Refining your plots

Daniel Anderson
Week 6, Class 1

UNIVERSITY OF OREGON

Agenda

- Axes and aspect ratios
- Labels and captions
- Annotations
- Themes

Agenda

- Axes and aspect ratios
- Labels and captions
- Annotations
- Themes

What we won't get to

Each of the following are pretty fundamental to good data viz, but we won't have time to go over them today. Please make sure to read the corresponding chapters:

- Handling high data density (lots of overlapping points)
- Compound figures
  - See {patchwork} and {cowplot}
- Exporting figures

Learning Objectives

- Understand how to make a wide variety of tweaks to ggplot to essentially make it look however you want it to.
- Understand common modifications to plots to make them more clear and reduce cognitive load

First

Things I learned from you

First

Things I learned from you

- From Teresa: `coord_fixed` works **really** well on heat maps (in most cases). Try it!

First

Things I learned from you

- From Teresa: `coord_fixed` works **really** well on heat maps (in most cases). Try it!

- From Monika: Use `NA` within `separate` to drop columns immediately (rather than naming them something temporary and removing with `select`)

    - e.g., `separate(col, c(NA, date, location, NA))` would keep only the middle two columns

First

Things I learned from you

- From Teresa: `coord_fixed` works **really** well on heat maps (in most cases). Try it!

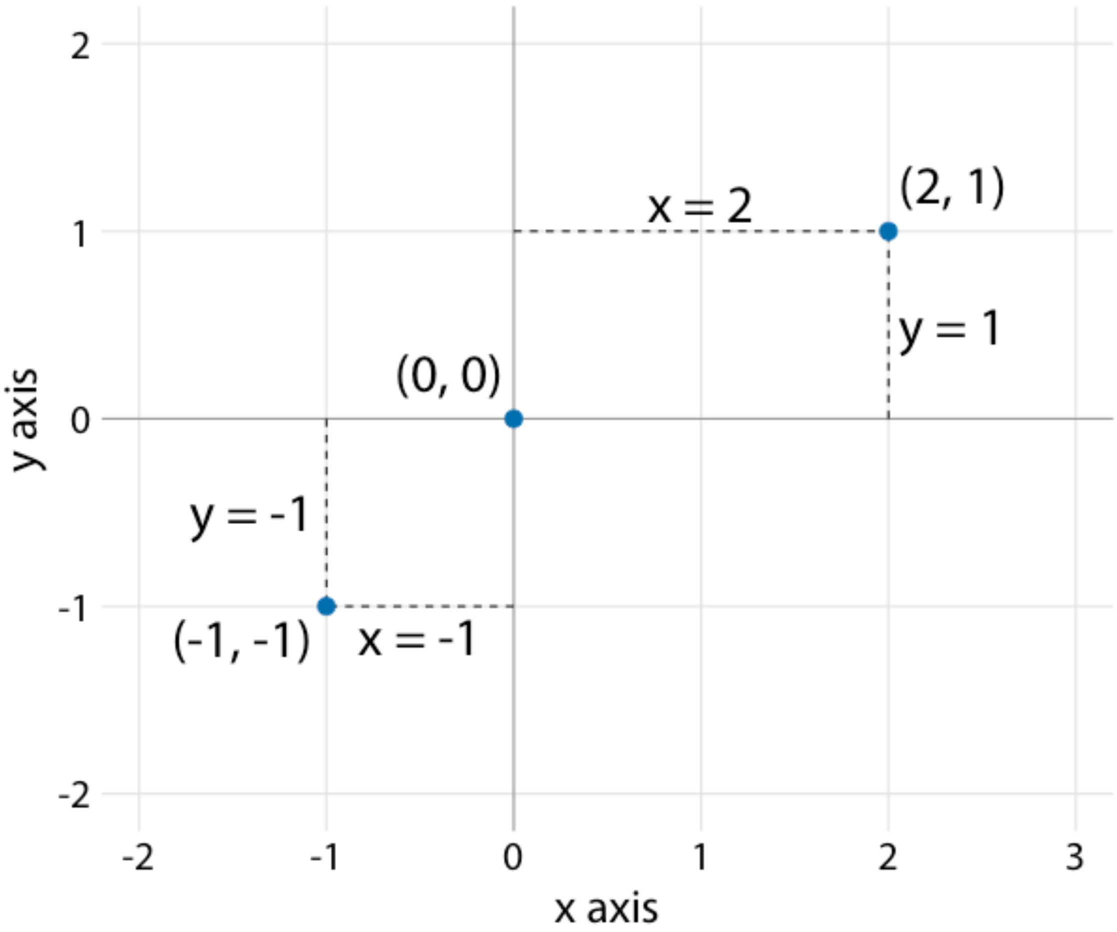
- From Monika: Use `NA` within `separate` to drop columns immediately (rather than naming them something temporary and removing with `select`)

  - e.g., `separate(col, c(NA, date, location, NA))` would keep only the middle two columns

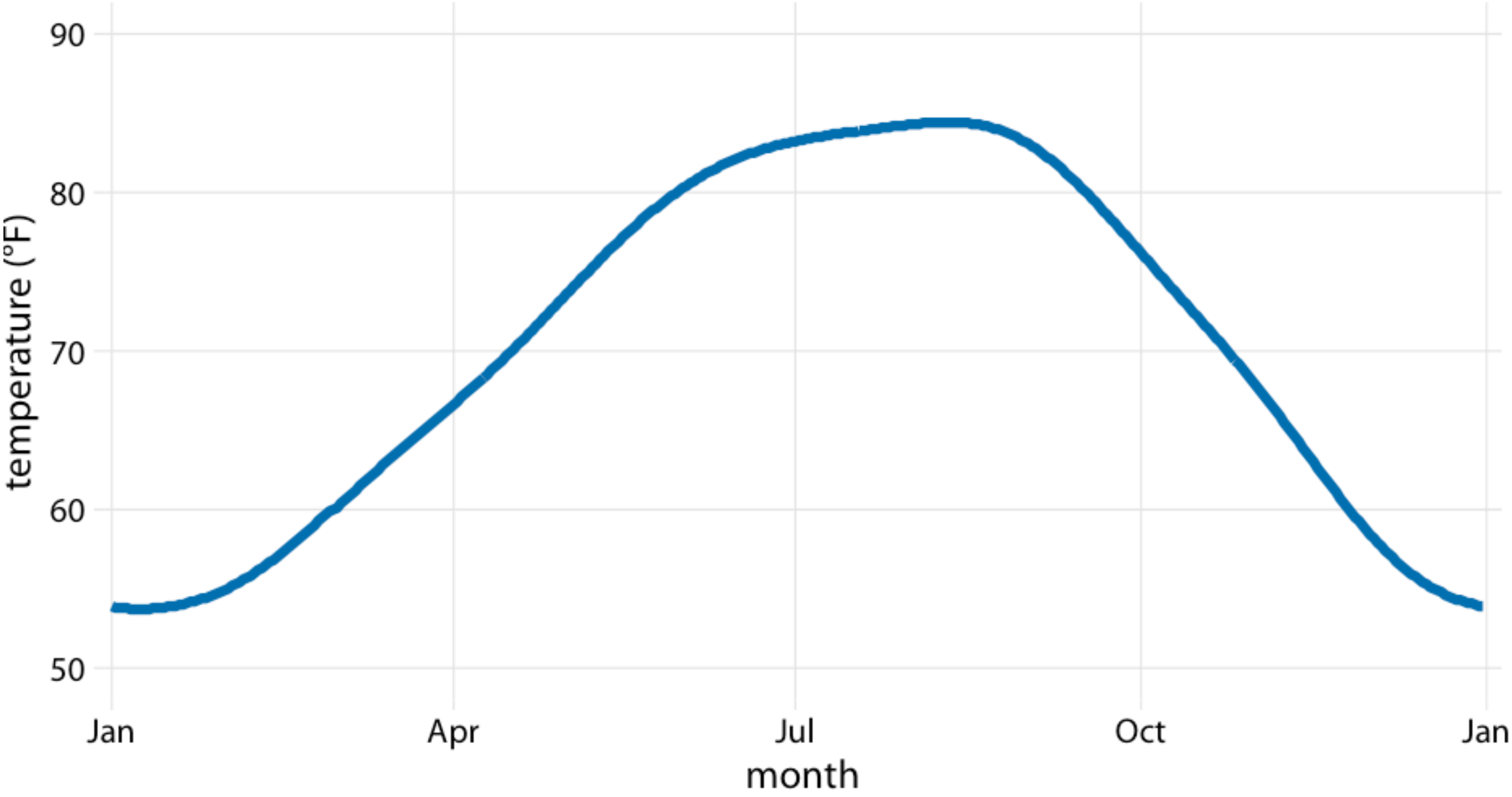
- From Andrew: {shadowtext} makes cool labels with `geom_shadowtext`! (see vignette here

Axes

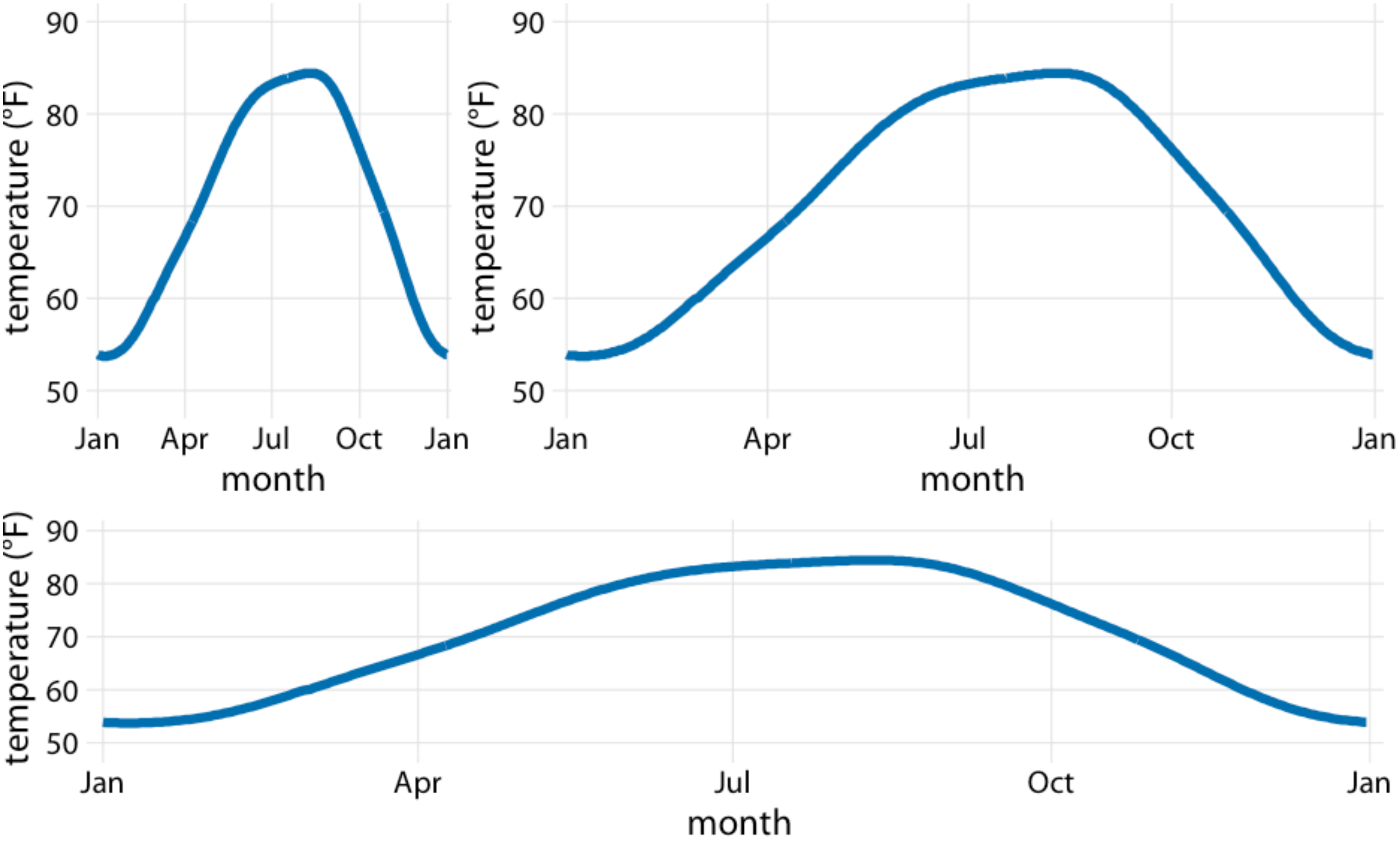
- Cartesian coordinates - what we generally use

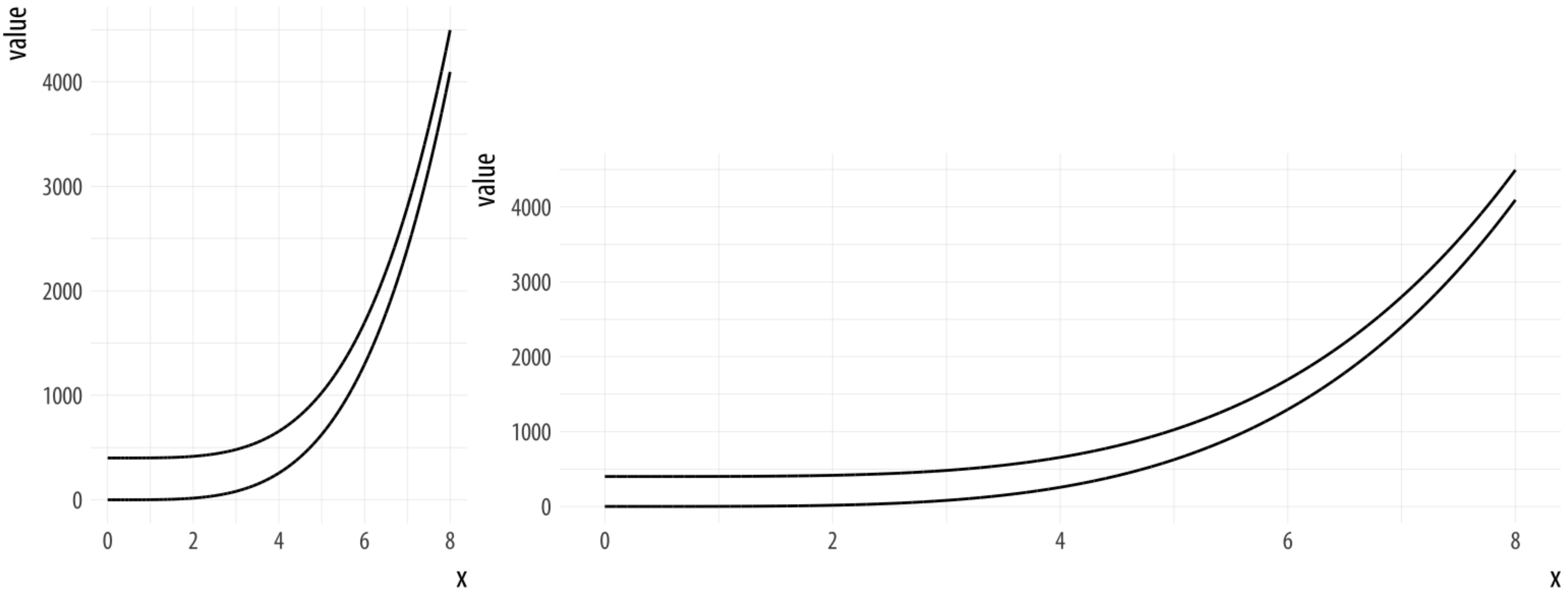


Different units



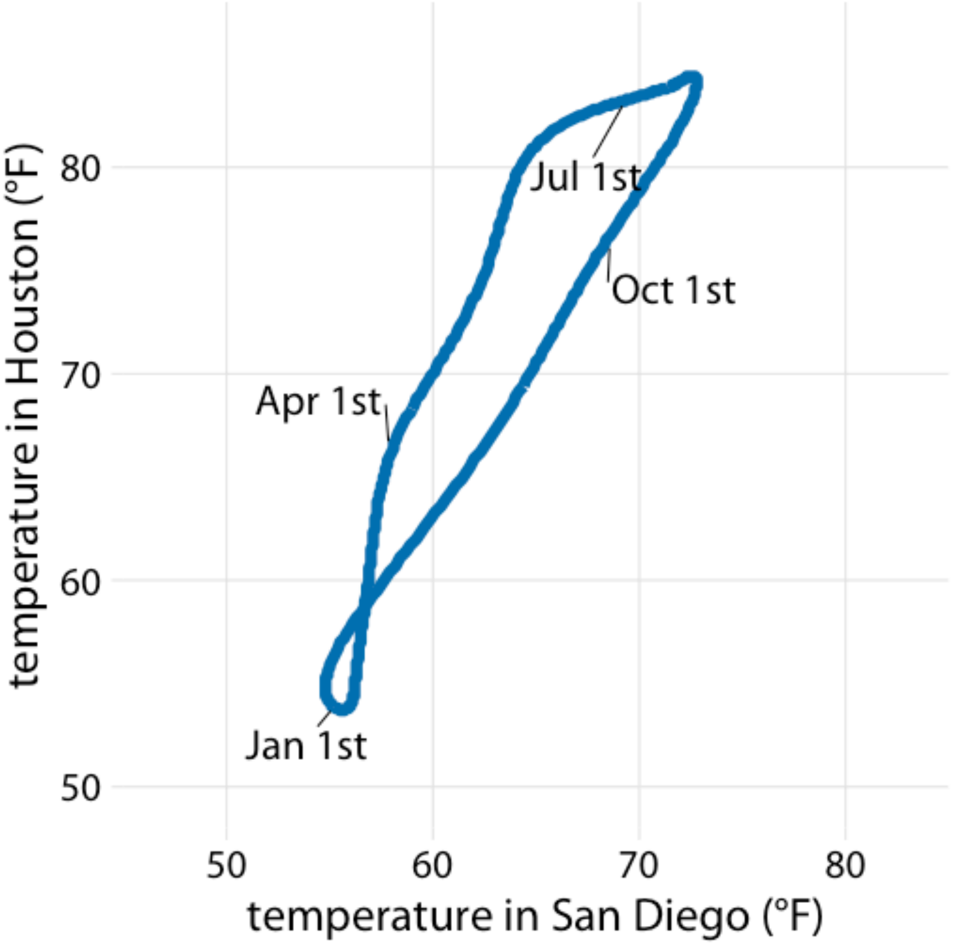
Aspect ratio




value
4000
3000
2000
1000
0
0
2
4
6
8
x
value
4000
3000
2000
1000
0
0
2
4
6
8
x

Same scales

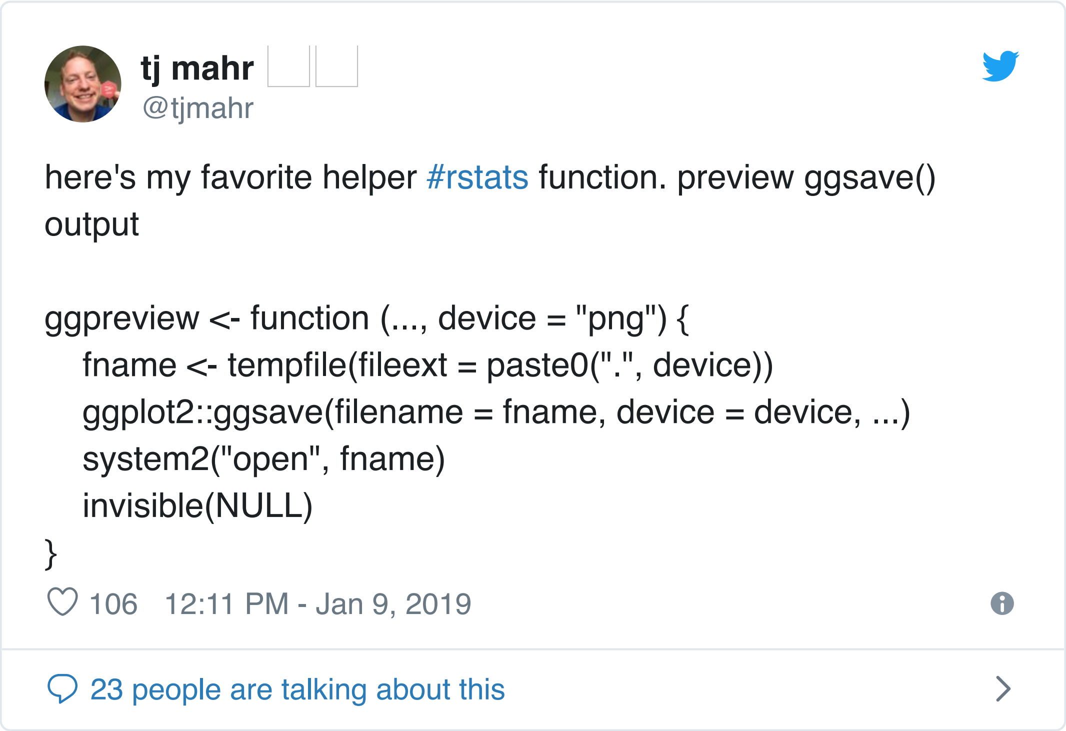
Use `coord_fixed()`



Changing aspect ratio

- Explore how your plot will look in its final size
- No hard/fast rules (if on different scales)
- Not even really rules of thumb
- Keep visual perception in mind
- Try your best to be truthful - show the trend/relation, but don't exaggerate/hide it

Handy function



tj mahr □□
@tjmahr

here's my favorite helper #rstats function. preview ggsave() output

```
ggpreview <- function (..., device = "png") {
   fname <- tempfile(fileext = paste0(".", device))
   ggplot2::ggsave(filename = fname, device = device, ...)
   system2("open", fname)
   invisible(NULL)
}
```

♡ 106   12:11 PM - Jan 9, 2019

23 people are talking about this

Gist

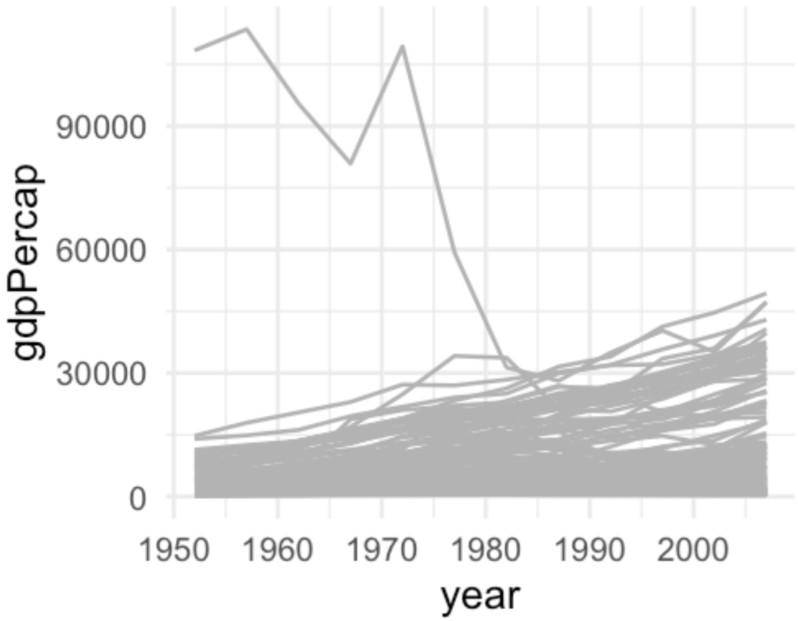
(side note: gists are a good way to share things)

- See the full code/example here
- Let's take 5-10 minutes to play around:
  - Create a plot (could even be the example in the gist)
  - Try different aspect ratios by changing the width/length

Scale transformations

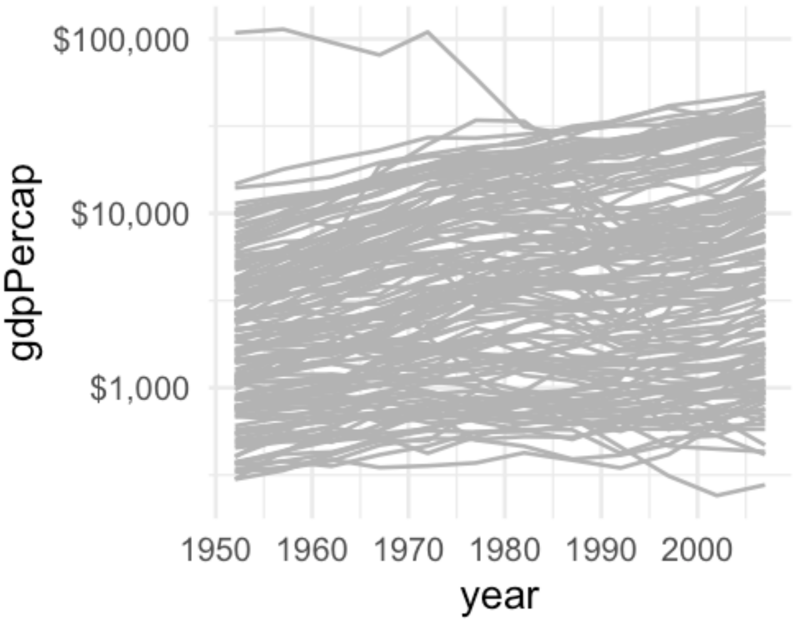
Raw scale

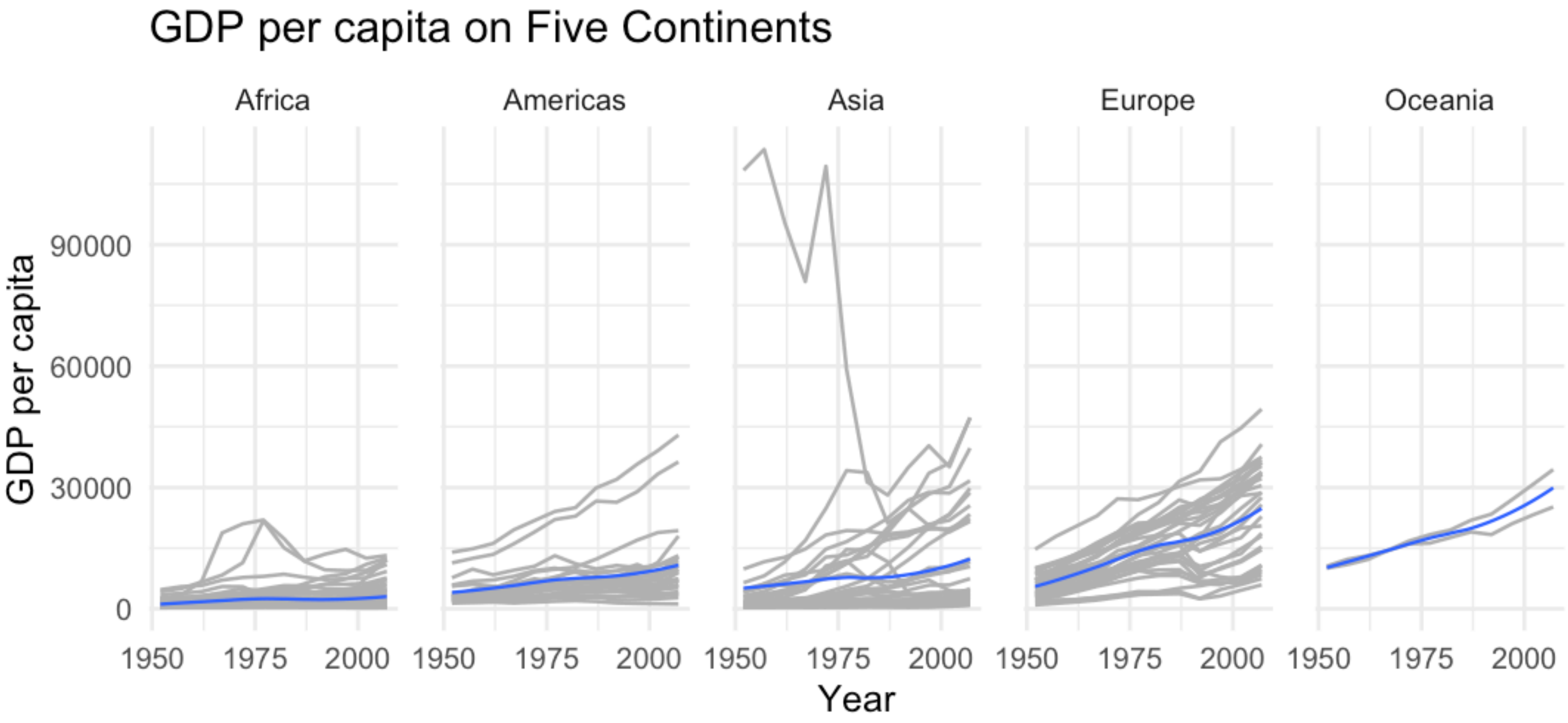
```
library(gapminder)
ggplot(gapminder, aes(year, gdpPerc
  geom_line(aes(group = country),
            color = "gray70")
```

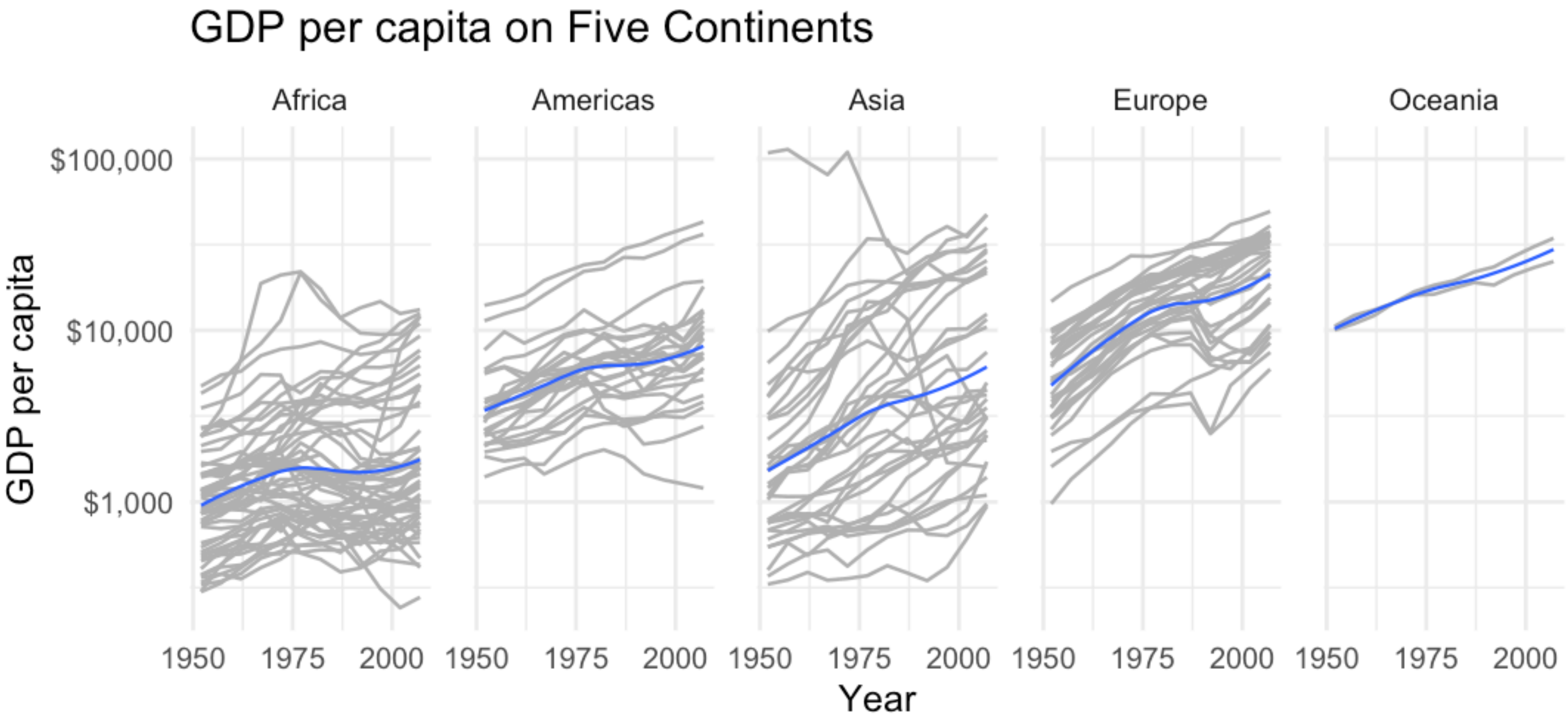


Log10 scale

```
ggplot(gapminder, aes(year, gdpPerc
  geom_line(aes(group = country),
            color = "gray70") +
  scale_y_log10(labels = scales::do
```



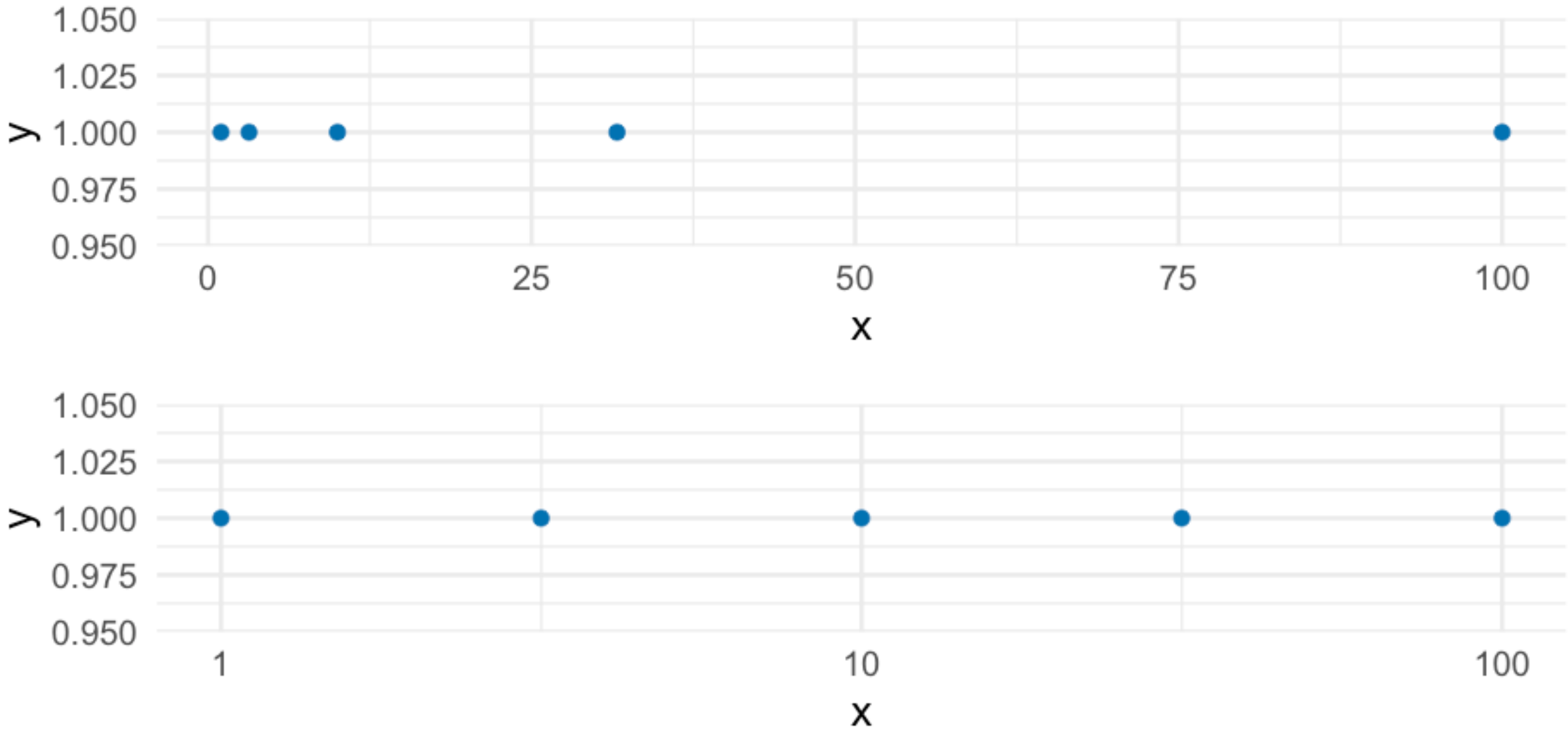

GDP per capita on Five Continents
Africa
Americas
Asia
Europe
Oceania
GDP per capita
90000
60000
30000
0
1950
1975
2000
Year


GDP per capita on Five Continents
Africa
Americas
Asia
Europe
Oceania
$100,000
$10,000
$1,000
GDP per capita
1950
1975
2000
Year

Scales

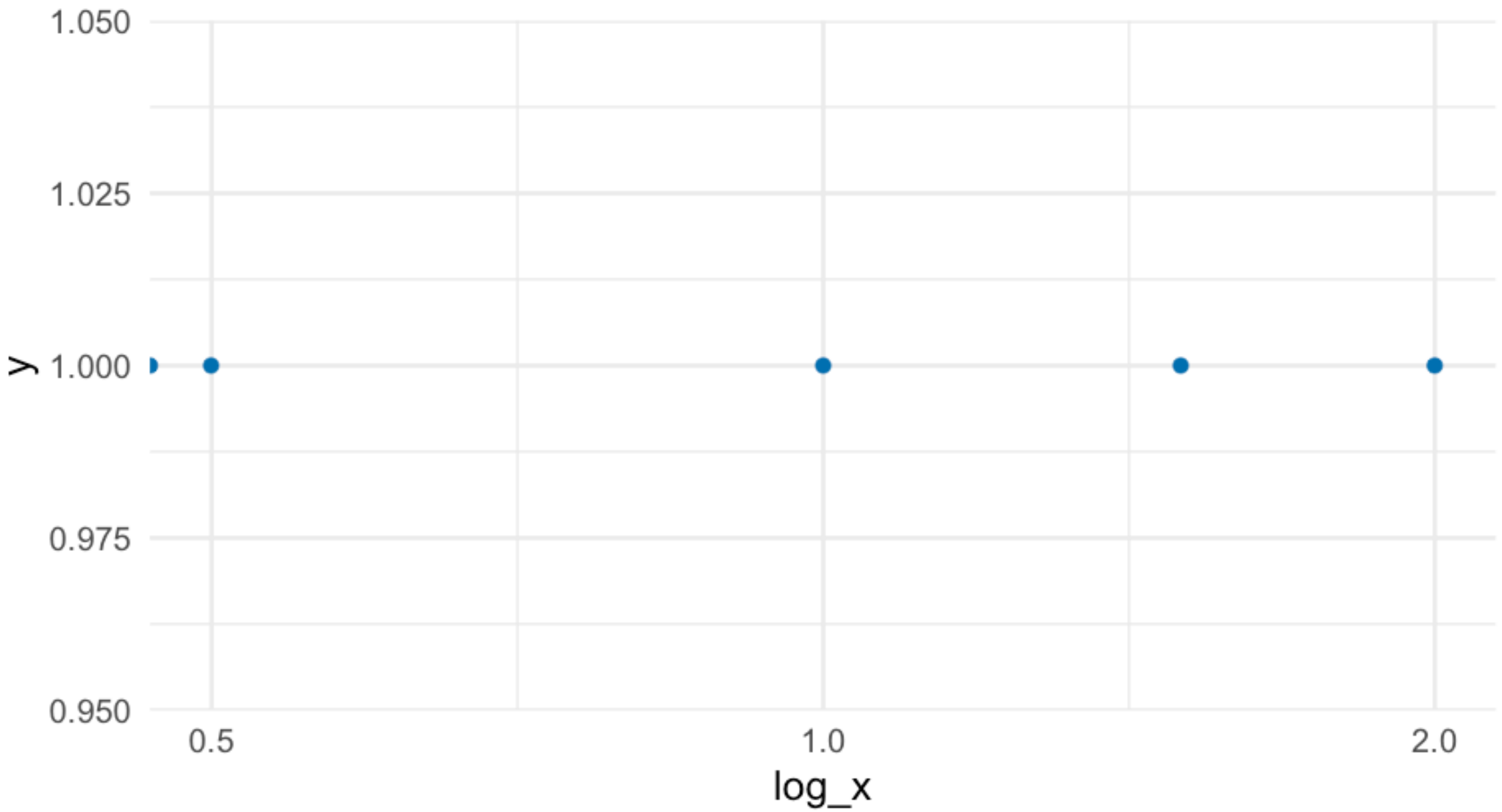
```r
d <- tibble(x = c(1, 3.16, 10, 31.6, 100),
            log_x = log10(x))

ggplot(d, aes(x, 1)) +
  geom_point(color = "#0072B2")

ggplot(d, aes(x, 1)) +
  geom_point(color = "#0072B2") +
  scale_x_log10()

ggplot(d, aes(log_x, 1)) +
  geom_point(color = "#0072B2")
```

Scales


1.050
1.025
1.000
0.975
0.950
y
0
25
50
75
100
x
1.050
1.025
1.000
0.975
0.950
y
1
10
100
x

Don't transform twice

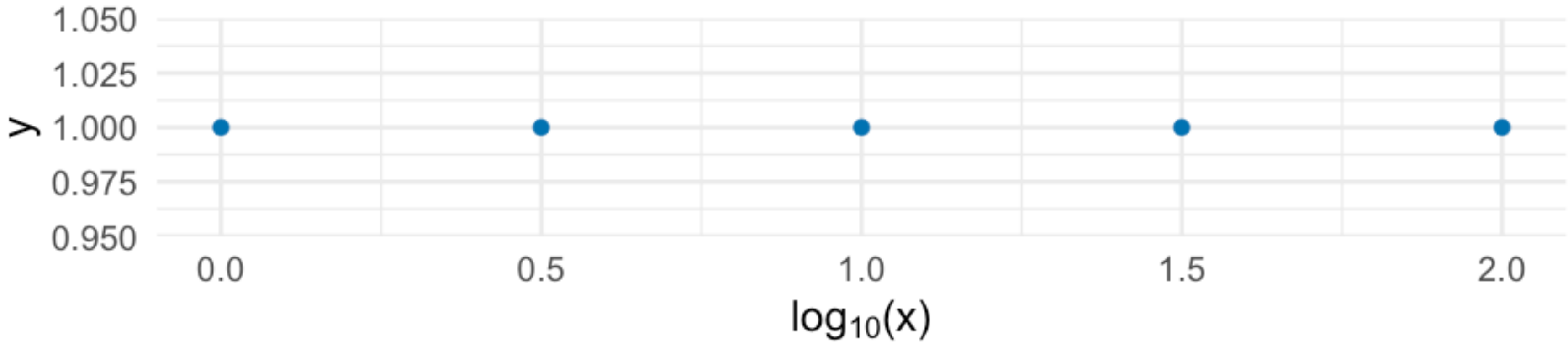
```
ggplot(d, aes(log_x, 1)) +
  geom_point(color = "#0072B2") +
  scale_x_log10()
```



Careful with labeling

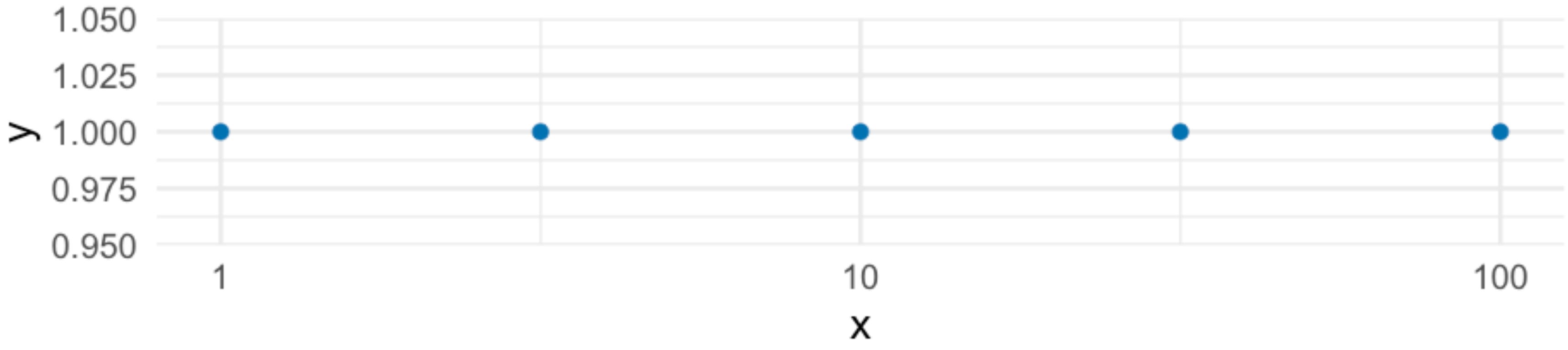
- Has the scale or the data been log transformed?
- Specify the base

```r
ggplot(d, aes(log_x, 1)) +
  geom_point(color = "#0072B2") +
  labs(x = expression(paste("log"["10"], "(x)")))
```



Labels should denote the data, not the scale of the axis

```r
ggplot(d, aes(x, 1)) +
  geom_point(color = "#0072B2") +
  scale_x_log10()
```



Labeling the above with $log_{10}(x)$ would be ambiguous and confusing

Labels and captions

Disclaimer

- APA style requires the labels be made in specific ways
- Much of the following discussion still applies
- Our book (Wilke) uses a similar style throughout

Title

What is the point of your figure?

Title

What is the point of your figure?

What are you trying to communicate

Title

What is the point of your figure?

What are you trying to communicate

- Figures should have only one title

Title

What is the point of your figure?

What are you trying to communicate

- Figures should have only one title
- Use integrated title/subtitles for sharing with a broad audience
    - Blog posts
    - Social media
    - Reports to stakeholders

Title

What is the point of your figure?

What are you trying to communicate

- Figures should have only one title
- Use integrated title/subtitles for sharing with a broad audience
  - Blog posts
  - Social media
  - Reports to stakeholders
- Keep figures in subtext when there's a designated format you must adhere to

Title

What is the point of your figure?

What are you trying to communicate

- Figures should have only one title
- Use integrated title/subtitles for sharing with a broad audience
    - Blog posts
    - Social media
    - Reports to stakeholders
- Keep figures in subtext when there's a designated format you must adhere to
- Make sure your figure has a title
    - Should not start with "This figure displays/shows..."

Caption

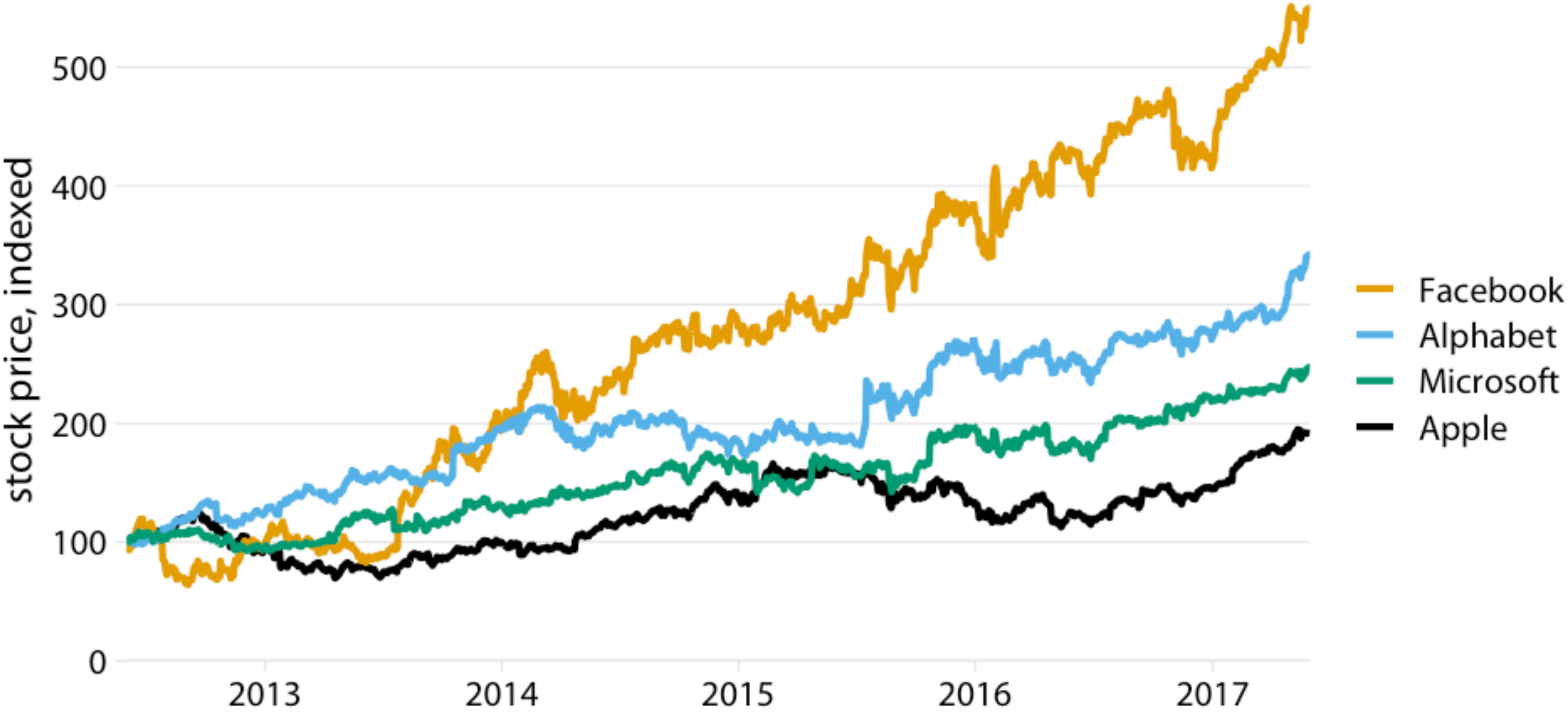
Consider stating the data source

Axis labels

- The title for the axis
- Critical for communication
- **Never** use variable names (very common and very poor practice)
- State the measure and the unit (if quantitative)
  - e.g., "Brain Mass (grams)", "Support for Measure (millions of people)", "Dollars spent"
  - Categorical variable likely will not need to the measurement unit

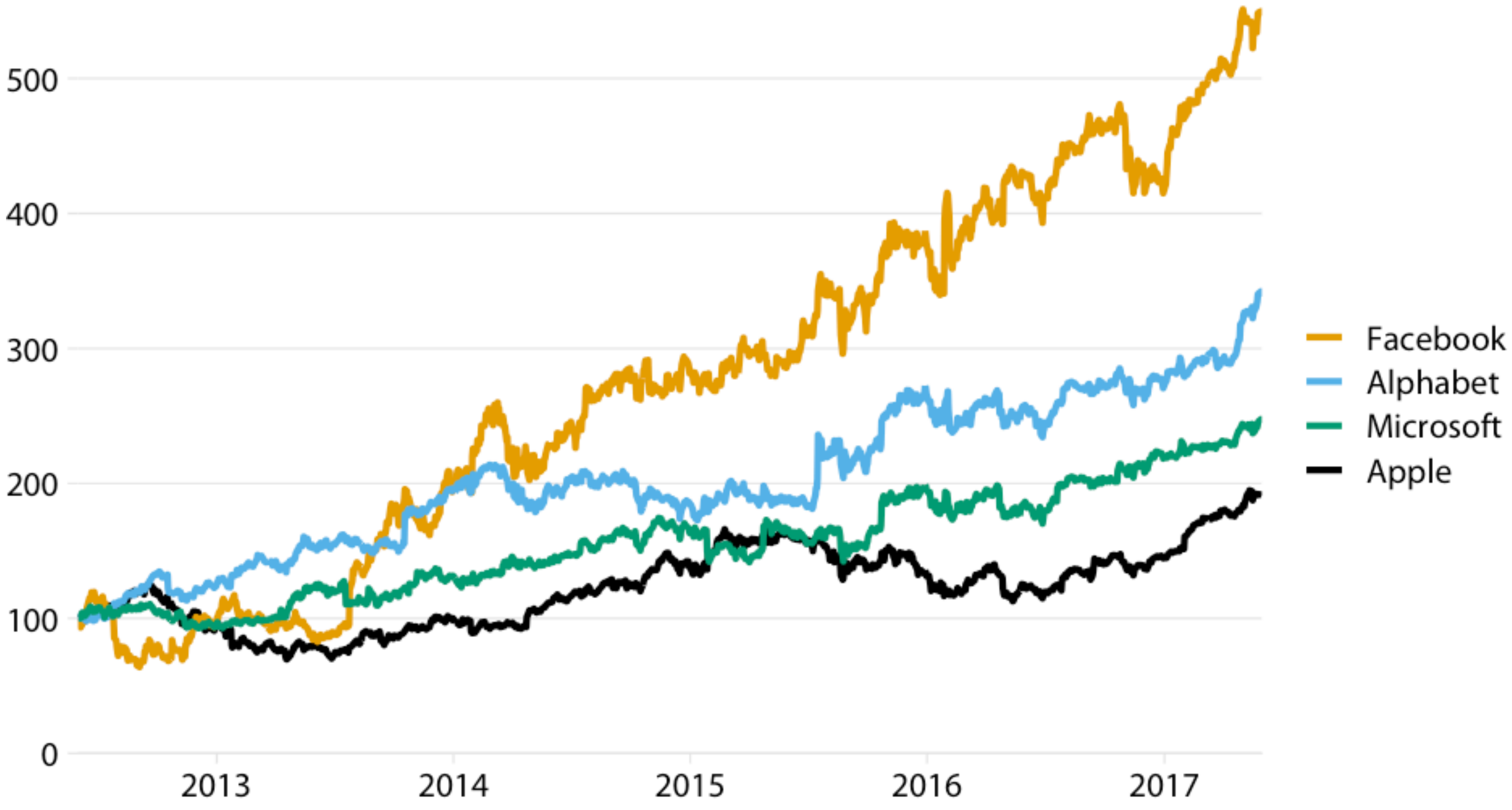
Omission

- Consider omitting obvious or redundant labels
  - Use `labs(x = NULL)` or `labs(x = "")`

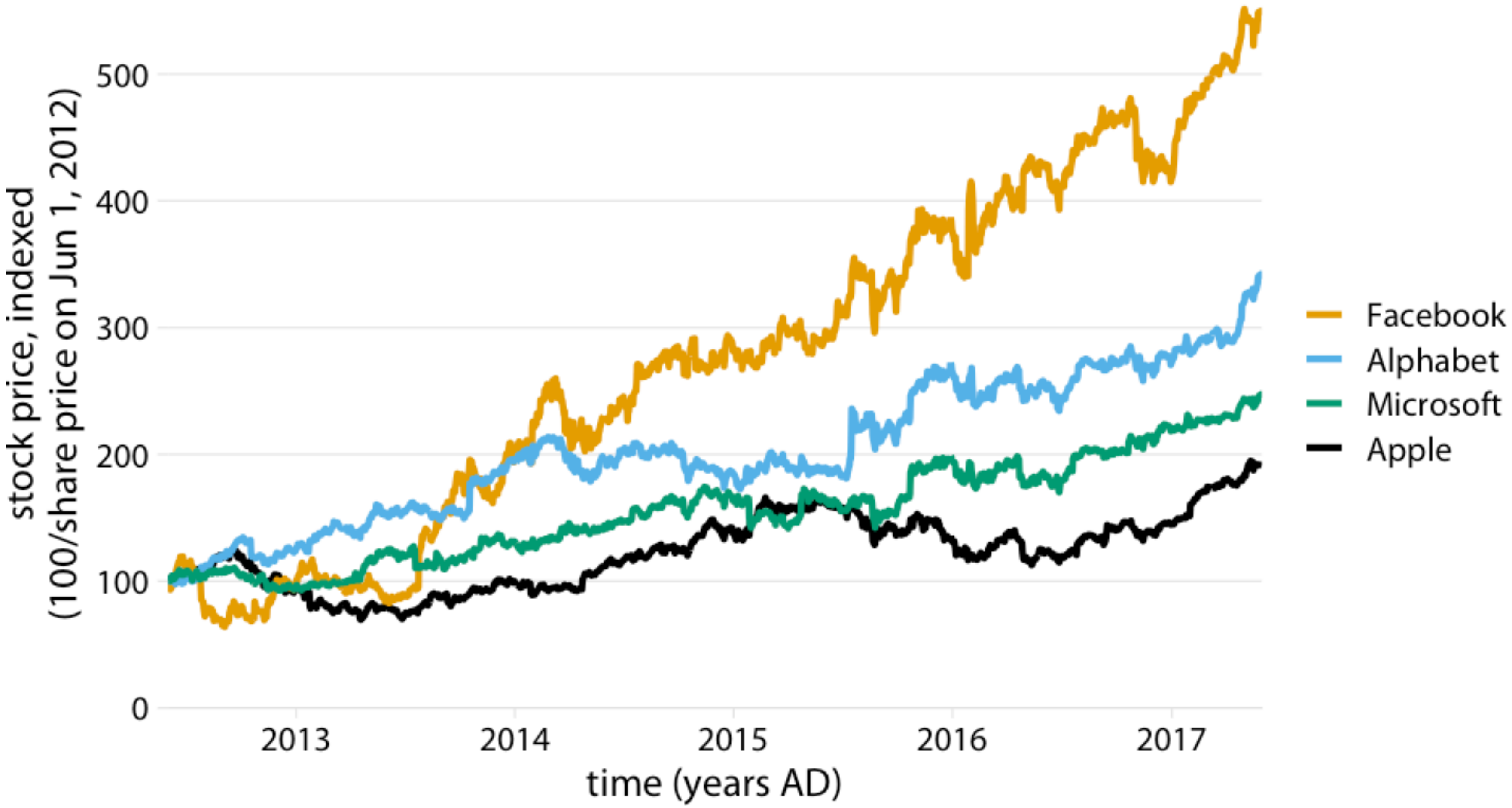


Omission

- Do not omit title that are not obvious



Don't overdo it



Annotations

Among the most effective

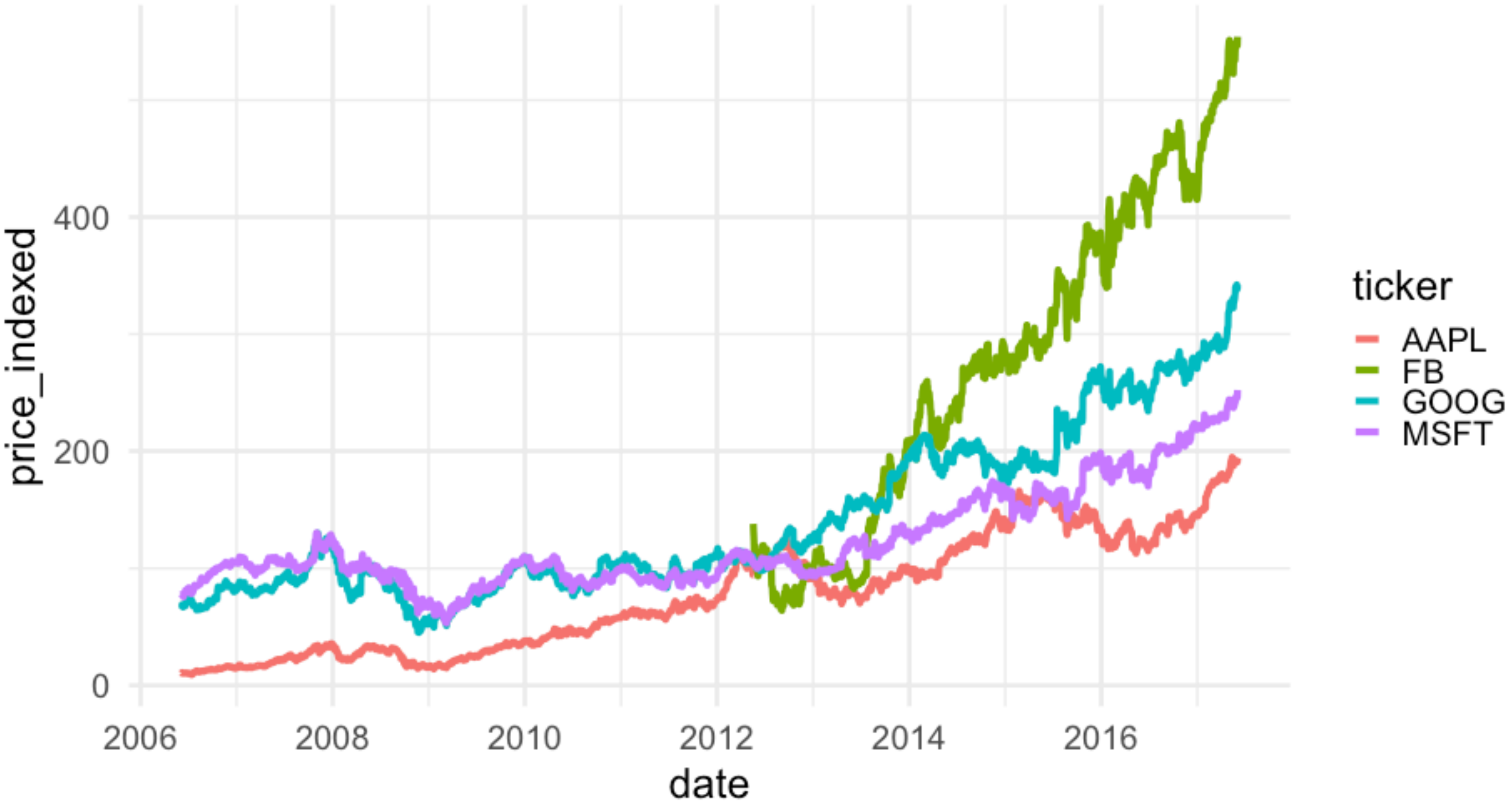
- If possible, try to remove legends, and just include annotations

Building up a plot

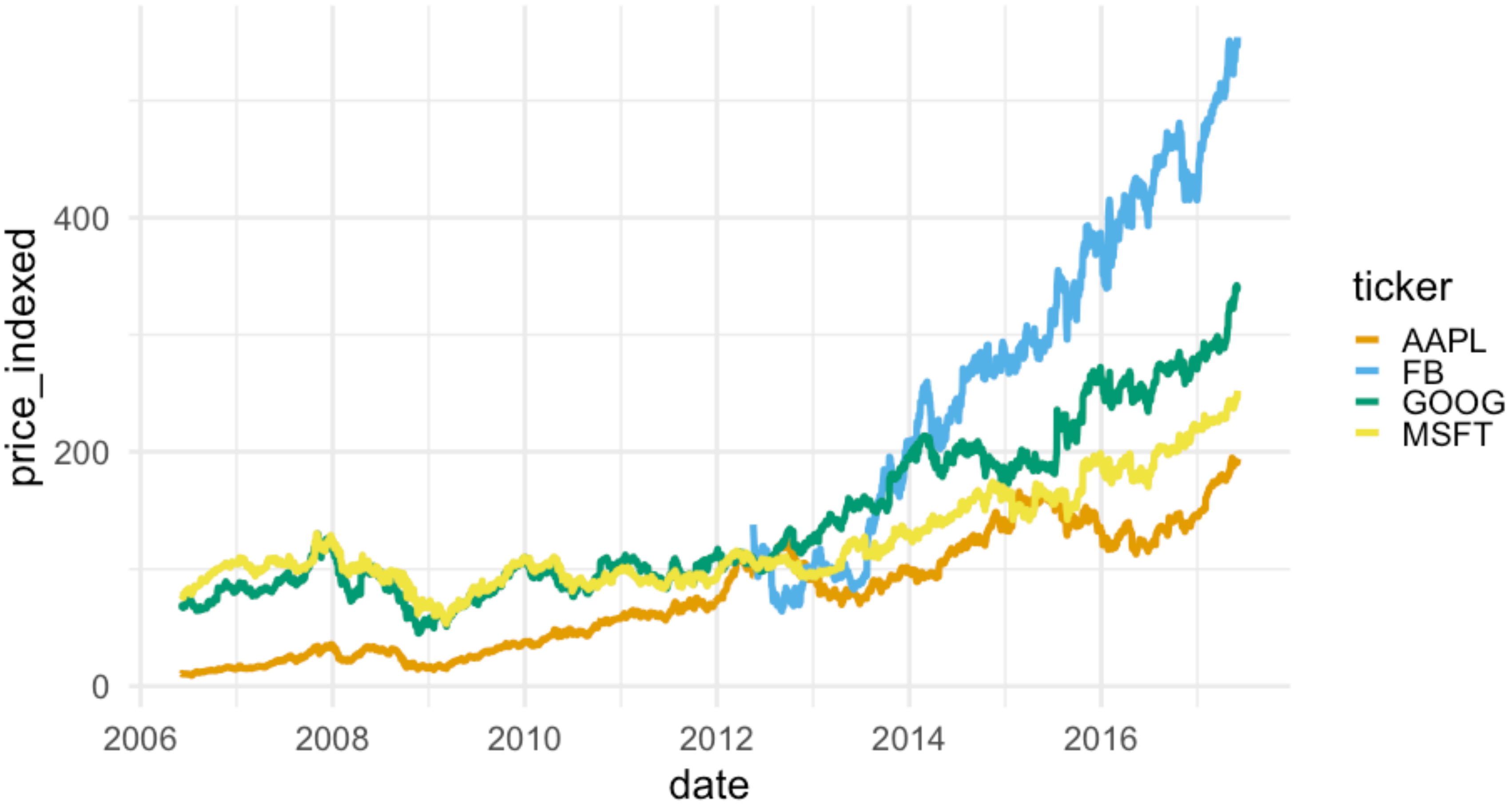
```
remotes::install_github("clauswilke/dviz.supp")
head(tech_stocks)
```

```
## # A tibble: 6 x 6
## # Groups:   ticker [1]
##   company  ticker date        price index_price price_indexed
##   <chr>    <chr>  <date>      <dbl>       <dbl>         <dbl>
## 1 Alphabet GOOG   2017-06-02 975.6        285.2      342.0757
## 2 Alphabet GOOG   2017-06-01 966.95       285.2      339.0428
## 3 Alphabet GOOG   2017-05-31 964.86       285.2      338.3100
## 4 Alphabet GOOG   2017-05-30 975.88       285.2      342.1739
## 5 Alphabet GOOG   2017-05-26 971.47       285.2      340.6276
## 6 Alphabet GOOG   2017-05-25 969.54       285.2      339.9509
```

```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line()
```

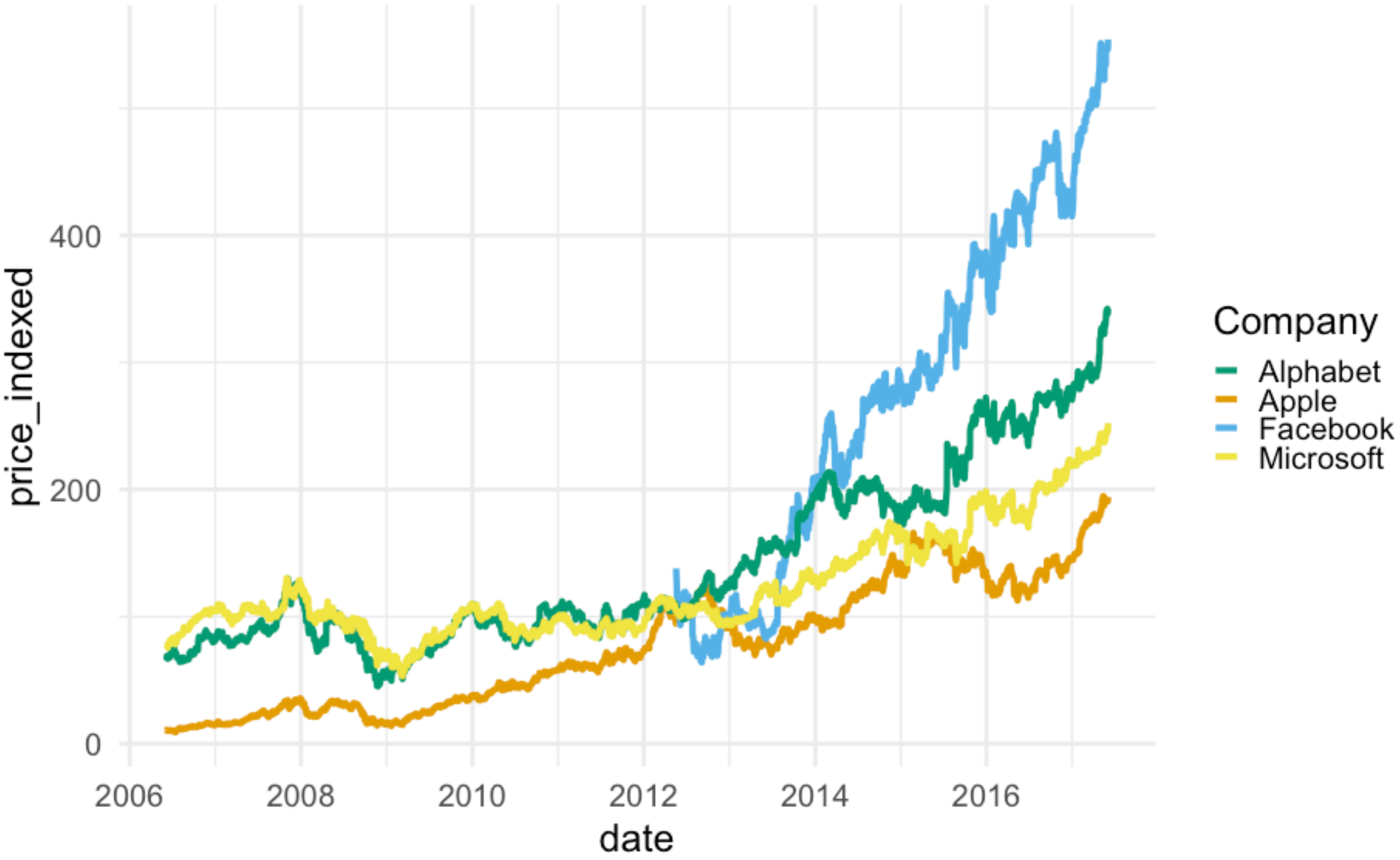


```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto()
```



```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(name = "Company",
                       breaks = c("GOOG", "AAPL", "FB", "MSFT"),
                       labels = c("Alphabet", "Apple", "Facebook", "Microsof
```

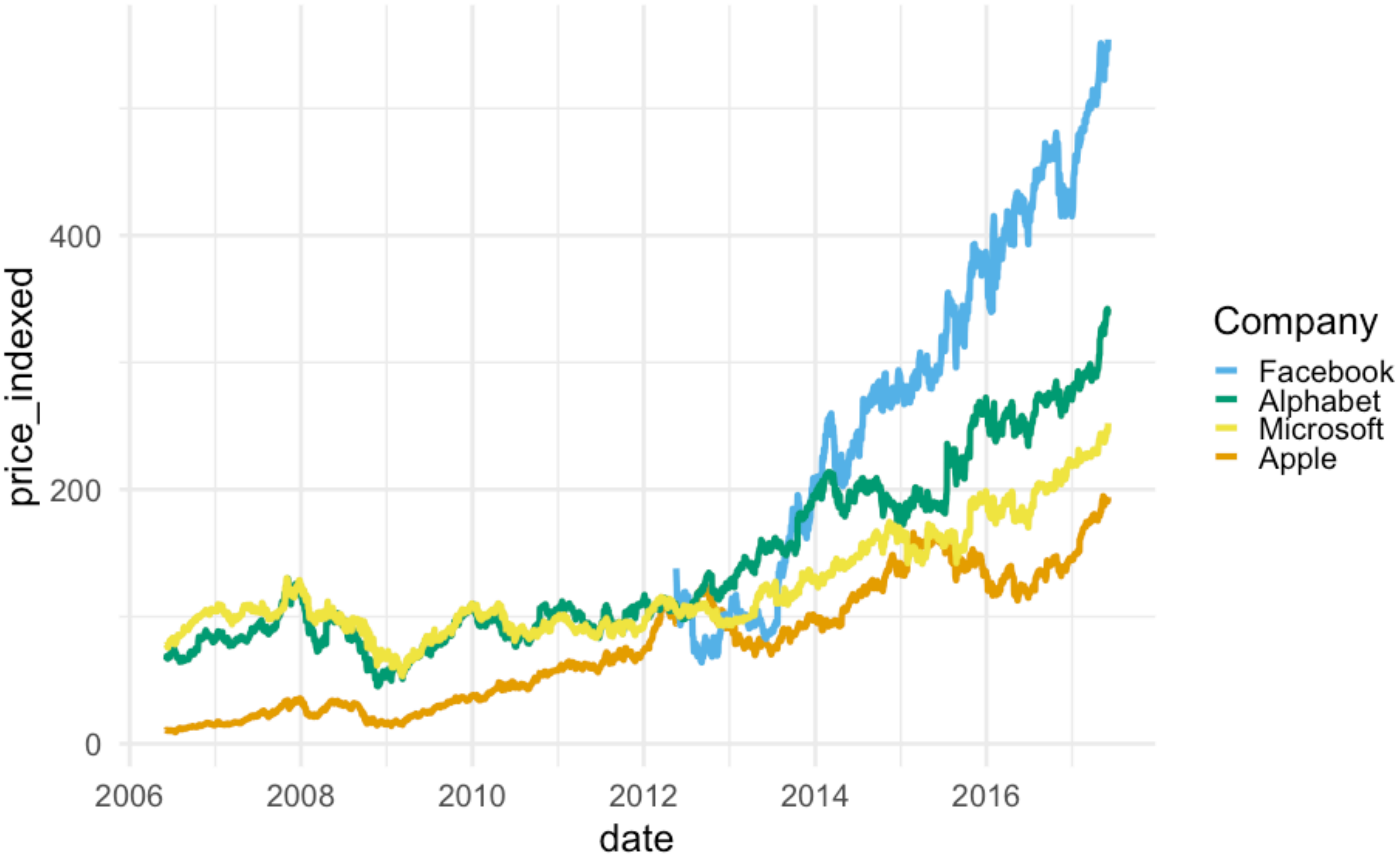
Bad



price_indexed
400
200
0
2006
2008
2010
2012
2014
2016
date
Company
Alphabet
Apple
Facebook
Microsoft

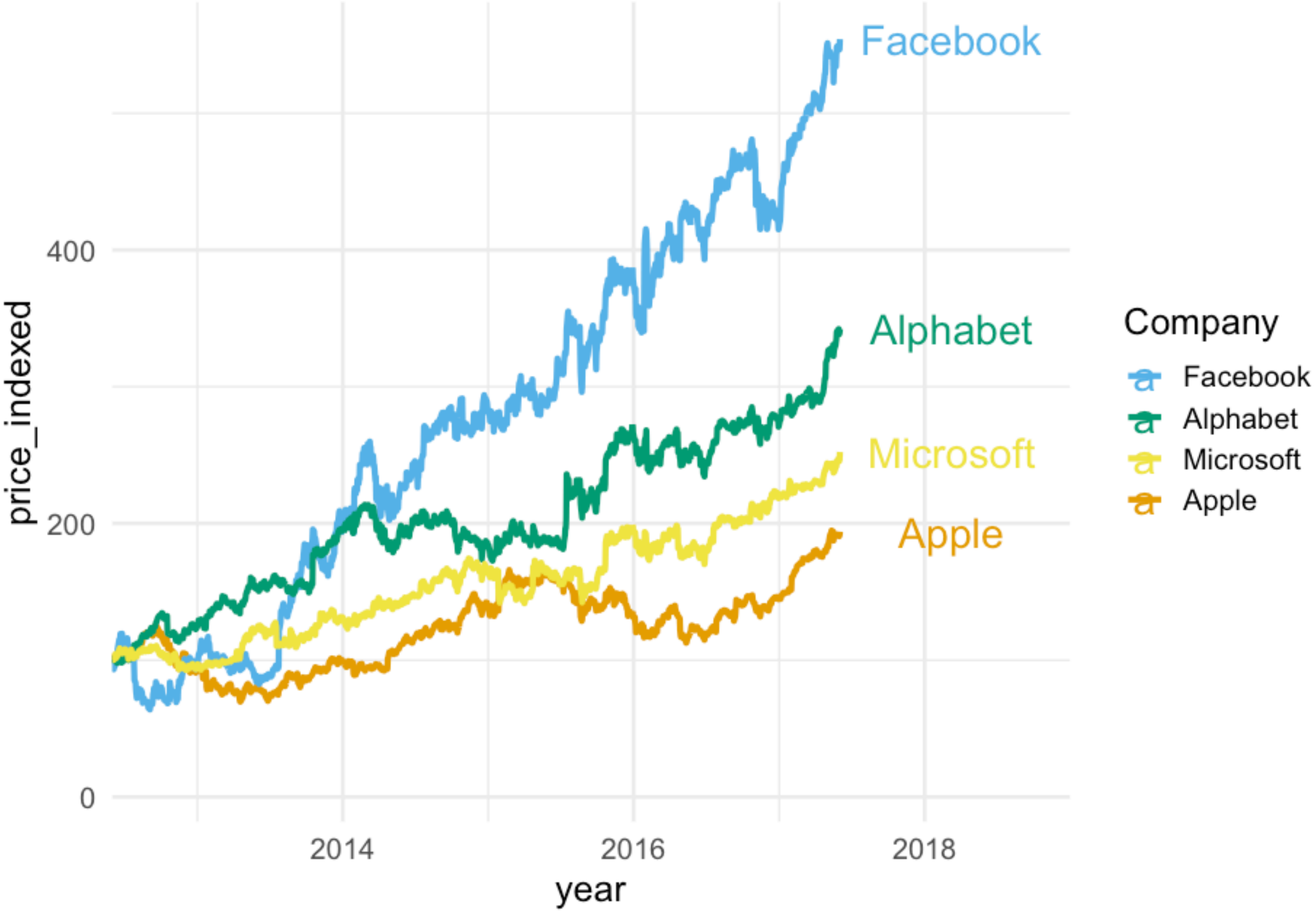
```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(name = "Company",
                       breaks = c("FB", "GOOG", "MSFT", "AAPL"),
                       labels = c("Facebook", "Alphabet", "Microsoft", "Appl
```

Good

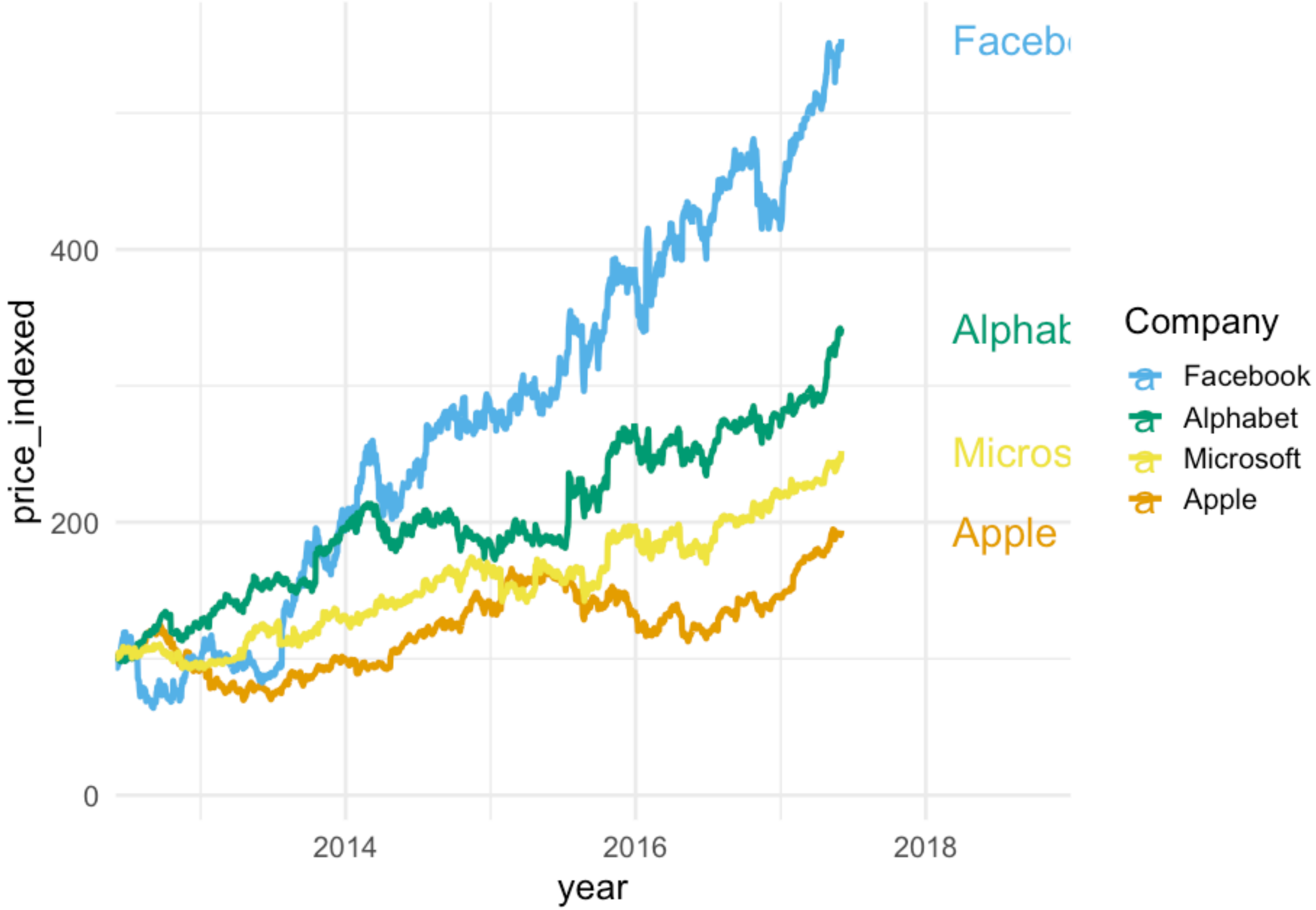


price_indexed
400
200
0
2006
2008
2010
2012
2014
2016
date
Company
Facebook
Alphabet
Microsoft
Apple

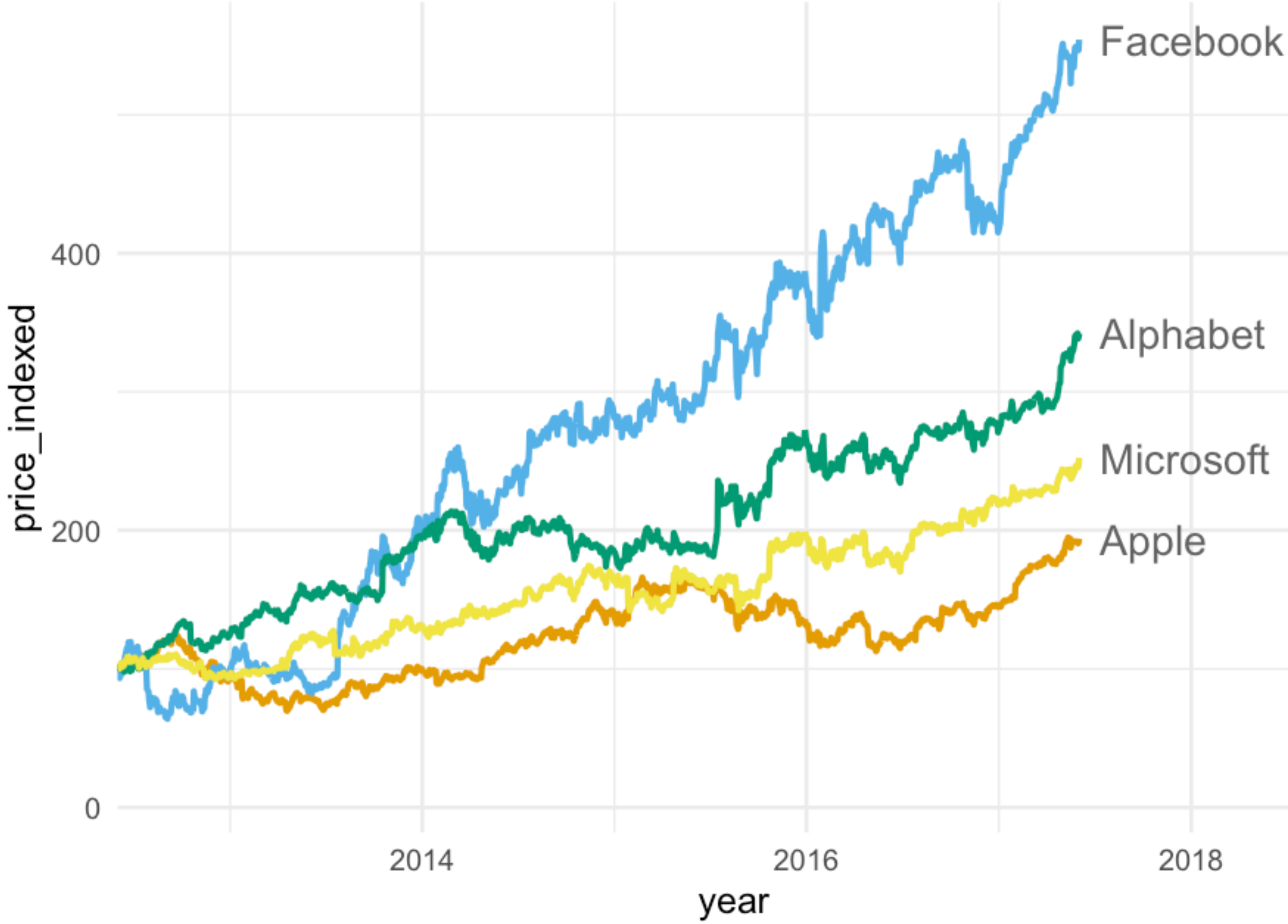
```r
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(name = "Company",
                       breaks = c("FB", "GOOG", "MSFT", "AAPL"),
                       labels = c("Facebook", "Alphabet", "Microsoft", "Appl
  scale_x_date(name = "year",
               limits = c(ymd("2012-06-01"), ymd("2018-12-31")),
               expand = c(0,0)) +
  geom_text(data = filter(tech_stocks, date == "2017-06-02"),
            aes(y = price_indexed, label = company),
            nudge_x = 280)
```


Facebook
Alphabet
Microsoft
Apple
400
200
0
price_indexed
2014
2016
2018
year
Company
Facebook
Alphabet
Microsoft
Apple

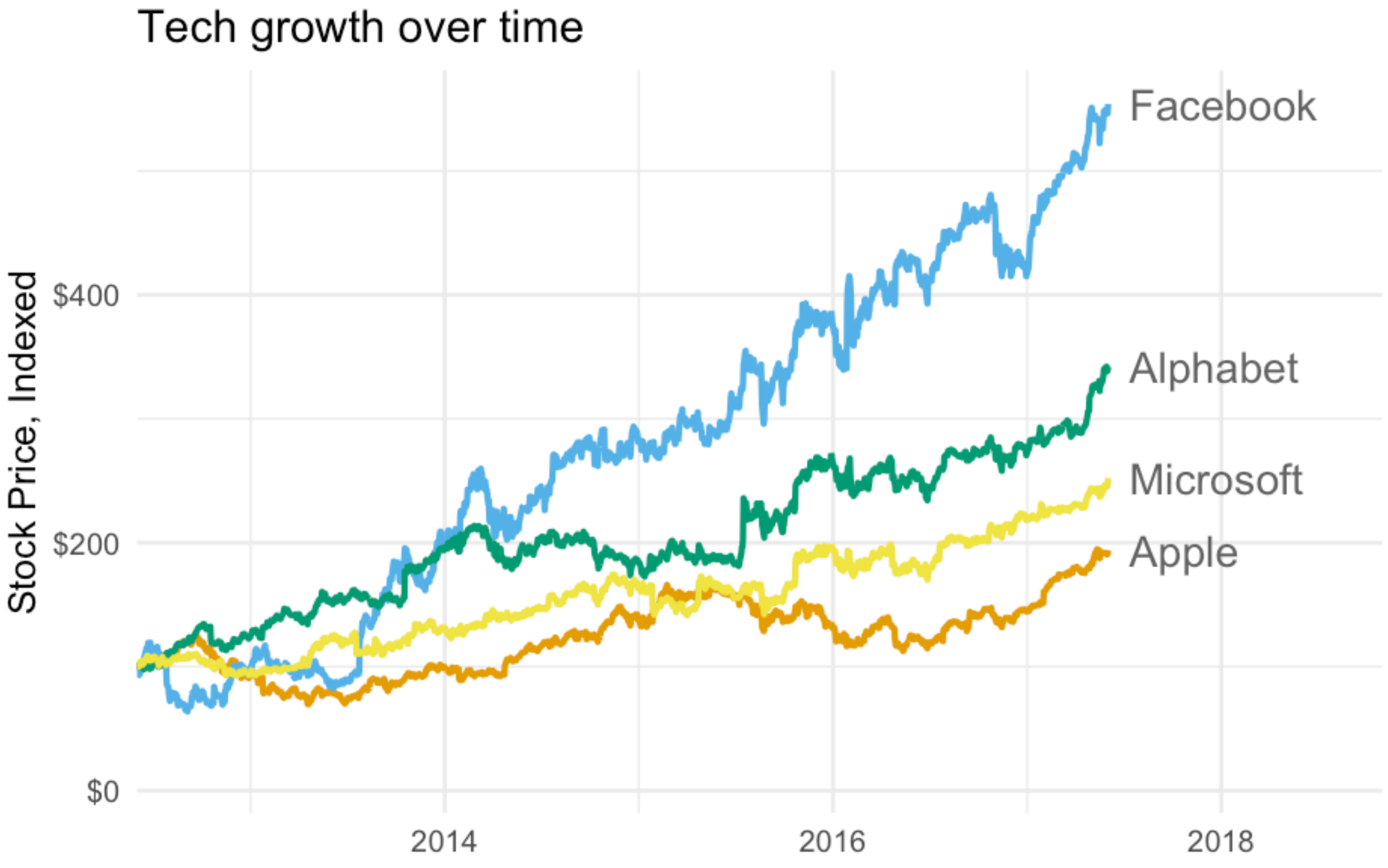
```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(name = "Company",
                       breaks = c("FB", "GOOG", "MSFT", "AAPL"),
                       labels = c("Facebook", "Alphabet", "Microsoft", "Appl
  scale_x_date(name = "year",
               limits = c(ymd("2012-06-01"), ymd("2018-12-31")),
               expand = c(0,0)) +
  geom_text(data = filter(tech_stocks, date == "2017-06-02"),
            aes(y = price_indexed, label = company),
            nudge_x = 280,
            hjust = 0)
```


Faceb
Alphab
Micros
Apple
400
200
0
price_indexed
2014
2016
2018
year
Company
Facebook
Alphabet
Microsoft
Apple

```r
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(name = "Company",
                       breaks = c("FB", "GOOG", "MSFT", "AAPL"),
                       labels = c("Facebook", "Alphabet", "Microsoft", "Appl
  scale_x_date(name = "year",
               limits = c(ymd("2012-06-01"), ymd("2018-10-31")),
               expand = c(0,0)) +
  geom_text(data = filter(tech_stocks, date == "2017-06-02"),
            aes(y = price_indexed, label = company),
            color = "gray40",
            nudge_x = 40,
            hjust = 0) +
  guides(color = "none")
```


Facebook
Alphabet
Microsoft
Apple
price_indexed
400
200
0
2014
2016
2018
year

```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(name = "Company",
                       breaks = c("FB", "GOOG", "MSFT", "AAPL"),
                       labels = c("Facebook", "Alphabet", "Microsoft", "Appl
  scale_x_date(name = "",
               limits = c(ymd("2012-06-01"), ymd("2018-10-31")),
               expand = c(0,0)) +
  scale_y_continuous(name = "Stock Price, Indexed",
                     labels = scales::dollar) +
  geom_text(data = filter(tech_stocks, date == "2017-06-02"),
            aes(y = price_indexed, label = company),
            color = "gray40",
            nudge_x = 40,
            hjust = 0,
            size = 10) +
  guides(color = "none") +
  labs(title = "Tech growth over time",
       caption = "Data from Wilke (2019): Fundamentals of Data Visualizatio
```



Data from Wilke (2019): Fundamentals of Data Visualization

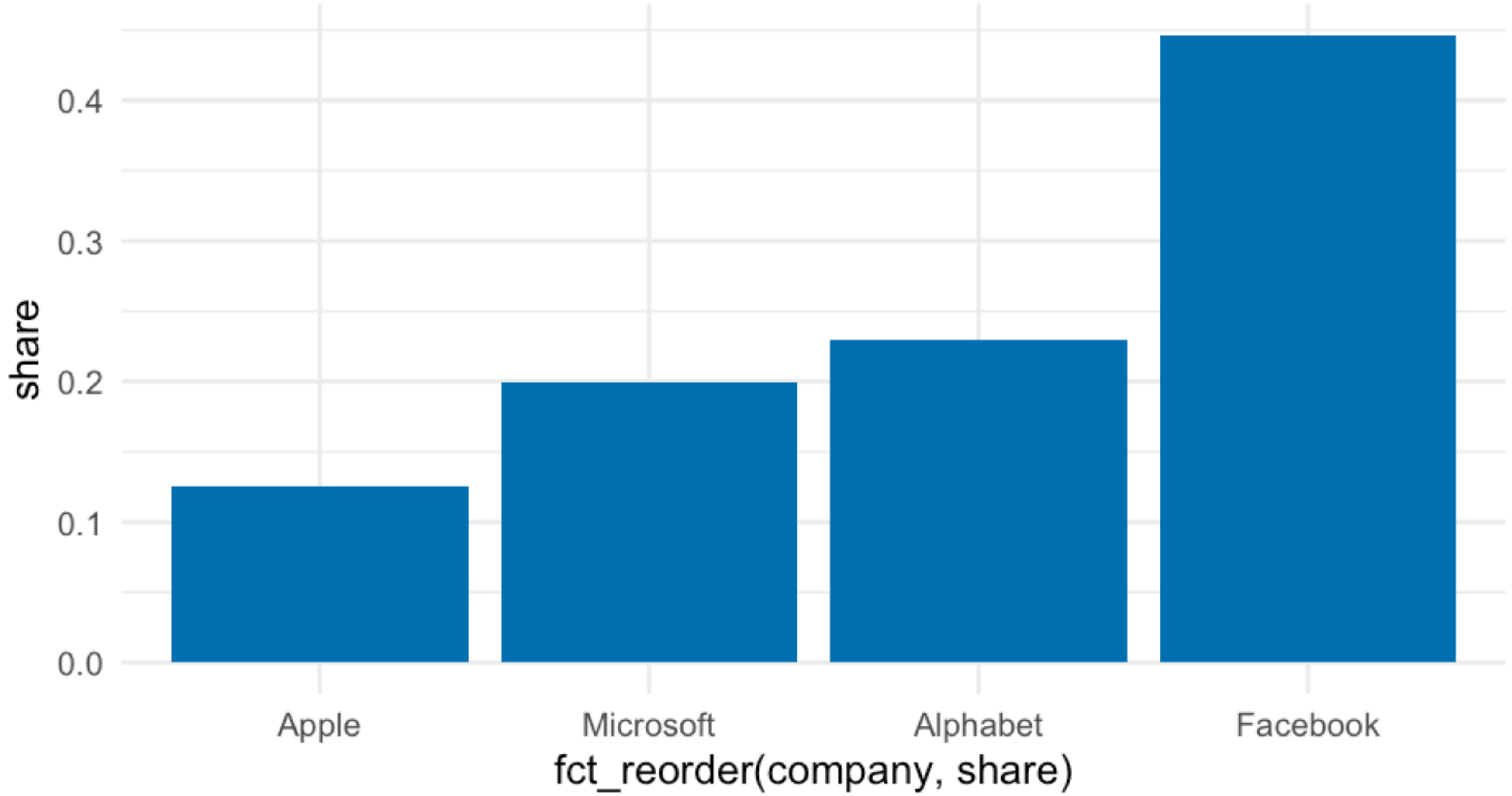
Labeling bars

```
avs <- tech_stocks %>%
  group_by(company) %>%
  summarize(stock_av = mean(price_indexed)) %>%
  ungroup() %>%
  mutate(share = stock_av / sum(stock_av))
avs
```

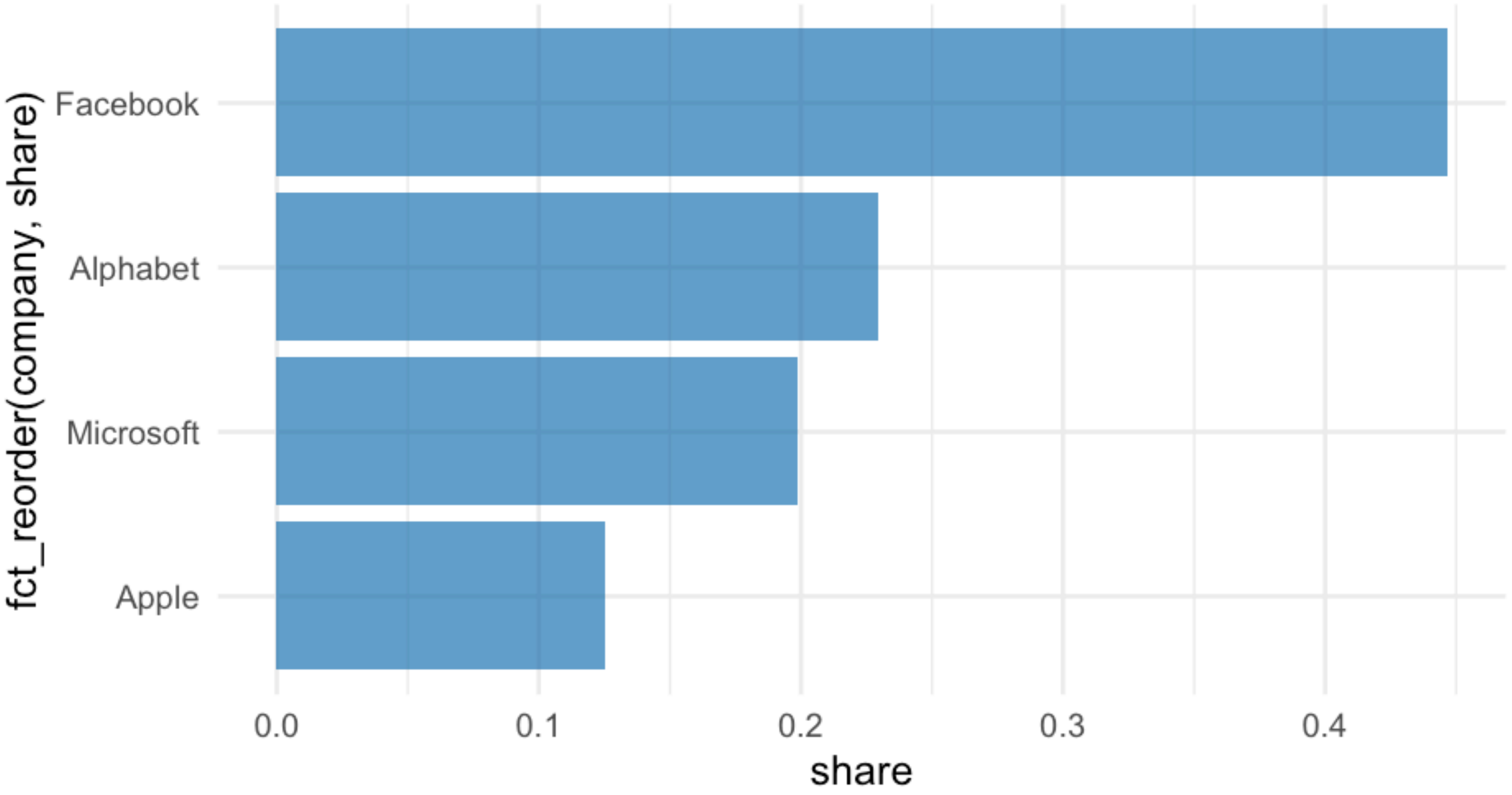
```
## # A tibble: 4 x 3
##   company    stock_av     share
##   <chr>         <dbl>     <dbl>
## 1 Alphabet  141.0205  0.2292441
## 2 Apple      77.08241 0.1253058
## 3 Facebook  274.7427  0.4466240
## 4 Microsoft 122.3088  0.1988261
```

Bar plot

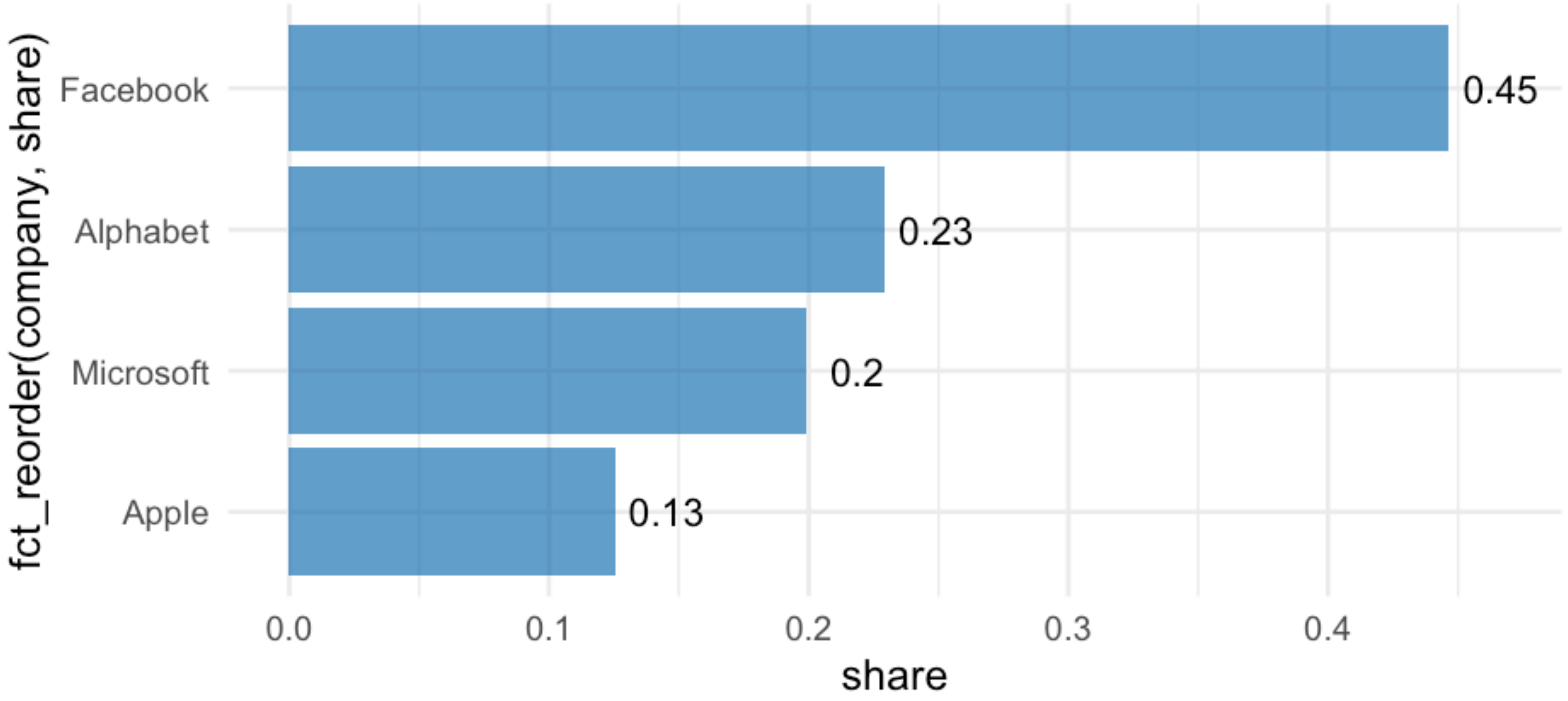
```
ggplot(avs, aes(fct_reorder(company, share), share)) +
  geom_col(fill = "#0072B2")
```



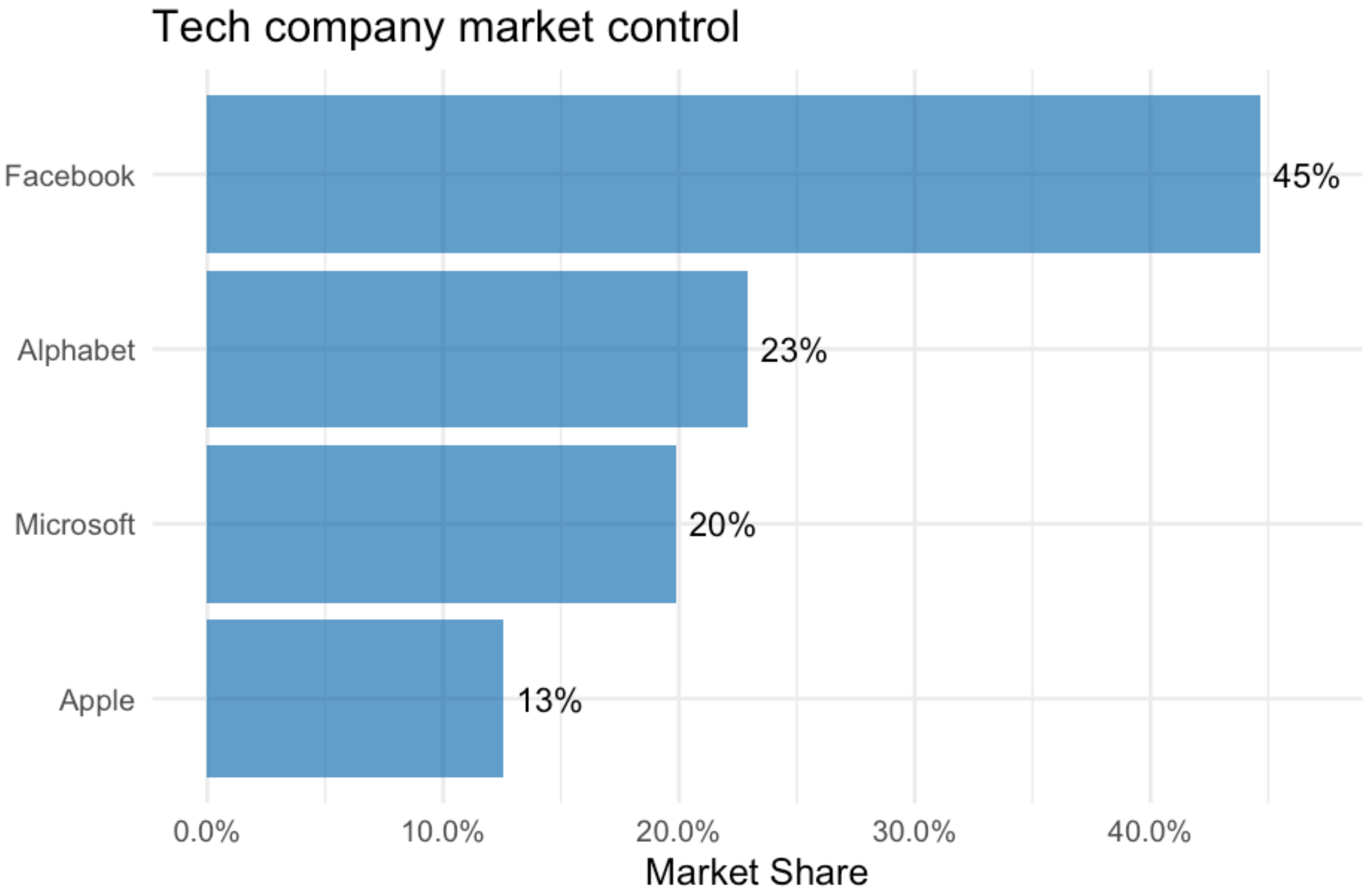
```
ggplot(avs, aes(fct_reorder(company, share), share)) +
  geom_col(fill = "#0072B2",
           alpha = 0.7) +
  coord_flip()
```



```
ggplot(avs, aes(fct_reorder(company, share), share)) +
  geom_col(fill = "#0072B2",
           alpha = 0.7) +
  geom_text(aes(company, share, label = round(share, 2)),
            nudge_y = 0.02,
            size = 8) +
  coord_flip()
```



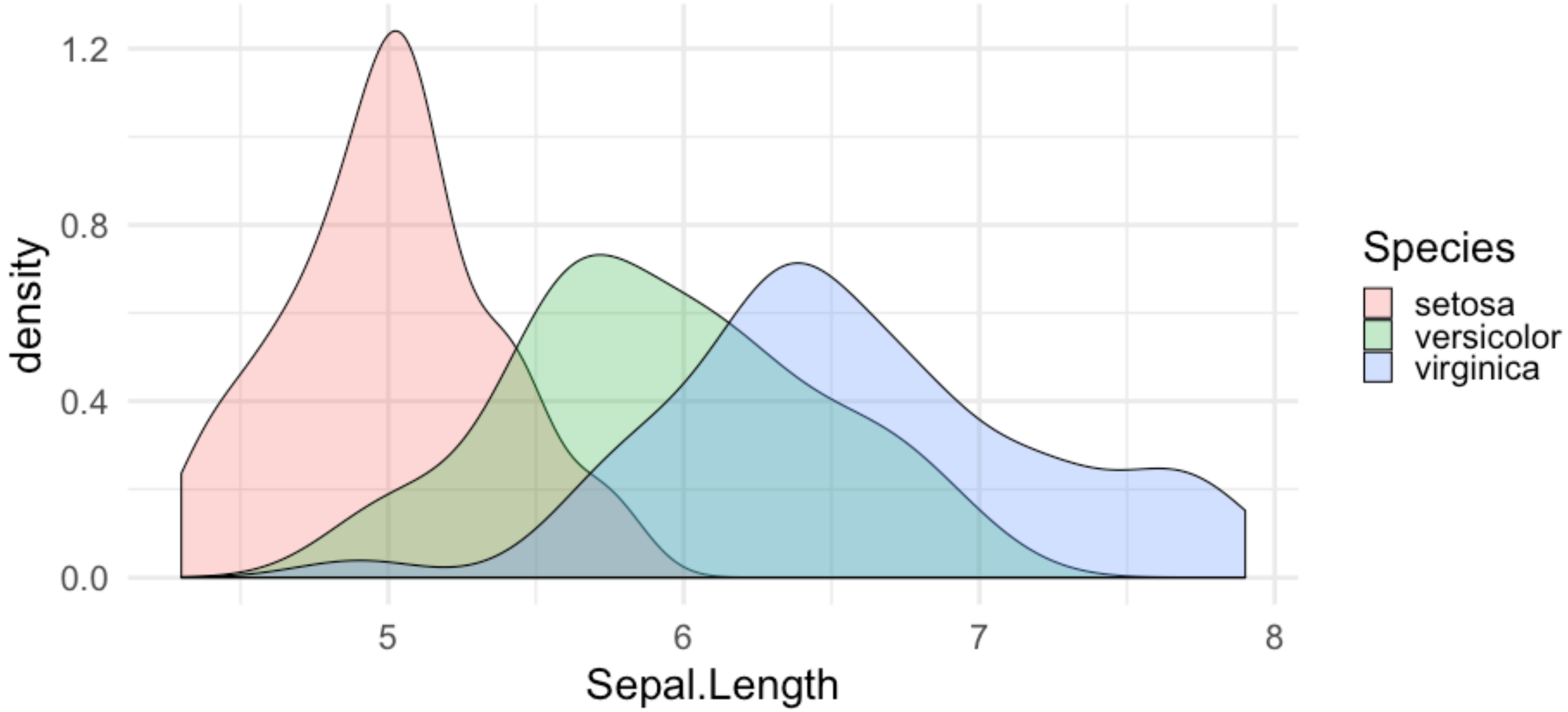
```r
ggplot(avs, aes(fct_reorder(company, share), share)) +
  geom_col(fill = "#0072B2",
           alpha = 0.7) +
  geom_text(aes(company, share, label = paste0(round(share*100), "%")),
            nudge_y = 0.02,
            size = 8) +
  coord_flip() +
  scale_y_continuous("Market Share", labels = scales::percent) +
  labs(x = NULL,
       title = "Tech company market control",
       caption = "Data from Clause Wilke Book: Fundamentals of Data Visualiz
```



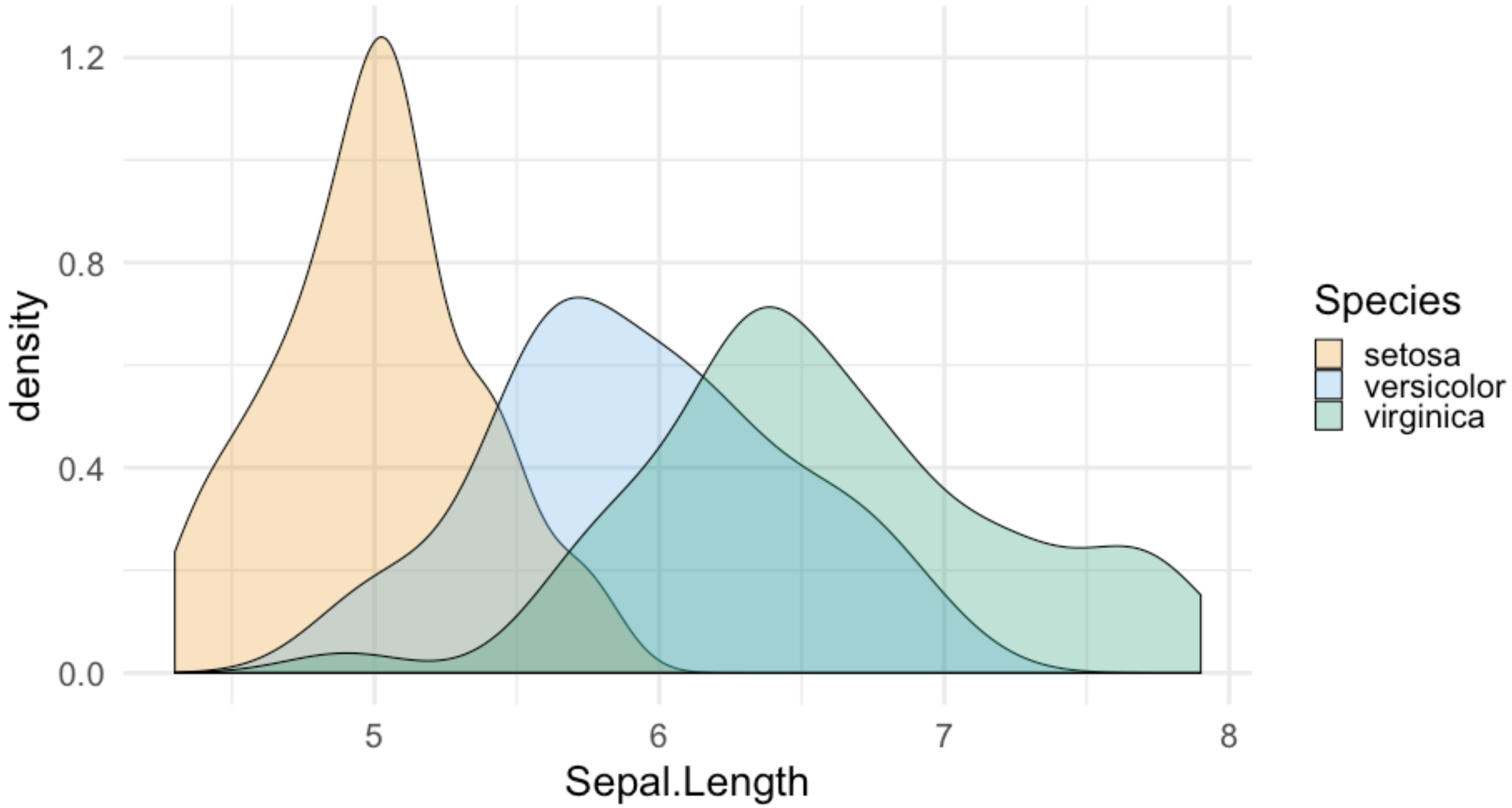
Data from Clause Wilke Book: Fundamentals of Data Visualizations

Distributions

```
ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.3)
```



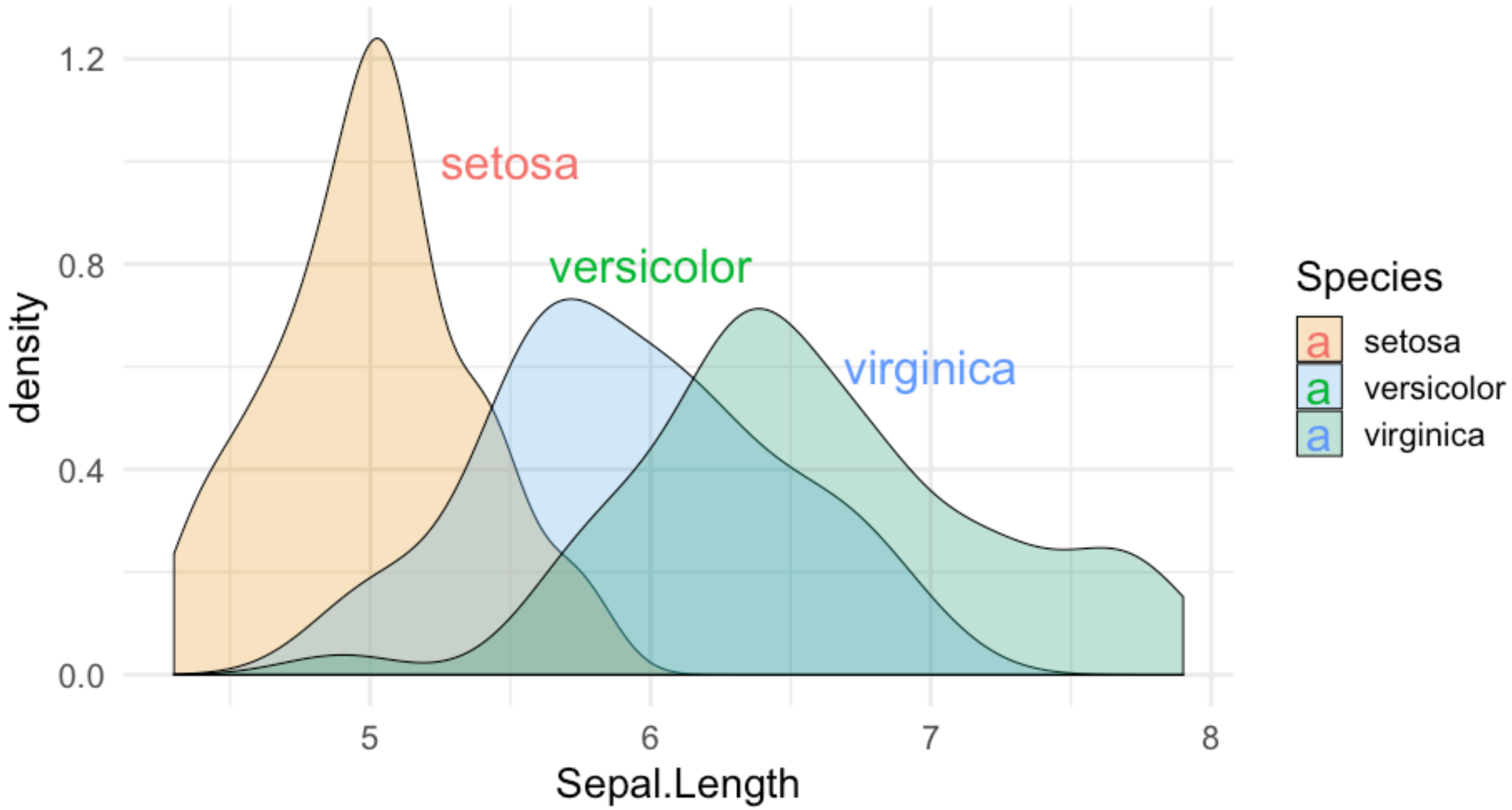
```r
ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.3) +
  scale_fill_OkabeIto()
```



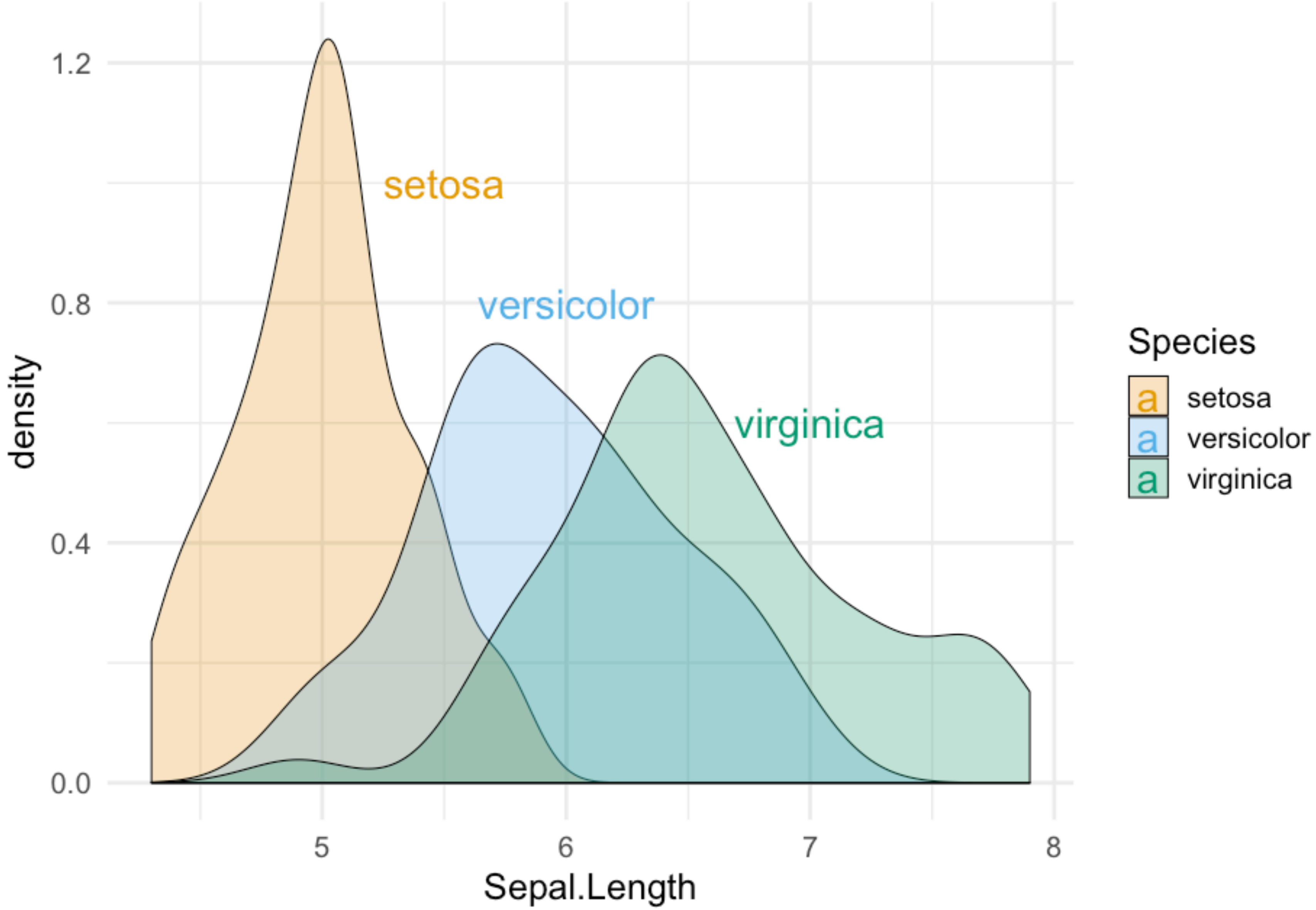
Labeling

One method

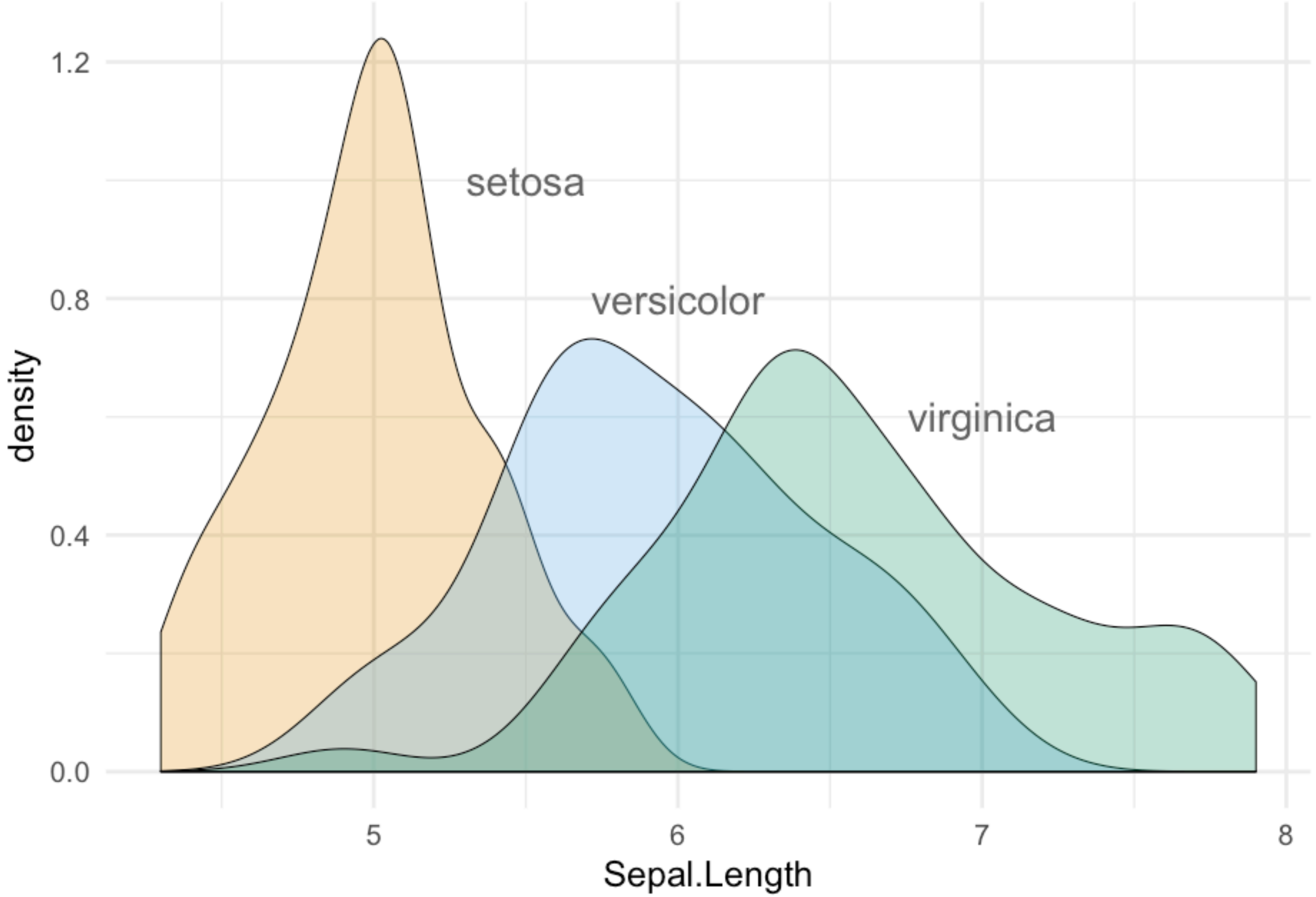
```r
label_locs <- tibble(Sepal.Length = c(5.5, 6, 7),
                     density = c(1, 0.8, 0.6),
                     Species = c("setosa", "versicolor", "virginica"))

ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.3) +
  scale_fill_OkabeIto() +
  geom_text(aes(label = Species, y = density, color = Species),
            data = label_locs)
```


1.2
0.8
0.4
0.0
density
5
6
7
8
Sepal.Length
setosa
versicolor
virginica
Species
a setosa
a versicolor
a virginica

```r
ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.3) +
  scale_fill_OkabeIto() +
  scale_color_OkabeIto() +
  geom_text(aes(label = Species, y = density, color = Species),
            data = label_locs) +
  guides(color = "none",
         fill = "none")
```


1.2
0.8
0.4
0.0
density
5
6
7
8
Sepal.Length
setosa
versicolor
virginica
Species
a setosa
a versicolor
a virginica

```r
label_locs <- tibble(Sepal.Length = c(5.5, 6, 7),
                     density = c(1, 0.8, 0.6),
                     Species = c("setosa", "versicolor", "virginica"))

ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.3) +
  scale_fill_OkabeIto() +
  scale_color_OkabeIto() +
  geom_text(aes(label = Species, y = density),
            color = "gray40",
            data = label_locs) +
  guides(fill = "none")
```

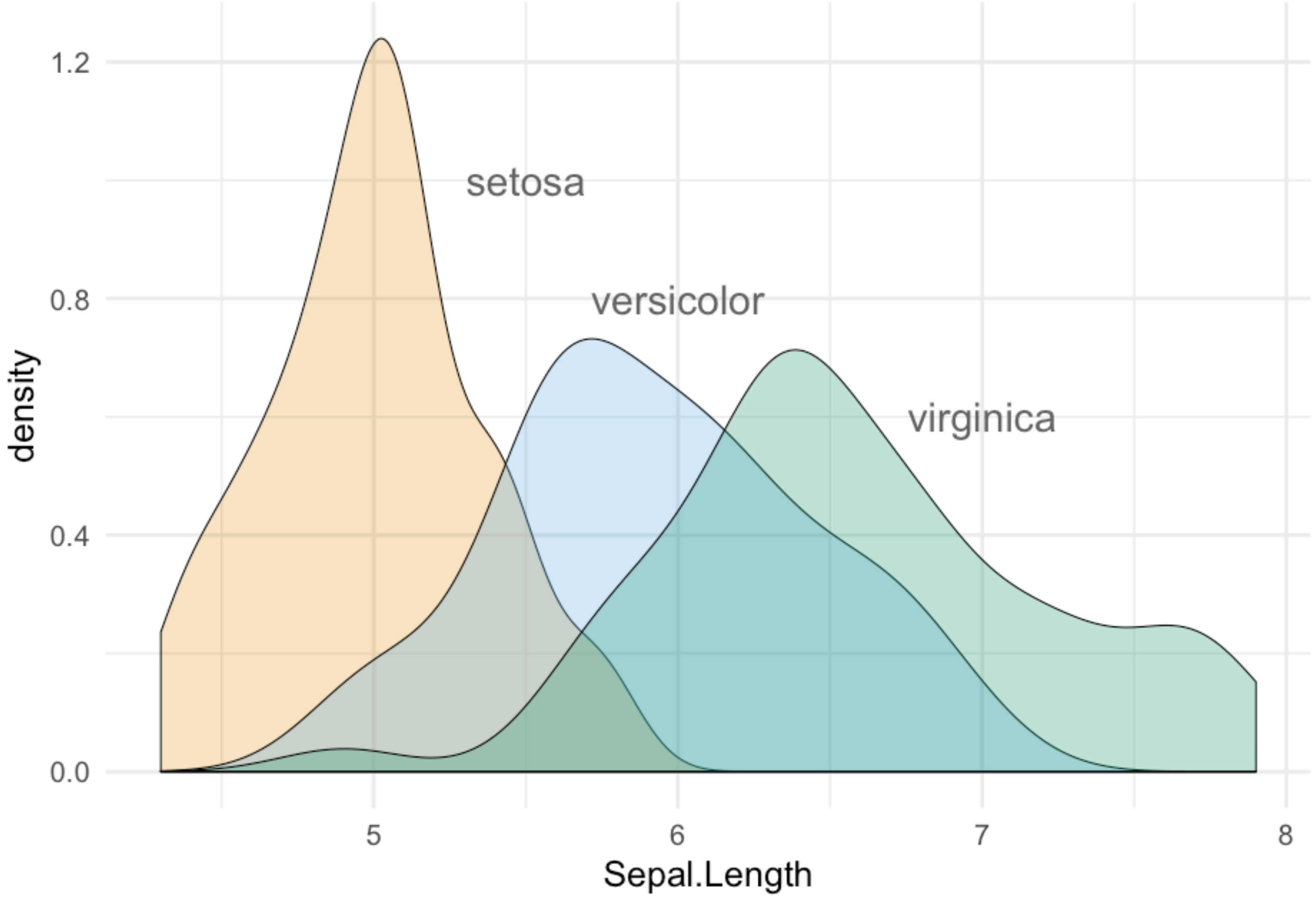

1.2
0.8
0.4
0.0
density
setosa
versicolor
virginica
5
6
7
8
Sepal.Length

Other options

- Rather than using a new data frame, you could use multiple calls to `annotate`.
- One is not necessarily better than the other, but I prefer the data frame method
- Keep in mind you can **always** use multiple data sources within a single plot
    - Each layer can have its own data source
    - Common in geographic data in particular

Annotate example

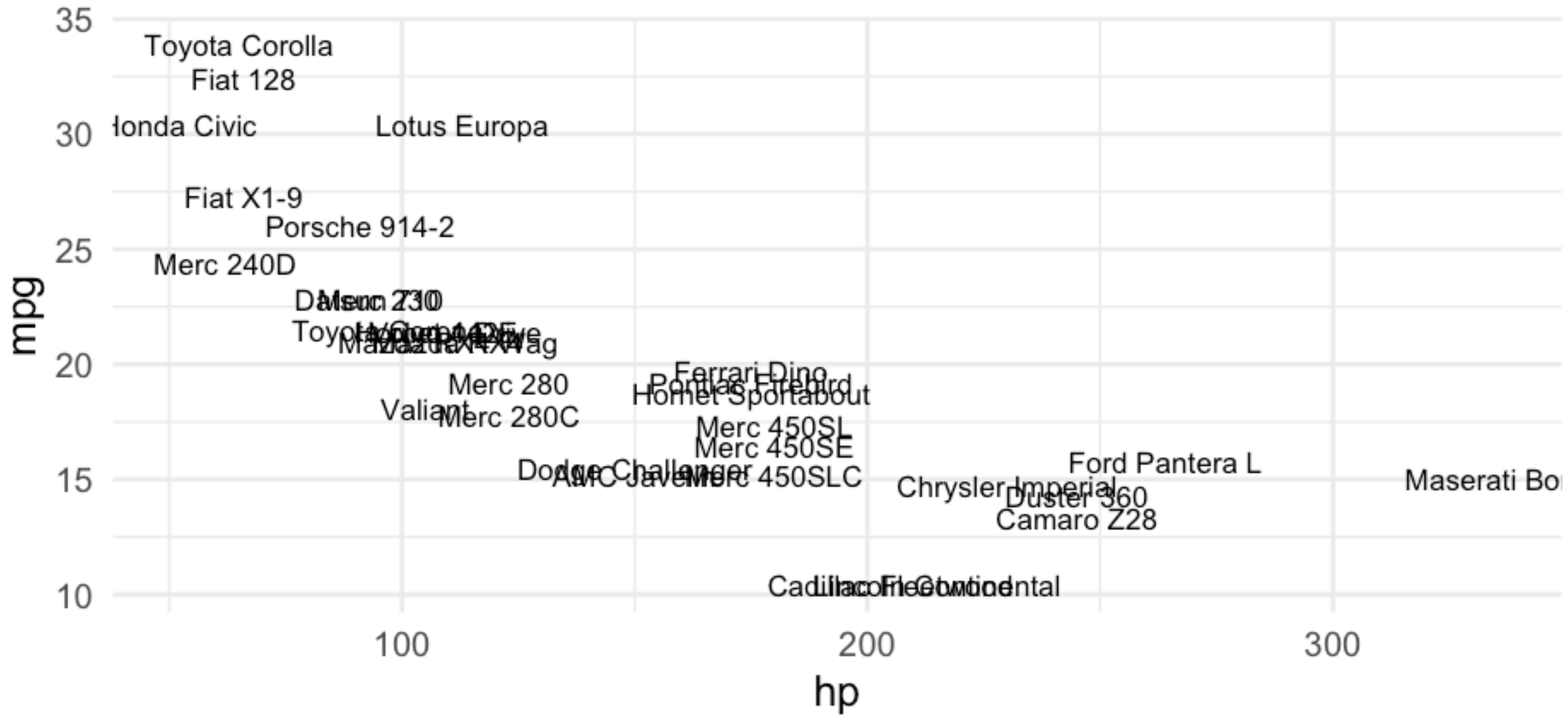
```
ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.3) +
  scale_fill_OkabeIto() +
  scale_color_OkabeIto() +
  annotate("text", label = "setosa", x = 5.5, y = 1, color = "gray40") +
  annotate("text", label = "versicolor", x = 6, y = 0.8, color = "gray40") +
  annotate("text", label = "virginica", x = 7, y = 0.6, color = "gray40") +
  guides(fill = "none")
```


1.2
0.8
0.4
0.0
density
setosa
versicolor
virginica
5
6
7
8
Sepal.Length

ggrepel

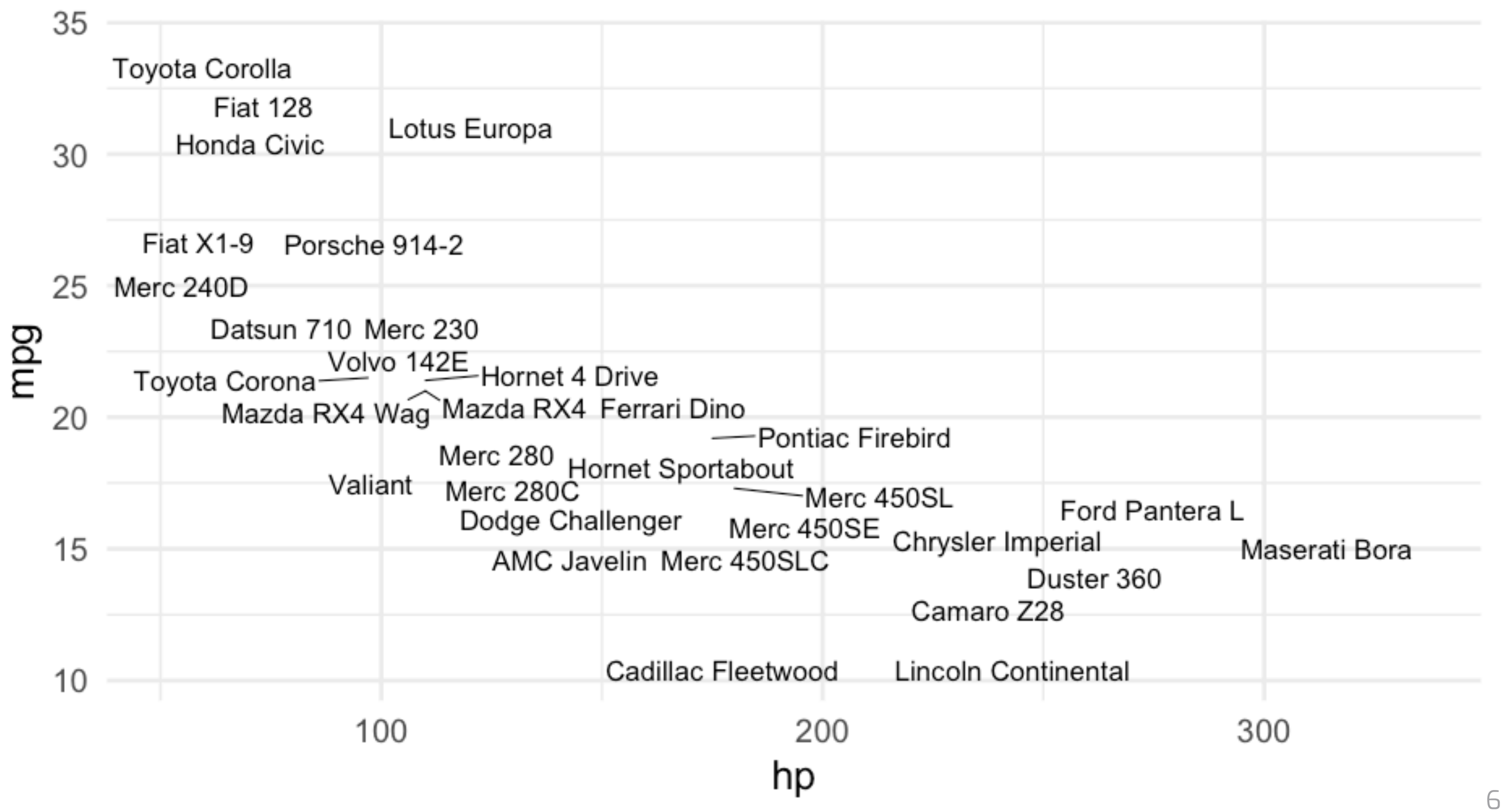
Plot text directly

```
cars <- rownames_to_column(mtcars)

ggplot(cars, aes(hp, mpg)) +
  geom_text(aes(label = rowname))
```



Repel text

```
library(ggrepel)
ggplot(cars, aes(hp, mpg)) +
  geom_text_repel(aes(label = rowname))
```



Slightly better

```
ggplot(cars, aes(hp, mpg)) +
  geom_point(color = "gray70") +
  geom_text_repel(aes(label = rowname),
                  min.segment.length = 0)
```

Common use cases

- Label some sample data that makes some theoretical sense (we've seen this before)
- Label outliers
- Label points from a specific group (e.g., similar to highlighting - can be used in conjunction)

Some new data

```
remotes::install_github("kjhealy/socviz")
library(socviz)
```

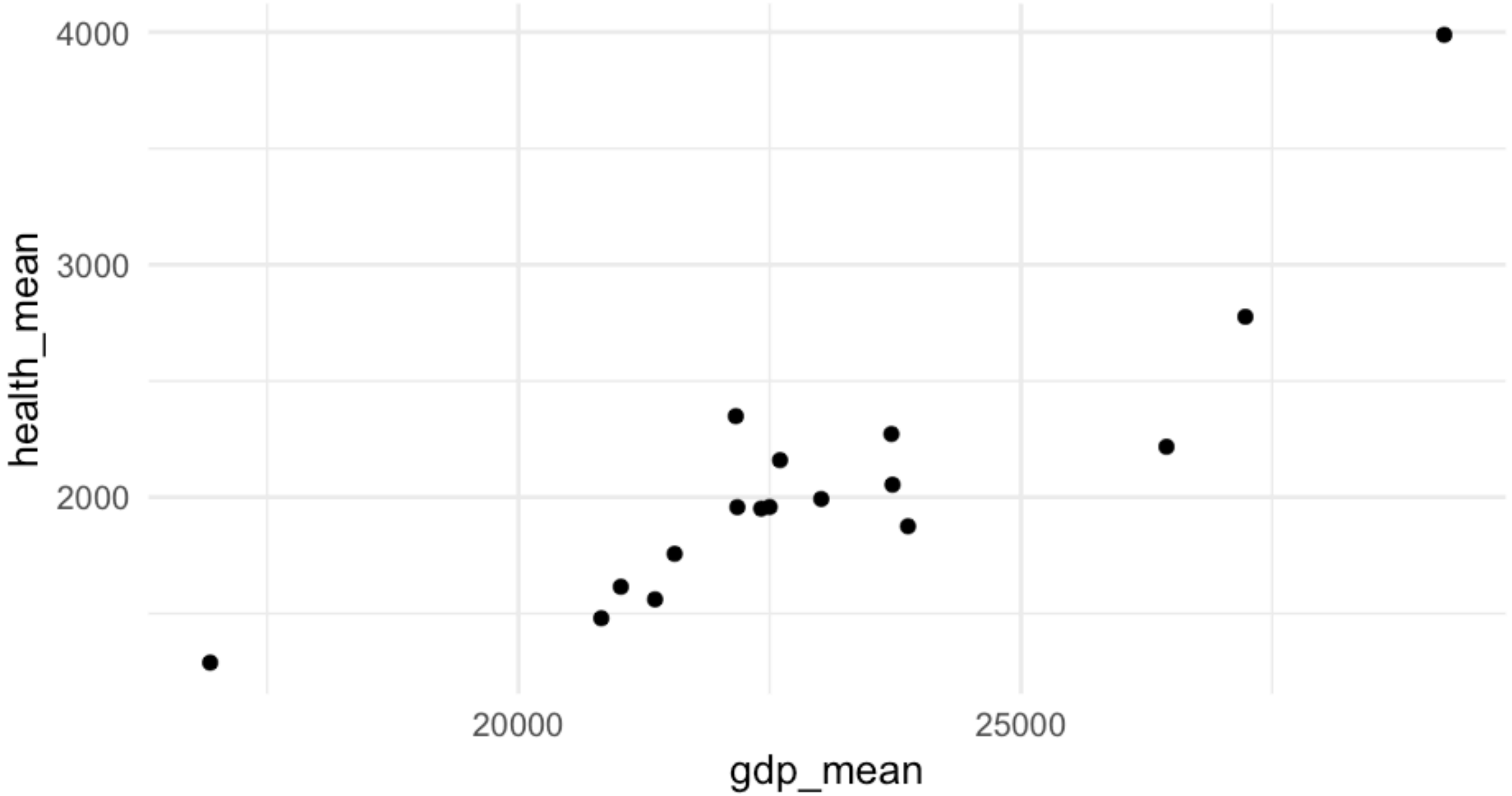
```
by_country <- organdata %>% group_by(consent_law, country) %>%
    summarize(donors_mean= mean(donors, na.rm = TRUE),
              donors_sd = sd(donors, na.rm = TRUE),
              gdp_mean = mean(gdp, na.rm = TRUE),
              health_mean = mean(health, na.rm = TRUE),
              roads_mean = mean(roads, na.rm = TRUE),
              cerebvas_mean = mean(cerebvas, na.rm = TRUE))
```

```
by_country
```

```
## # A tibble: 17 x 8
## # Groups:   consent_law [?]
##   consent_law country donors_mean donors_sd gdp_mean health_mean roads_mean
##   <chr>       <chr>         <dbl>     <dbl>    <dbl>       <dbl>      <dbl>
## 1 Informed    Austra…    10.635   1.142808  22178.54    1957.5    104.8757
## 2 Informed    Canada     13.96667 0.7511607 23711.08    2271.929  109.2601
## 3 Informed    Denmark    13.09167 1.468121  23722.31    2054.071  101.6363
## 4 Informed    Germany    13.04167 0.6111960 22163.23    2348.75   112.7887
## 5 Informed    Ireland    19.79167 2.478437  20824.38    1479.929  117.7742
## 6 Informed    Nether…    13.65833 1.551807  23013.15    1992.786   76.09357
## # … with 11 more rows, and 1 more variable: cerebvas_mean <dbl>
```

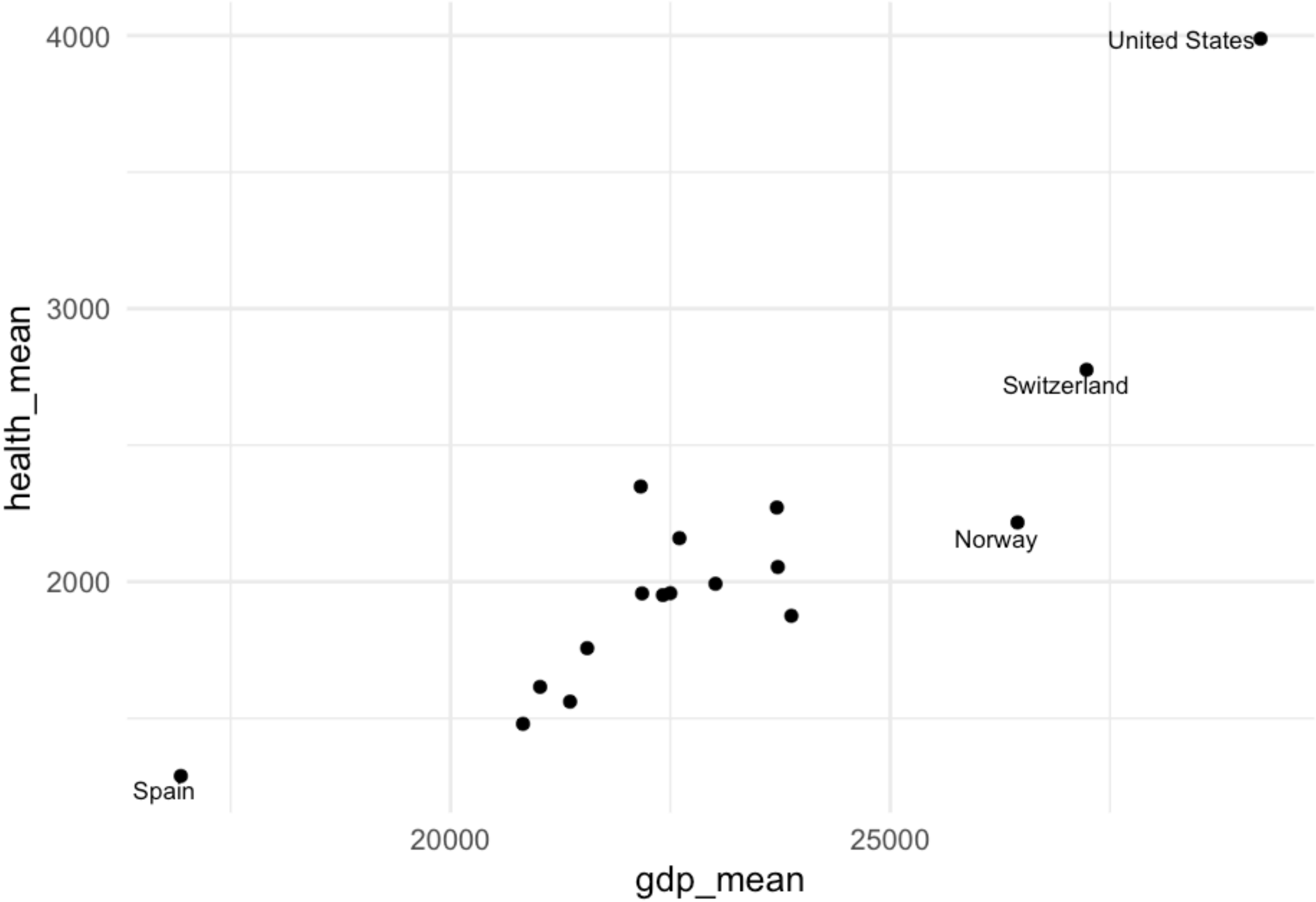
Scatterplot

```
ggplot(by_country, aes(gdp_mean, health_mean)) +
  geom_point()
```



Outliers

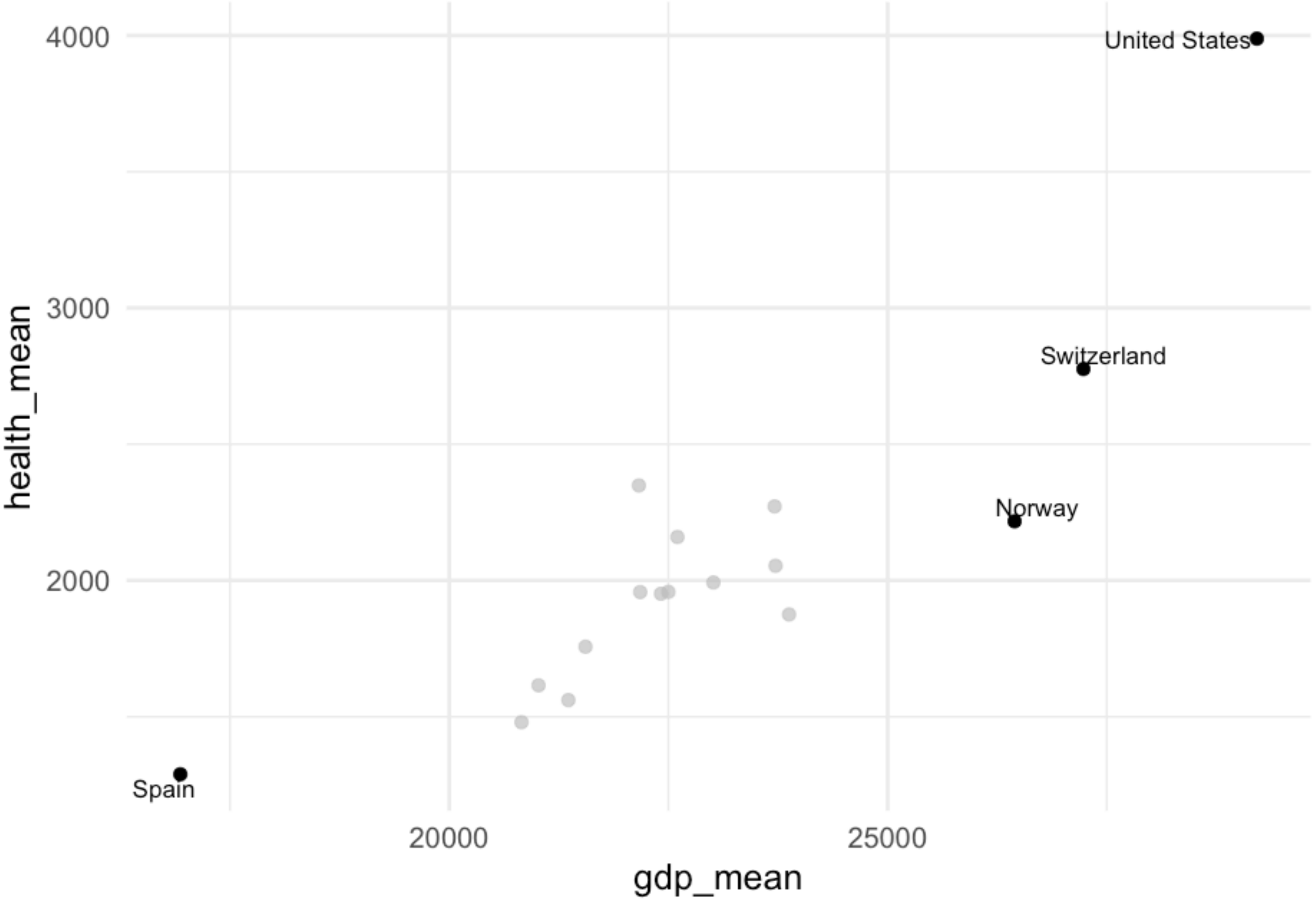
```
ggplot(by_country, aes(gdp_mean, health_mean)) +
  geom_point() +
  geom_text_repel(data = filter(by_country,
                                gdp_mean > 25000 |
                                gdp_mean < 20000),
                  aes(label = country))
```


United States
Switzerland
Norway
Spain
4000
3000
2000
20000
25000
health_mean
gdp_mean

Combine with highlighting

```r
library(gghighlight)
ggplot(by_country, aes(gdp_mean, health_mean)) +
  geom_point() +
  gghighlight(gdp_mean > 25000 | gdp_mean < 20000) +
  geom_text_repel(aes(label = country))
```

- Notice you only have to specify the points to highlight and `geom_text_repel` will then only label those points

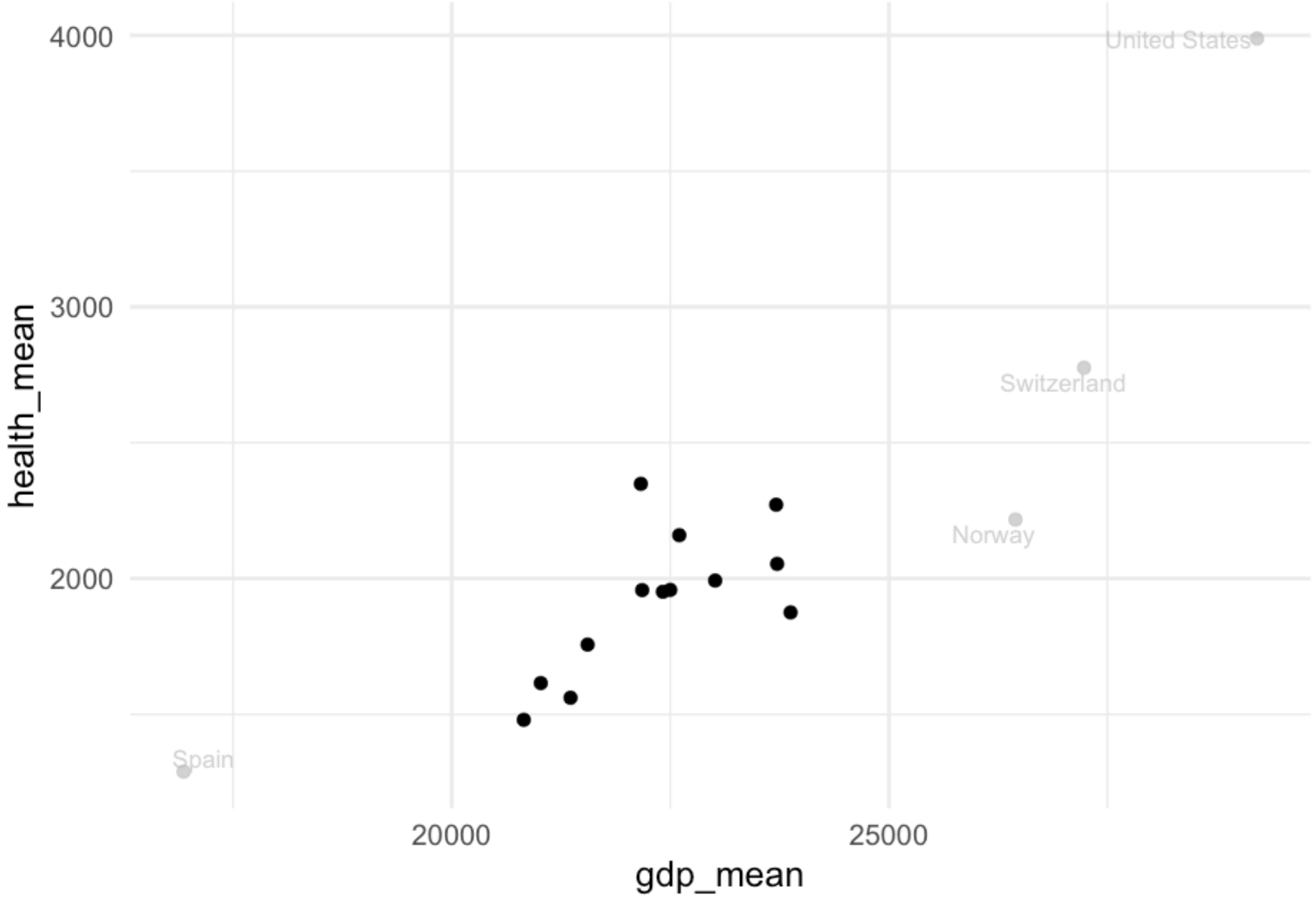

United States
Switzerland
Norway
Spain
4000
3000
2000
health_mean
20000
25000
gdp_mean

Combine with highlighting

Switch to make outliers grayed out and labeled

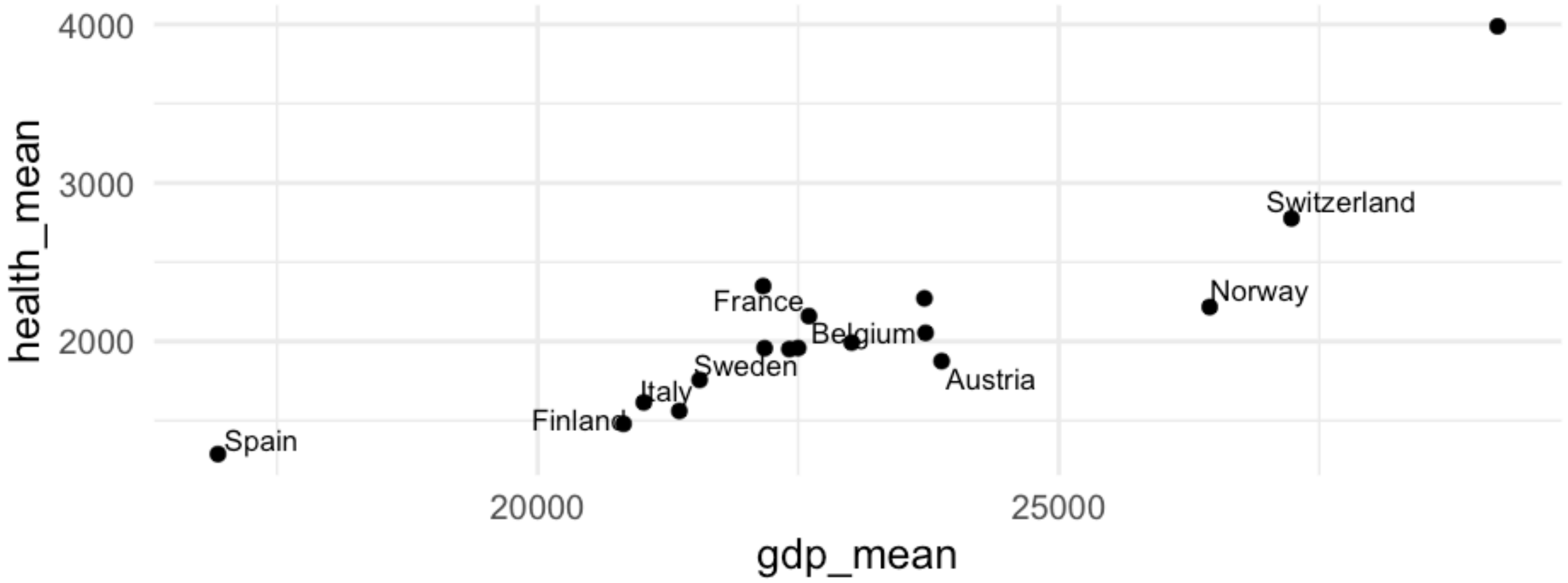
```
ggplot(by_country, aes(gdp_mean, health_mean)) +
  geom_point() +
  gghighlight(gdp_mean > 20000 & gdp_mean < 25000 ) +
  geom_text_repel(data = filter(by_country,
                                gdp_mean > 25000 |
                                gdp_mean < 20000),
                  aes(label = country),
                  color = "#BEBEBEB3")
```

Note I found the exact gray color by looking at the source code. Specifically, it is the output from `ggplot2::alpha("grey", 0.7)`


4000
3000
2000
health_mean
United States
Switzerland
Norway
Spain
20000
25000
gdp_mean

By group

```
ggplot(by_country, aes(gdp_mean, health_mean)) +
  geom_point() +
  geom_text_repel(data = filter(by_country, consent_law == "Presumed"),
                  aes(label = country))
```

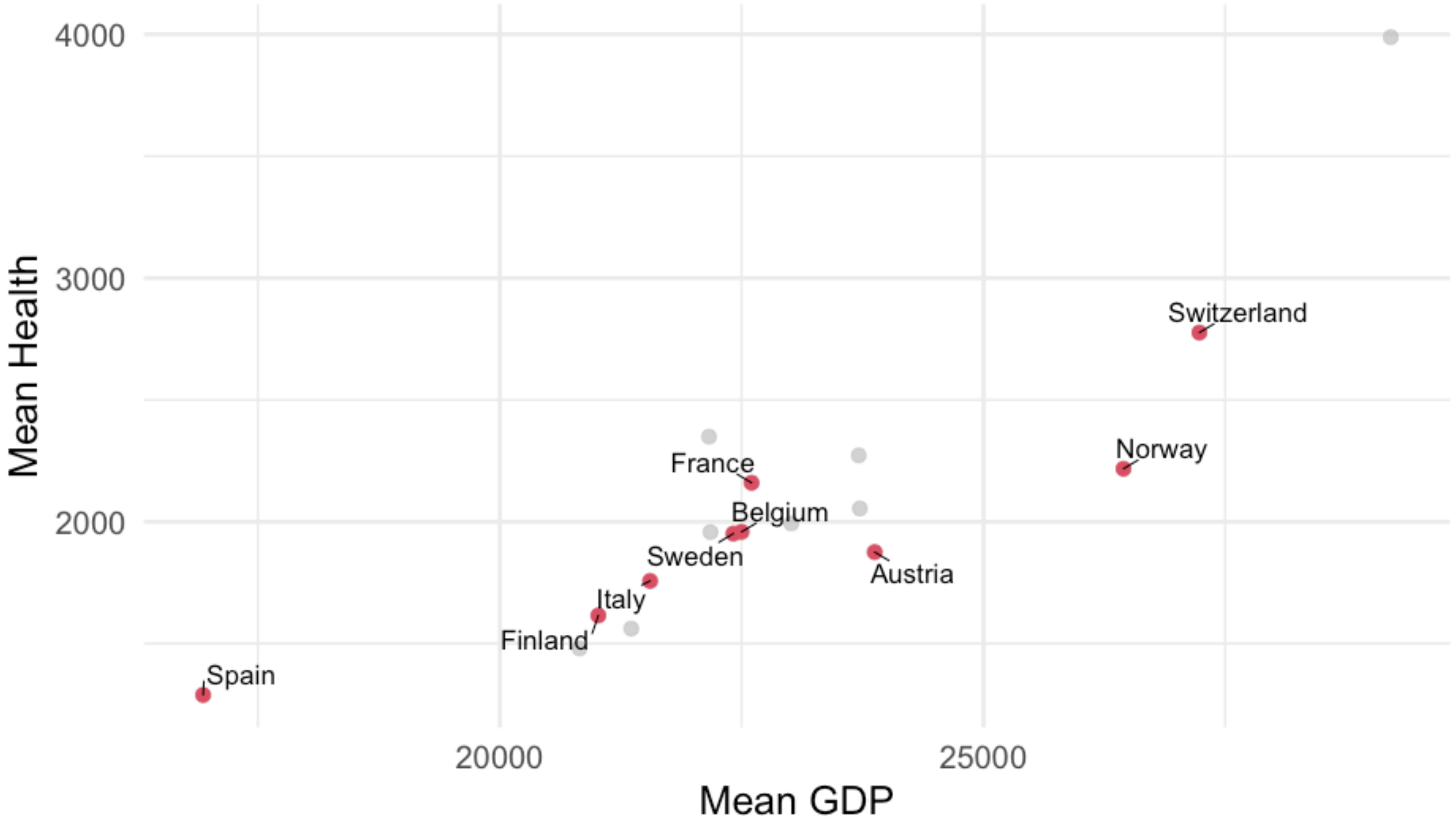


By group

```
ggplot(by_country, aes(gdp_mean, health_mean)) +
  geom_point(color = "#DC5265") +
  gghighlight(consent_law == "Presumed") +
  geom_text_repel(aes(label = country),
                  min.segment.length = 0,
                  box.padding = 0.5) +
  labs(title = "GDP and Health",
       subtitle = "Countries with a presumed organ donation consent are hi
       caption = "Data from the General Social Science Survey, Distributed
       x = "Mean GDP",
       y = "Mean Health")
```

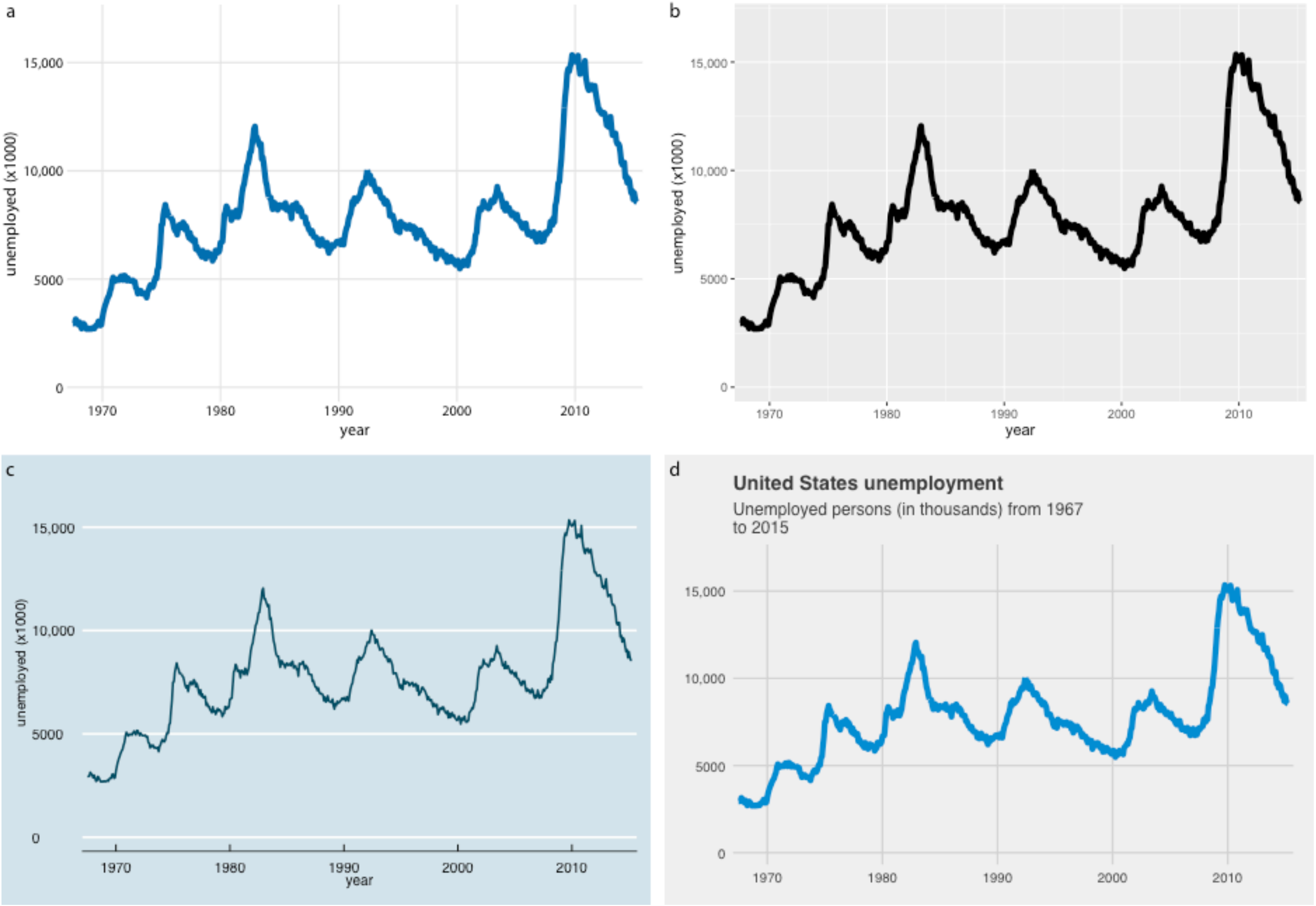
GDP and Health

Countries with a presumed organ donation consent are highlighted



Data from the General Social Science Survey, Distributed through the socviz R package

Themes (quickly)


a
15,000
10,000
5000
0
unemployed (x1000)
1970
1980
1990
2000
2010
year
b
15,000
10,000
5000
0
unemployed (x1000)
1970
1980
1990
2000
2010
year
c
15,000
10,000
5000
0
unemployed (x1000)
1970
1980
1990
2000
2010
year
d
United States unemployment
Unemployed persons (in thousands) from 1967 to 2015
15,000
10,000
5000
0
1970
1980
1990
2000
2010

ggthemes

- Great place to start. All sorts of themes.
- Includes color scales, etc., that align with themes
- You can even conform with other software
    - fit into an economics conference with `theme_stata`

See the themes here

ggthemeassist

- Another great place to start with making major modifications/creating your own custom theme
- Can't do everything, but can do a lot
- See here

(demo)

`theme()` for everything else

- I almost never touch `theme` unless I'm sharing the plot publicly
- You can basically change your plot to look however you want through `theme`
- Generally a bit more complicated

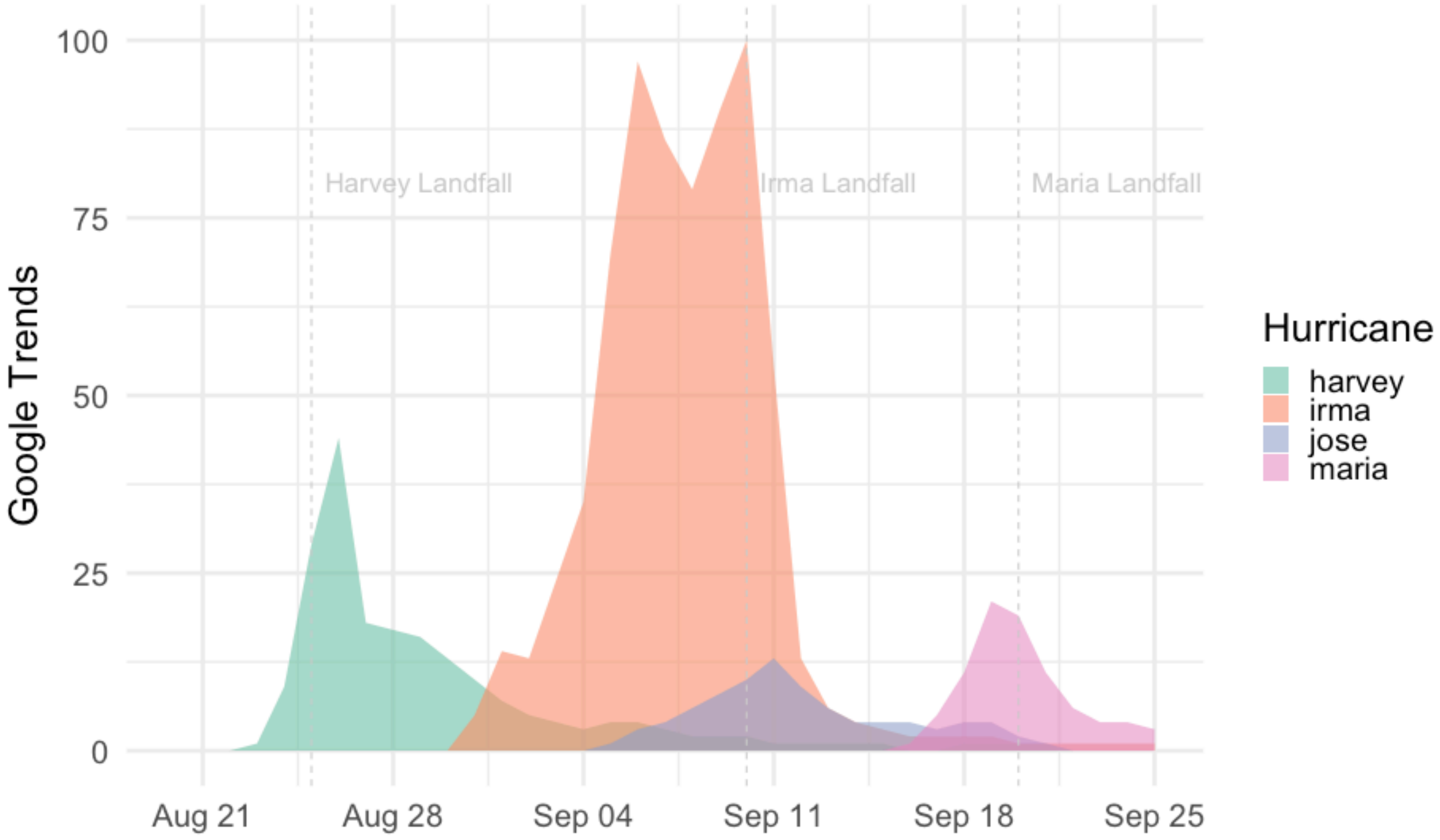
Quick example

From Lab 3

```r
g <- fivethirtyeight::google_trends %>%
  gather(hurricane, interest, -date) %>%
  separate(hurricane, c(NA, "hurricane", NA))

landfall <- tibble(date = lubridate::mdy(c("August 25, 2017",
                                           "September 10, 2017",
                                           "September 20, 2017")),
                   hurricane = c("Harvey Landfall",
                                 "Irma Landfall",
                                 "Maria Landfall"))
```

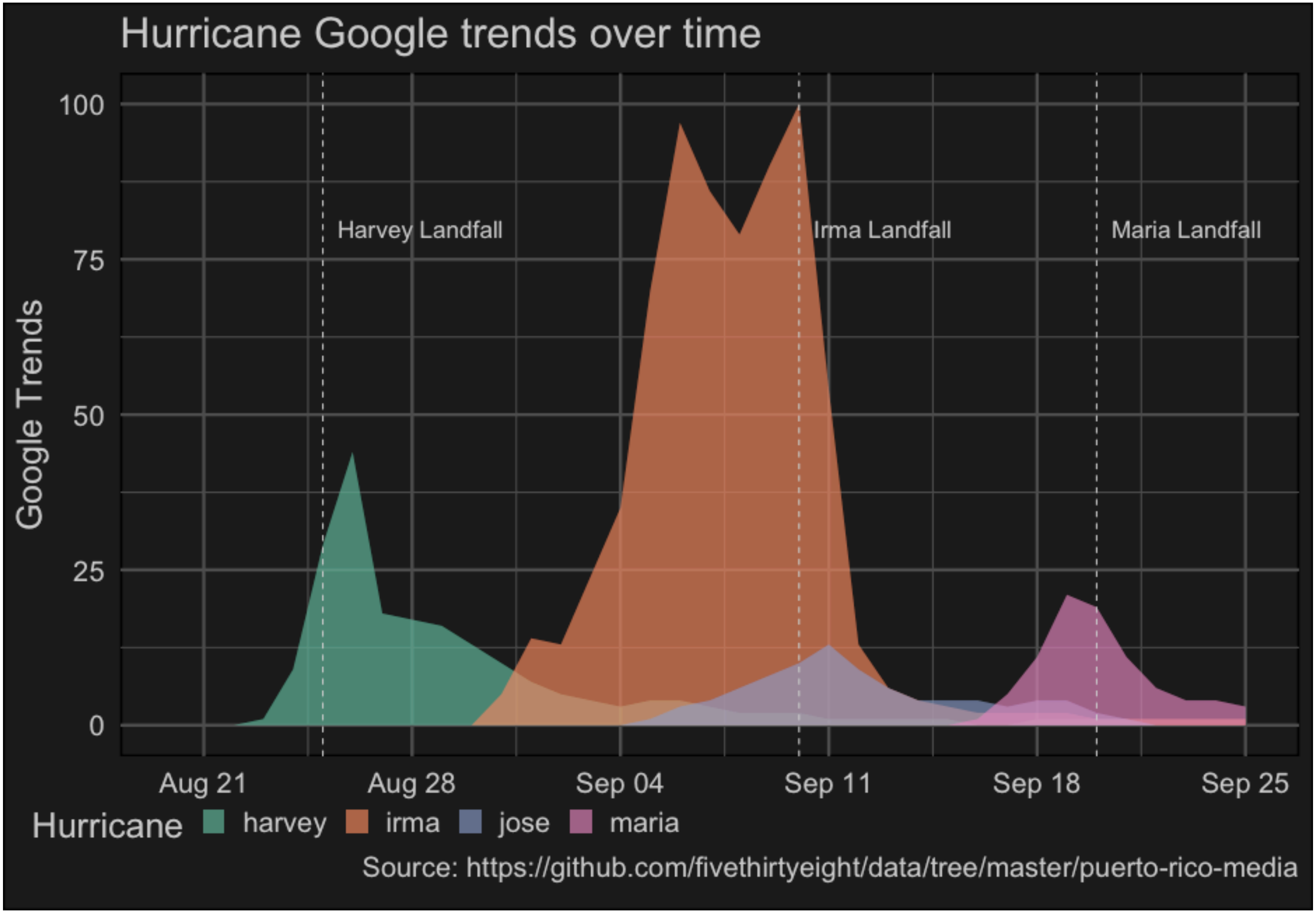
```
p <- ggplot(g, aes(date, interest)) +
  geom_ribbon(aes(fill = hurricane, ymin = 0, ymax = interest),
              alpha = 0.6) +
  geom_vline(aes(xintercept = date), landfall,
             color = "gray80",
             lty = "dashed") +
  geom_text(aes(x = date, y = 80, label = hurricane), landfall,
            color = "gray80",
            nudge_x = 0.5,
            hjust = 0) +
  labs(x = "",
       y = "Google Trends",
       title = "Hurricane Google trends over time",
       caption = "Source: https://github.com/fivethirtyeight/data/tree/maste
  scale_fill_brewer("Hurricane", palette = "Set2")
```

Hurricane Google trends over time



Source: https://github.com/fivethirtyeight/data/tree/master/puerto-rico-media

```
p + theme(panel.grid.major = element_line(colour = "gray30"),
          panel.grid.minor = element_line(colour = "gray30"),
          axis.text = element_text(colour = "gray80"),
          axis.text.x = element_text(colour = "gray80"),
          axis.text.y = element_text(colour = "gray80"),
          axis.title = element_text(colour = "gray80"),
          legend.text = element_text(colour = "gray80"),
          legend.title = element_text(colour = "gray80"),
          panel.background = element_rect(fill = "gray10"),
          plot.background = element_rect(fill = "gray10"),
          legend.background = element_rect(fill = NA, color = NA),
          legend.position = c(0.20, -0.1),
          legend.direction = "horizontal",
          plot.margin = margin(10, 10, b = 20, 10),
          plot.caption = element_text(colour = "gray80", vjust = 1),
          plot.title = element_text(colour = "gray80"))
```



Source: https://github.com/fivethirtyeight/data/tree/master/puerto-rico-media

Next time

Alison Hill

alison@rstudio.com

apreshill.rbind.io

@apreshill

@apreshill