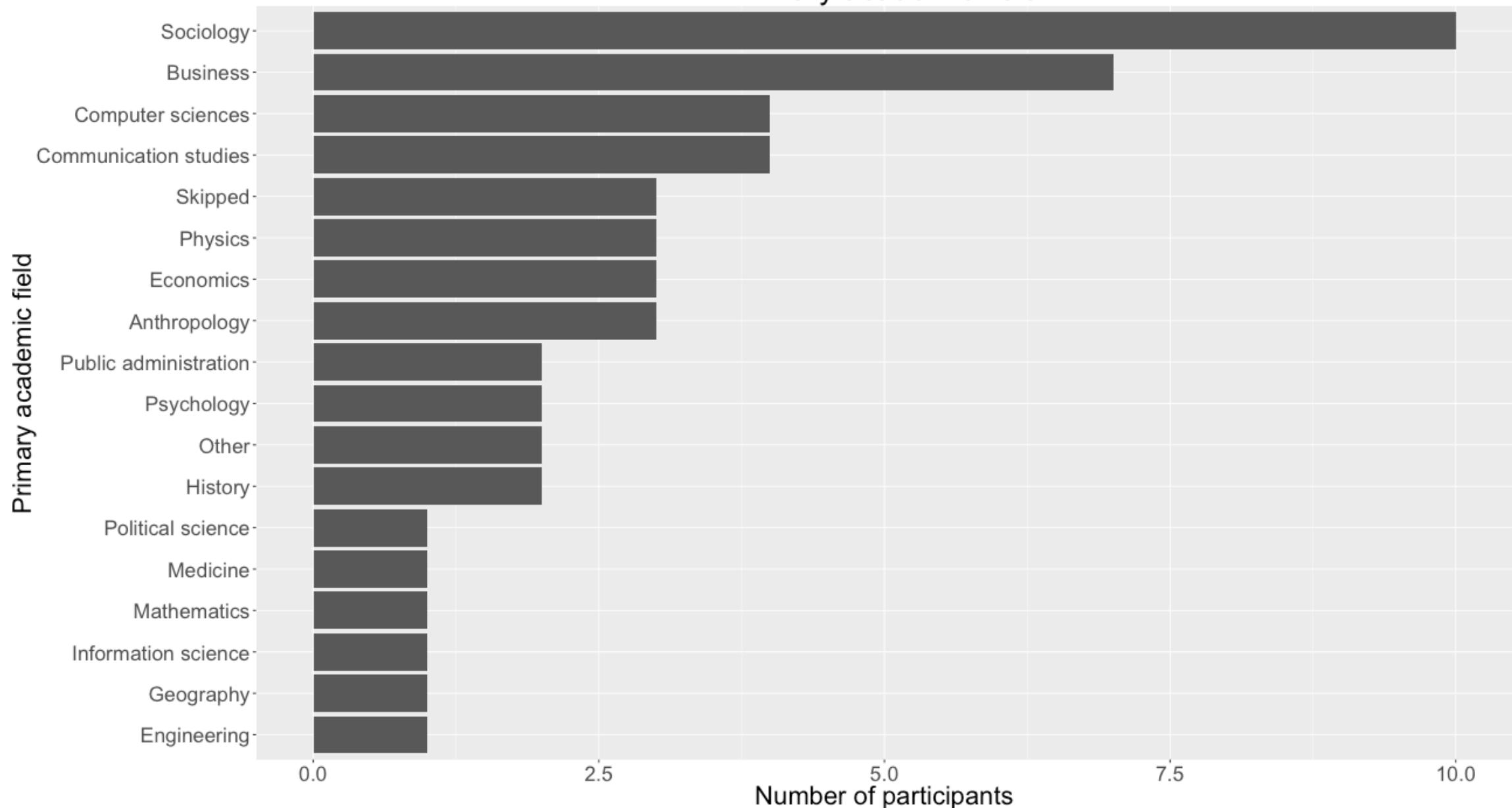
Oops!

```
data$academic_field <-  
  fct_rev(fct_infreq(  
    as_factor(data$academic_field) ))
```



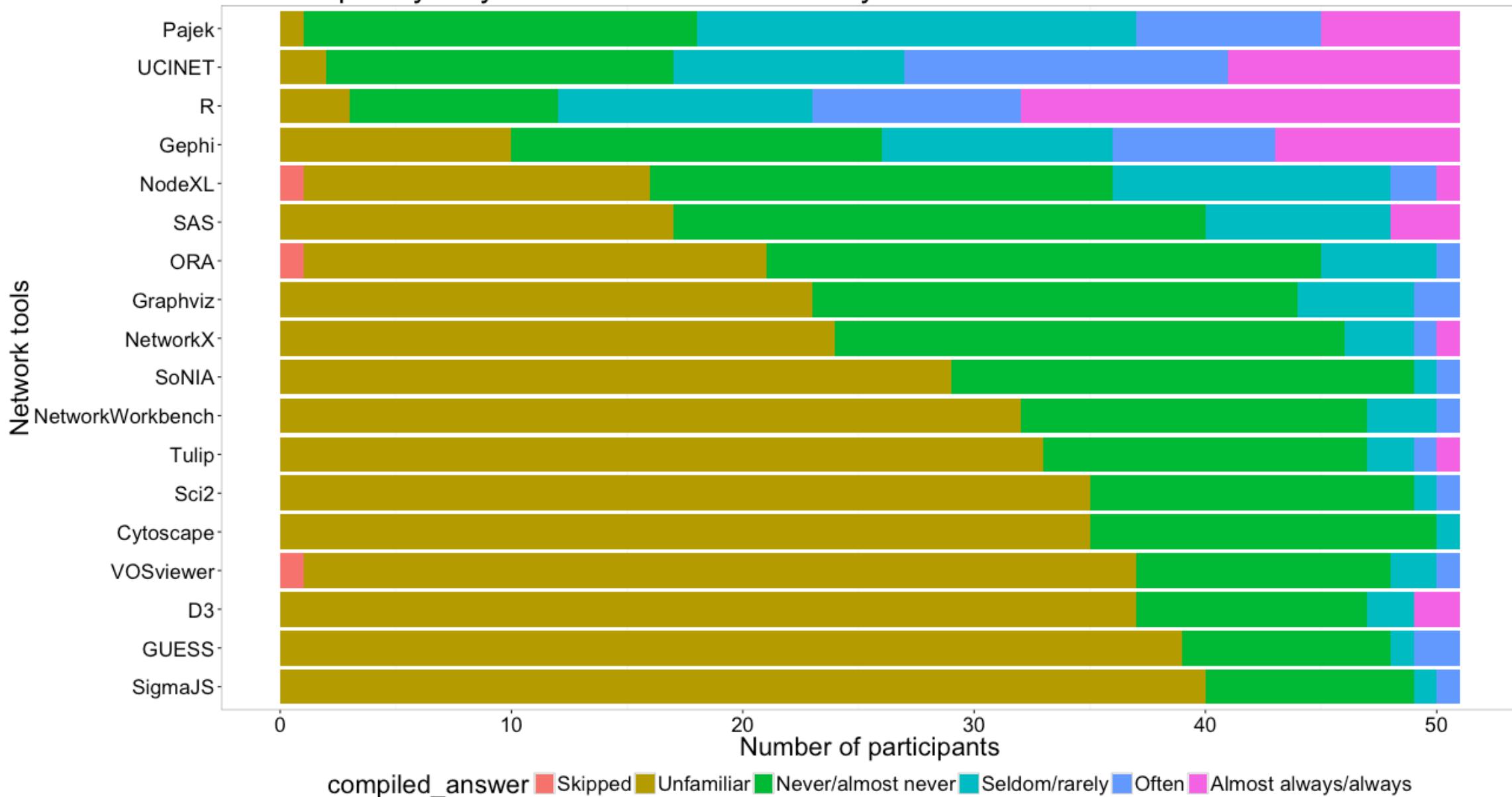
Have to reverse the
order of the levels

Primary academic field



Principle 3:
Select meaningful colors

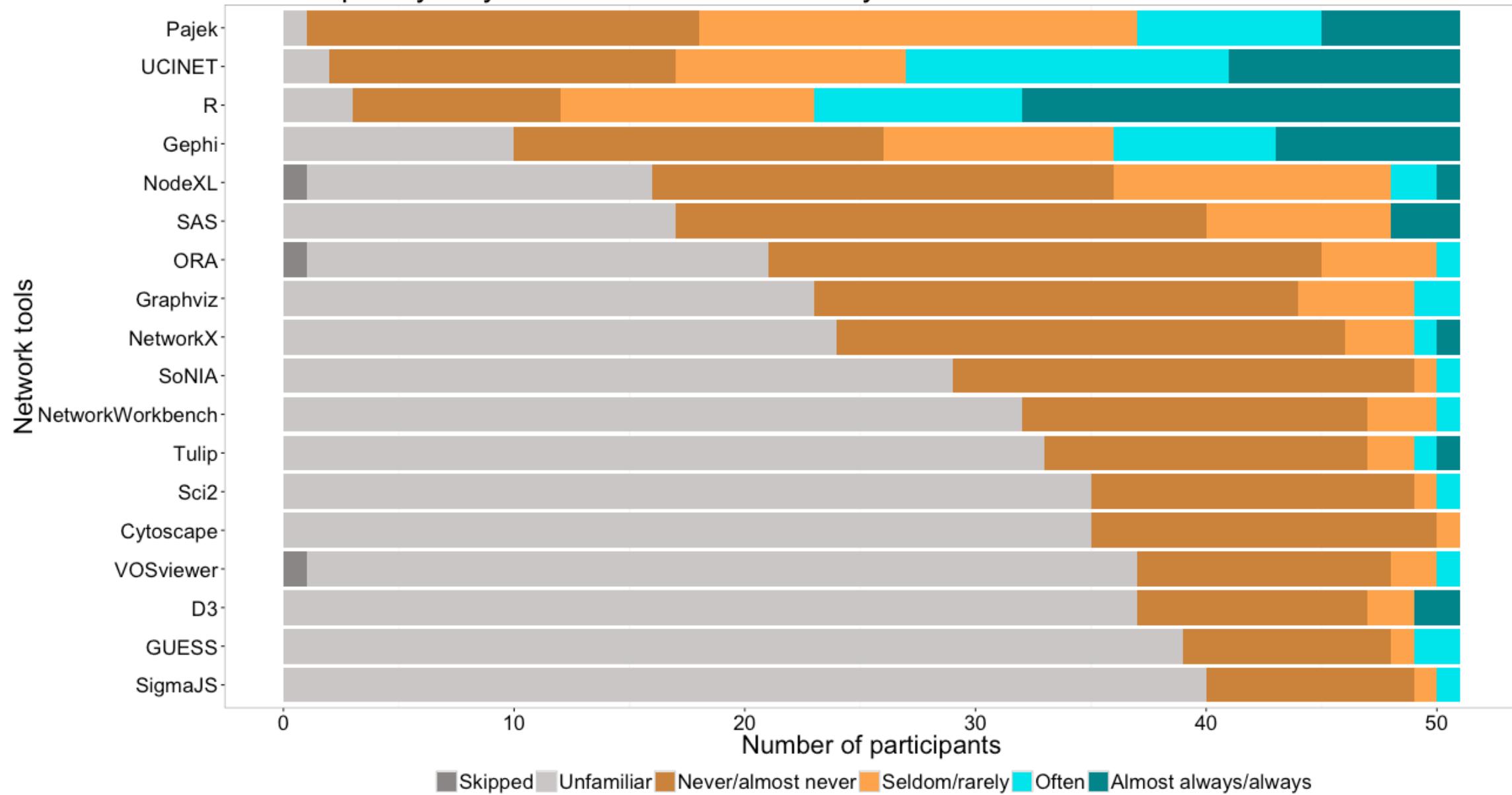
How frequently do you use these tools for analysis?



Select colors manually, or use alternate palette

```
scale_fill_manual(  
  values=c("snow4", "snow3",  
          "tan3", "tan1",  
          "turquoise2", "turquoise4"))  
  
# Also see package RColorBrewer  
scale_fill_brewer(palette="BrBG")
```

How frequently do you use these tools for analysis?



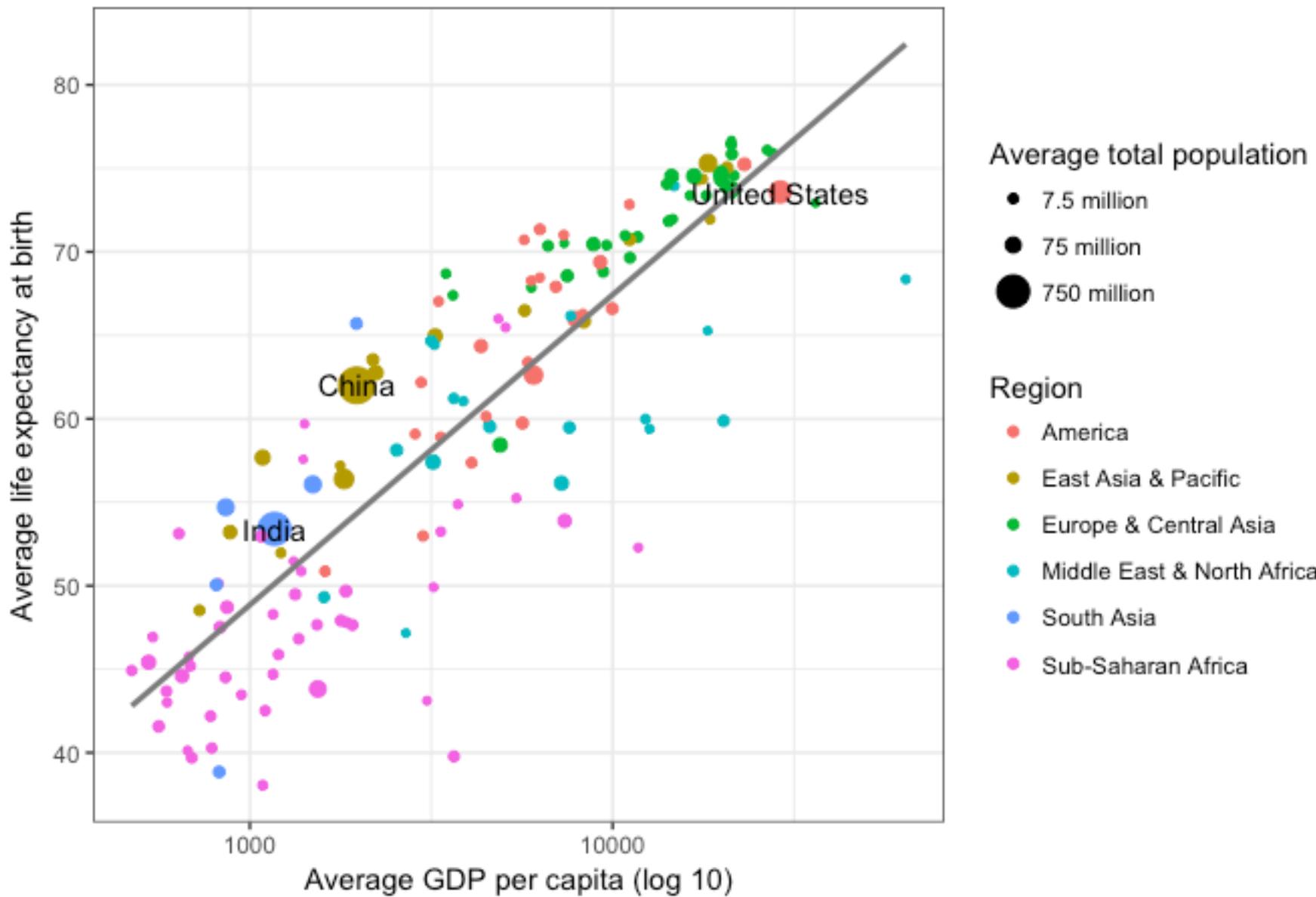
Dataset 3: Star Wars opinion survey

<https://fivethirtyeight.com/features/americas-favorite-star-wars-movies-and-least-favorite-characters/>

Dataset 4: Gapminder Data

<http://www.gapminder.org/>

Averages across all years of the traditional Gapminder dataset



Advanced topics:
Mapping, saving charts out

Mapping resources

- [tigris](#) for downloading TIGER/Line shapefiles
- [sf \(simple features\)](#) for spatial tables
 - [Spatial Data Science book](#)
 - [Spatial Data Science in the tidyverse slides](#)
 - [Spatial Data Science in the tidyverse video](#)

Other helper packages

- [`gganonymize`](#) to randomize text in `ggplot2` figures
- [`visdat`](#) to visualize variable classes and missing data
- [`ggthemes`](#) for additional themes and scales, especially ones that match software defaults (e.g., Tableau)
- [`esquisse`](#) for building `ggplot2` charts interactively
- [`colorblindr`](#) for simulating color vision deficiency
- [`ggpubr`](#) for publication-ready plots

ggplot2 Resources

- General ggplot2 information
<http://ggplot2.tidyverse.org/>
- R Graphics Cookbook (recipes for plots)
<http://www.cookbook-r.com/Graphs/index.html>
- R for Data Science (online book that includes ggplot2)
<http://r4ds.had.co.nz/>
- ggplot2: Elegant Graphs for Data Analysis (book by Hadley Wickham)
<http://ggplot2.org/book/>
- ggplot2 cheatsheet (also in RStudio)
<http://bit.ly/ggplot2-cheatsheet>
- [Data Carpentry lesson on ggplot2](#)
- [Data Visualization: A Practical Introduction, by Kieran Healy](#)
- [RStudio “Visualize Data” Primer](#)

Videos of past workshops

The image shows a screenshot of a Panopto video player interface. At the top left is the Panopto logo and the title "Figures and Posters". To the right are links for "Help" and "Sign in". Below the title, there is a thumbnail image of two people, a man and a woman, standing in front of a whiteboard in a classroom setting. The whiteboard has some handwritten text on it. To the right of the thumbnail, the main title "Designing Academic Figures and Posters" is displayed in large bold letters. Below the title is the date "March 4, 2016". Underneath the date is a link "Slides: <http://duke.box.com/PostersSpring2016>". Further down, two speakers are introduced: "Angela Zoss" and "Eric Monson", each with their titles: "Data Visualization Coordinator" and "Data Visualization Analyst" respectively, both from "Data and Visualization Services". At the bottom of the player, there is a control bar with a play button, a timer showing "0:03", a circular progress bar, and other controls like "Speed" (set to 1x) and "Quality". Below the control bar, there are four small thumbnail images of posters, labeled "Good Posters", "Causal Observation", "Purpose of a poster", and another unlabeled one. Each thumbnail has a timestamp at the bottom right: "1:32", "4:32", "7:32", and "10:32" respectively.

<http://bit.ly/DVSvideos>

Thanks for your feedback!

angela.zoss@duke.edu

ggplot2: Chart quirks

See [“templates” file](#)

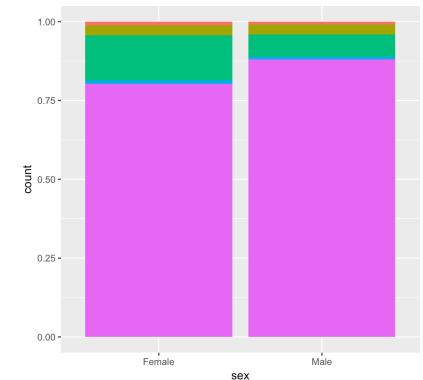
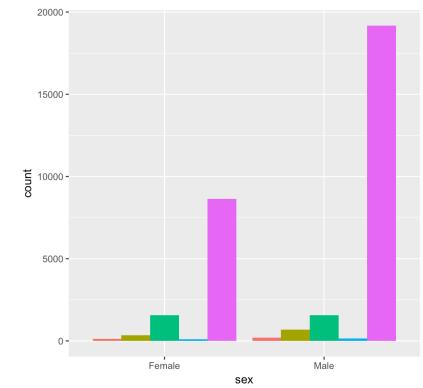
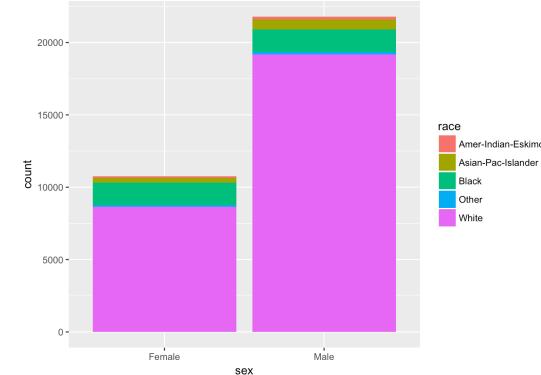
Chart components/slots

Bar chart, for example:

- x
category (the names of the bars)
- y (optional)
*default is count, but can also specify a number
(the length of the bars)*
- color (optional)
category (grouped or stacked bars)

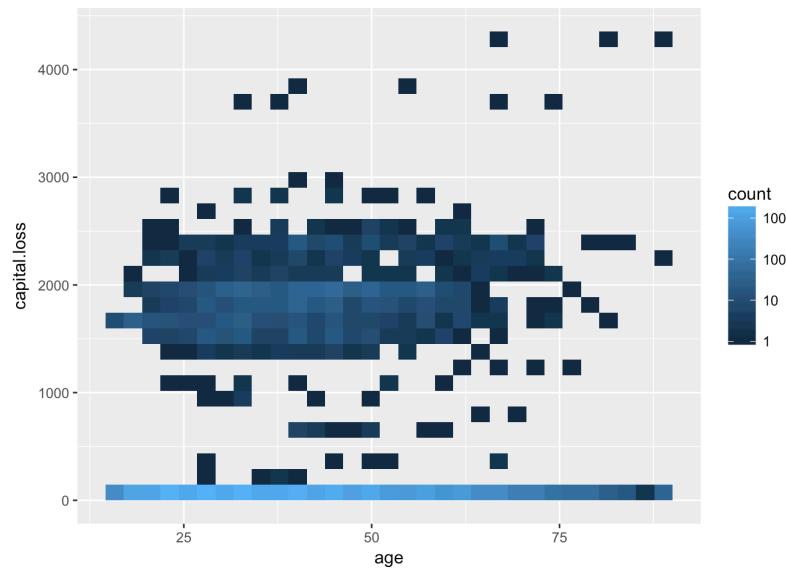
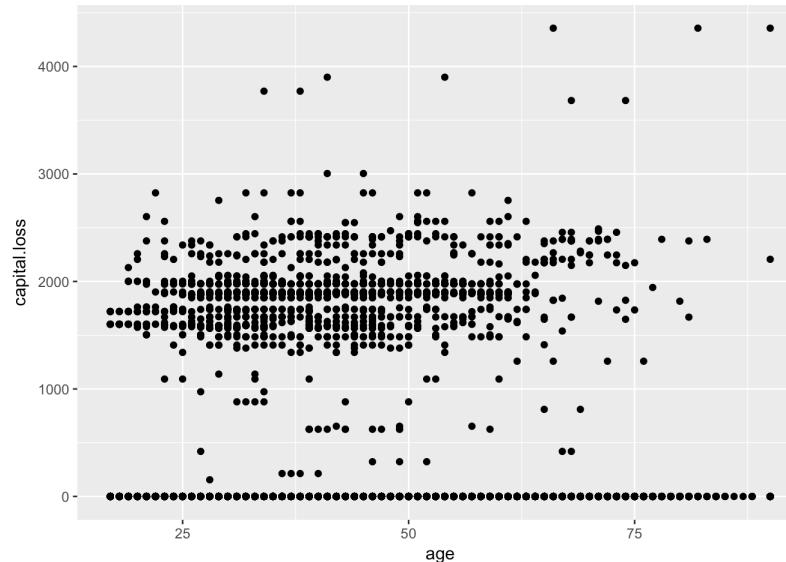
Bar chart

- geom_bar() vs. geom_col()
- count vs. identity vs. summary
- categorical vs. continuous
- stack vs. dodge vs. fill
- bar vs. pie



Scatter plot

- Overplotting
- point vs. bin2d



Line chart

- identity vs. summary
- line vs. smooth

