Putting the R in Reed and in Lewis and ClaRk

Chester Ismay

GitHub: ismayc
Twitter: @old_man_chester

2017-05-25 & 2017-05-26

Bootcamp website at http://bit.ly/rbootcamp17
Slides available at http://bit.ly/rbootcamp17-slides

Pre-bootcamp HW

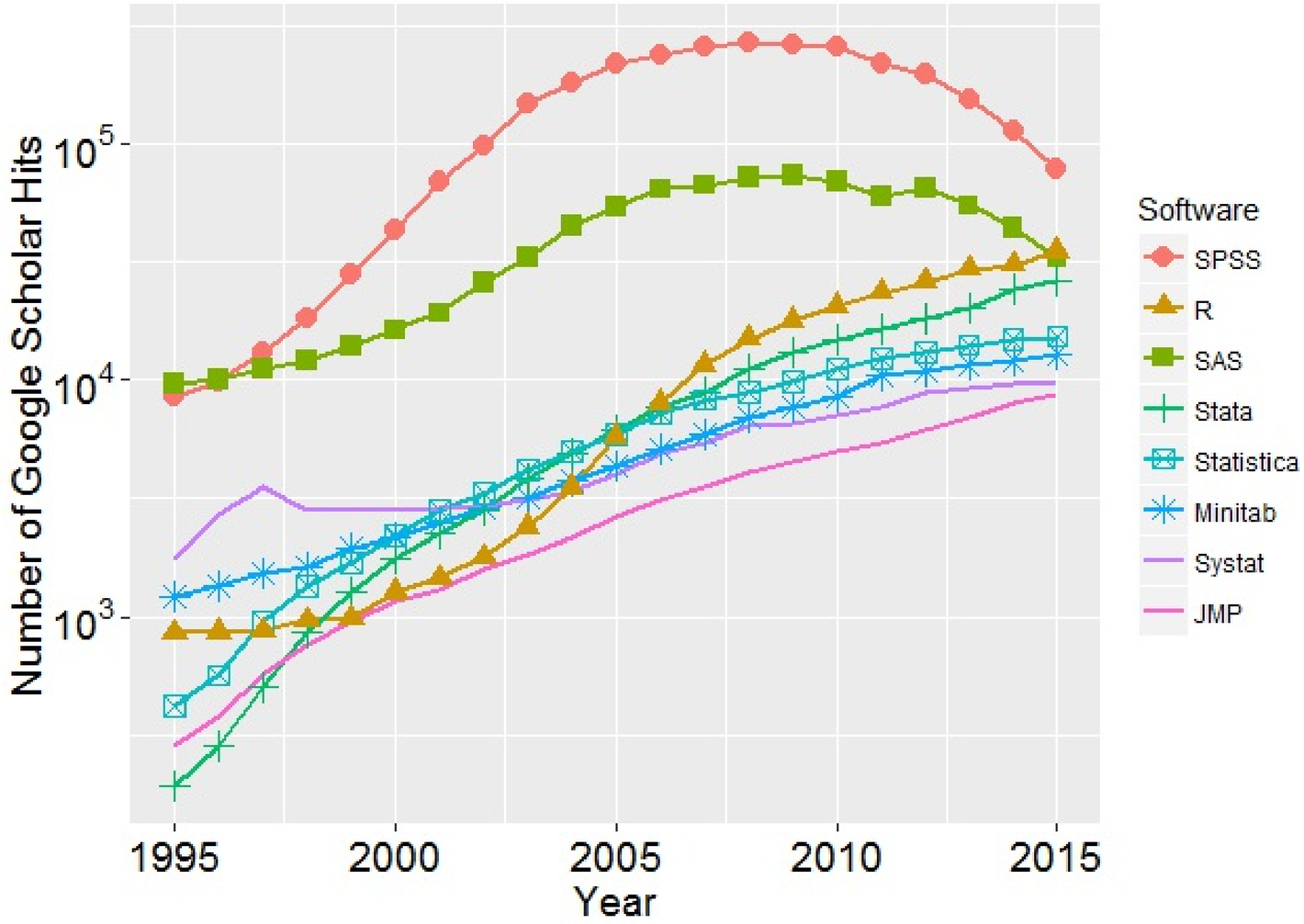
Talk about your analysis on the
`weather` data
in groups of 2-3

Let's get it all out there

When you think of R what comes to mind?

Why are you here wanting to learn R?

- ~~Because you enjoy partaking in~~ Schadenfreude



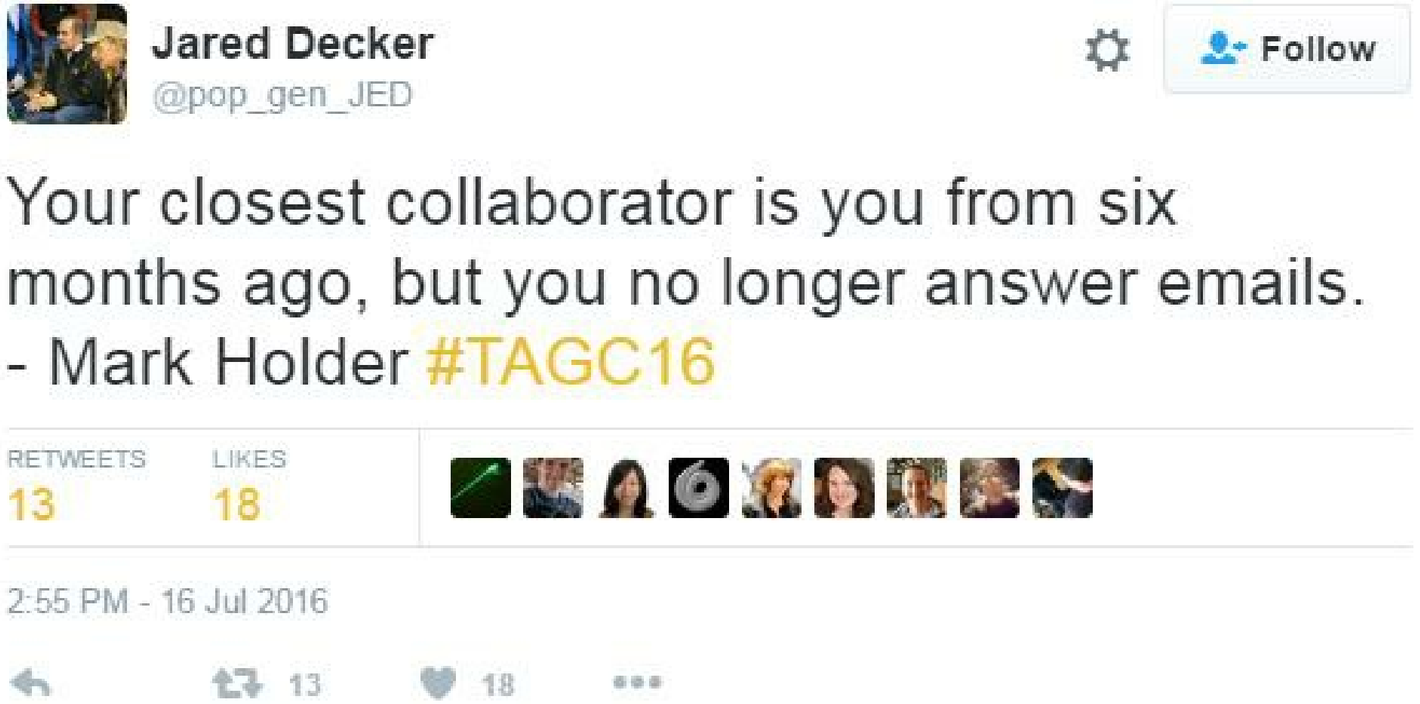
Why should you learn R?

- R is free, R improves, R has existed for 40+ years
- Students can show what they have learned to a potential employer easily
- The support system for R is actually much better than some people say

But as a professor, why should you learn R?

- R and R Markdown will save you TONS of time. Long term thinking is key.
  - You can easily tweak your code if you need to do another analysis
  - Remembering what drop-down menu some thing is in two years from now in a different program will be hard
  - Remembering to copy-and-paste your updated plots/analysis into your word processor is a pain and error prone

But as a professor, why should you learn R?

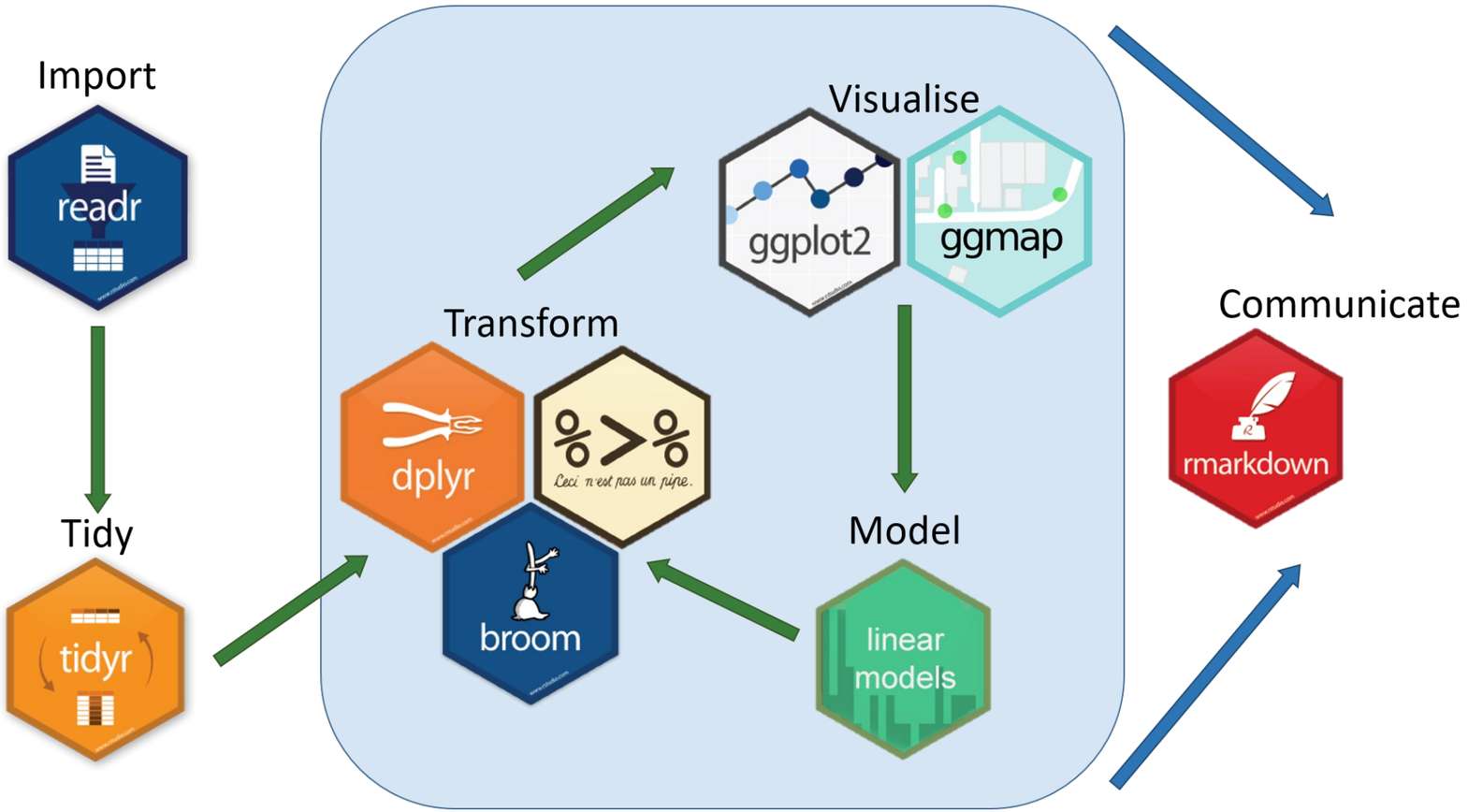


DEMO in RStudio with R Markdown

Analogy

R	RStudio	DataCamp
		

What this bootcamp is



- Designed for all levels of knowledge and background

What this bootcamp is not

- A thorough development of machine learning techniques applied to big data

What this bootcamp is not

- A thorough development of machine learning techniques applied to big data
- A discussion on the role of p-values in science

What this bootcamp is not

- A thorough development of machine learning techniques applied to big data
- A discussion on the role of p-values in science
- A tutorial on how to work with base R subsetting and base R graphics

What I hope you'll learn

- That R is not as scary as it used to be.

What I hope you'll learn

- That R is not as scary as it used to be.
- Learning how to Google is an incredibly valuable skill.

What I hope you'll learn

- That R is not as scary as it used to be.
- Learning how to Google is an incredibly valuable skill.
- That having students turn in assignments using R scripts and R Markdown provides an efficient way to check their work and their analyses easily.

What I hope you'll learn

- That the R packages you will use in this workshop are the same ones that are used by scientists and graduate programs all over the world

What I hope you'll learn

- That the R packages you will use in this workshop are the same ones that are used by scientists and graduate programs all over the world
- That teaching students how to use open-source tools is what is best for them long term

What I hope you'll learn

- That the R packages you will use in this workshop are the same ones that are used by scientists and graduate programs all over the world
- That teaching students how to use open-source tools is what is best for them long term
- That students with no programming background can do great things in only a few weeks
    - Sociology/Criminal Justice majors at Pacific U.
    - A wide range of students at Middlebury College

Pieces of advice

- Scaffold & support as a foreign languages do

Pieces of advice

- Scaffold & support as a foreign languages do
- To be able to use R (and really any other language), students need more than a few assignments and more than a weekly lab

Pieces of advice

- Scaffold & support as a foreign languages do
- To be able to use R (and really any other language), students need more than a few assignments and more than a weekly lab
- Make use of RStudio Projects (students have a really hard time navigating directory structures)

Pieces of advice

My opinion

- Have students work on writing their code in R script files and documenting their errors
    - This is the same workflow that DataCamp uses

Pieces of advice

My opinion

- Have students work on writing their code in R script files and documenting their errors
    - This is the same workflow that DataCamp uses
- Show students R Markdown after a few weeks of working with the script file
    - I've found it is hard for students to learn R and R Markdown from the start
    - Better to have them use R Markdown in groups initially

R Data Types

The bare minimum needed for understanding today

Vector/variable

- Type of vector (`int`, `num`, `chr`, `lgl`, `date`)

The bare minimum needed for understanding today

Vector/variable

- Type of vector (`int`, `num`, `chr`, `lgl`, `date`)

Data frame

- Vectors of (potentially) different types
- Each vector has the same number of rows

The bare minimum needed for understanding today

```
library(tibble)
library(lubridate)
ex1 <- data_frame(
    vec1 = c(1980, 1990, 2000, 2010),
    vec2 = c(1L, 2L, 3L, 4L),
    vec3 = c("low", "low", "high", "high"),
    vec4 = c(TRUE, FALSE, FALSE, FALSE),
    vec5 = ymd(c("2017-05-23", "1776/07/04", "1983-05/31", "1908/04-01"))
  )
ex1
```

The bare minimum needed for understanding today

```
library(tibble)
library(lubridate)
ex1 <- data_frame(
    vec1 = c(1980, 1990, 2000, 2010),
    vec2 = c(1L, 2L, 3L, 4L),
    vec3 = c("low", "low", "high", "high"),
    vec4 = c(TRUE, FALSE, FALSE, FALSE),
    vec5 = ymd(c("2017-05-23", "1776/07/04", "1983-05/31", "1908/04-01"))
  )
ex1
```

```
# A tibble: 4 x 5
   vec1  vec2  vec3  vec4       vec5
  <dbl> <int> <chr> <lgl>     <date>
1  1980     1   low  TRUE 2017-05-23
2  1990     2   low FALSE 1776-07-04
3  2000     3  high FALSE 1983-05-31
4  2010     4  high FALSE 1908-04-01
```

Welcome to the tidyverse!

The `tidyverse` is a collection of R packages that share common philosophies and are designed to work together.

Beginning steps

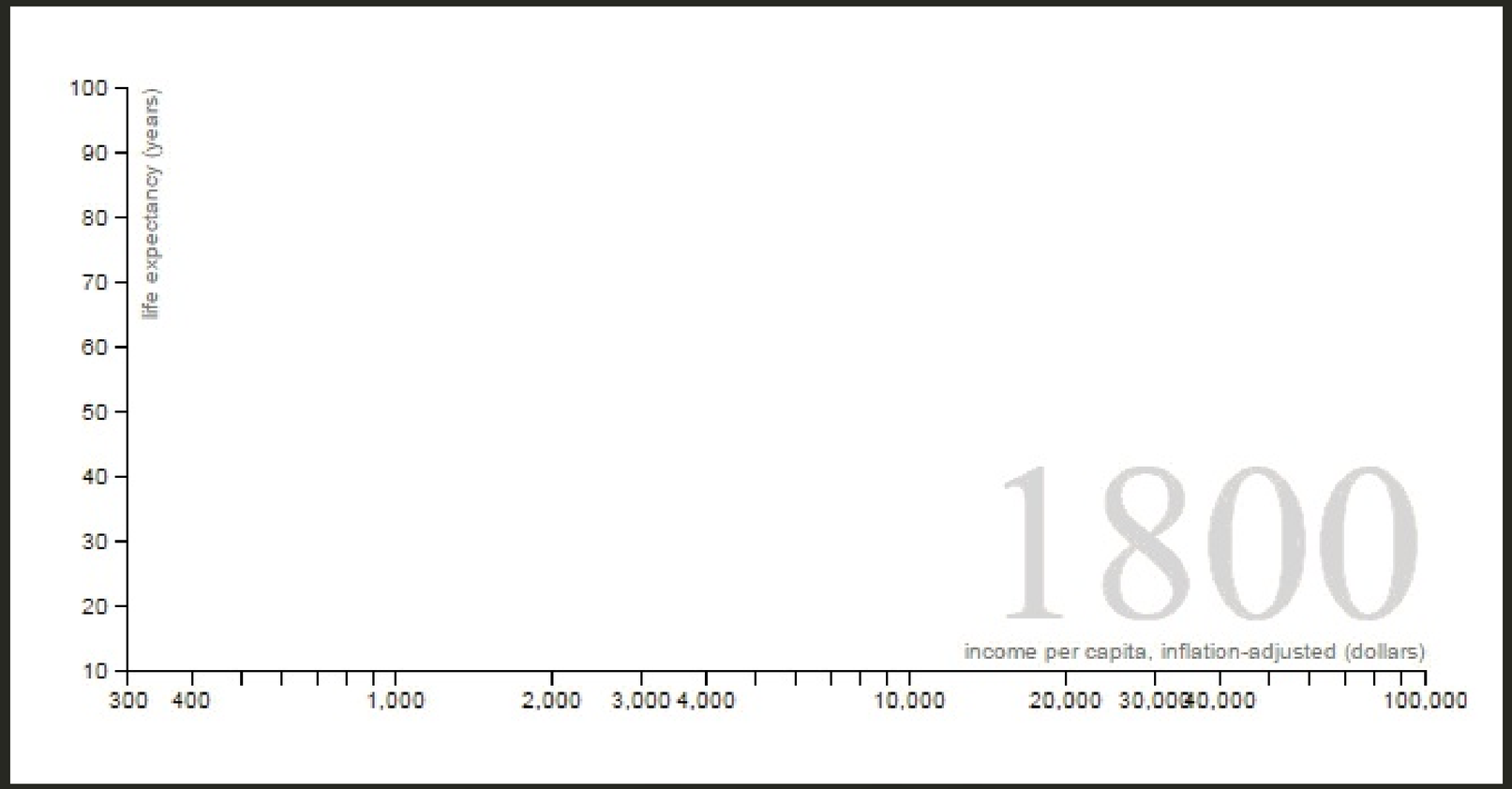
Frequently the first thing to do when given a dataset is to

- identify the observational unit,
- specify the variables,
- give the types of variables you are presented with, and
- check that the data is tidy. (TO COME)

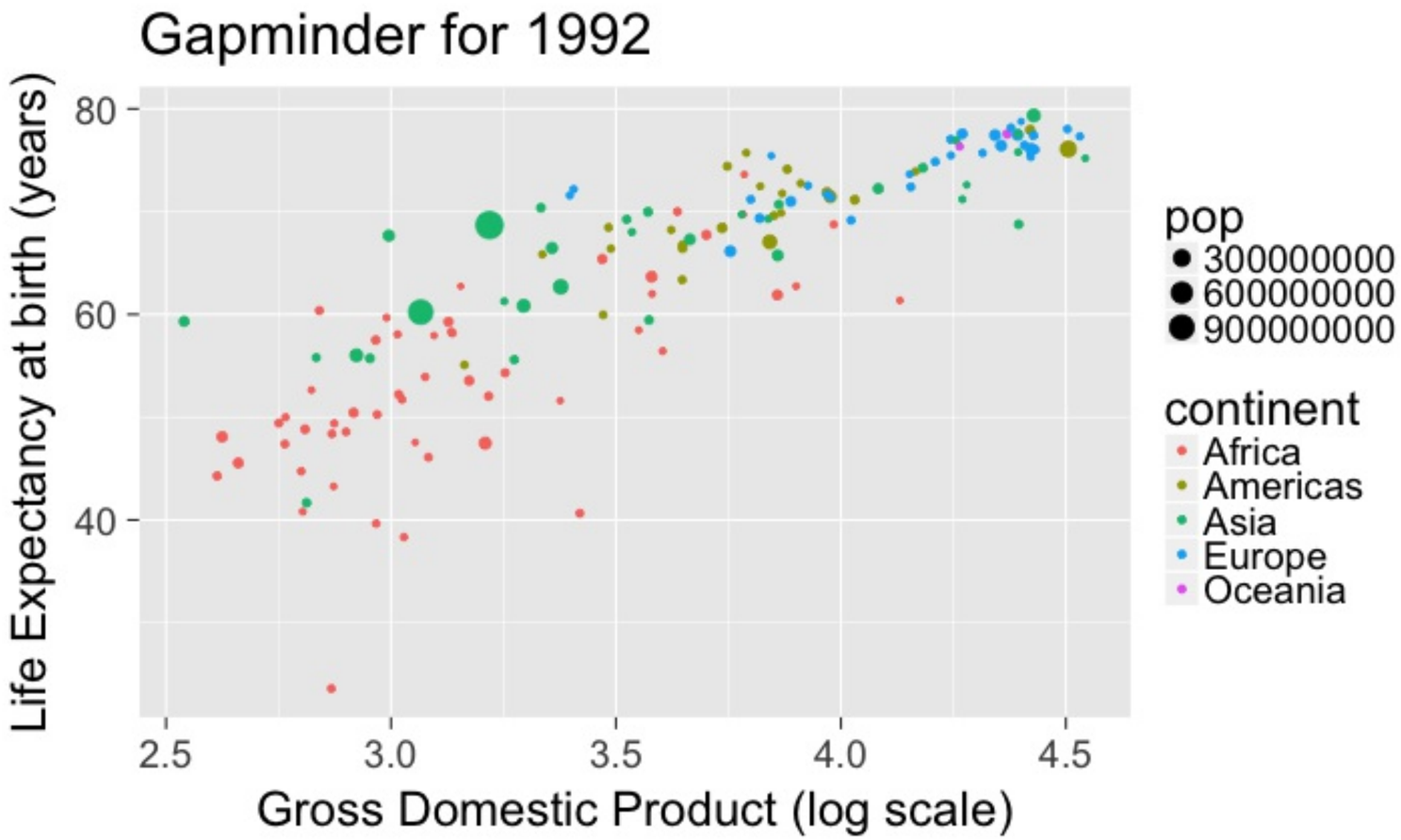
This will help with

- choosing the appropriate plot,
- summarizing the data, and
- understanding which inferences/models can be applied.

Data Visualization



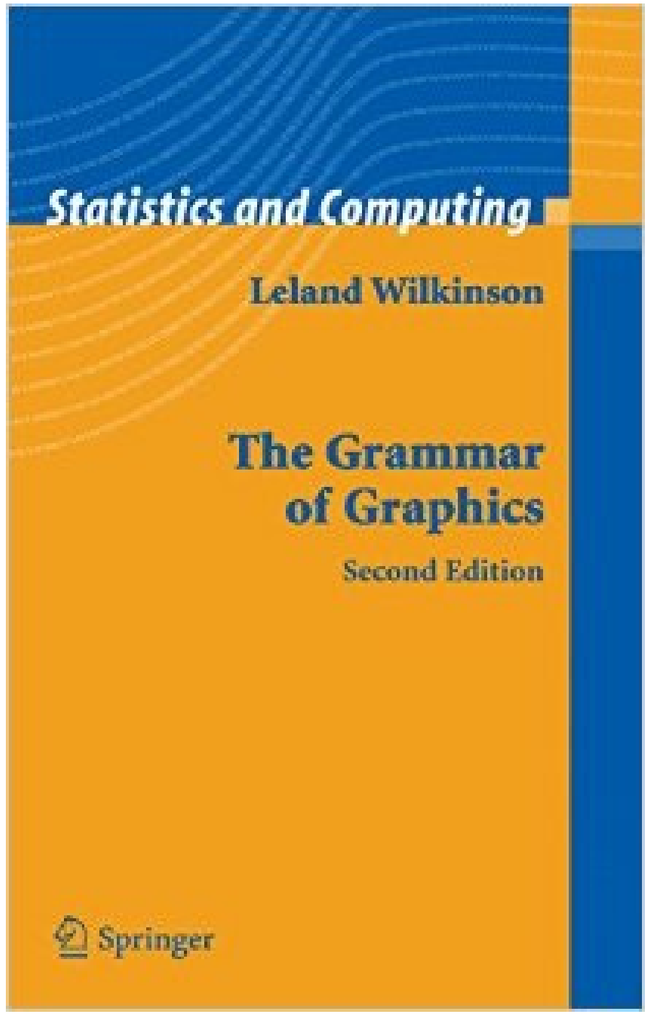
Inspired by Hans Rosling



- What are the variables here?
- What is the observational unit?
- How are the variables mapped to aesthetics?

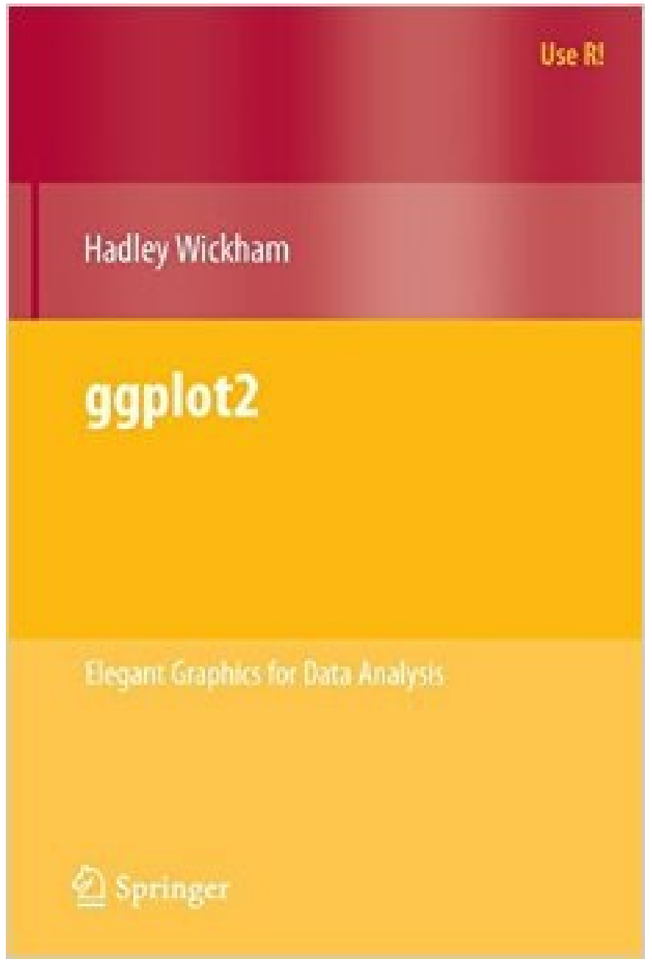
Grammar of Graphics

Wilkinson (2005) laid out the proposed "Grammar of Graphics"



Grammar of Graphics in R

Wickham implemented the grammar in R in the `ggplot2` package



Grammar of Graphics elsewhere

- Make plots online with plotly
- Leland Wilkinson works at Tableau
- The Grammar of Graphics provides a theoretical framework for building and deciphering statistical graphics

What is a statistical graphic?

What is a statistical graphic?

A `mapping` of
`data` variables

What is a statistical graphic?

A `mapping` of
`data` variables

to
`aes()`thetic attributes

What is a statistical graphic?

A `mapping` of
`data` variables

to
`aes()`thetic attributes

of
`geom_`etric objects.

Back to basics

Consider the following data in tidy format:

```
simple_ex <-
  data_frame(
    A = c(1980, 1990, 2000, 2010),
    B = c(1, 2, 3, 4),
    C = c(3, 2, 1, 2),
    D = c("low", "low", "high", "high")
  )
simple_ex
```

```
# A tibble: 4 x 4
      A     B     C     D
  <dbl> <dbl> <dbl> <chr>
1  1980     1     3   low
2  1990     2     2   low
3  2000     3     1  high
4  2010     4     2  high
```

Consider the following data in tidy format:

A	B	C	D
1980	1	3	low
1990	2	2	low
2000	3	1	high
2010	4	2	high

- Sketch the graphics below on paper, where the `x`-axis is variable `A` and the `y`-axis is variable `B`

1. A scatter plot
2. A scatter plot where the `color` of the points corresponds to `D`
3. A scatter plot where the `size` of the points corresponds to `C`
4. A line graph
5. A line graph where the `color` of the line corresponds to `D` with points added that are all green of size 4.

Reproducing the plots in `ggplot2`

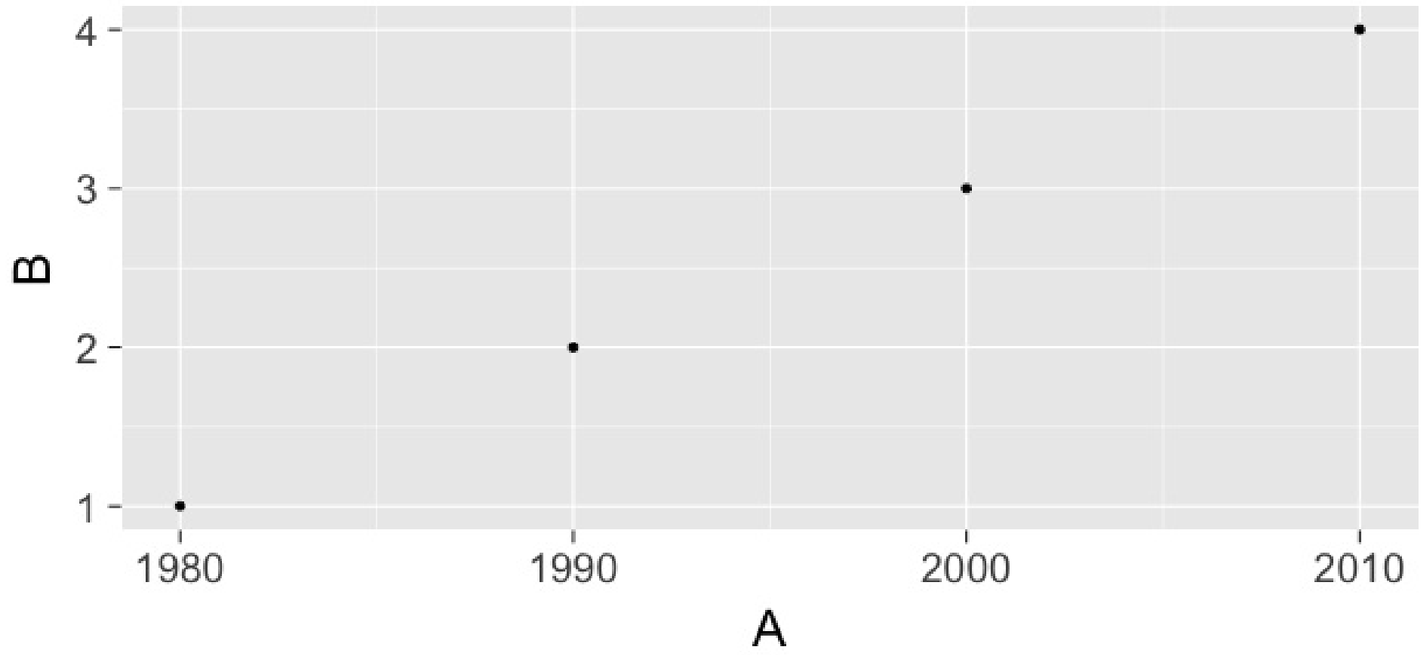
1. A scatterplot

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B)) +
  geom_point()
```

Reproducing the plots in `ggplot2`

1. A scatterplot

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B)) +
  geom_point()
```



Reproducing the plots in `ggplot2`

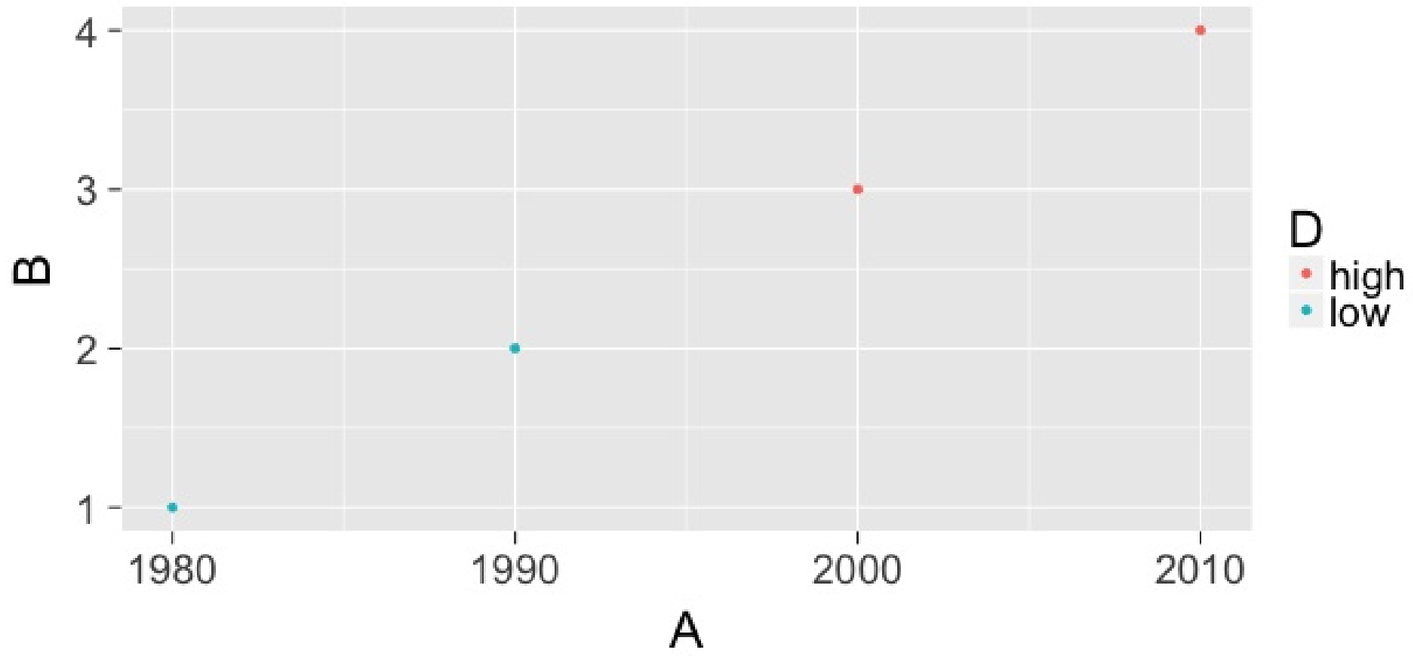
2. A scatter plot where the `color` of the points corresponds to `group`

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B)) +
  geom_point(mapping = aes(color = D))
```

Reproducing the plots in `ggplot2`

2. A scatter plot where the `color` of the points corresponds to `group`

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B)) +
  geom_point(mapping = aes(color = D))
```



Reproducing the plots in `ggplot2`

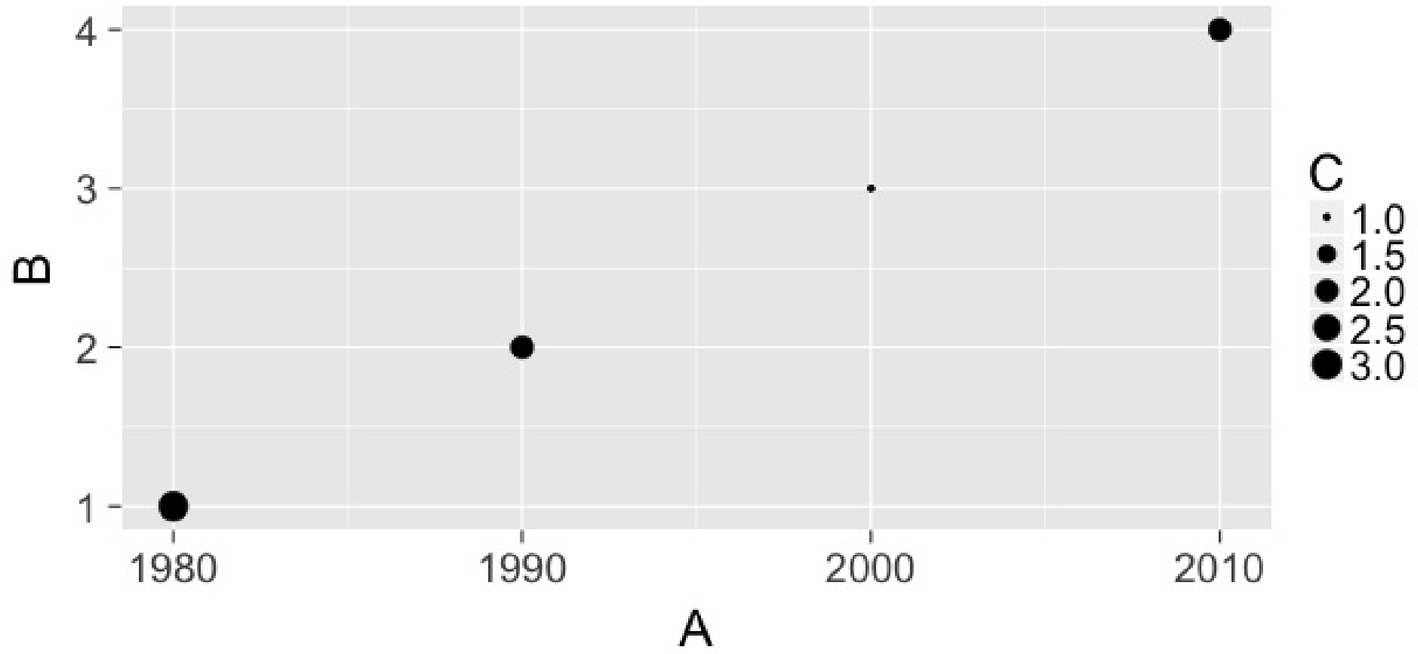
3. A scatter plot where the `size` of the points corresponds to `C`

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B, size = C)) +
  geom_point()
```

Reproducing the plots in `ggplot2`

3. A scatter plot where the `size` of the points corresponds to `C`

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B, size = C)) +
  geom_point()
```



Reproducing the plots in `ggplot2`

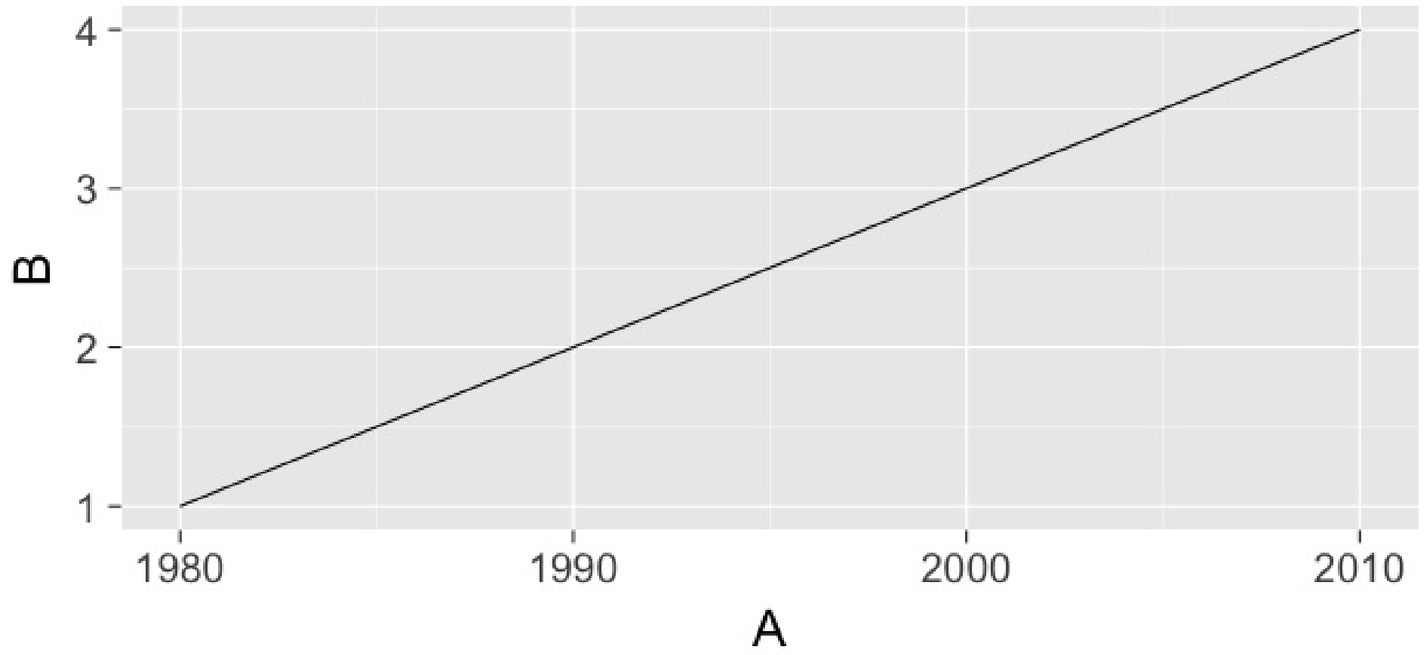
4. A line graph

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B)) +
  geom_line()
```

Reproducing the plots in `ggplot2`

4. A line graph

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B)) +
  geom_line()
```



Reproducing the plots in `ggplot2`

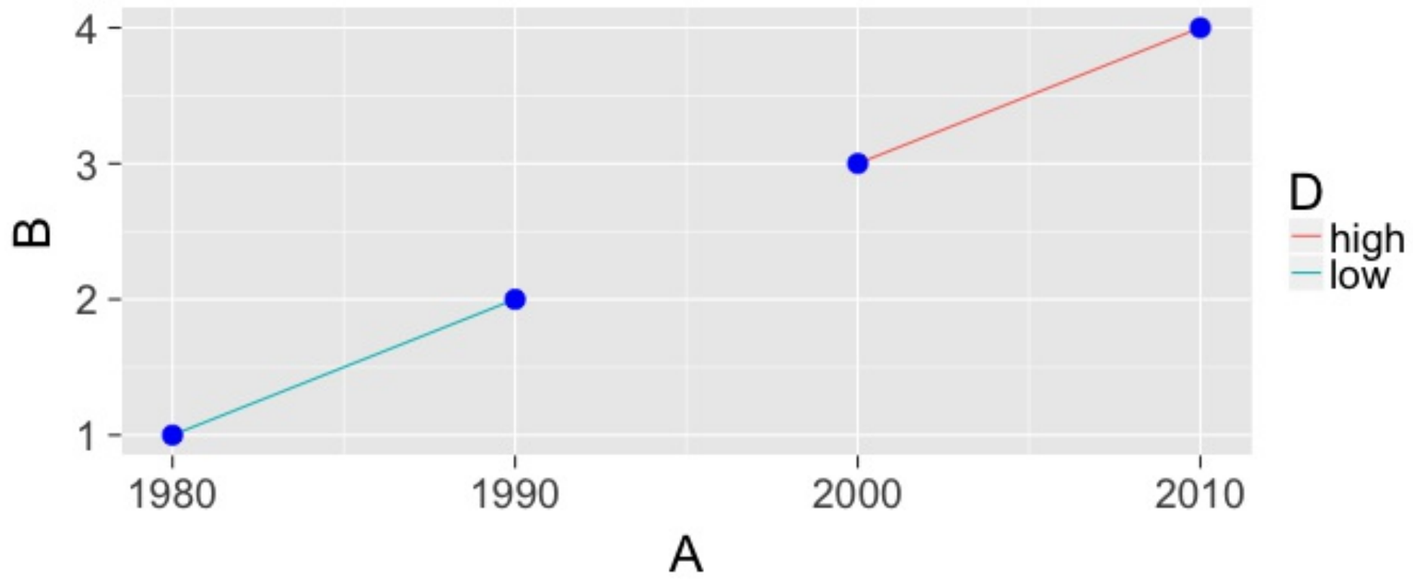
5. A line graph where the `color` of the line corresponds to `D` with points added that are all blue of size 4.

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B)) +
  geom_line(mapping = aes(color = D)) +
  geom_point(color = "blue", size = 4)
```

Reproducing the plots in `ggplot2`

5. A line graph where the `color` of the line corresponds to `D` with points added that are all blue of size 4.

```
library(ggplot2)
ggplot(data = simple_ex, mapping = aes(x = A, y = B)) +
  geom_line(mapping = aes(color = D)) +
  geom_point(color = "blue", size = 4)
```



The Five-Named Graphs

The 5NG of data viz

- Scatterplot: `geom_point()`
- Line graph: `geom_line()`

The Five-Named Graphs

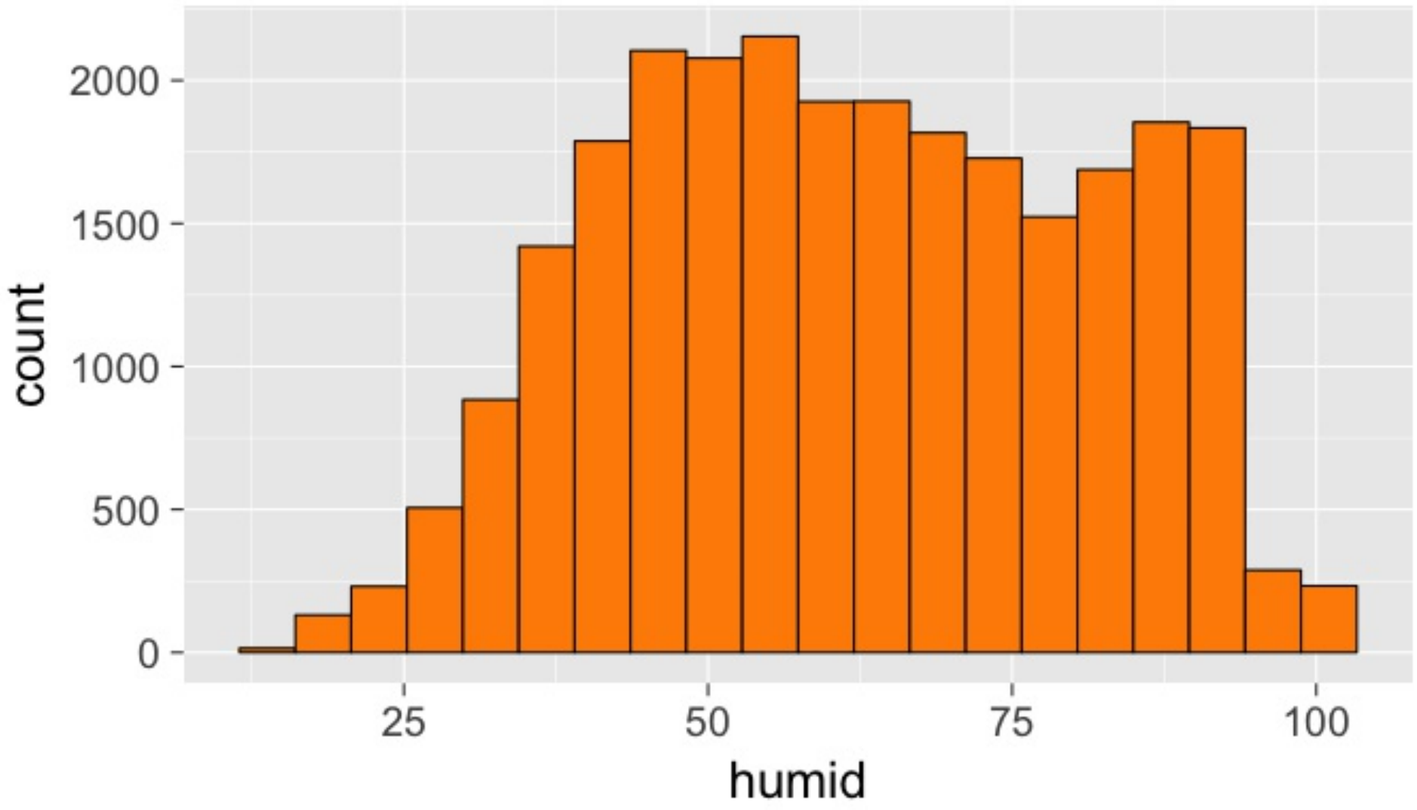
The 5NG of data viz

- Scatterplot: `geom_point()`
- Line graph: `geom_line()`
- Histogram: `geom_histogram()`
- Boxplot: `geom_boxplot()`
- Bar graph: `geom_bar()`

More `ggplot2` examples

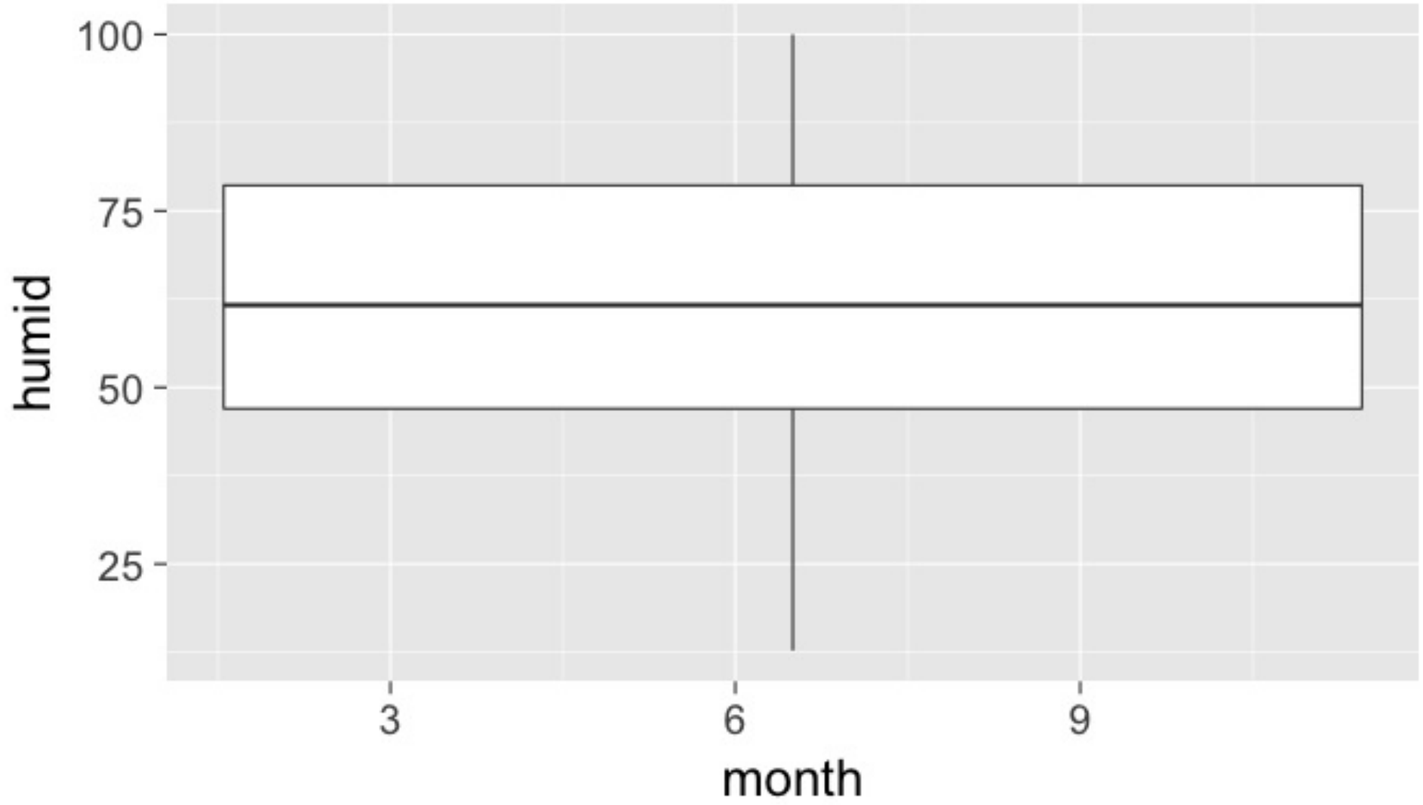
Histogram

```
library(nycflights13)
ggplot(data = weather, mapping = aes(x = humid)) +
  geom_histogram(bins = 20, color = "black", fill = "darkorange")
```



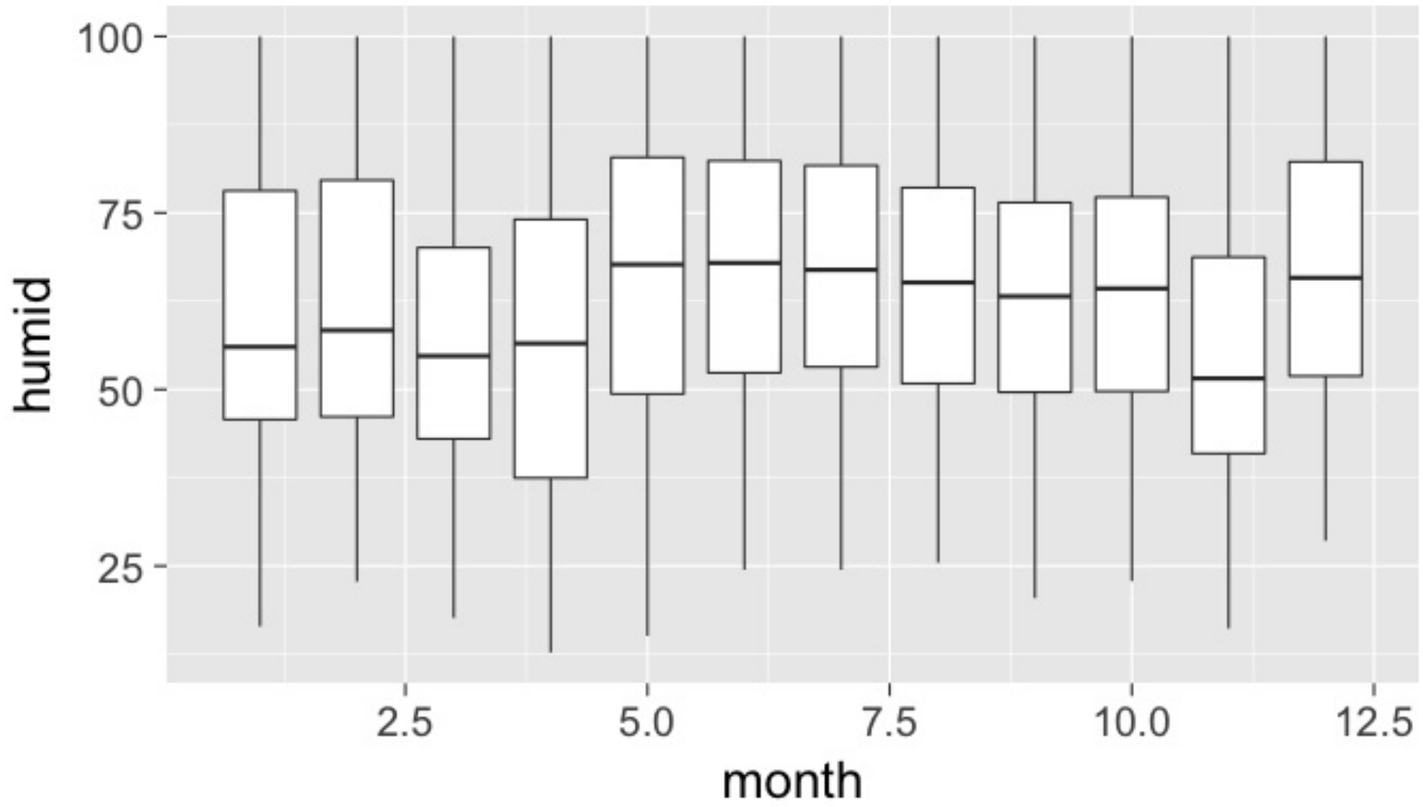
Boxplot (broken)

```
library(nycflights13)
ggplot(data = weather, mapping = aes(x = month, y = humid)) +
  geom_boxplot()
```



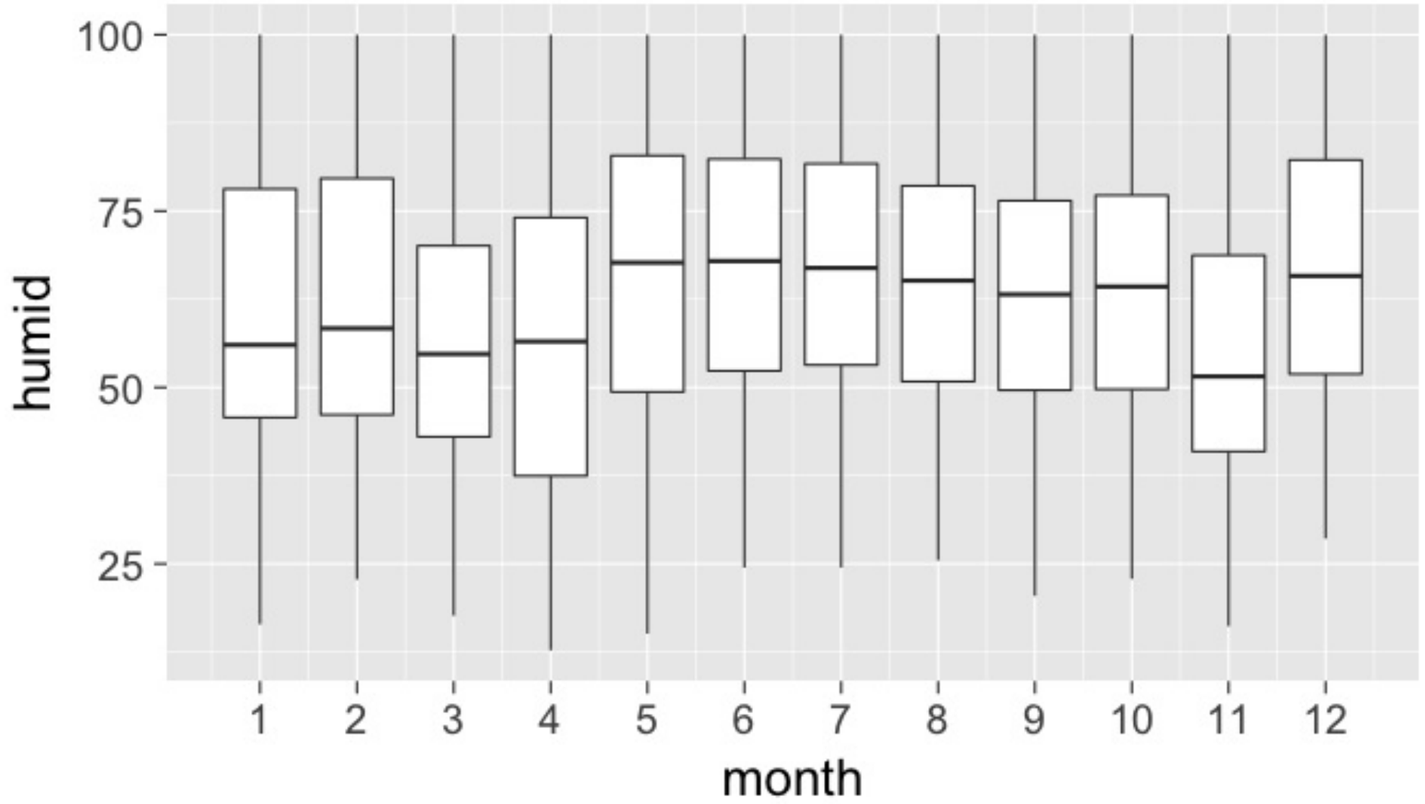
Boxplot (almost fixed)

```
library(nycflights13)
ggplot(data = weather, mapping = aes(x = month, group = month, y = humid)) +
  geom_boxplot()
```



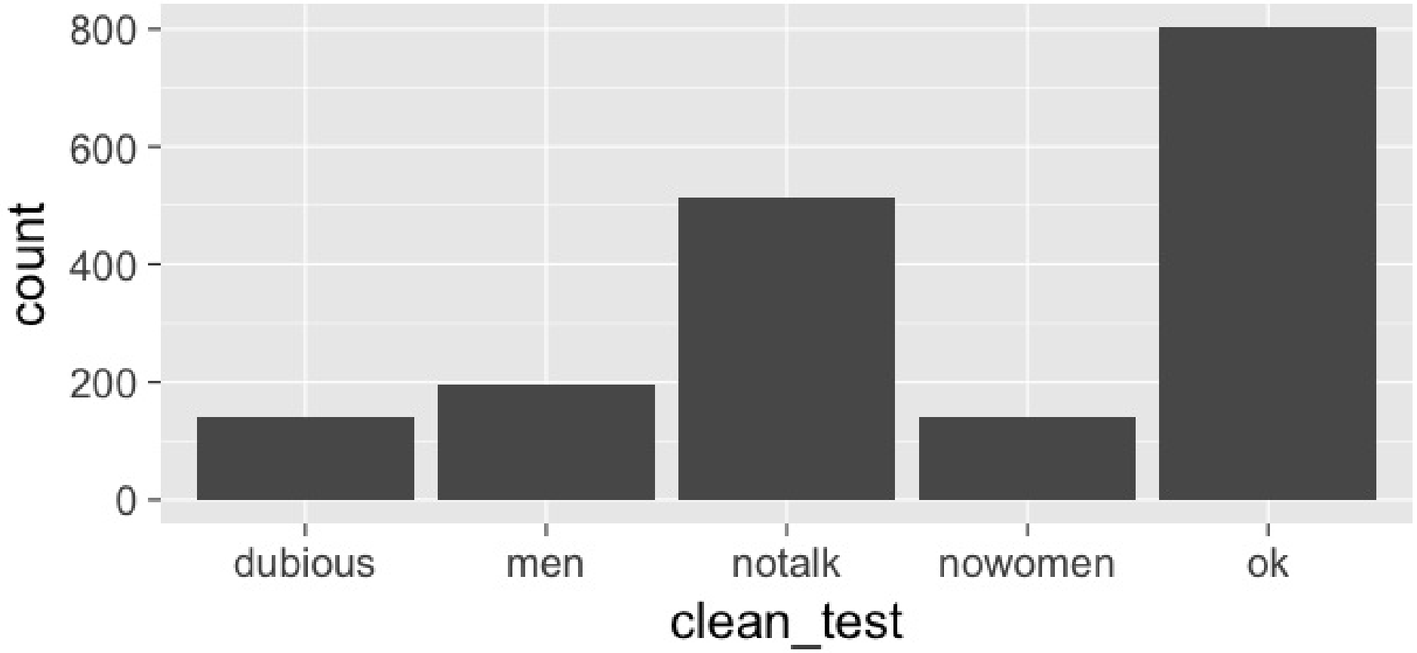
Boxplot (fixed)

```
library(nycflights13)
ggplot(data = weather, mapping = aes(x = month, group = month, y = humid)) +
  geom_boxplot() +
  scale_x_continuous(breaks = 1:12)
```



Bar graph

```
library(fivethirtyeight)
ggplot(data = bechdel, mapping = aes(x = clean_test)) +
  geom_bar()
```



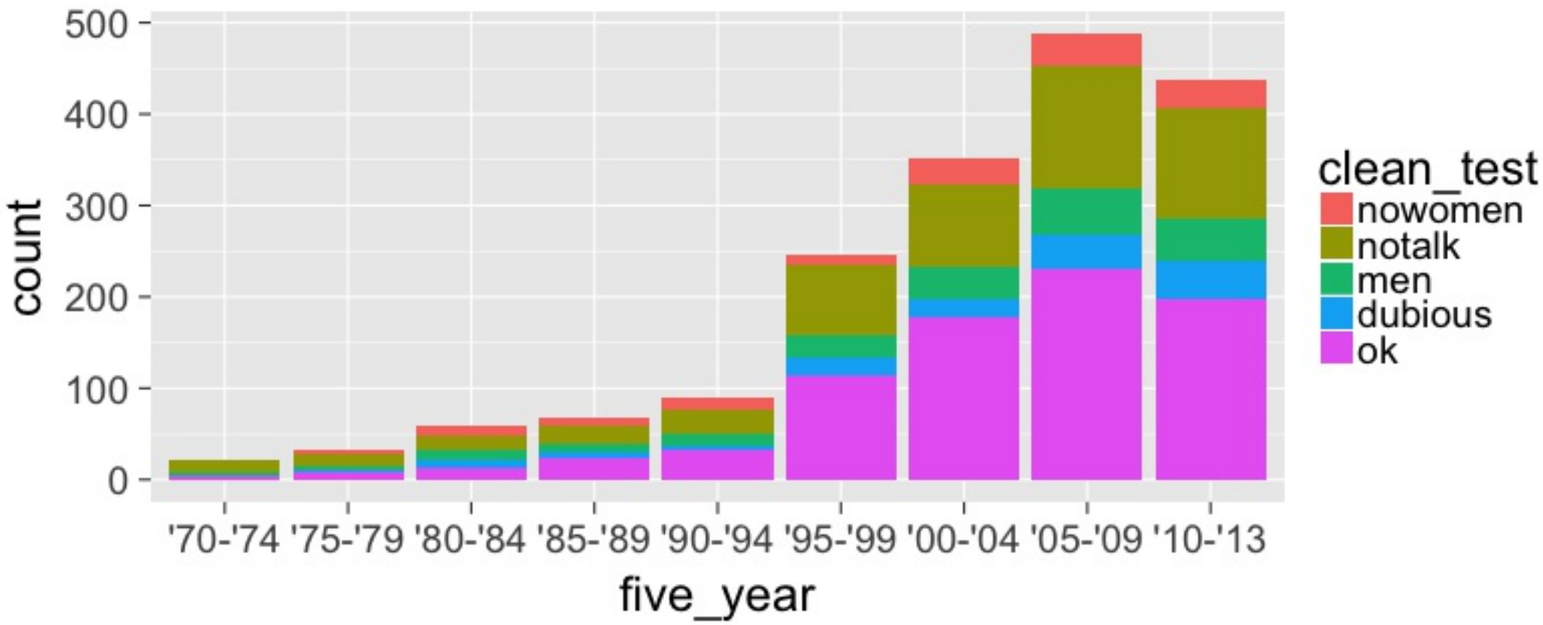
How about over time?

- Hop into `dplyr`

```
library(dplyr)
year_bins <- c("'70-'74", "'75-'79", "'80-'84", "'85-'89",
               "'90-'94", "'95-'99", "'00-'04", "'05-'09",
               "'10-'13")
bechdel <- bechdel %>%
  mutate(five_year = cut(year,
                         breaks = seq(1969, 2014, 5),
                         labels = year_bins)) %>%
  mutate(clean_test = factor(clean_test,
                             levels = c("nowomen", "notalk", "men",
                                        "dubious", "ok")))
```

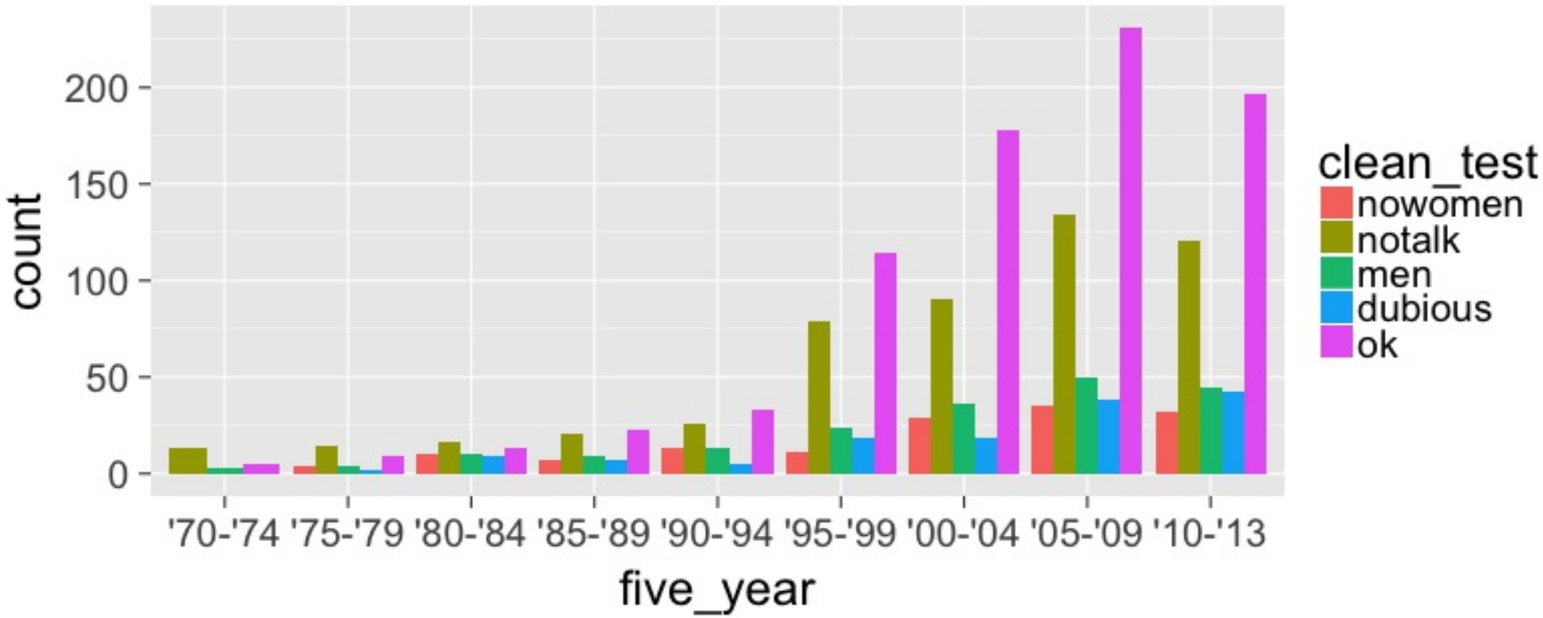
How about over time? (Stacked)

```
library(fivethirtyeight)
library(ggplot2)
ggplot(data = bechdel,
       mapping = aes(x = five_year, fill = clean_test)) +
  geom_bar()
```



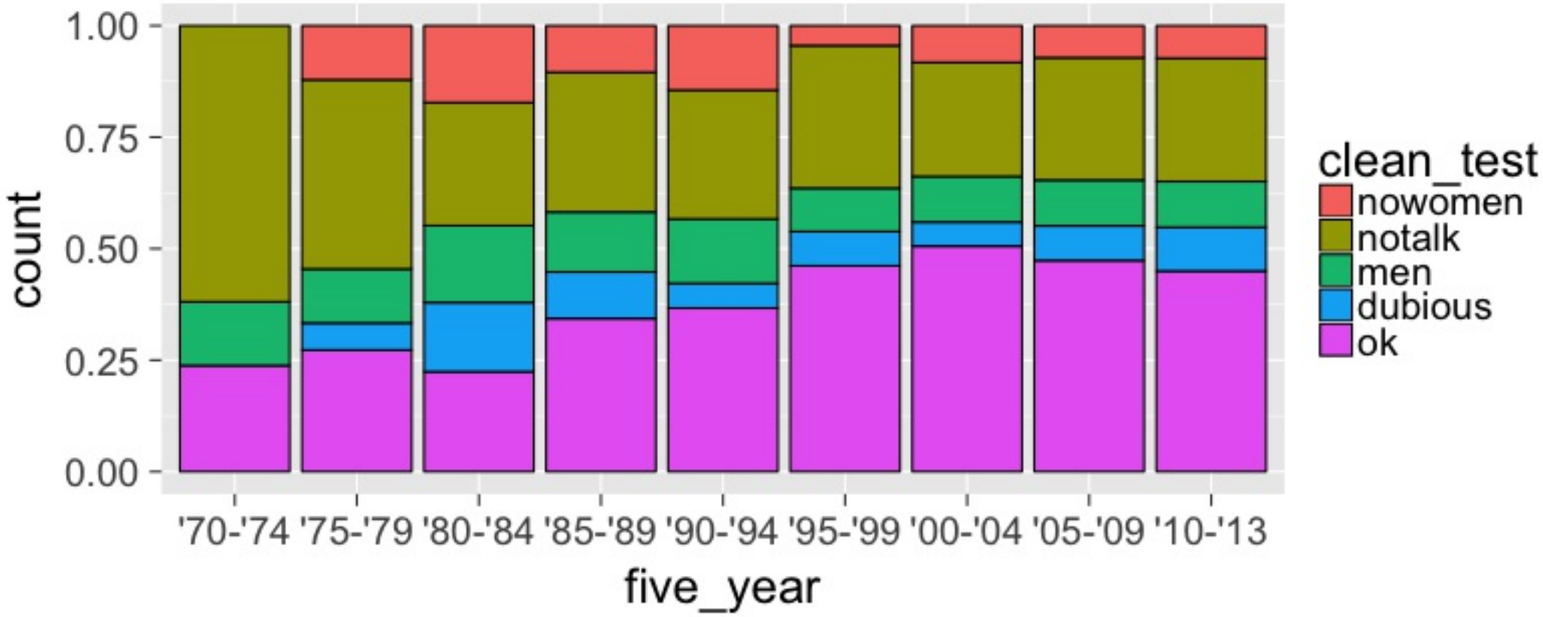
How about over time? (Side-by-side)

```
library(fivethirtyeight)
library(ggplot2)
ggplot(data = bechdel,
       mapping = aes(x = five_year, fill = clean_test)) +
  geom_bar(position = "dodge")
```

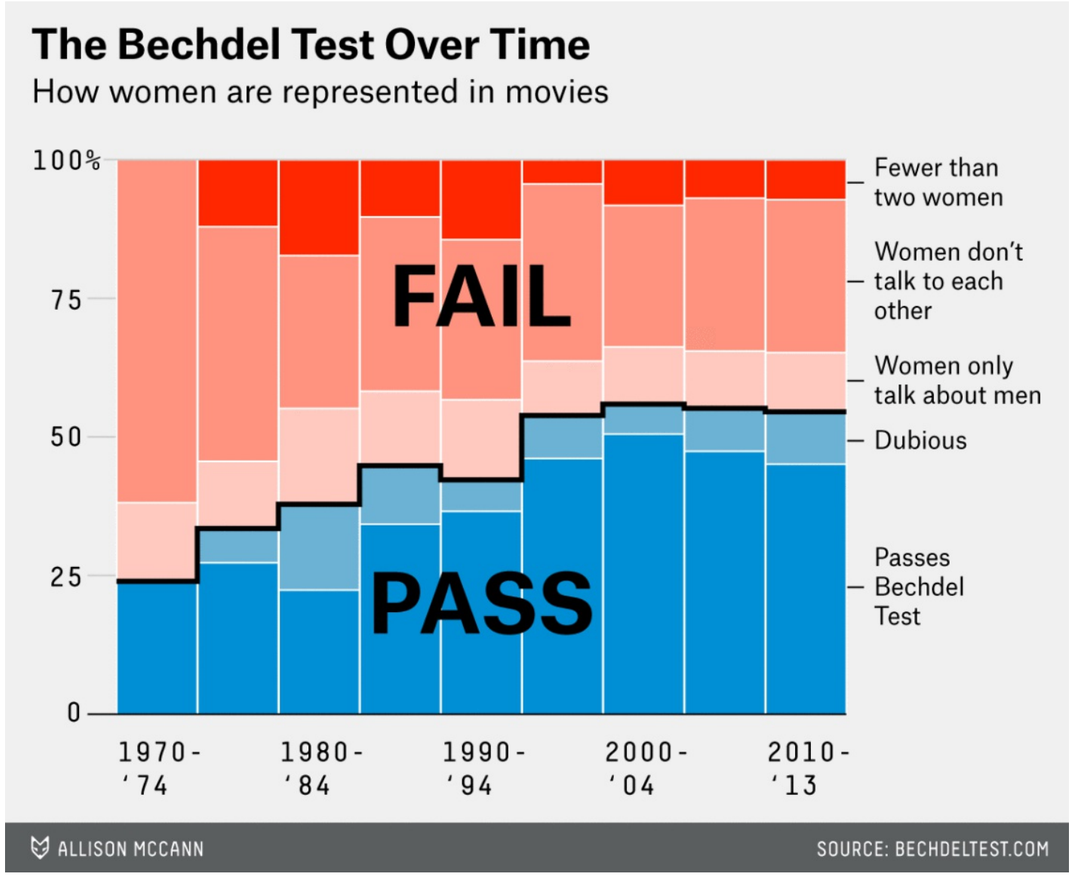


How about over time? (Stacked proportional)

```
library(fivethirtyeight)
library(ggplot2)
ggplot(data = bechdel,
       mapping = aes(x = five_year, fill = clean_test)) +
  geom_bar(position = "fill", color = "black")
```



The `tidyverse`/`ggplot2` is for beginners and for data science professionals!

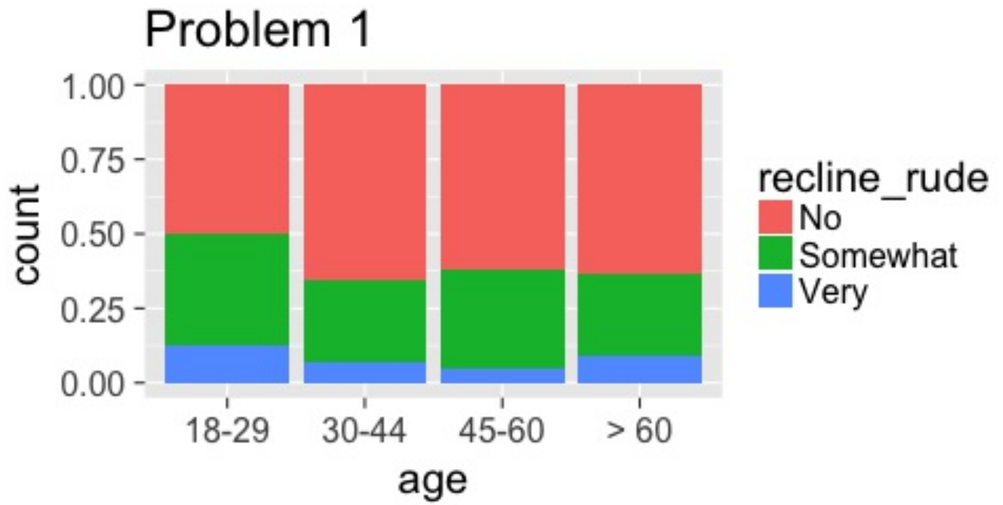


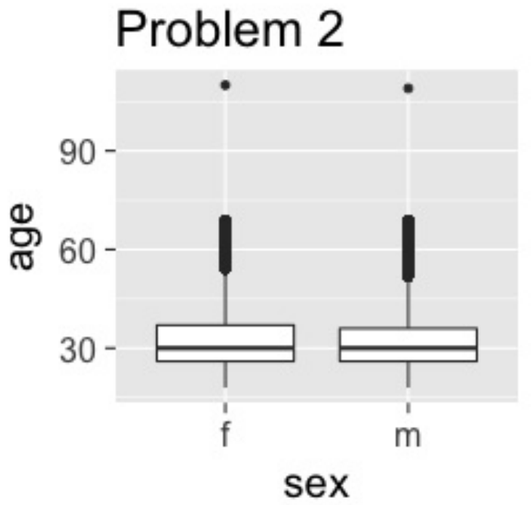
Practice

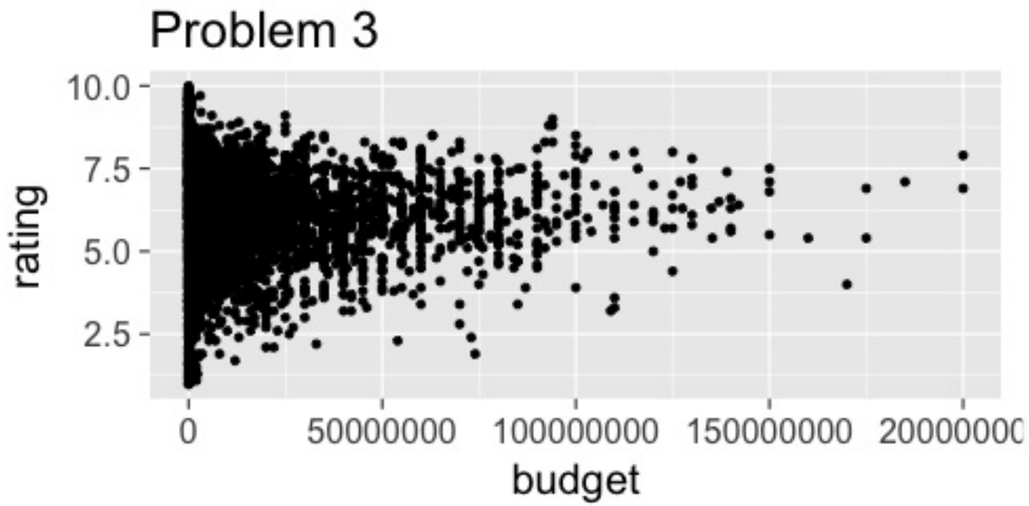
Produce appropriate 5NG with R package & data set in [ ],
e.g., [`nycflights13` → `weather`]

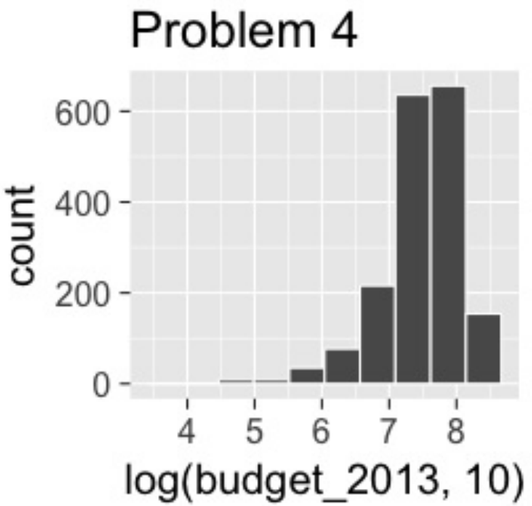
1. Does `age` predict `recline_rude`?
   [`fivethirtyeight` → `na.omit(flying)`]
2. Distribution of `age` by `sex`
   [`okcupiddata` → `profiles`]
3. Does `budget` predict `rating`?
   [`ggplot2movies` → `movies`]
4. Distribution of log base 10 scale of `budget_2013`
   [`fivethirtyeight` → `bechdel`]

HINTS


Problem 1
1.00
0.75
0.50
0.25
0.00
count
18-29
30-44
45-60
> 60
age
recline_rude
No
Somewhat
Very

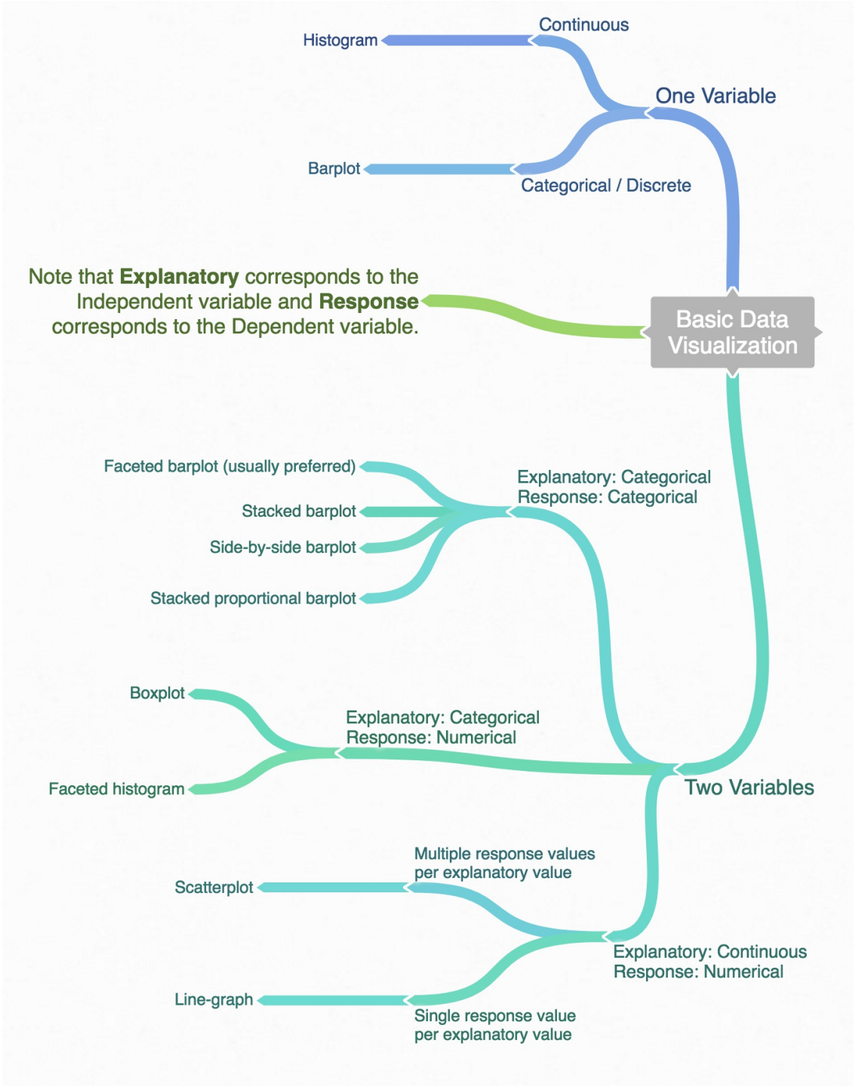

Problem 2
90
60
30
age
f
m
sex


Problem 3
10.0
7.5
5.0
2.5
rating
0
50000000
100000000
150000000
budget


Problem 4
600
400
200
0
count
4
5
6
7
8
log(budget_2013, 10)

DEMO of `ggplot2` in RStudio

Determining the appropriate plot



Data Wrangling

`gapminder` data frame in the `gapminder` package

```
library(gapminder)
gapminder
```

```
# A tibble: 1,704 x 6
       country continent  year lifeExp      pop gdpPercap
        <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
 1 Afghanistan      Asia  1952  28.801  8425333  779.4453
 2 Afghanistan      Asia  1957  30.332  9240934  820.8530
 3 Afghanistan      Asia  1962  31.997 10267083  853.1007
 4 Afghanistan      Asia  1967  34.020 11537966  836.1971
 5 Afghanistan      Asia  1972  36.088 13079460  739.9811
 6 Afghanistan      Asia  1977  38.438 14880372  786.1134
 7 Afghanistan      Asia  1982  39.854 12881816  978.0114
 8 Afghanistan      Asia  1987  40.822 13867957  852.3959
 9 Afghanistan      Asia  1992  41.674 16317921  649.3414
10 Afghanistan      Asia  1997  41.763 22227415  635.3414
# ... with 1,694 more rows
```

Base R versus the `tidyverse`

Say we wanted mean life expectancy across all years for Asia

Base R versus the `tidyverse`

Say we wanted mean life expectancy across all years for Asia

```
# Base R
asia <- gapminder[gapminder$continent == "Asia", ]
mean(asia$lifeExp)
```

```
[1] 60.0649
```

Base R versus the `tidyverse`

Say we wanted mean life expectancy across all years for Asia

```
# Base R
asia <- gapminder[gapminder$continent == "Asia", ]
mean(asia$lifeExp)
```

```
[1] 60.0649
```

```
library(dplyr)
gapminder %>% filter(continent == "Asia") %>%
  summarize(mean_exp = mean(lifeExp))
```

```
# A tibble: 1 x 1
  mean_exp
     <dbl>
1  60.0649
```

The pipe %>%

The pipe %>%

- A way to chain together commands

The pipe %>%

- A way to chain together commands
- It is essentially the `dplyr` equivalent to the `+` in `ggplot2`

The 5NG of data viz

The 5NG of data viz

```
geom_point()
geom_line()
geom_histogram()
geom_boxplot()
geom_bar()
```

The Five Main Verbs (5MV) of data wrangling

```
filter()
summarize()
group_by()
mutate()
arrange()
```

`filter()`

- Select a subset of the rows of a data frame.
- The arguments are the "filters" that you'd like to apply.

`filter()`

- Select a subset of the rows of a data frame.
- The arguments are the "filters" that you'd like to apply.

```
library(gapminder); library(dplyr)
gap_2007 <- gapminder %>% filter(year == 2007)
head(gap_2007)
```

```
# A tibble: 6 x 6
      country continent  year lifeExp      pop  gdpPercap
       <fctr>    <fctr> <int>   <dbl>    <int>      <dbl>
1 Afghanistan      Asia  2007  43.828 31889923   974.5803
2     Albania    Europe  2007  76.423  3600523  5937.0295
3     Algeria    Africa  2007  72.301 33333216  6223.3675
4      Angola    Africa  2007  42.731 12420476  4797.2313
5   Argentina  Americas  2007  75.320 40301927 12779.3796
6   Australia   Oceania  2007  81.235 20434176 34435.3674
```

- Use == to compare a variable to a value

Logical operators

- Use | to check for any in multiple filters being true:

Logical operators

- Use | to check for any in multiple filters being true:

```
gapminder %>%
  filter(year == 2002 | continent == "Europe")
```

Logical operators

- Use | to check for any in multiple filters being true:

```
gapminder %>%
  filter(year == 2002 | continent == "Europe")
```

```
# A tibble: 472 x 6
       country continent  year lifeExp      pop gdpPercap
        <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
 1 Afghanistan      Asia  2002  42.129 25268405  726.7341
 2     Albania    Europe  1952  55.230  1282697 1601.0561
 3     Albania    Europe  1957  59.280  1476505 1942.2842
 4     Albania    Europe  1962  64.820  1728137 2312.8890
 5     Albania    Europe  1967  66.220  1984060 2760.1969
 6     Albania    Europe  1972  67.690  2263554 3313.4222
 7     Albania    Europe  1977  68.930  2509048 3533.0039
 8     Albania    Europe  1982  70.420  2780097 3630.8807
 9     Albania    Europe  1987  72.000  3075321 3738.9327
10     Albania    Europe  1992  71.581  3326498 2497.4379
# ... with 462 more rows
```

Logical operators

- Use & or , to check for all of multiple filters being true:

Logical operators

- Use `&` or `,` to check for all of multiple filters being true:

```
gapminder %>%
  filter(year == 2002, continent == "Europe")
```

```
# A tibble: 30 x 6
                   country continent  year lifeExp      pop gdpPercap
                    <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
 1                 Albania    Europe  2002  75.651  3508512  4604.212
 2                 Austria    Europe  2002  78.980  8148312 32417.608
 3                 Belgium    Europe  2002  78.320 10311970 30485.884
 4 Bosnia and Herzegovina    Europe  2002  74.090  4165416  6018.975
 5                Bulgaria    Europe  2002  72.140  7661799  7696.778
 6                 Croatia    Europe  2002  74.876  4481020 11628.389
 7          Czech Republic    Europe  2002  75.510 10256295 17596.210
 8                 Denmark    Europe  2002  77.180  5374693 32166.500
 9                 Finland    Europe  2002  78.370  5193039 28204.591
10                  France    Europe  2002  79.590 59925035 28926.032
# ... with 20 more rows
```

Logical operators

- Use `%in%` to check for any being true (shortcut to using `|` repeatedly with `==`)

Logical operators

- Use `%in%` to check for any being true (shortcut to using `|` repeatedly with `==`)

```
gapminder %>%
  filter(country %in% c("Argentina", "Belgium", "Mexico"),
         year %in% c(1987, 1992))
```

Logical operators

- Use `%in%` to check for any being true (shortcut to using `|` repeatedly with `==`)

```
gapminder %>%
  filter(country %in% c("Argentina", "Belgium", "Mexico"),
         year %in% c(1987, 1992))
```

```
# A tibble: 6 x 6
    country continent  year lifeExp      pop gdpPercap
     <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
1 Argentina  Americas  1987  70.774 31620918  9139.671
2 Argentina  Americas  1992  71.868 33958947  9308.419
3   Belgium    Europe  1987  75.350  9870200 22525.563
4   Belgium    Europe  1992  76.460 10045622 25575.571
5    Mexico  Americas  1987  69.498 80122492  8688.156
6    Mexico  Americas  1992  71.455 88111030  9472.384
```

`summarize()`

- Any numerical summary that you want to apply to a column of a data frame is specified within `summarize()`.

```
max_exp_1997 <- gapminder %>%
  filter(year == 1997) %>%
  summarize(max_exp = max(lifeExp))
max_exp_1997
```

`summarize()`

- Any numerical summary that you want to apply to a column of a data frame is specified within `summarize()`.

```
max_exp_1997 <- gapminder %>%
  filter(year == 1997) %>%
  summarize(max_exp = max(lifeExp))
max_exp_1997
```

```
# A tibble: 1 x 1
  max_exp
    <dbl>
1   80.69
```

Combining `summarize()` with `group_by()`

When you'd like to determine a numerical summary for all levels of a different categorical variable

```
max_exp_1997_by_cont <- gapminder %>%
  filter(year == 1997) %>%
  group_by(continent) %>%
  summarize(max_exp = max(lifeExp))
max_exp_1997_by_cont
```

Combining `summarize()` with `group_by()`

When you'd like to determine a numerical summary for all levels of a different categorical variable

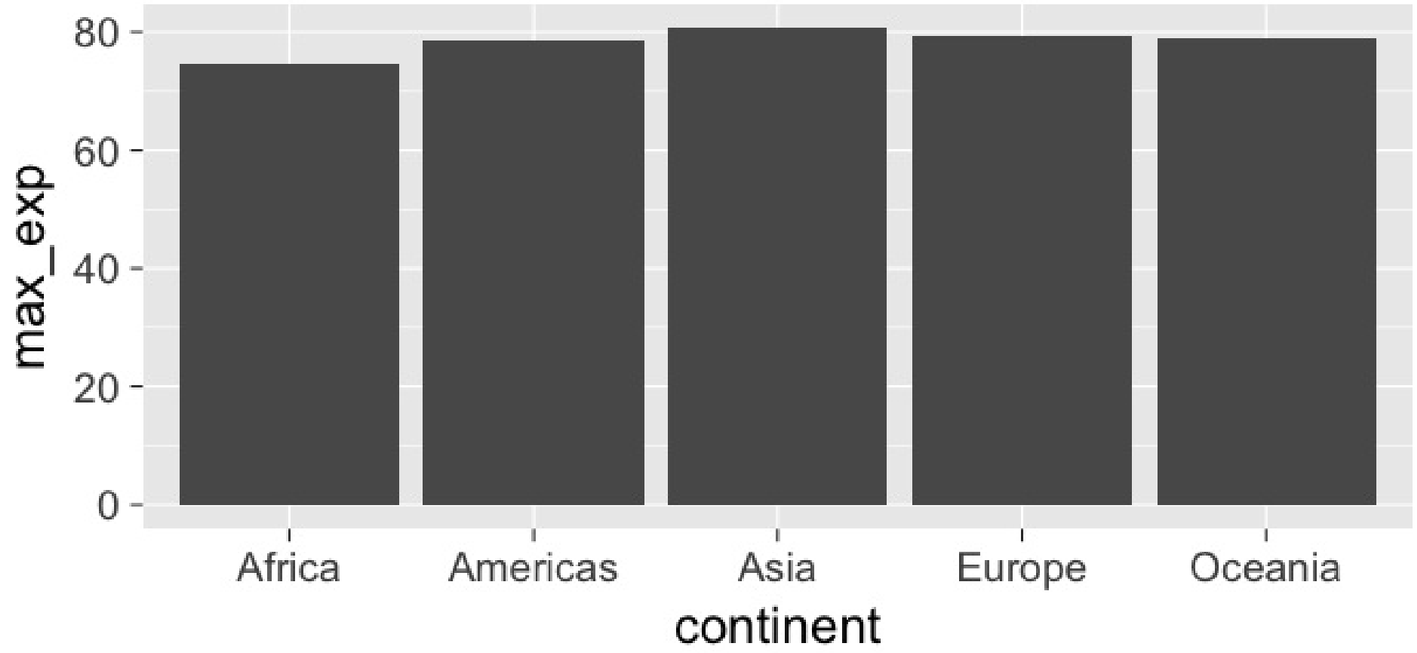
```
max_exp_1997_by_cont <- gapminder %>%
  filter(year == 1997) %>%
  group_by(continent) %>%
  summarize(max_exp = max(lifeExp))
max_exp_1997_by_cont
```

```
# A tibble: 5 x 2
  continent max_exp
     <fctr>   <dbl>
1    Africa  74.772
2  Americas  78.610
3      Asia  80.690
4    Europe  79.390
5   Oceania  78.830
```

`ggplot2` revisited

For aggregated data, use `geom_col`

```
ggplot(data = max_exp_1997_by_cont,
       mapping = aes(x = continent, y = max_exp)) +
  geom_col()
```



The 5MV

- `filter()`
- `summarize()`
- `group_by()`

The 5MV

- `filter()`
- `summarize()`
- `group_by()`
- `mutate()`

The 5MV

- `filter()`
- `summarize()`
- `group_by()`
- `mutate()`
- `arrange()`

`mutate()`

- Allows you to
  1. create a new variable with a specific value OR
  2. create a new variable based on other variables OR
  3. change the contents of an existing variable

mutate()

- Allows you to
  1. create a new variable with a specific value OR
  2. create a new variable based on other variables OR
  3. change the contents of an existing variable

```
gap_plus <- gapminder %>% mutate(just_one = 1)
head(gap_plus)
```

```
# A tibble: 6 x 7
      country continent  year lifeExp      pop gdpPercap just_one
       <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>    <dbl>
1 Afghanistan      Asia  1952  28.801  8425333  779.4453        1
2 Afghanistan      Asia  1957  30.332  9240934  820.8530        1
3 Afghanistan      Asia  1962  31.997 10267083  853.1007        1
4 Afghanistan      Asia  1967  34.020 11537966  836.1971        1
5 Afghanistan      Asia  1972  36.088 13079460  739.9811        1
6 Afghanistan      Asia  1977  38.438 14880372  786.1134        1
```

`mutate()`

- Allows you to
  1. create a new variable with a specific value OR
  2. create a new variable based on other variables OR
  3. change the contents of an existing variable

mutate()

- Allows you to
  1. create a new variable with a specific value OR
  2. create a new variable based on other variables OR
  3. change the contents of an existing variable

```
gap_w_gdp <- gapminder %>% mutate(gdp = pop * gdpPercap)
head(gap_w_gdp)
```

```
# A tibble: 6 x 7
      country continent  year lifeExp      pop gdpPercap         gdp
       <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>       <dbl>
1 Afghanistan      Asia  1952  28.801  8425333  779.4453  6567086330
2 Afghanistan      Asia  1957  30.332  9240934  820.8530  7585448670
3 Afghanistan      Asia  1962  31.997 10267083  853.1007  8758855797
4 Afghanistan      Asia  1967  34.020 11537966  836.1971  9648014150
5 Afghanistan      Asia  1972  36.088 13079460  739.9811  9678553274
6 Afghanistan      Asia  1977  38.438 14880372  786.1134 11697659231
```

`mutate()`

- Allows you to
  1. create a new variable with a specific value OR
  2. create a new variable based on other variables OR
  3. change the contents of an existing variable

mutate()

- Allows you to
  1. create a new variable with a specific value OR
  2. create a new variable based on other variables OR
  3. change the contents of an existing variable

```
gap_weird <- gapminder %>% mutate(pop = pop + 1000)
head(gap_weird)
```

```
# A tibble: 6 x 6
      country continent  year lifeExp      pop gdpPercap
       <fctr>    <fctr> <int>   <dbl>    <dbl>     <dbl>
1 Afghanistan      Asia  1952  28.801  8426333  779.4453
2 Afghanistan      Asia  1957  30.332  9241934  820.8530
3 Afghanistan      Asia  1962  31.997 10268083  853.1007
4 Afghanistan      Asia  1967  34.020 11538966  836.1971
5 Afghanistan      Asia  1972  36.088 13080460  739.9811
6 Afghanistan      Asia  1977  38.438 14881372  786.1134
```

`arrange()`

- Reorders the rows in a data frame based on the values of one or more variables

arrange()

- Reorders the rows in a data frame based on the values of one or more variables

```
gapminder %>%
  arrange(year, country)
```

```
# A tibble: 1,704 x 6
       country continent  year lifeExp      pop  gdpPercap
        <fctr>    <fctr> <int>   <dbl>    <int>      <dbl>
 1 Afghanistan      Asia  1952  28.801  8425333   779.4453
 2     Albania    Europe  1952  55.230  1282697  1601.0561
 3     Algeria    Africa  1952  43.077  9279525  2449.0082
 4      Angola    Africa  1952  30.015  4232095  3520.6103
 5   Argentina  Americas  1952  62.485 17876956  5911.3151
 6   Australia   Oceania  1952  69.120  8691212 10039.5956
 7     Austria    Europe  1952  66.800  6927772  6137.0765
 8     Bahrain      Asia  1952  50.939   120447  9867.0848
 9  Bangladesh      Asia  1952  37.484 46886859   684.2442
10     Belgium    Europe  1952  68.000  8730405  8343.1051
# ... with 1,694 more rows
```

`arrange()`

- Can also put into descending order

arrange()

- Can also put into descending order

```
gapminder %>%
  filter(year > 2000) %>%
  arrange(desc(lifeExp))
```

Don't mix up `arrange` and `group_by`

- `group_by` is used (mostly) with `summarize` to calculate summaries over groups
- `arrange` is used for sorting

Don't mix up `arrange` and `group_by`

This doesn't really do anything useful

```
gapminder %>% group_by(year)
```

```
Source: local data frame [1,704 x 6]
Groups: year [12]

# A tibble: 1,704 x 6
       country continent  year lifeExp      pop gdpPercap
        <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
 1 Afghanistan      Asia  1952  28.801  8425333  779.4453
 2 Afghanistan      Asia  1957  30.332  9240934  820.8530
 3 Afghanistan      Asia  1962  31.997 10267083  853.1007
 4 Afghanistan      Asia  1967  34.020 11537966  836.1971
 5 Afghanistan      Asia  1972  36.088 13079460  739.9811
 6 Afghanistan      Asia  1977  38.438 14880372  786.1134
 7 Afghanistan      Asia  1982  39.854 12881816  978.0114
 8 Afghanistan      Asia  1987  40.822 13867957  852.3959
 9 Afghanistan      Asia  1992  41.674 16317921  649.3414
10 Afghanistan      Asia  1997  41.763 22227415  635.3414
# ... with 1,694 more rows
```

Don't mix up `arrange` and `group_by`

But this does

```
gapminder %>% arrange(year)
```

```
# A tibble: 1,704 x 6
       country continent  year lifeExp      pop  gdpPercap
        <fctr>    <fctr> <int>   <dbl>    <int>      <dbl>
 1 Afghanistan      Asia  1952  28.801  8425333   779.4453
 2     Albania    Europe  1952  55.230  1282697  1601.0561
 3     Algeria    Africa  1952  43.077  9279525  2449.0082
 4      Angola    Africa  1952  30.015  4232095  3520.6103
 5   Argentina  Americas  1952  62.485 17876956  5911.3151
 6   Australia   Oceania  1952  69.120  8691212 10039.5956
 7     Austria    Europe  1952  66.800  6927772  6137.0765
 8     Bahrain      Asia  1952  50.939   120447  9867.0848
 9  Bangladesh      Asia  1952  37.484 46886859   684.2442
10     Belgium    Europe  1952  68.000  8730405  8343.1051
# ... with 1,694 more rows
```

Changing of observation unit

True or False

Each of `filter`, `mutate`, and `arrange` change the observational unit.

Changing of observation unit

True or False

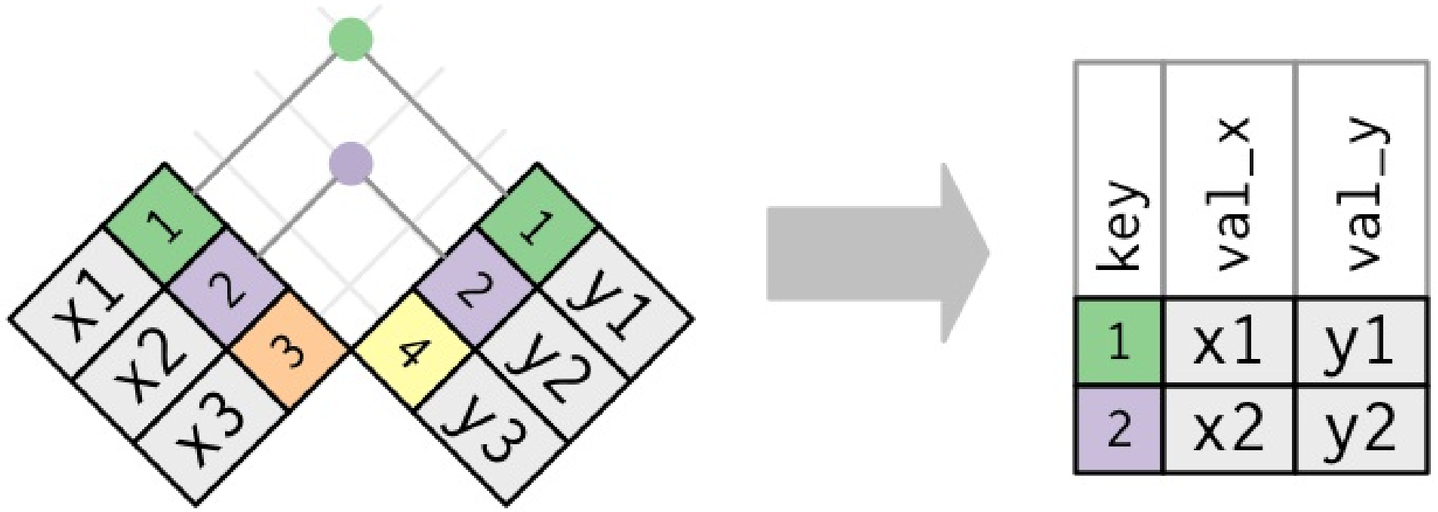
Each of `filter`, `mutate`, and `arrange` change the observational unit.

True or False

`group_by() %>% summarize()` changes the observational unit.

What is meant by "joining data frames" and why is it useful?

What is meant by "joining data frames" and why is it useful?



key	val_x	val_y
1	x1	y1
2	x2	y2

Does cost of living in a state relate to whether police officers live in the cities they patrol? What about state political ideology?

```
library(fivethirtyeight); library(readr)
police2 <- police_locals %>% select(city, all)
ideology <- read_csv("https://ismayc.github.io/Effective-Data-Storytelling-us
police_join <- inner_join(x = police2, y = ideology, by = "city")
police_join
```

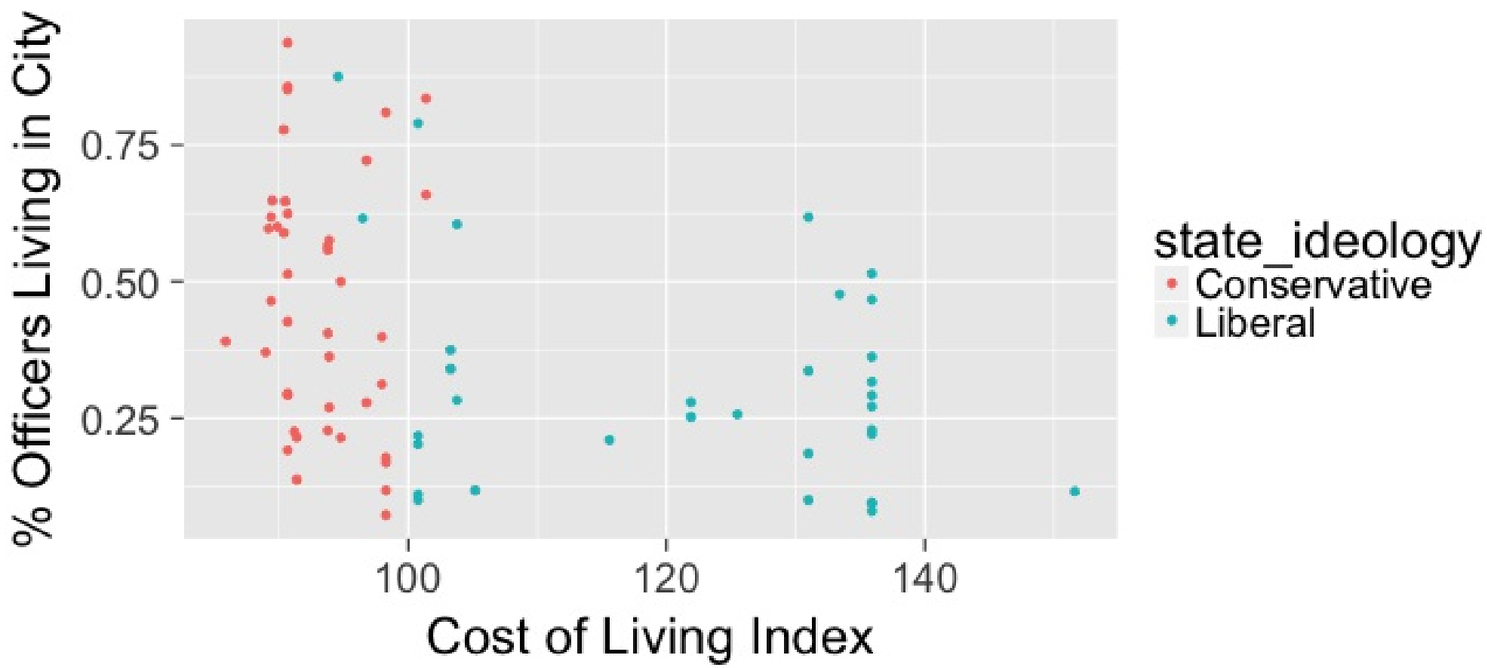
```
# A tibble: 75 x 4
           city       all                state state_ideology
          <chr>     <dbl>                <chr>          <chr>
 1     New York 0.6179567             New York        Liberal
 2      Chicago 0.8750000             Illinois        Liberal
 3  Los Angeles 0.2282178           California        Liberal
 4   Washington 0.1156317 District of Columbia        Liberal
 5      Houston 0.2922078                Texas   Conservative
 6 Philadelphia 0.8354012         Pennsylvania   Conservative
 7      Phoenix 0.3117318              Arizona   Conservative
 8    San Diego 0.3621076           California        Liberal
 9       Dallas 0.1914008                Texas   Conservative
10      Detroit 0.3705972             Michigan   Conservative
# ... with 65 more rows
```

```
url <- paste0("https://ismayc.github.io/",
              "Effective-Data-Storytelling-using-the-tidyverse/",
              "datasets/cost_of_living.csv")
cost_of_living <- read_csv(url)
police_join_cost <- inner_join(
    x = police_join,
    y = cost_of_living,
    by = "state"
  )
police_join_cost %>% select(-state) %>% arrange(city)
```

```
# A tibble: 75 x 5
                   city       all state_ideology index col_group
                  <chr>     <dbl>          <chr> <dbl>     <chr>
 1       Albany, N.Y. 0.1853933        Liberal 131.0      high
 2  Albuquerque, N.M. 0.6156716        Liberal  96.5       mid
 3     Arlington, Va. 0.2022059        Liberal 100.8       mid
 4            Atlanta 0.1372881   Conservative  91.4       low
 5      Austin, Texas 0.2947103   Conservative  90.7       low
 6          Baltimore 0.2571429        Liberal 125.5      high
 7   Baton Rouge, La. 0.2142857   Conservative  94.8       low
 8   Birmingham, Ala. 0.2252252   Conservative  91.2       low
 9            Boston 0.4765625        Liberal 133.4      high
10 Brownsville, Texas 0.5135135   Conservative  90.7       low
# ... with 65 more rows
```

Does cost of living in a state relate to whether police officers live in the cities they patrol? What about state political ideology?

```
ggplot(data = police_join_cost,
       mapping = aes(x = index, y = all)) +
  geom_point(mapping = aes(color = state_ideology)) +
  labs(x = "Cost of Living Index", y = "% Officers Living in City")
```



Practice (Lay out what the resulting table should look like on paper first.)

Use the 5MV to answer problems from R data packages, e.g., [`nycflights13` → `weather`]

1. What is the maximum arrival delay for each carrier departing JFK? [`nycflights13` → `flights`]
2. Determine the top five movies in terms of domestic return on investment for 2013 scaled data [`fivethirtyeight` → `bechdel`]
3. Include the name of the destination airport as a column in the `flights` data frame [`nycflights13` → `flights, airports`]

DEMO of `dplyr` in RStudio

Homework for tomorrow

- Review the Data Import cheatsheet and use the `readr`, `readxl`, and `haven` packages to read in data from CSVs, Excel XLS/XLSX files, STATA/SPSS files
- Use `ggplot2` and `dplyr` to make graphics and produce data wrangling on data sets of interest to you read in from the previous step

Before you go

- Tell me on the sheet of paper you have what types of models you usually run (i.e., linear regression, logistic regression, mixed models, etc.)