Comparing Categories - Refinements

Maithreyi Gopalan

Week 5

Data viz in Wild

Saratessa

Sophia

Will and Maiko on deck

Reviewing Lab PS-2

Agenda

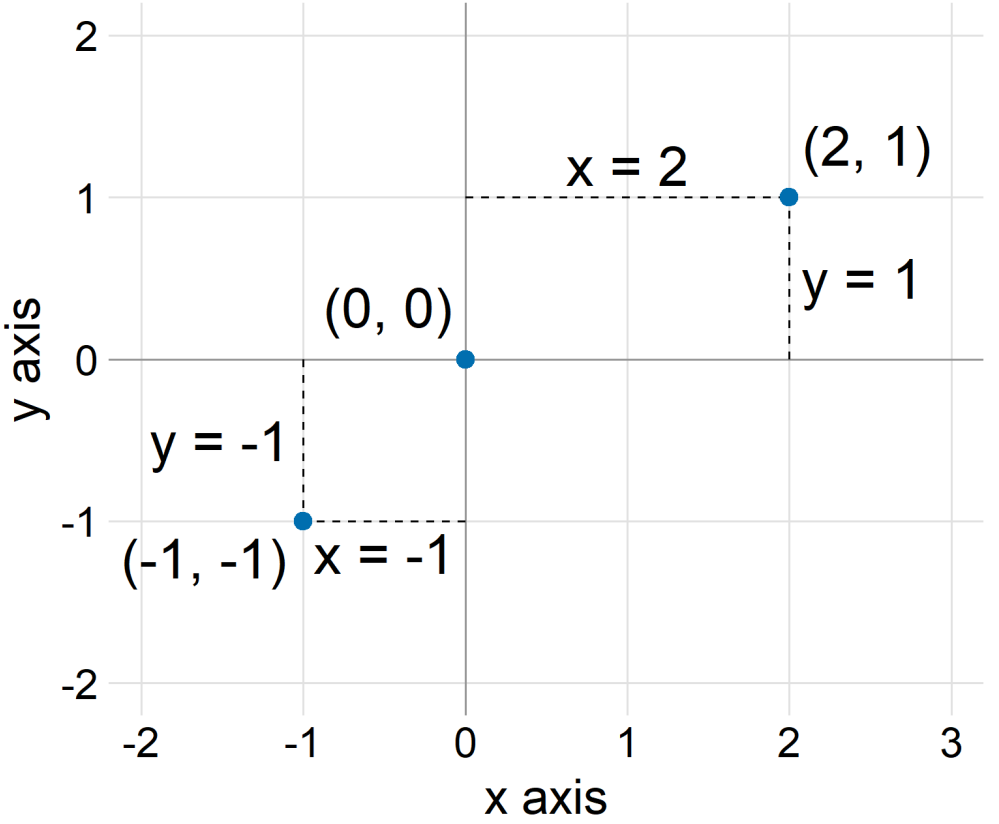
- Visualization for comparing categories - more considerations
- Aspect ratios, scales, and labels
- Annotations (most of the day)
- Saving plots (pretty quick)
- Compound figures
- Themes refresher (if time permits)
- Introduce Lab-PS3 quickly
- Lab 5
- Q & A with Dr. Daniel Anderson

Learning Objectives

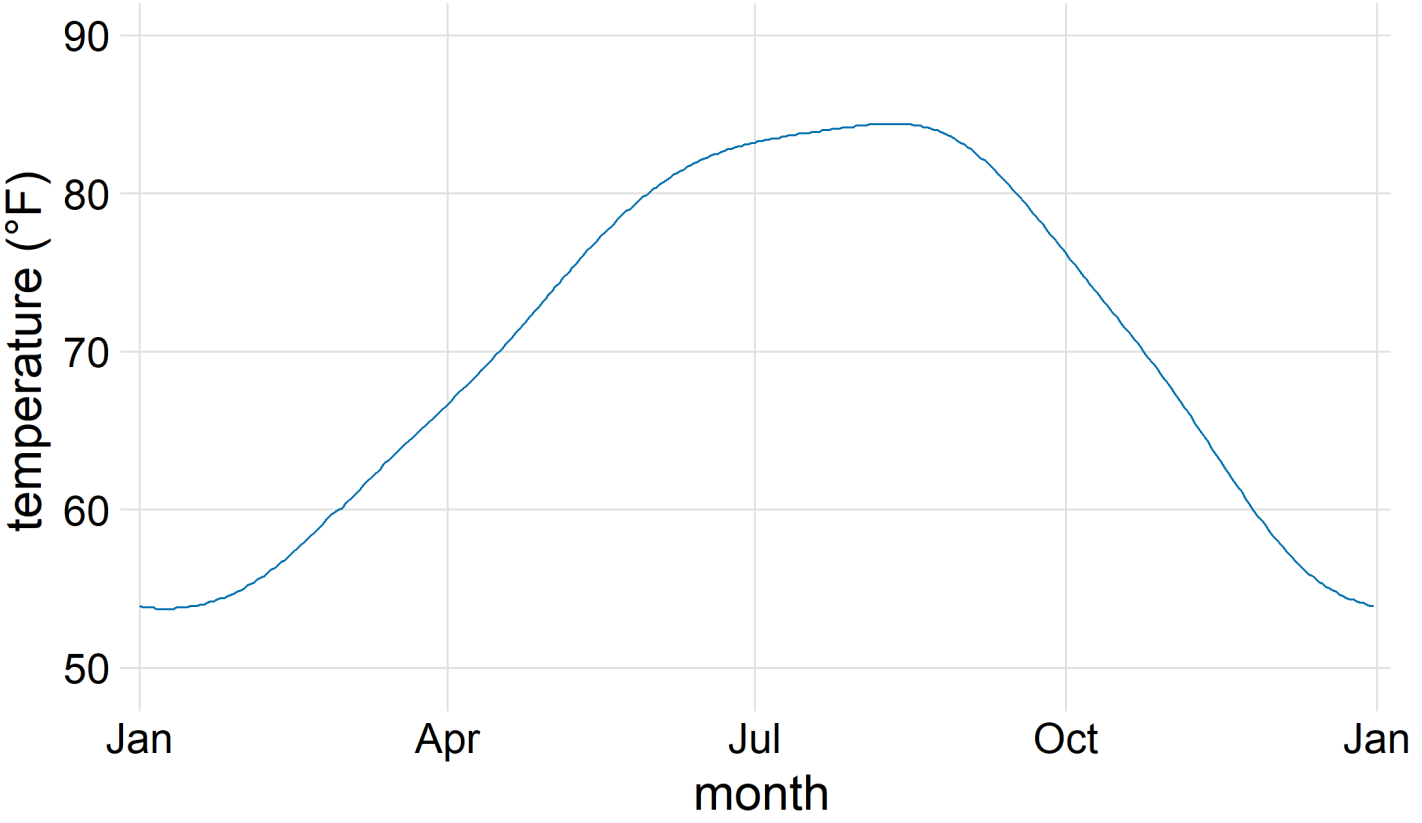
- Additional Considerations when visualizing/comparing categories including over time
- Understand how to make a wide variety of tweaks to ggplot.
- Understand common modifications to plots to make them more clear and reduce cognitive load
    - And ways to implement them

Axes

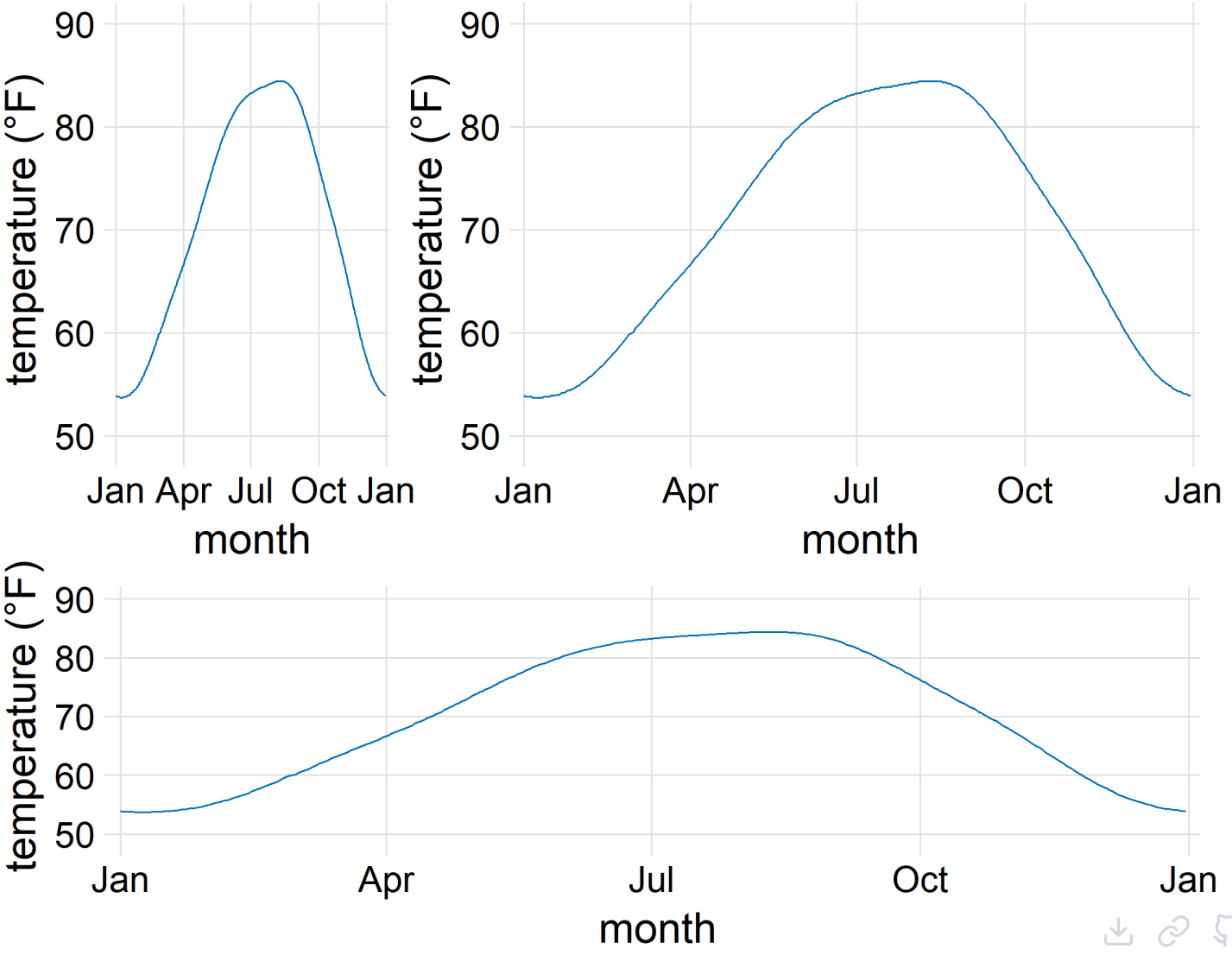
- Cartesian coordinates - what we generally use

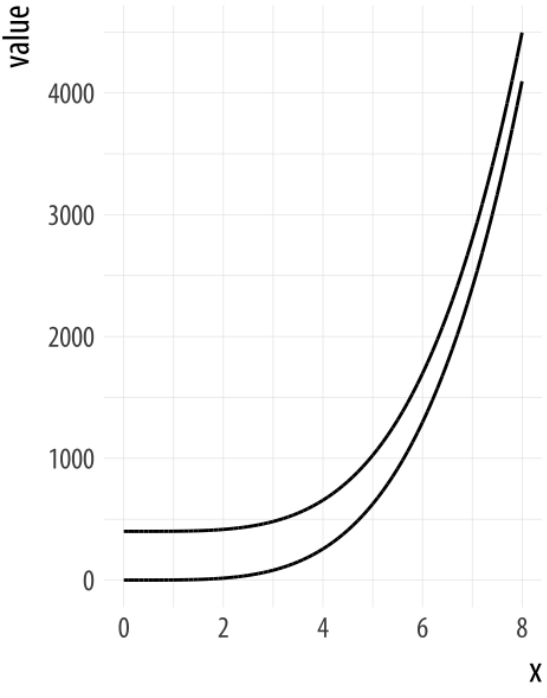


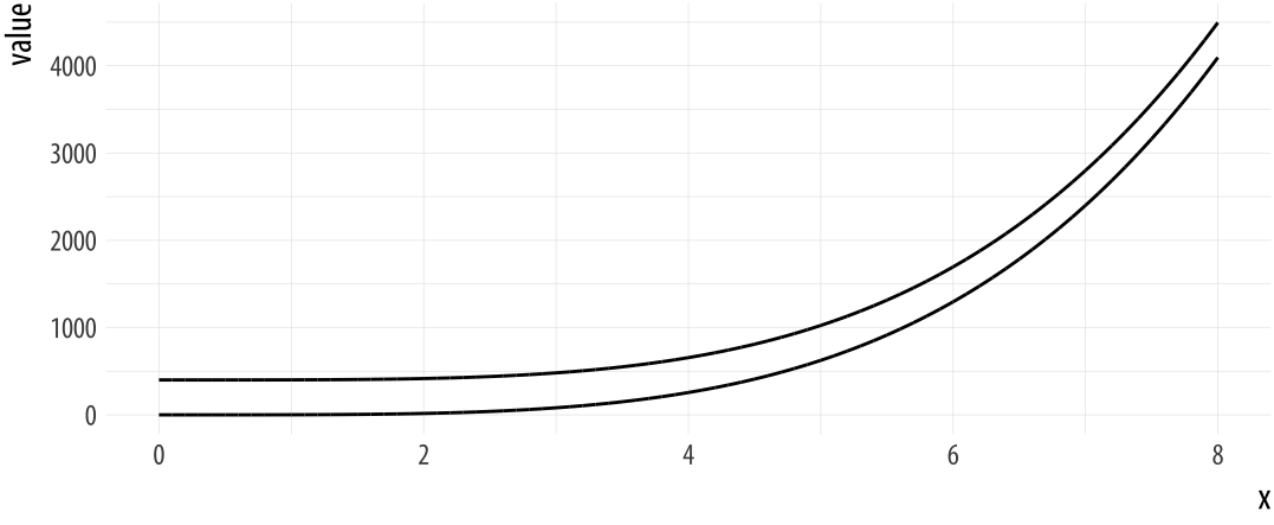
Different units



Aspect ratio

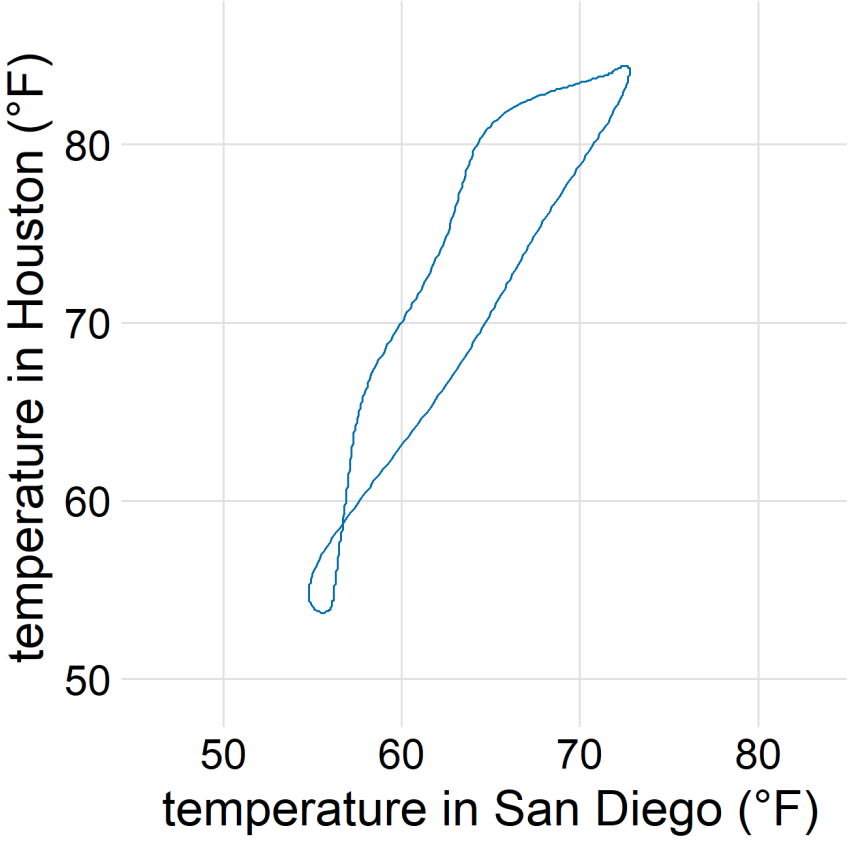



value
4000
3000
2000
1000
0
0
2
4
6
8
x


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

Same scales

Use `coord_fixed()`



Note - I think this is a weird plot, but the point remains

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

(from an apparently deleted tweet from @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)
> }
>
> — tj mahr 🍕🍍 (@tjmahr)

Gist

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

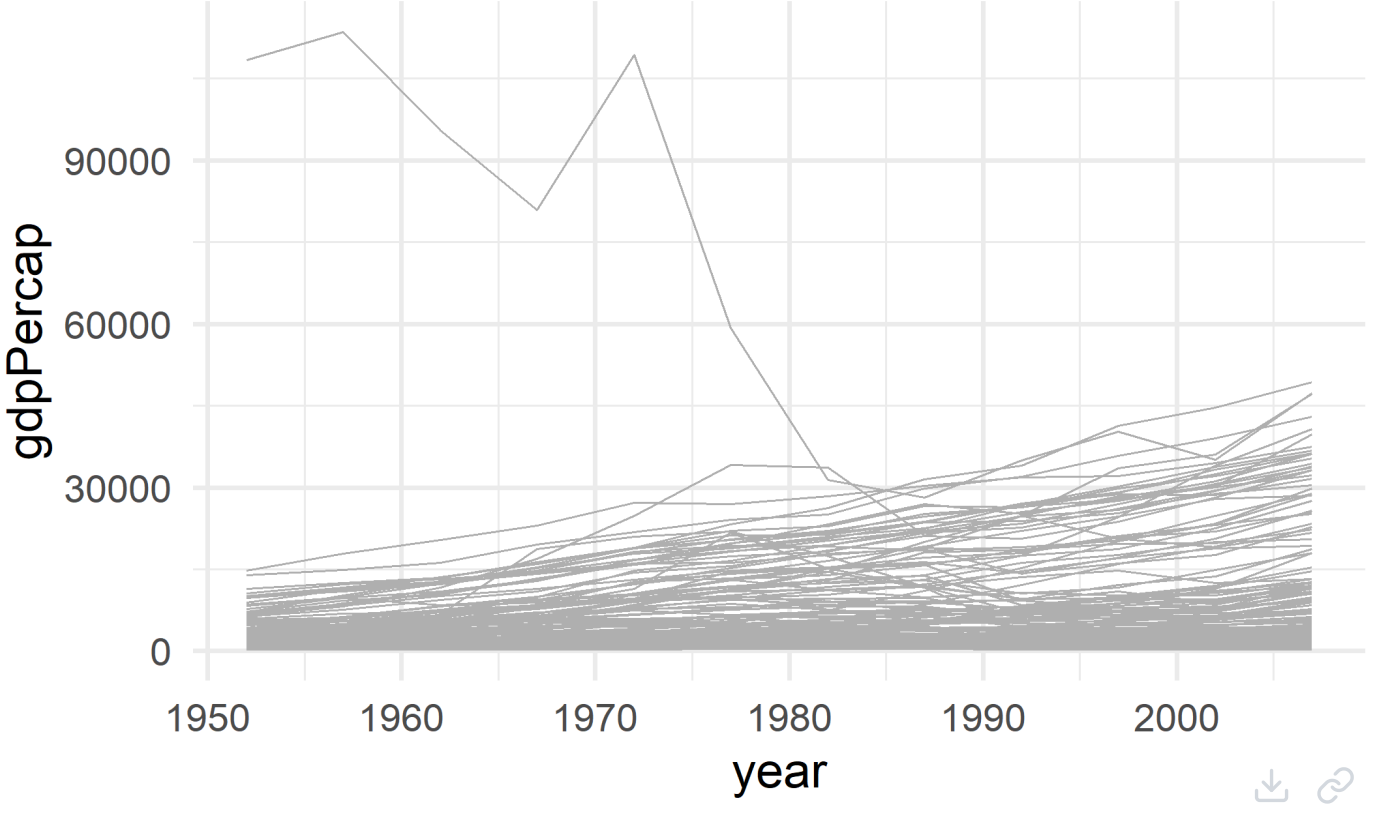
- See the full code/example here
- Let's take 3 minutes to play around:
  - Create a plot (could even be the example in the gist)
  - Try different aspect ratios by changing the width/length

03:00

Scale transformation

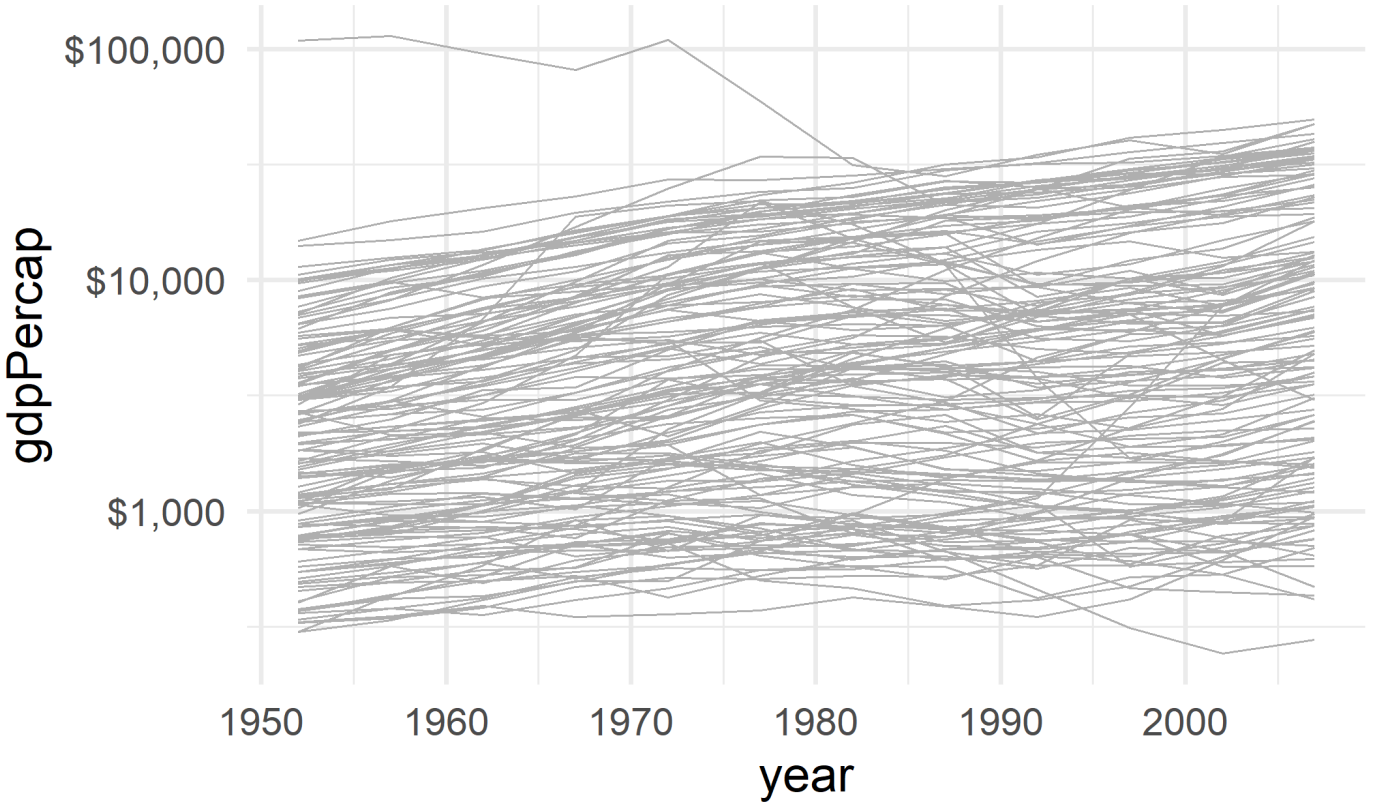
Raw scale

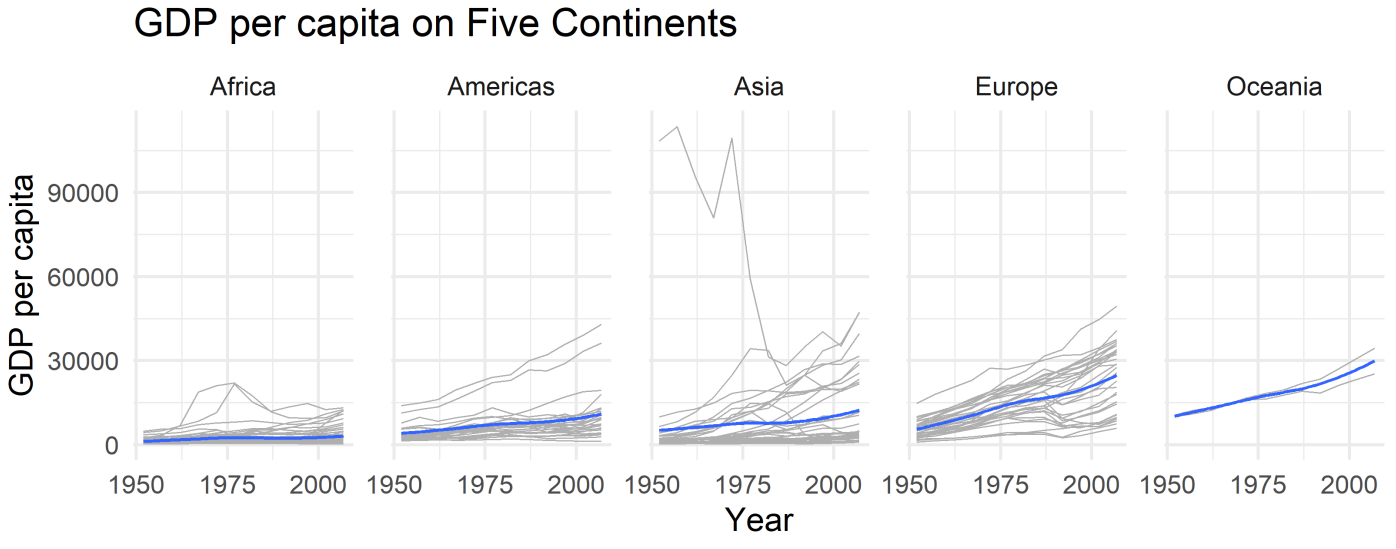
```
library(gapminder)
ggplot(gapminder, aes(year, gdpPercap)) +
  geom_line(aes(group = country),
            color = "gray70")
```

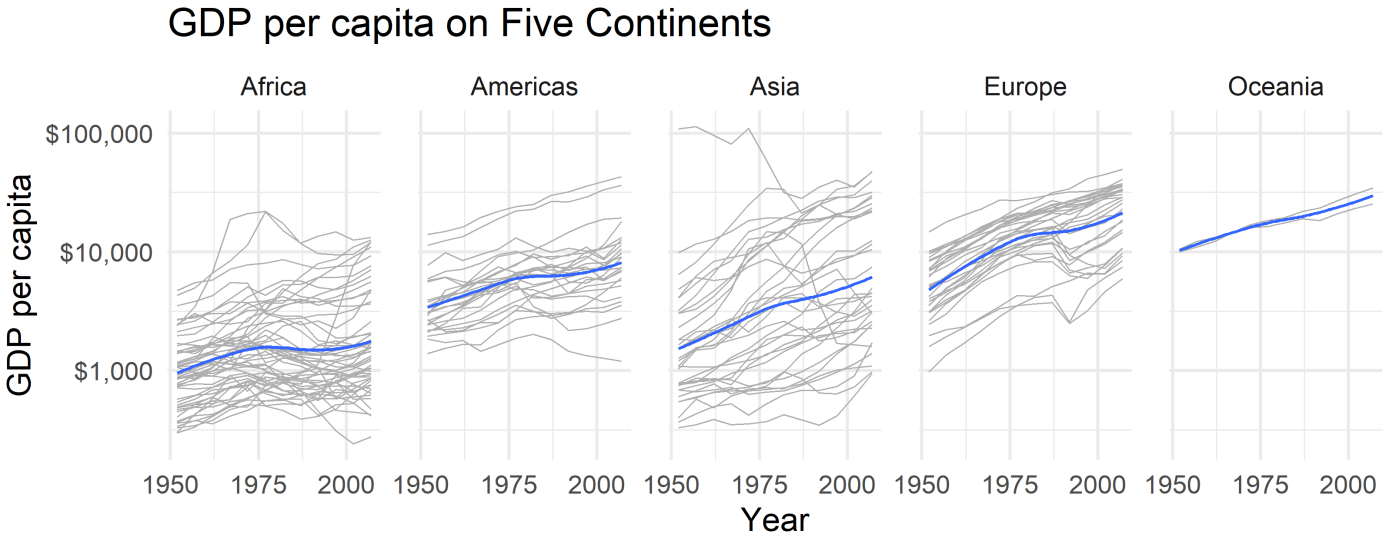


Log10 scale

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



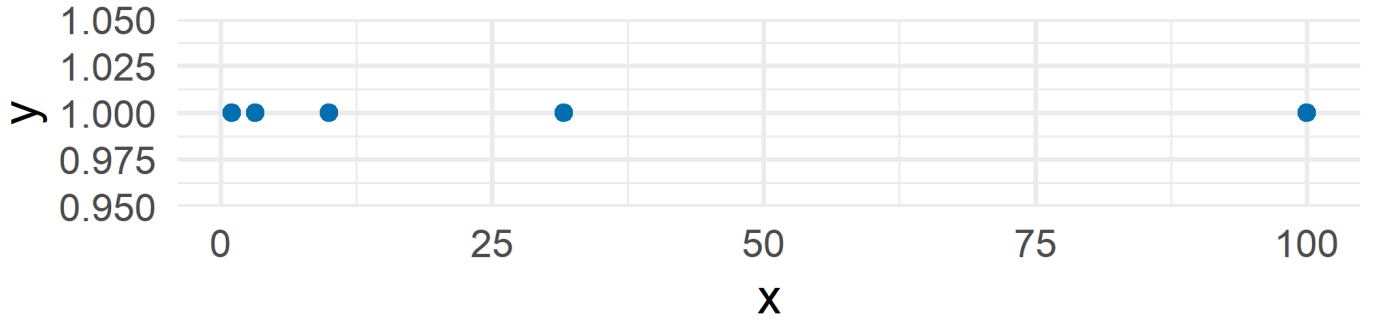

GDP per capita on Five Continents
Africa
Americas
Asia
Europe
Oceania
GDP per capita
90000
60000
30000
0
1950
1975
2000
Year

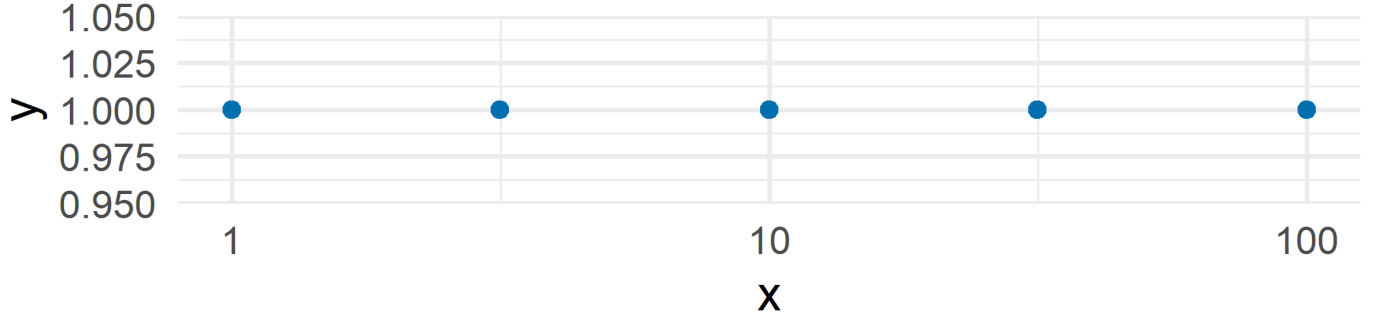

GDP per capita on Five Continents
Africa
Americas
Asia
Europe
Oceania
$100,000
$10,000
$1,000
GDP per capita
1950
1975
2000
Year

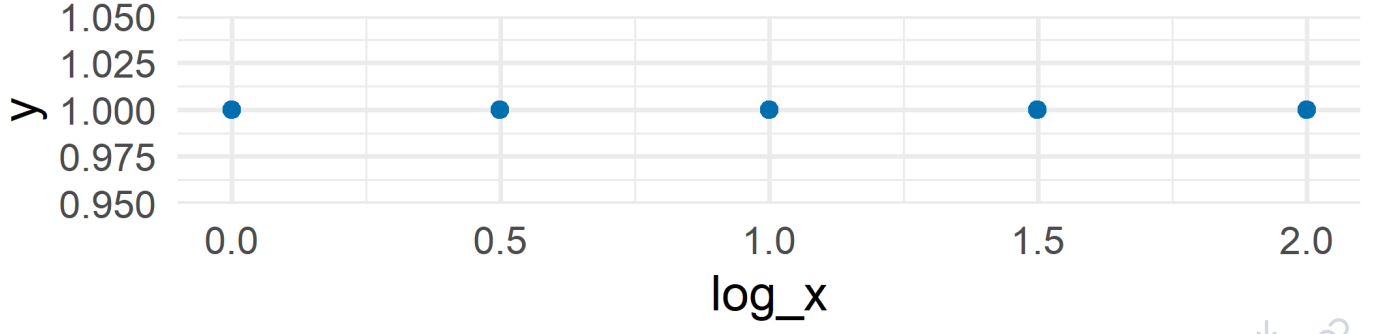
Scales

```
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

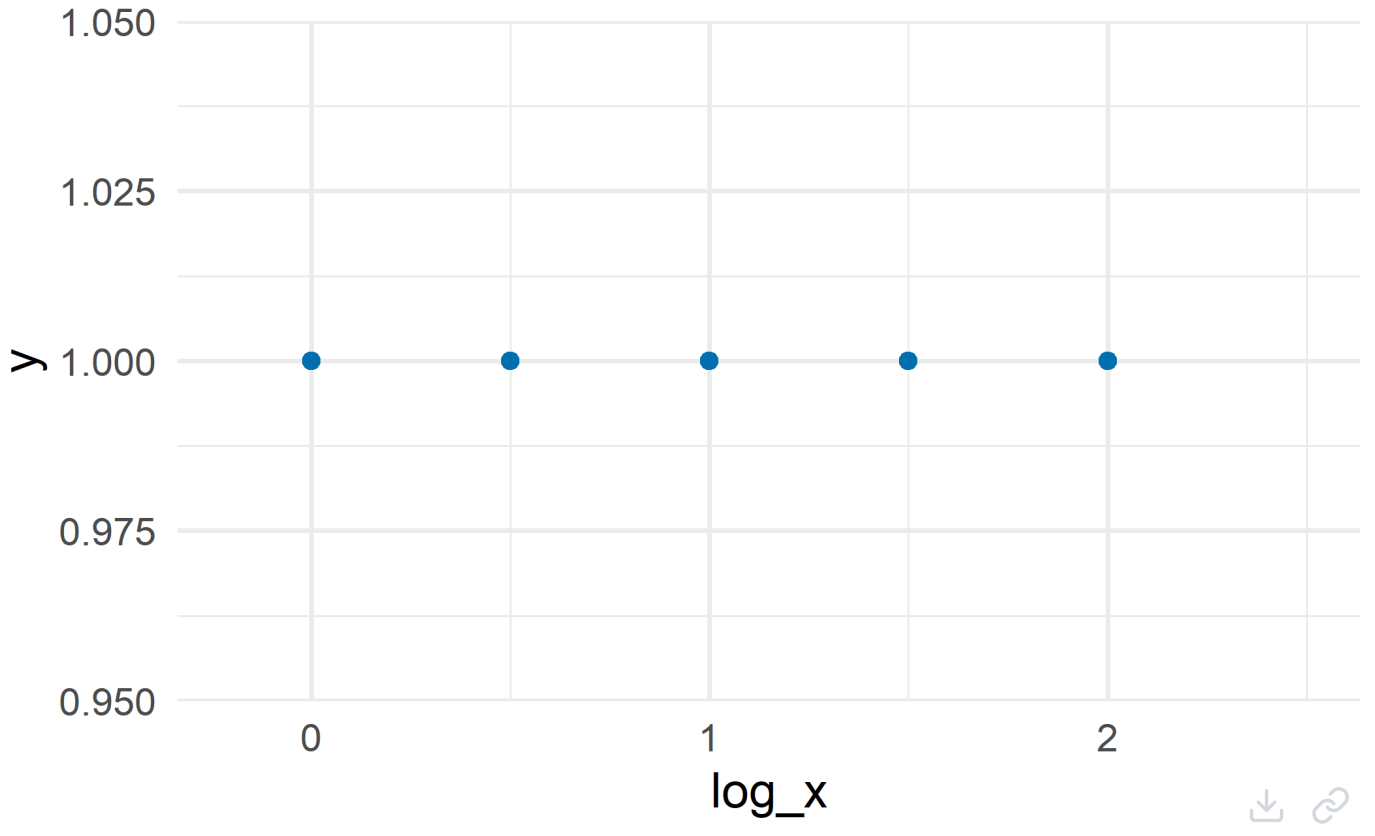

1.050
1.025
1.000
0.975
0.950
y
0
25
50
75
100
x


1.050
1.025
1.000
0.975
0.950
y
1
10
100
x


1.050
1.025
1.000
0.975
0.950
y
0.0
0.5
1.0
1.5
2.0
log_x

Don't transform twice

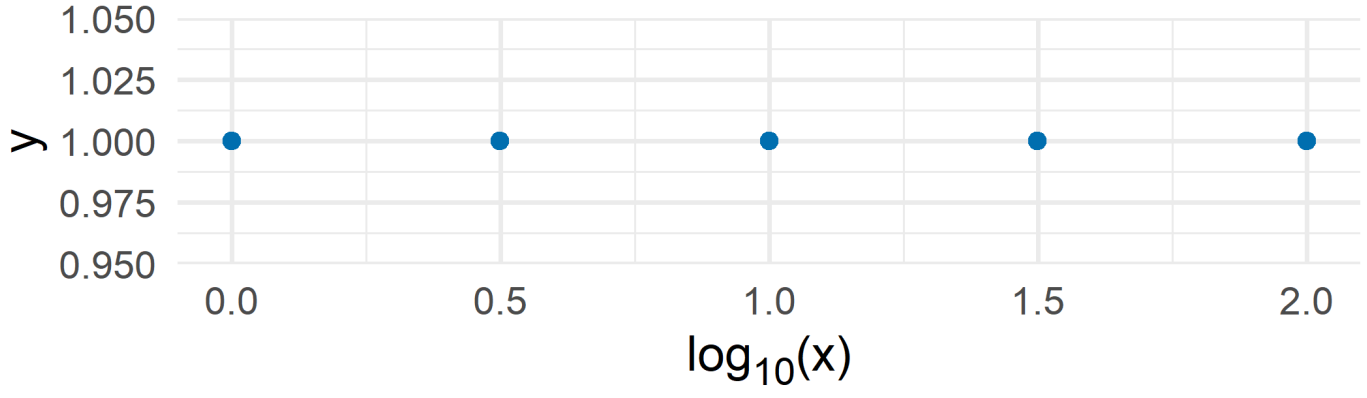
```
ggplot(d, aes(log_x, 1)) +
  geom_point(color = "#0072B2") +
  scale_x_log10() +
  xlim(-0.2, 2.5)
```



Careful with labeling

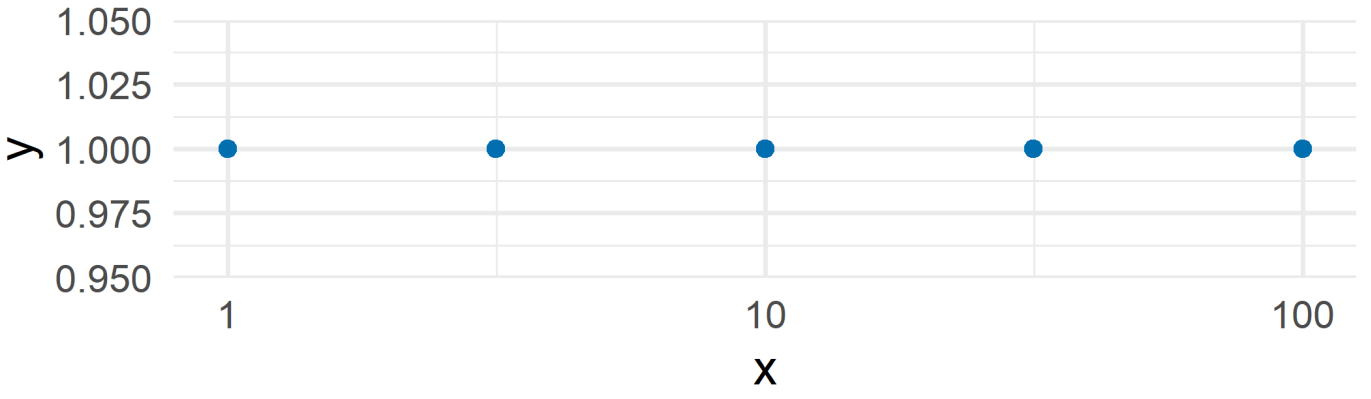
- Has the scale or the data been log transformed?
- Specify the base

```
library(ggtext)
ggplot(d, aes(log_x, 1)) +
  geom_point(color = "#0072B2") +
  labs(x = "log<sub>10</sub>(x)") +
  theme(axis.title.x = element_markdown())
```



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

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



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

Interpretation

Log scales show relative numbers, not raw

Difference between 100 and 101 (1% change) will be much smaller than difference between 1 and 2 (100% change)

More resources to learn more

- Blog post by Lisa Charlotte Muth
- The "Logorithmic or Linear scales" section here

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?

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
- Make sure your figure has a title
  - Should not start with "This figure displays/shows..."

Caption

Consider stating the data source

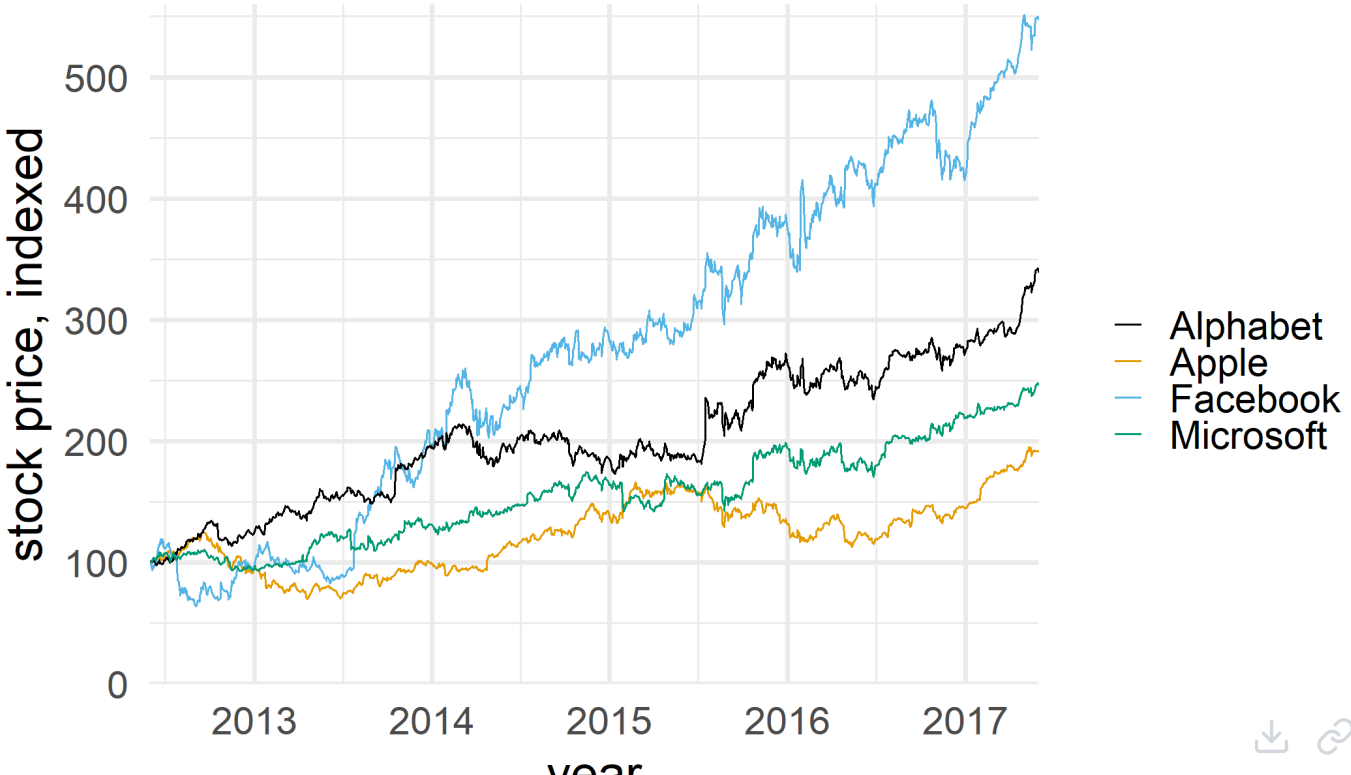
Other details relevant to the figure but not important enough for a subtitle

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

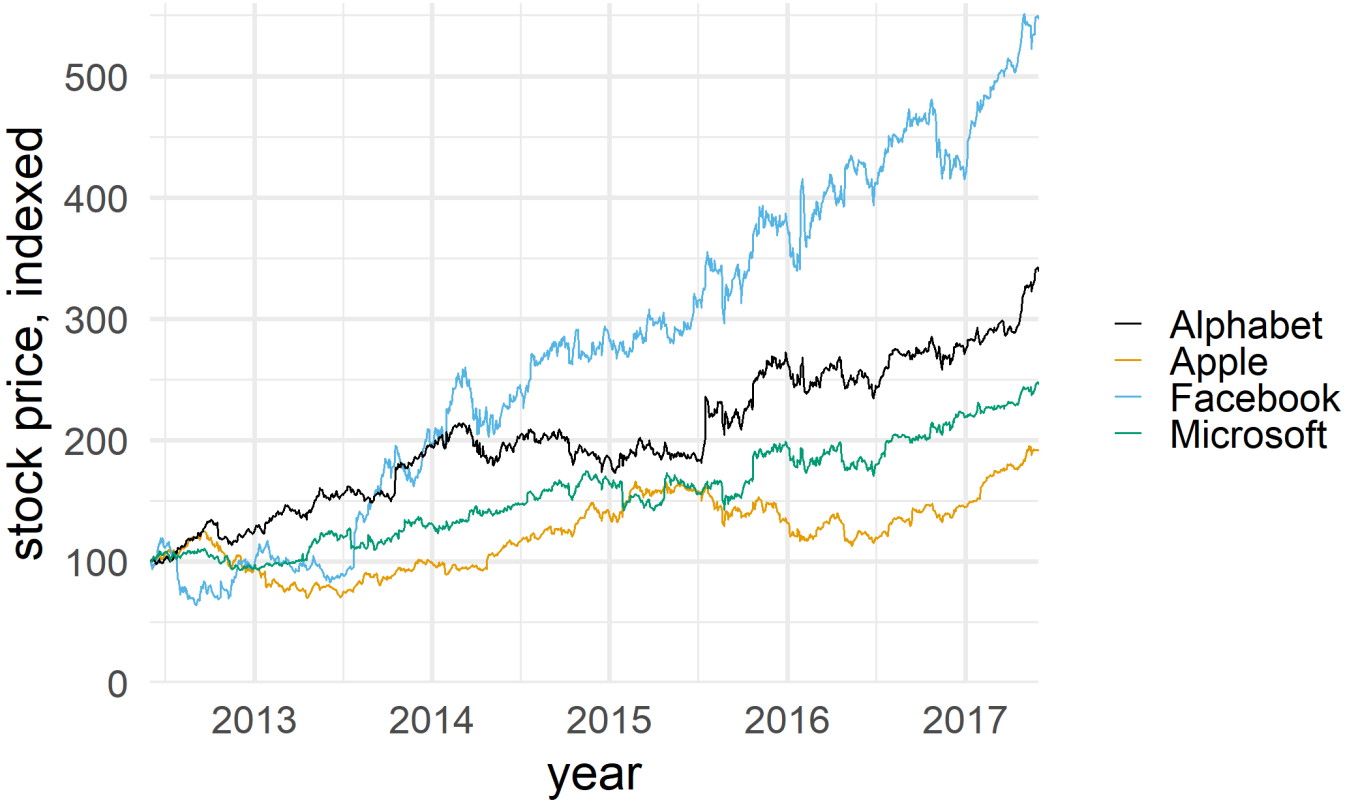
Omission

- Consider omitting obvious or redundant labels
  - Use `labs(x = NULL)` or `labs(x = "")`
  - If already using `scale_x/y_*()` just supply the `name` argument

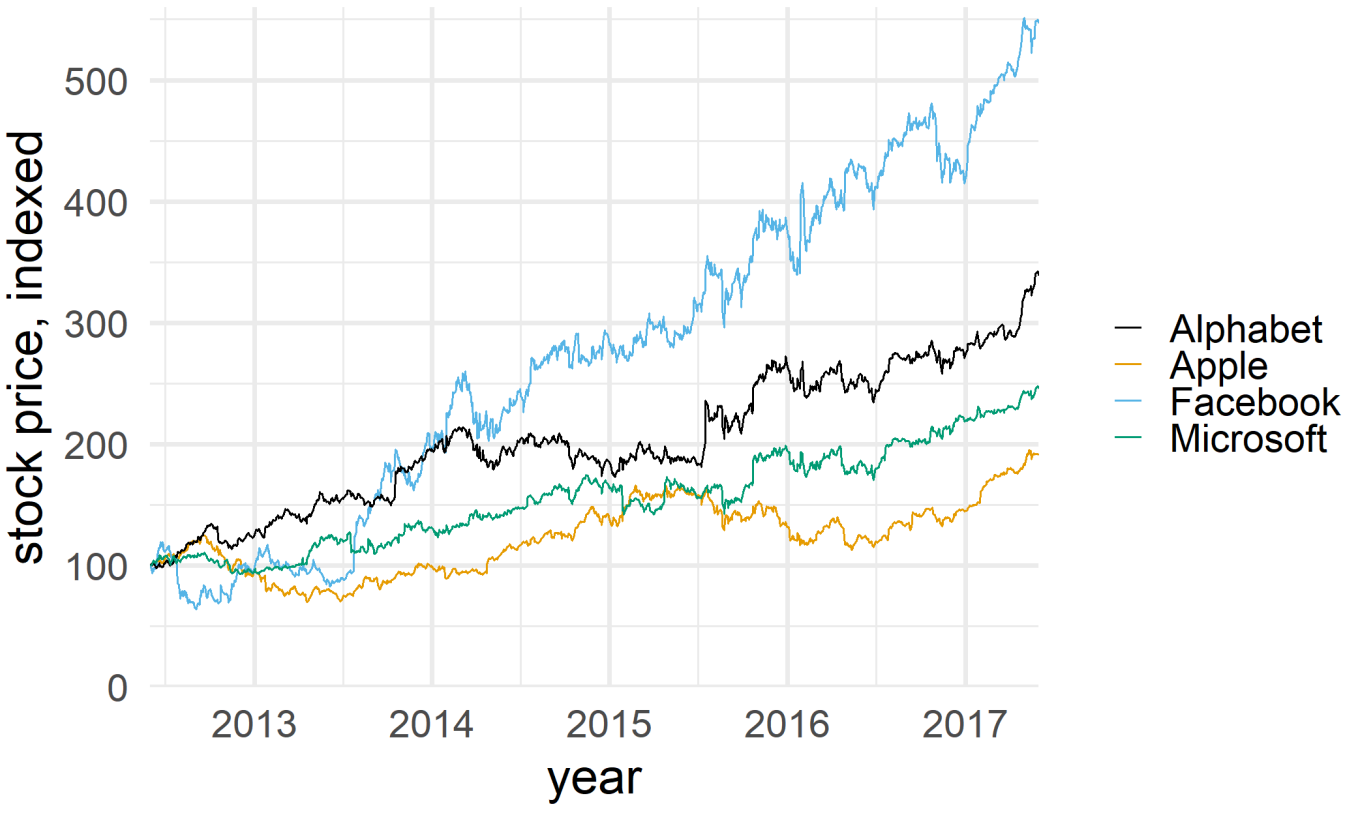


Omission

- Do not omit axis titles that are not obvious



Don't overdo it



Practice

Let's use the `ggplot2::diamonds` dataset.

- Plot the relationship between `carat` and `price`
- Play with scale transformations
- Give it some good labels
- Make any other modifications you'd like that you think makes it prettier and/or easier to interpret.

05:00

Annotations

The big topic for the day

Among the most effective

- If possible, try to remove legends, and just include annotations
- Warning - this is often fairly difficult in ggplot (requires a lot of fiddling)

- Consider saving and making final annotations outside of R

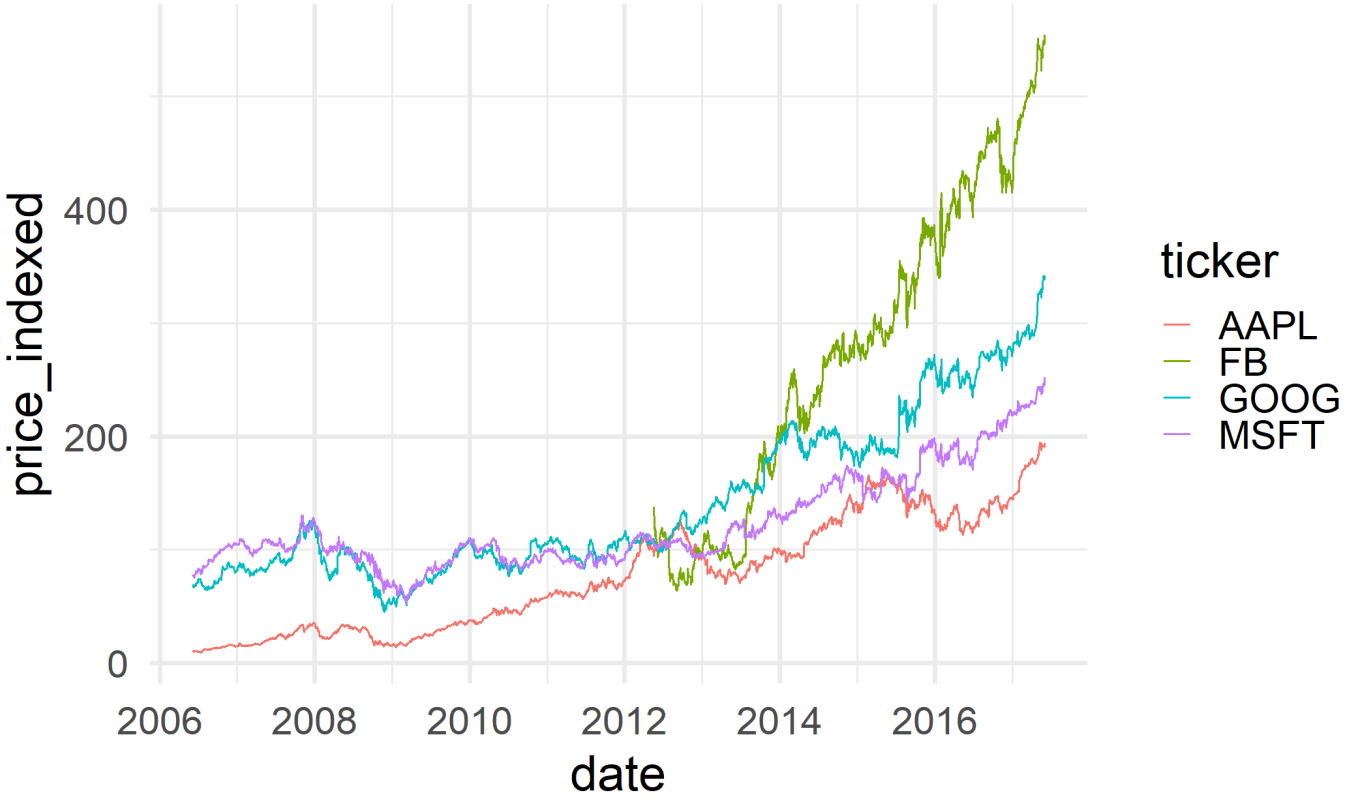
    - Bad for reproducibility, but good for interpretability
- There are some packages that can help

Building up a plot

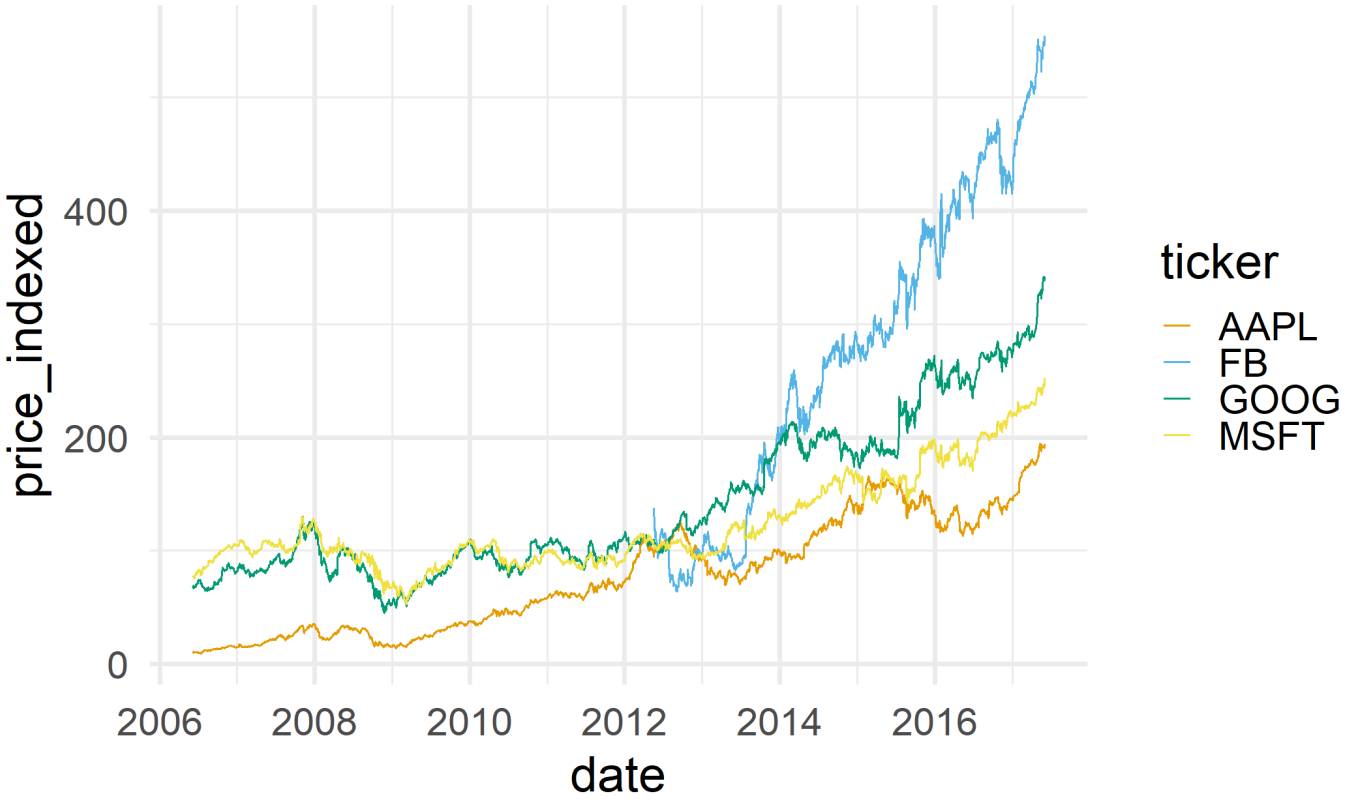
```
remotes::install_github("clauswilke/dviz.supp")
head(tech_stocks)
```

```
## # A tibble: 6 × 6
##   company  ticker date       price index_price price_indexed
##   <chr>    <chr>  <date>     <dbl>       <dbl>          <dbl>
## 1 Alphabet GOOG   2017-06-02  976.        285.           342.
## 2 Alphabet GOOG   2017-06-01  967.        285.           339.
## 3 Alphabet GOOG   2017-05-31  965.        285.           338.
## 4 Alphabet GOOG   2017-05-30  976.        285.           342.
## 5 Alphabet GOOG   2017-05-26  971.        285.           341.
## 6 Alphabet GOOG   2017-05-25  970.        285.           340.
```

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

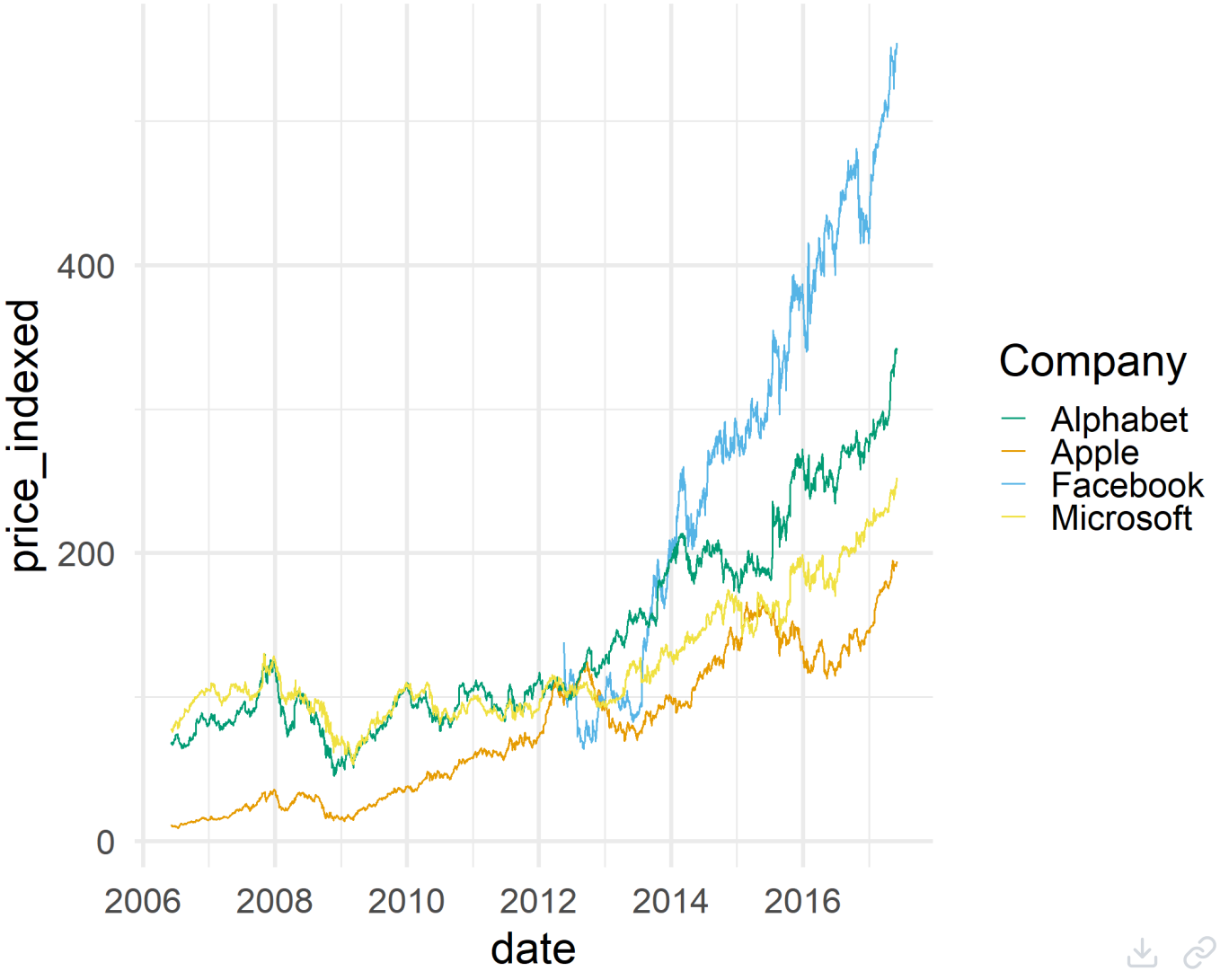


```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto()
```



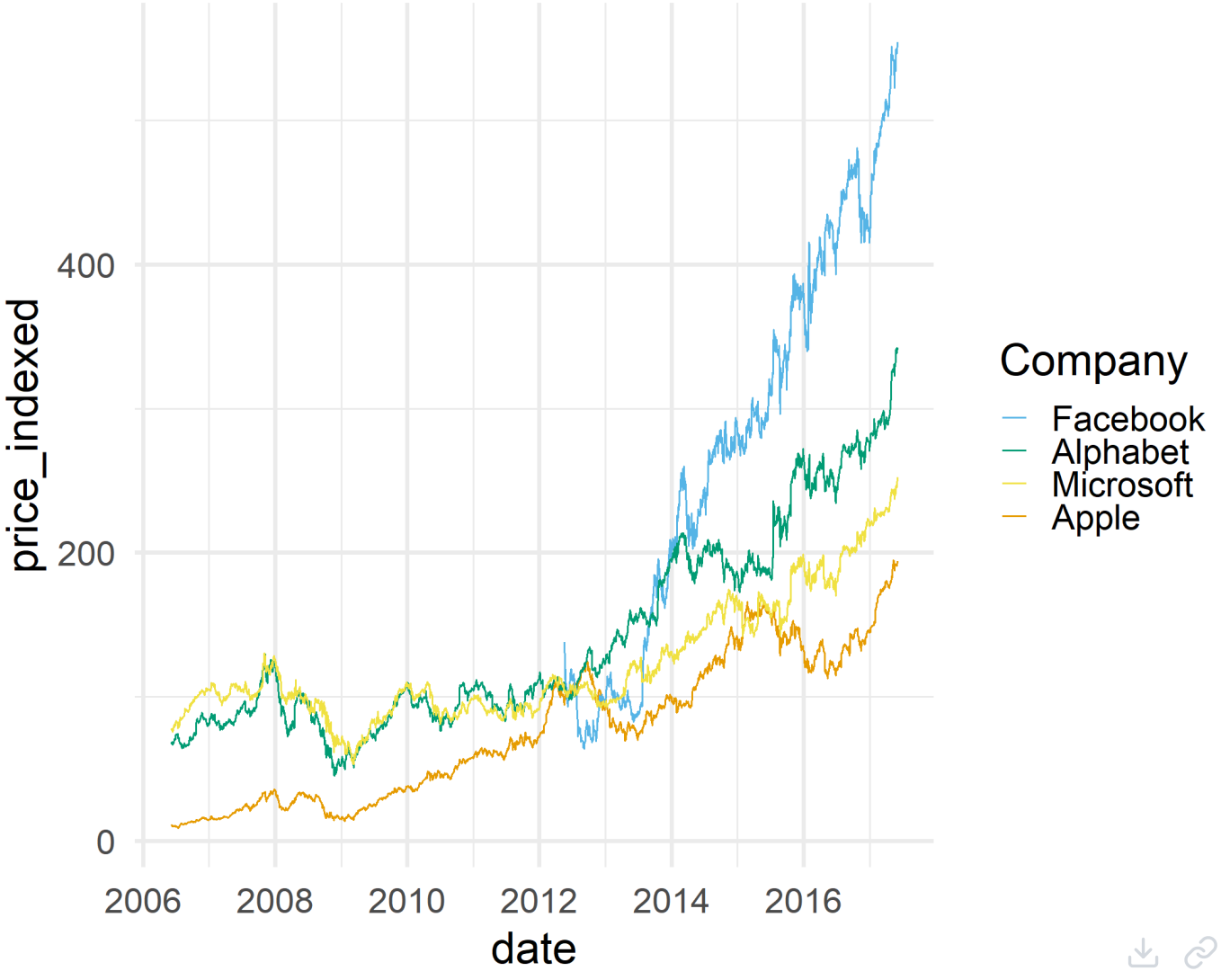
```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(
    name = "Company",
    breaks = c("GOOG", "AAPL", "FB", "MSFT"),
    labels = c("Alphabet", "Apple", "Facebook", "Microsoft")
  )
```

Bad

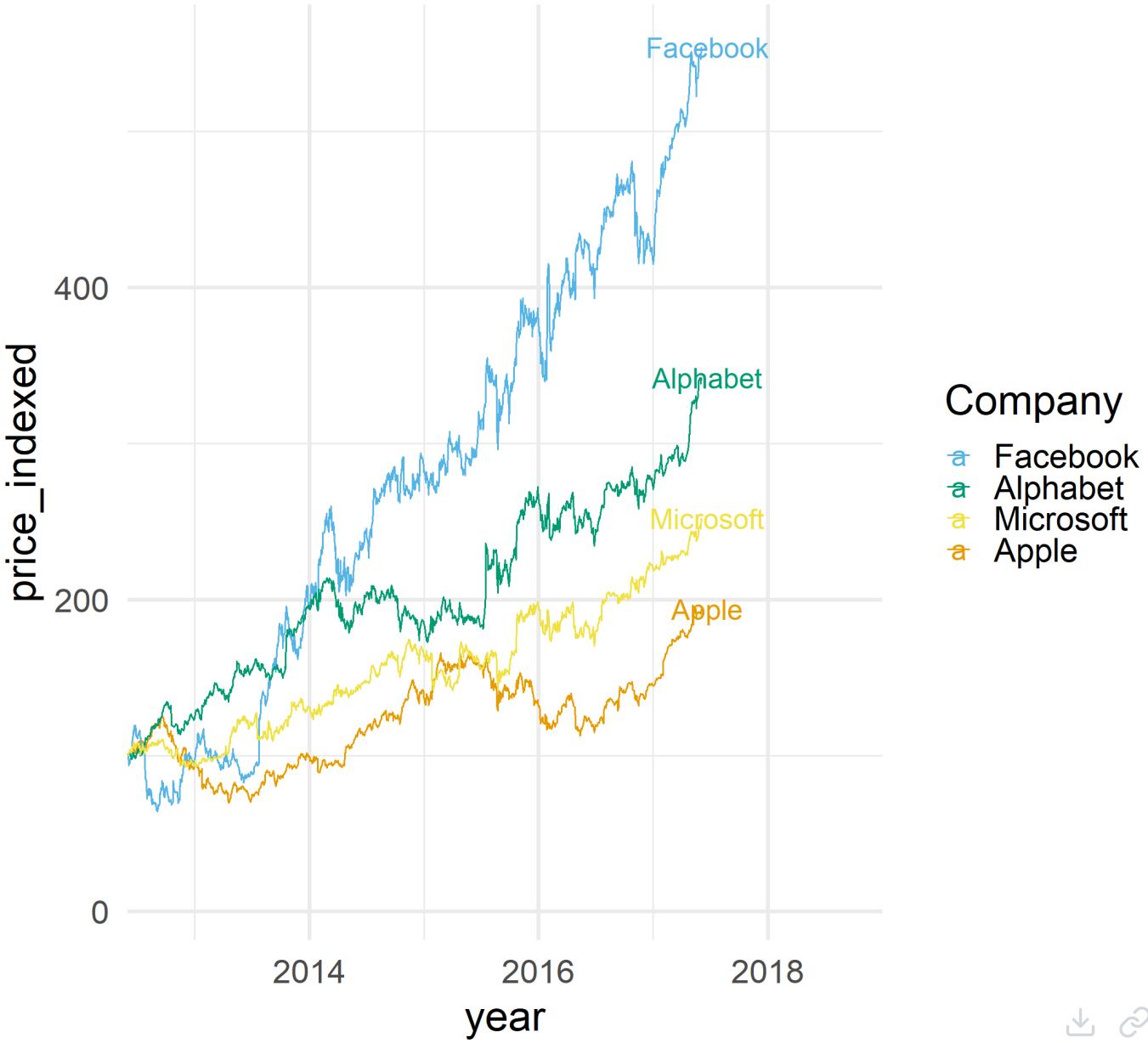

price_indexed
400
200
0
2006
2008
2010
2012
2014
2016
date
Company
Alphabet
Apple
Facebook
Microsoft

```
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", "Apple")
  )
```

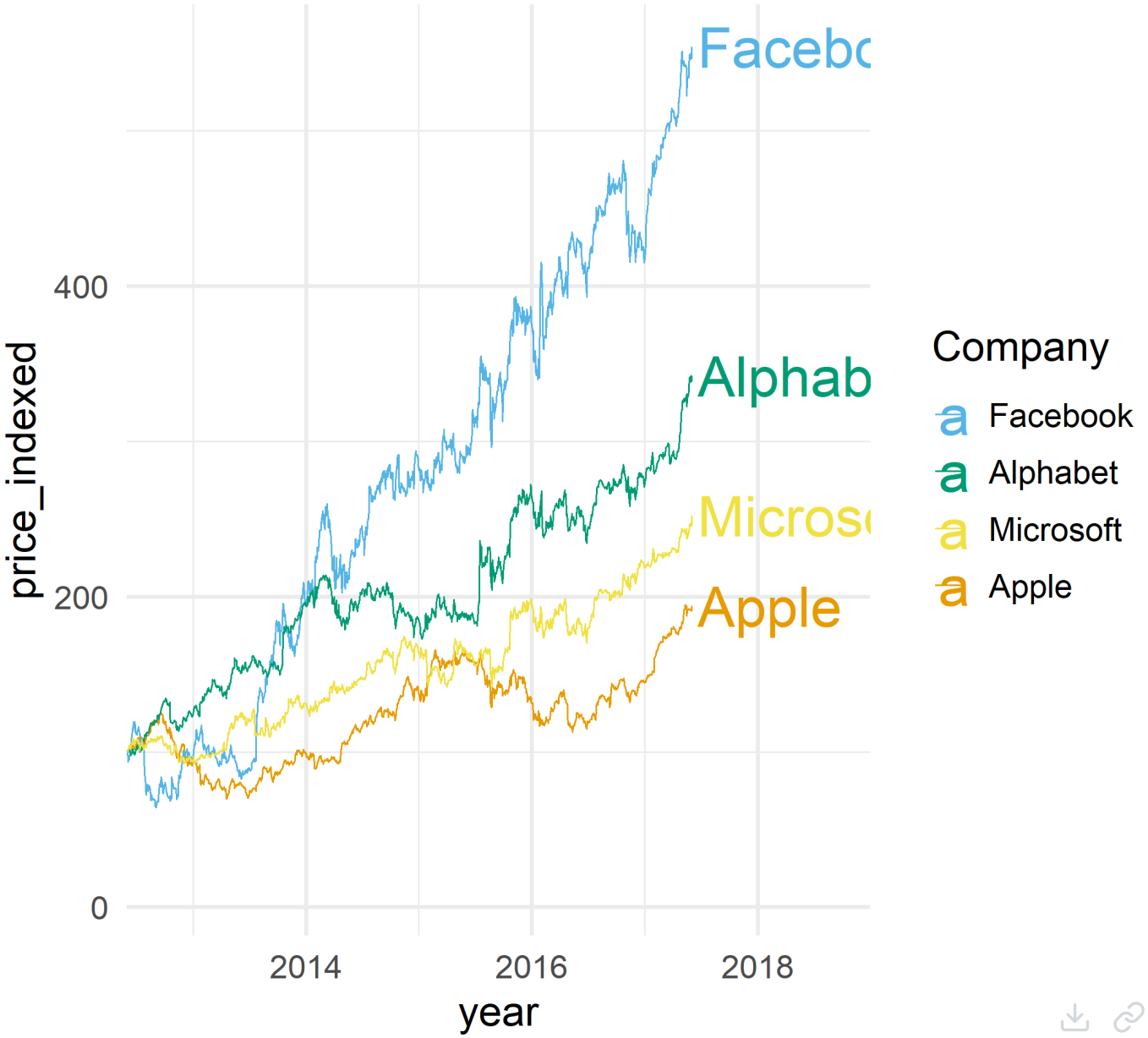
Good


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

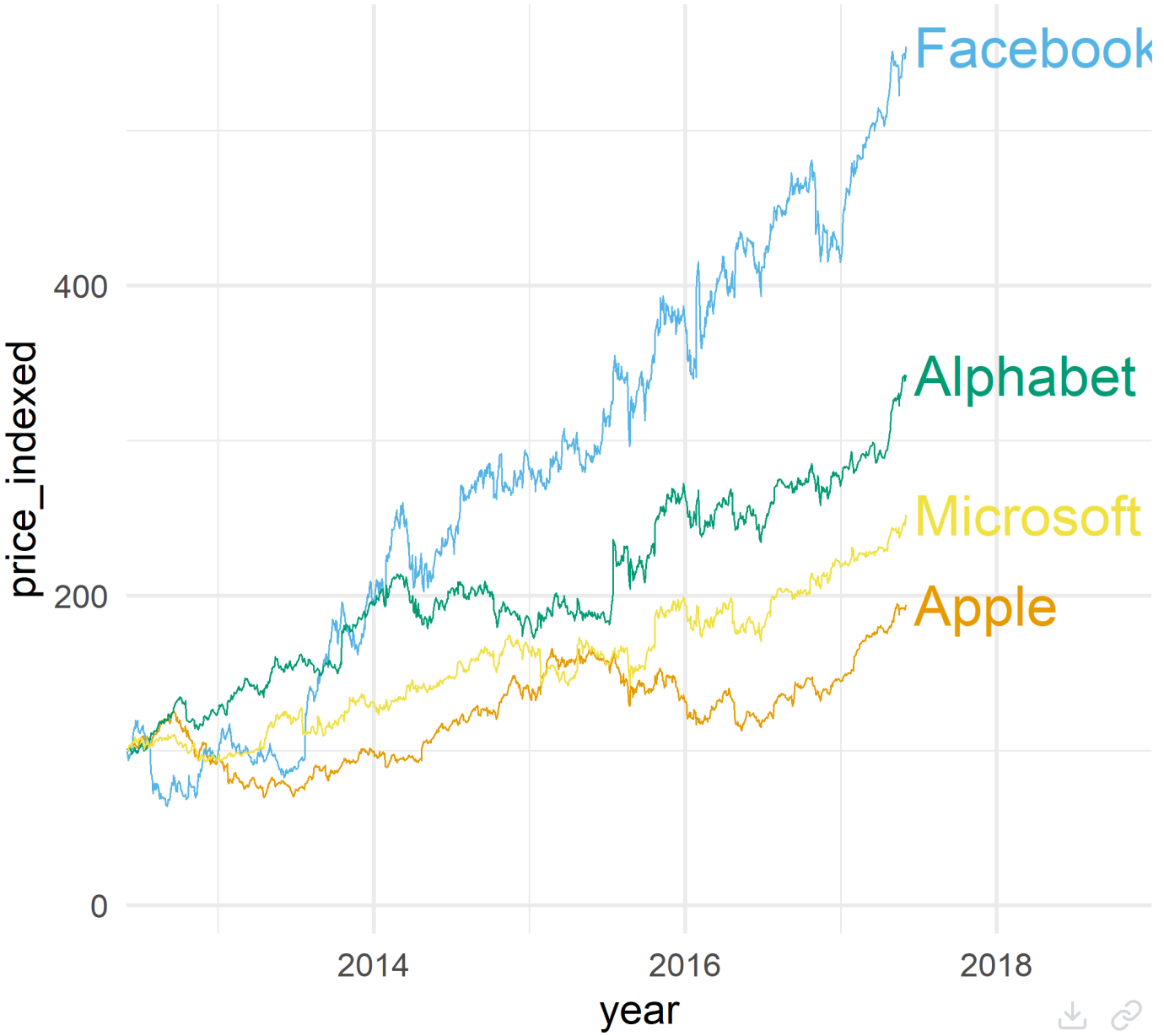
```r
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(
    name = "Company",
    breaks = c("FB", "GOOG", "MSFT", "AAPL"),
    labels = c("Facebook", "Alphabet", "Microsoft", "Apple")
  ) +
  scale_x_date(
    name = "year",
    limits = c(ymd("2012-06-01"), 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 = 20
  )
```


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

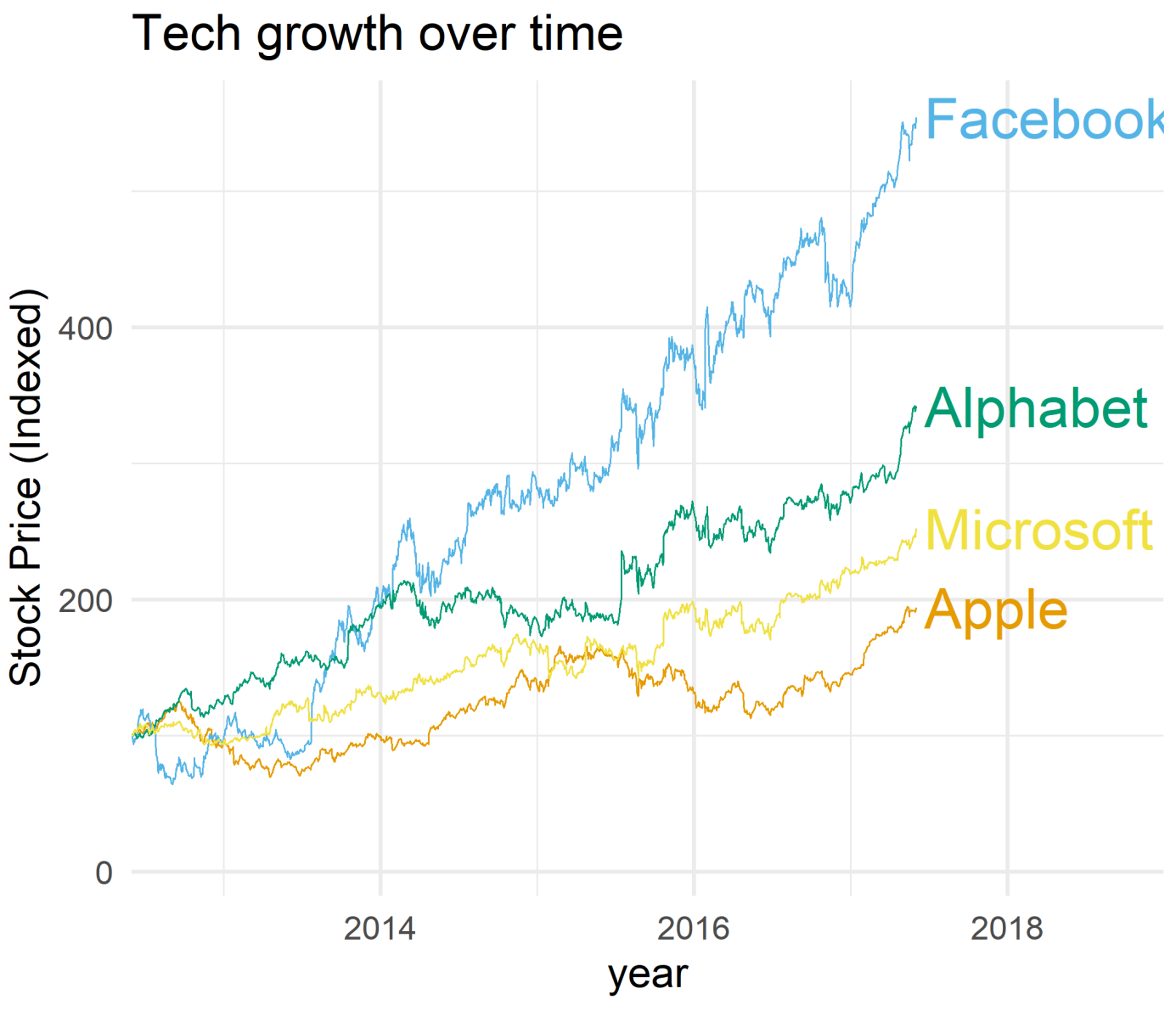
```
ggplot(tech_stocks, aes(date, price_indexed, color = ticker)) +
  geom_line() +
  scale_color_OkabeIto(
    name = "Company",
    breaks = c("FB", "GOOG", "MSFT", "AAPL"),
    labels = c("Facebook", "Alphabet", "Microsoft", "Apple")
  ) +
  scale_x_date(
    name = "year",
    limits = c(ymd("2012-06-01"), 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 = 20,
    hjust = 0,
    size = 12
  )
```


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", "Apple")
  ) +
  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 = 20,
    hjust = 0,
    size = 12
  ) +
  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", "Apple")
  ) +
  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 = 20,
    hjust = 0,
    size = 12
  ) +
  guides(color = "none") +
  labs(
    title = "Tech growth over time",
    caption = "Data from Wilke (2019): Fundamentals of Data Visu
    y = "Stock Price (Indexed)"
  )
```

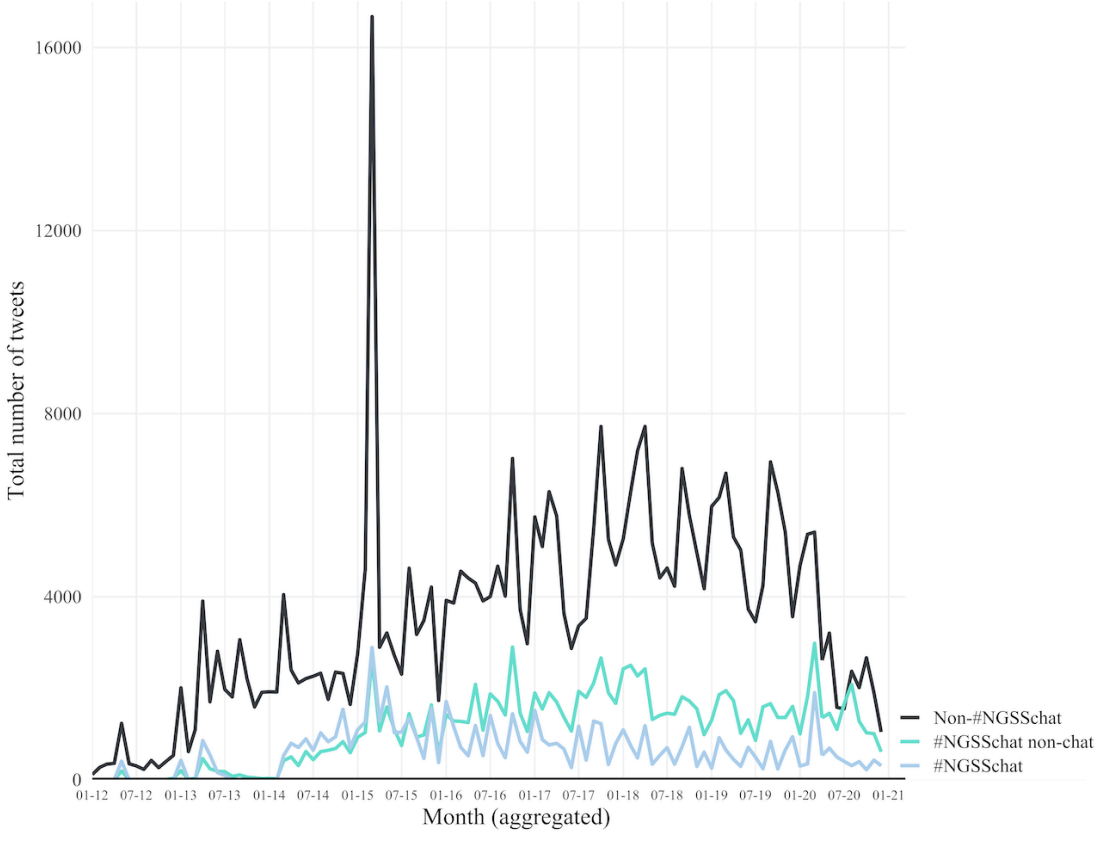

Tech growth over time
Facebook
Alphabet
Microsoft
Apple
Stock Price (Indexed)
400
200
0
2014
2016
2018
year
Data from Wilke (2019): Fundamentals of Data Visualization

A few more notes

- You might want to try `geom_label()` instead of `geom_text()`, or perhaps layering them with the first providing the white space for the second (as we saw in Lab 2)
- Could consider not making the font color vary with the lines (the labels are close enough)
- Depending on you you use the legend, it can work almost as well.

Example

From an actual publication, where I used the legend instead of direct annotations



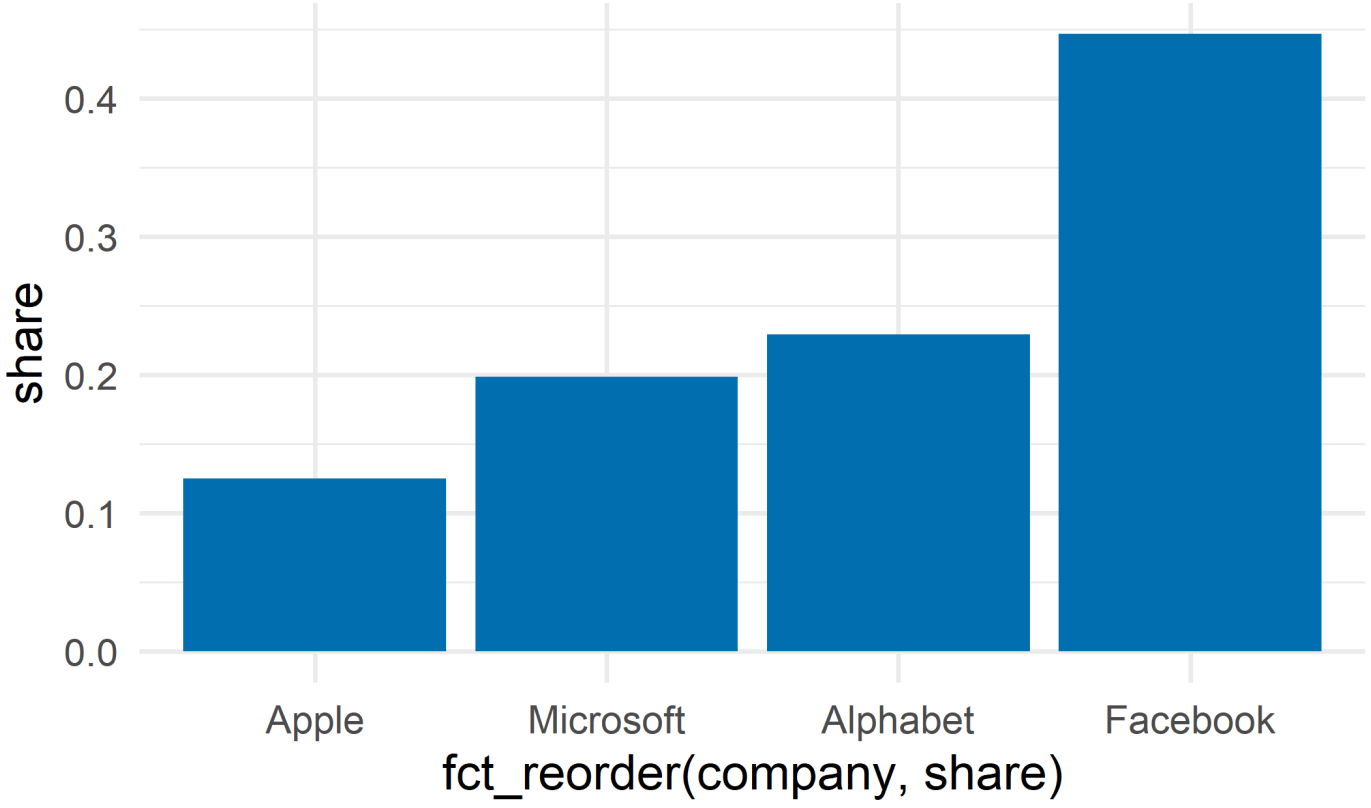
Labeling bars

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

```
## # A tibble: 4 × 3
##   company   stock_av share
##   <chr>        <dbl> <dbl>
## 1 Alphabet     141.  0.229
## 2 Apple         77.1 0.125
## 3 Facebook     275.  0.447
## 4 Microsoft    122.  0.199
```

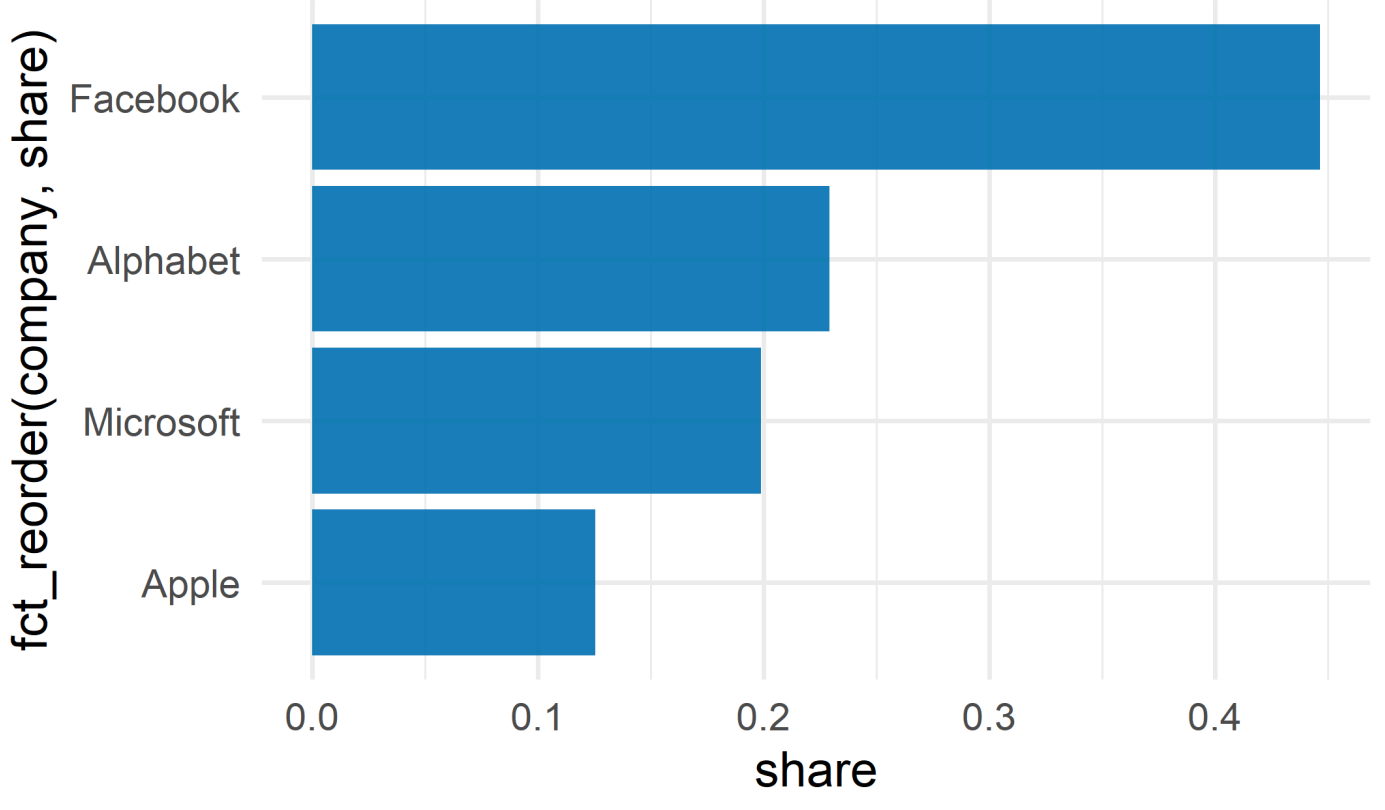
Bar plot

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

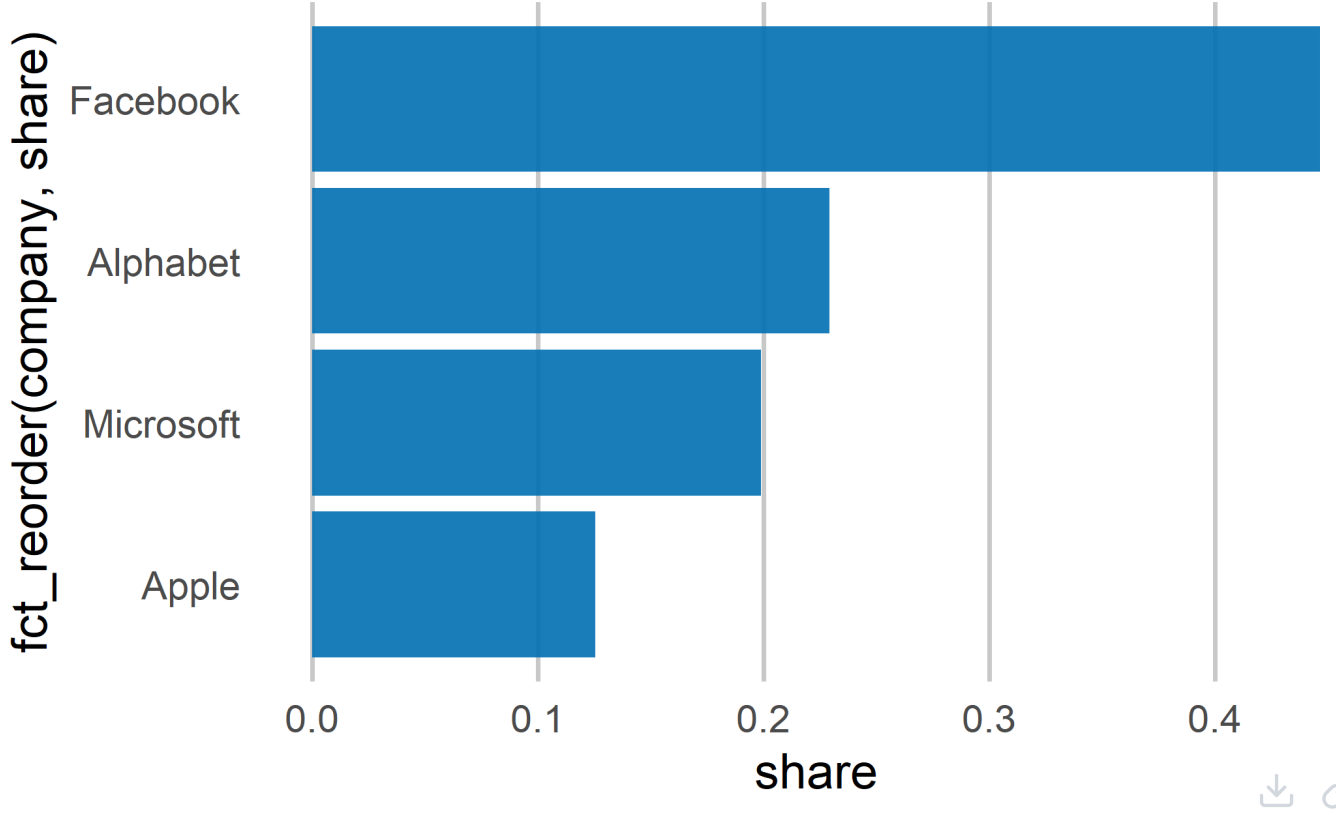


Horizontal

```
ggplot(avs, aes(share, fct_reorder(company, share))) +
  geom_col(fill = "#0072B2", alpha = 0.9)
```



```
ggplot(avs, aes(share, fct_reorder(company, share))) +
  geom_col(fill = "#0072B2", alpha = 0.9) +
  theme(
    panel.grid.major.y = element_blank(),
    panel.grid.minor.x = element_blank(),
    panel.grid.major.x = element_line(color = "gray80")
  )
```

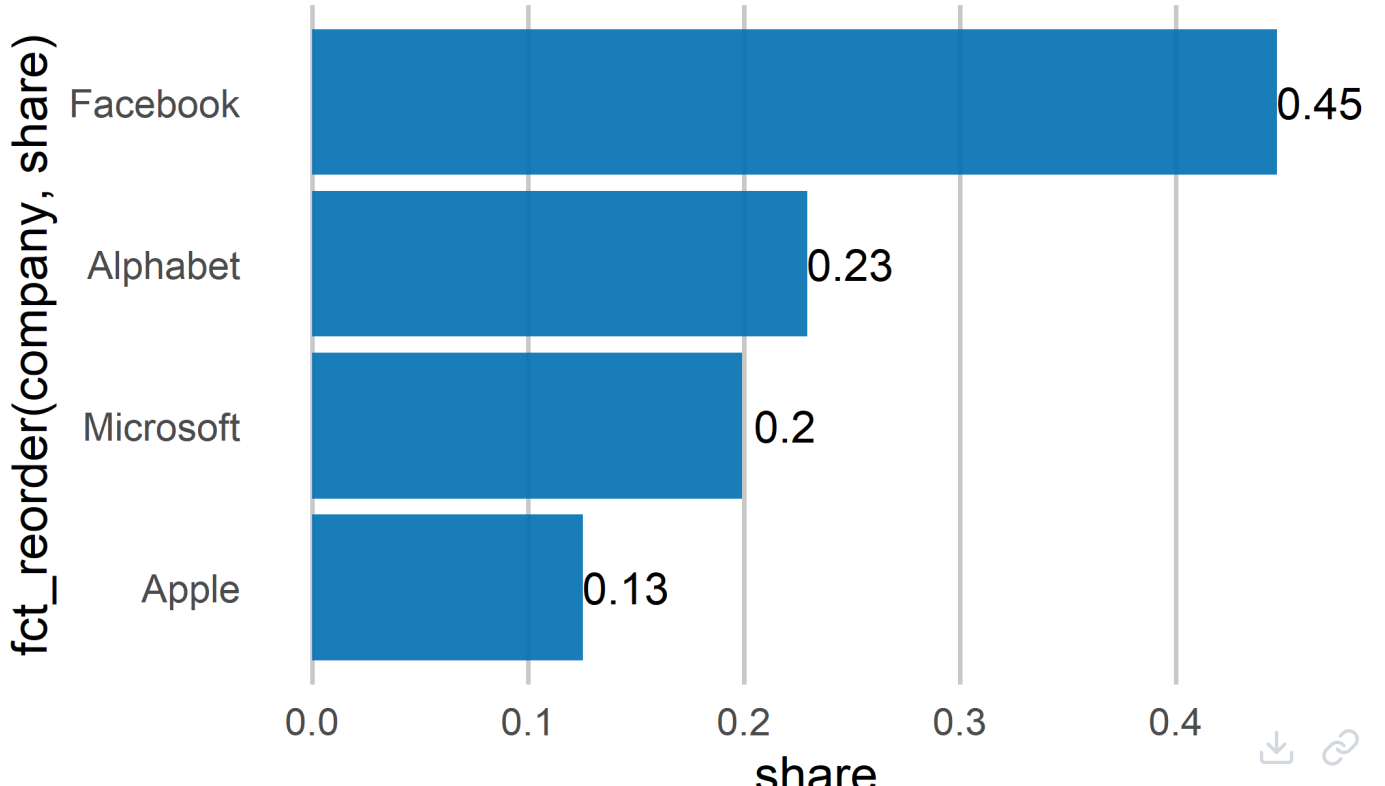


Quick aside

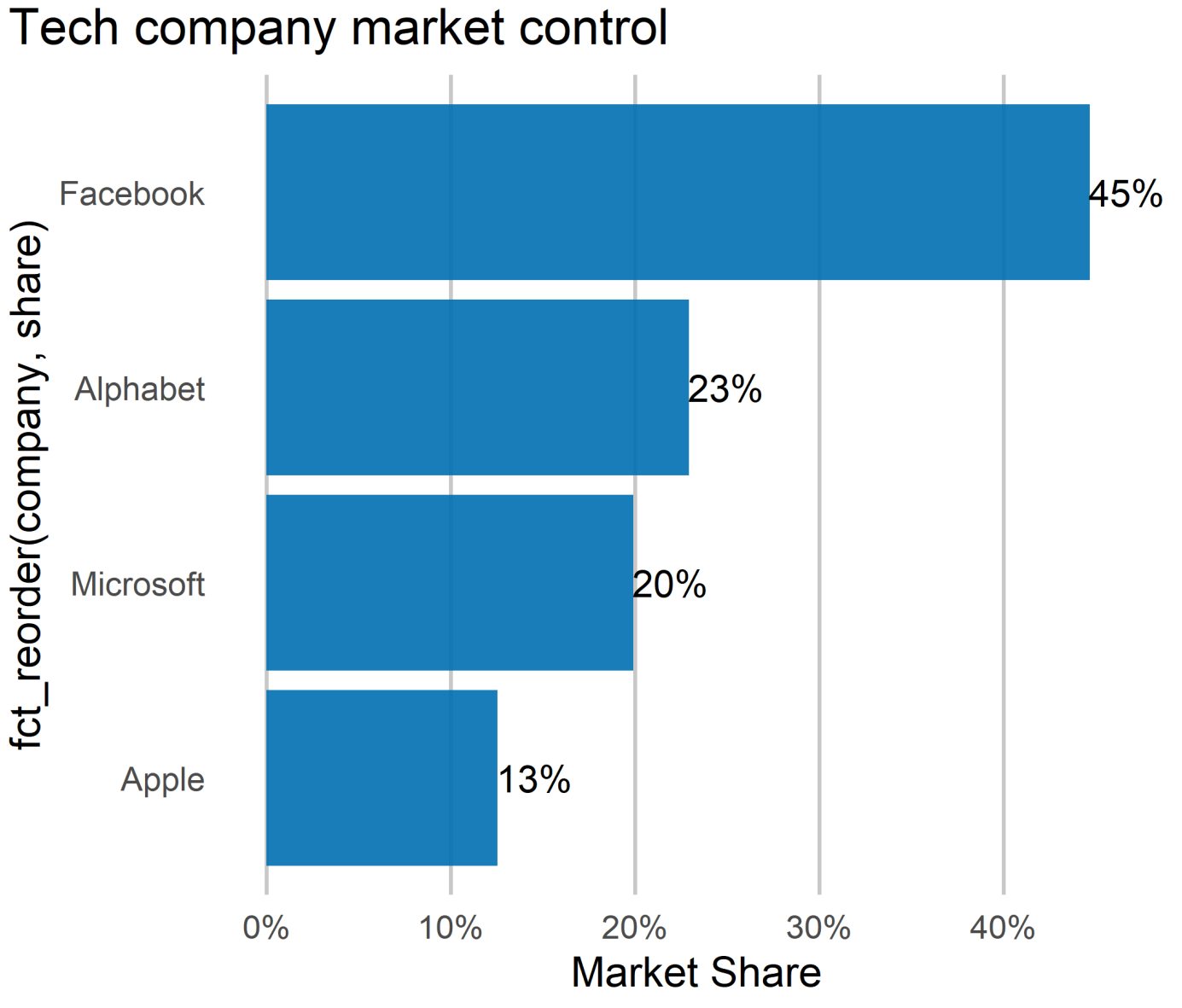
Let's actually make a bar plot theme

```
bp_theme <- function(...) {
  theme_minimal(...) +
    theme(
      panel.grid.major.y = element_blank(),
      panel.grid.minor.x = element_blank(),
      panel.grid.major.x = element_line(color = "gray80"),
      plot.title.position = "plot"
    )
}
```

```
ggplot(avs, aes(share, fct_reorder(company, share))) +
  geom_col(fill = "#0072B2",alpha = 0.9) +
  geom_text(
    aes(share, company, label = round(share, 2)),
    nudge_x = 0.02,
    size = 8
  ) +
  bp_theme(base_size = 25)
```

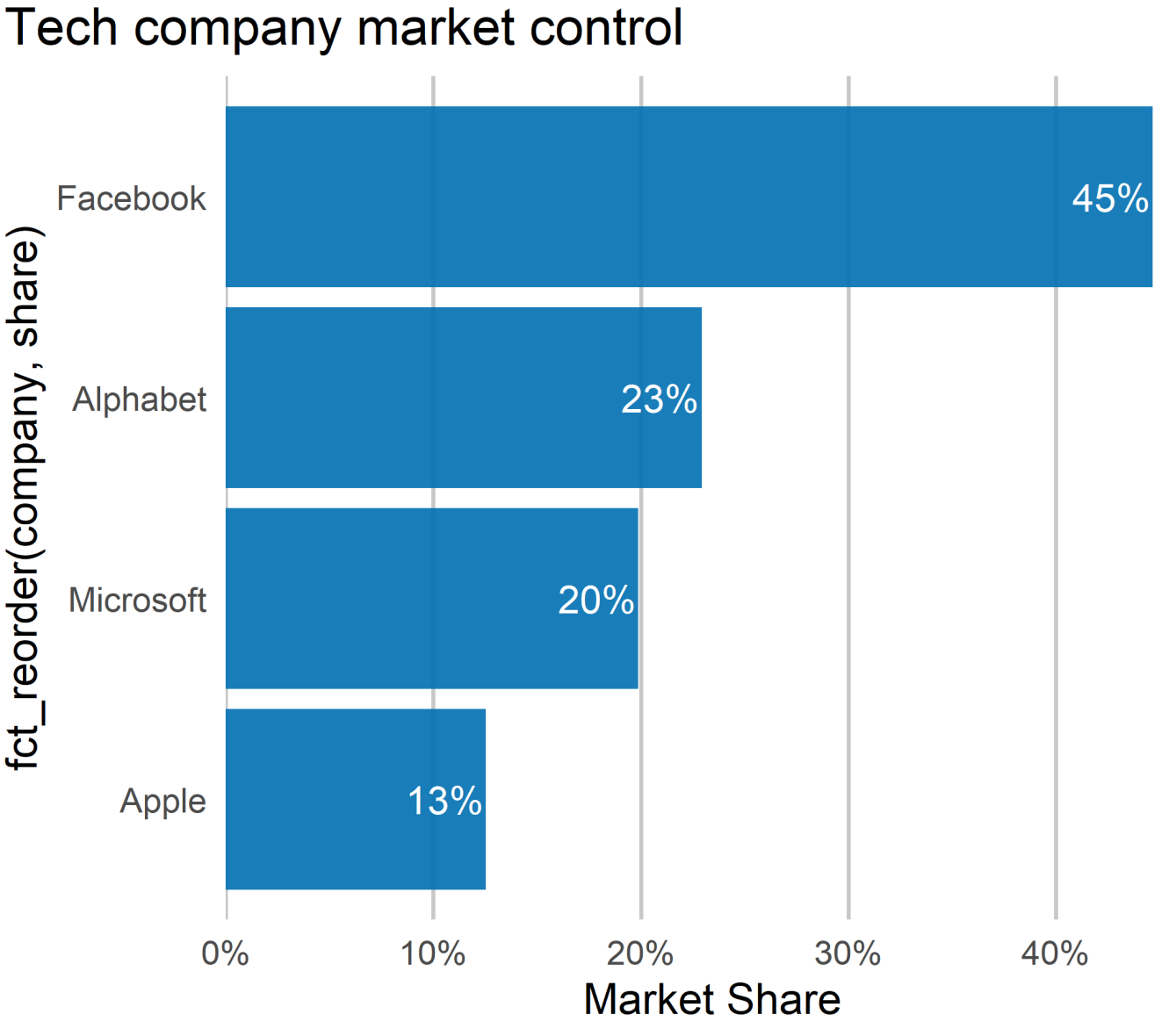


```
ggplot(avs, aes(share, fct_reorder(company, share))) +
  geom_col(fill = "#0072B2", alpha = 0.9) +
  geom_text(
    aes(share, company, label = paste0(round(share*100), "%")),
    nudge_x = 0.02,
    size = 8
  ) +
  scale_x_continuous(
    name = "Market Share",
    labels = scales::percent
  ) +
  labs(
    x = NULL,
    title = "Tech company market control",
    caption = "Data from Clause Wilke Book: Fundamentals of Data
  ) +
  bp_theme(base_size = 25)
```


Tech company market control
Facebook
Alphabet
Microsoft
Apple
fct_reorder(company, share)
45%
23%
20%
13%
0%
10%
20%
30%
40%
Market Share
Data from Clause Wilke Book: Fundamentals of Data Visualizations

Another alternative

```
ggplot(avs, aes(share, fct_reorder(company, share))) +
  geom_col(fill = "#0072B2", alpha = 0.9) +
  geom_text(
    aes(share, company, label = paste0(round(share*100), "%")),
    nudge_x = -0.02,
    size = 8,
    color = "white"
  ) +
  scale_x_continuous(
    "Market Share",
    labels = scales::percent,
    expand = c(0, 0, 0.05, 0)
  ) +
  labs(
    x = NULL,
    title = "Tech company market control",
    caption = "Data from Clause Wilke Book: Fundamentals of Data
  ) +
  bp_theme(base_size = 25)
```


Tech company market control
Facebook
Alphabet
Microsoft
Apple
45%
23%
20%
13%
fct_reorder(company, share)
0%
10%
20%
30%
40%
Market Share
Data from Clause Wilke Book: Fundamentals of Data Visualizations

Last example

This is a bit artificial in this case, but...

It is very common to have small bars. You may want most labels inside, but some outside

First, create variables specifying what you want.

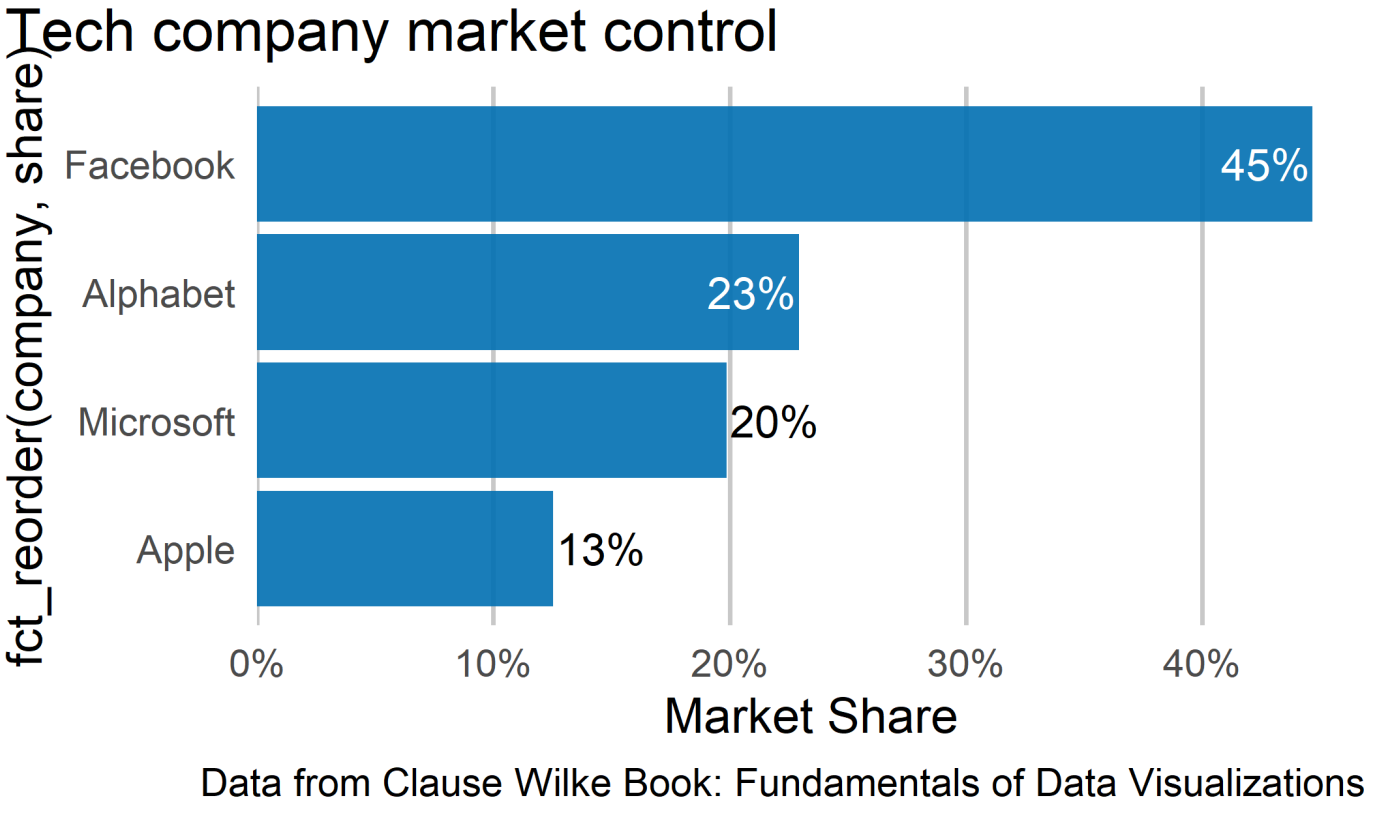
Here I'm using 0.2 as my cutoff for whether the label is inside the bar or outside

```
avs <- avs %>%
  mutate(
    nudge_amount = ifelse(share < 0.2, 0.02, -0.02),
    label_color = ifelse(share < 0.2, "black", "white")
  )
avs
```

```
## # A tibble: 4 × 5
##   company   stock_av share nudge_amount label_color
##   <chr>        <dbl> <dbl>        <dbl> <chr>
## 1 Alphabet     141.  0.229        -0.02 white
## 2 Apple         77.1 0.125         0.02 black
## 3 Facebook     275.  0.447        -0.02 white
## 4 Microsoft    122.  0.199         0.02 black
```

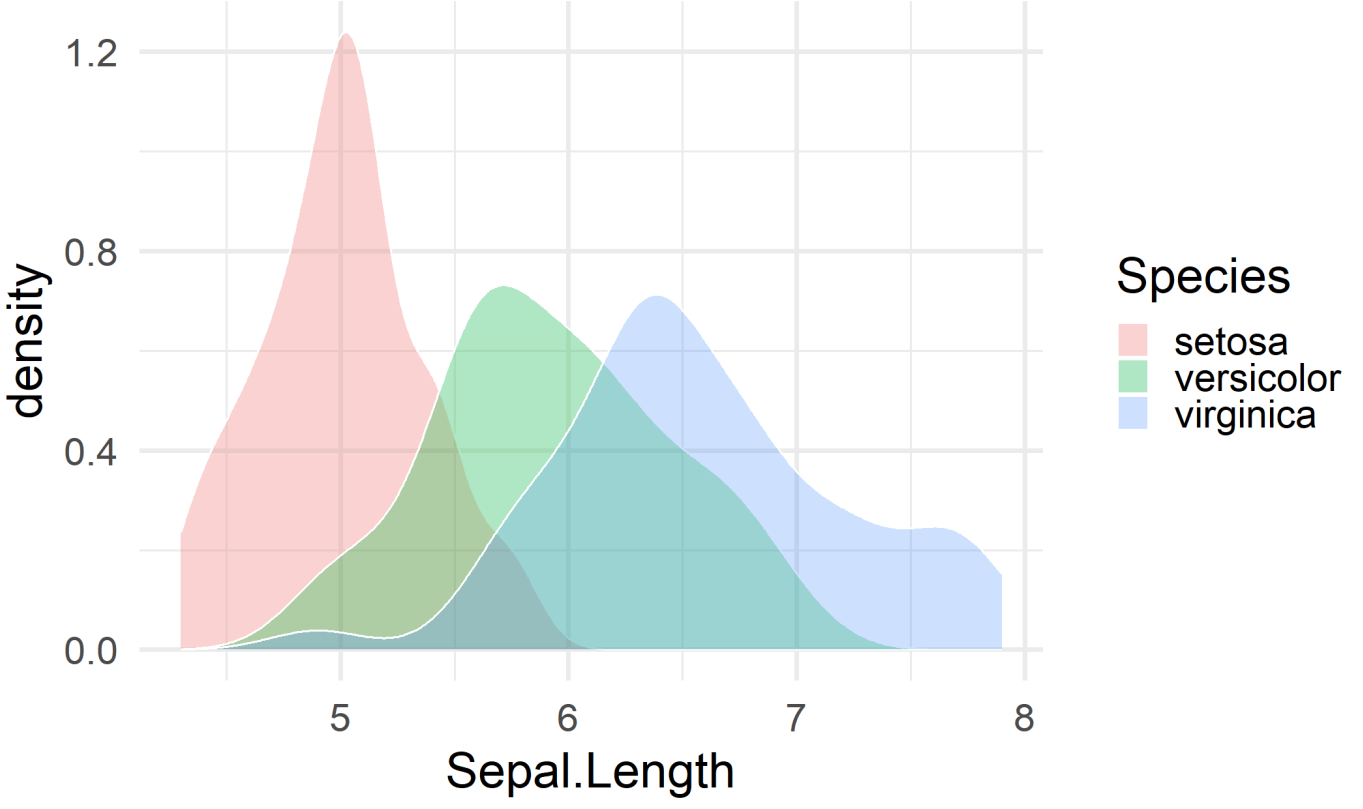
nudge_* doesn't work inside aes 🤷

```
ggplot(avs, aes(share, fct_reorder(company, share))) +
  geom_col(fill = "#0072B2", alpha = 0.9) +
  geom_text(
    aes(
      share,
      company,
      label = paste0(round(share*100), "%"),
      color = label_color
    ),
    nudge_x = avs$nudge_amount,
    size = 8,
  ) +
  scale_x_continuous(
    "Market Share",
    labels = scales::percent,
    expand = c(0, 0, 0.05, 0)
  ) +
  scale_color_identity() +
  labs(
    x = NULL,
    title = "Tech company market control",
    caption = "Data from Clause Wilke Book: Fundamentals of Data
  ) +
  bp_theme(base_size = 25)
```

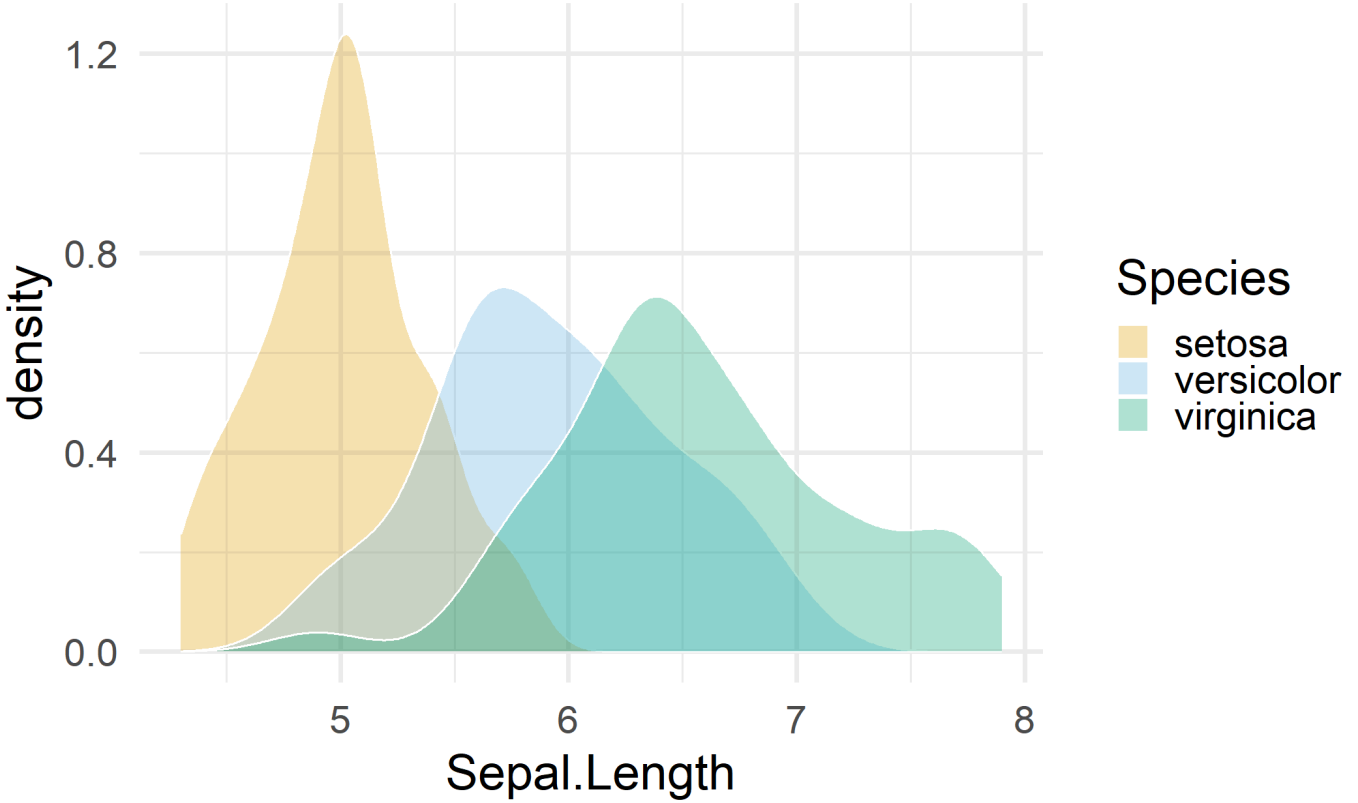

Tech company market control
fct_reorder(company, share)
Facebook
Alphabet
Microsoft
Apple
45%
23%
20%
13%
0%
10%
20%
30%
40%
Market Share
Data from Clause Wilke Book: Fundamentals of Data Visualizations

Distributions

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



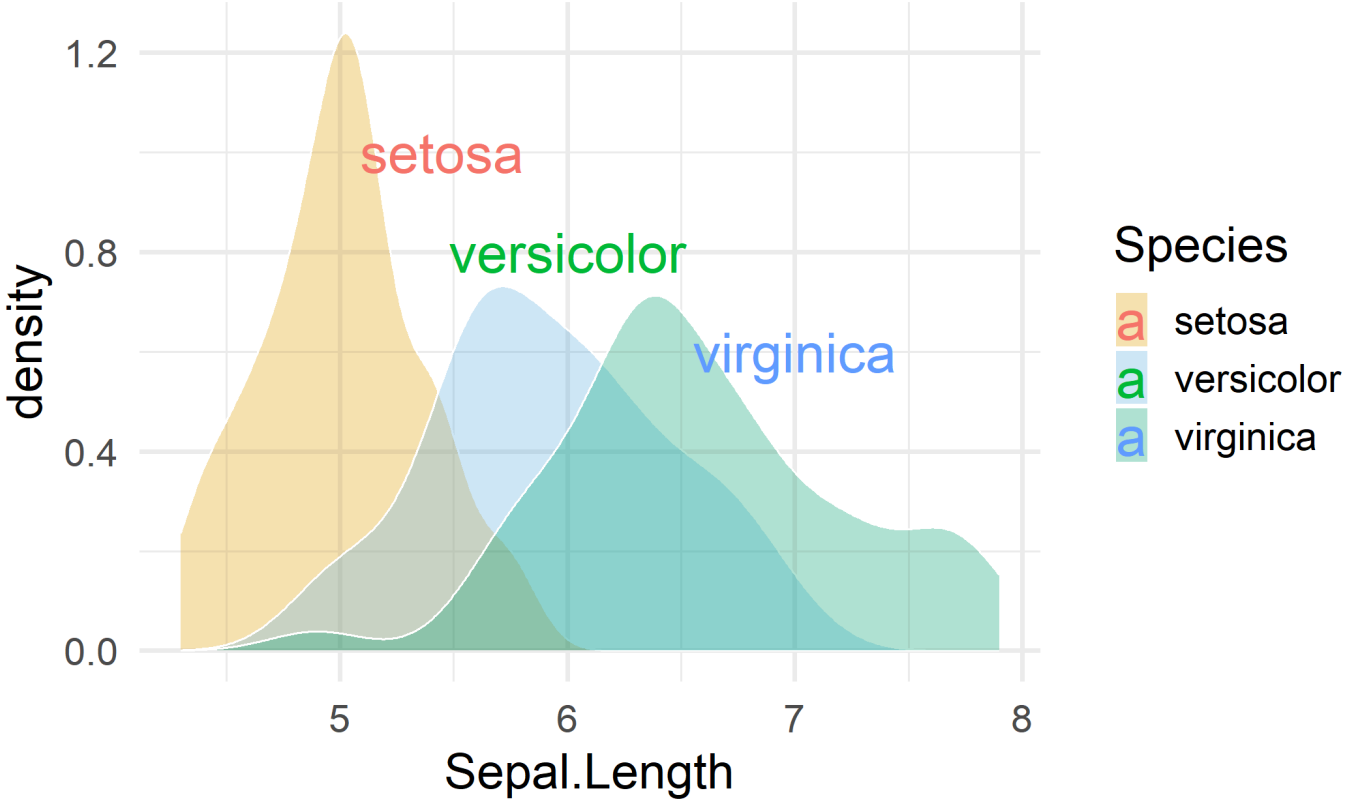
```
ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.3,
               color = "white") +
  scale_fill_OkabeIto()
```



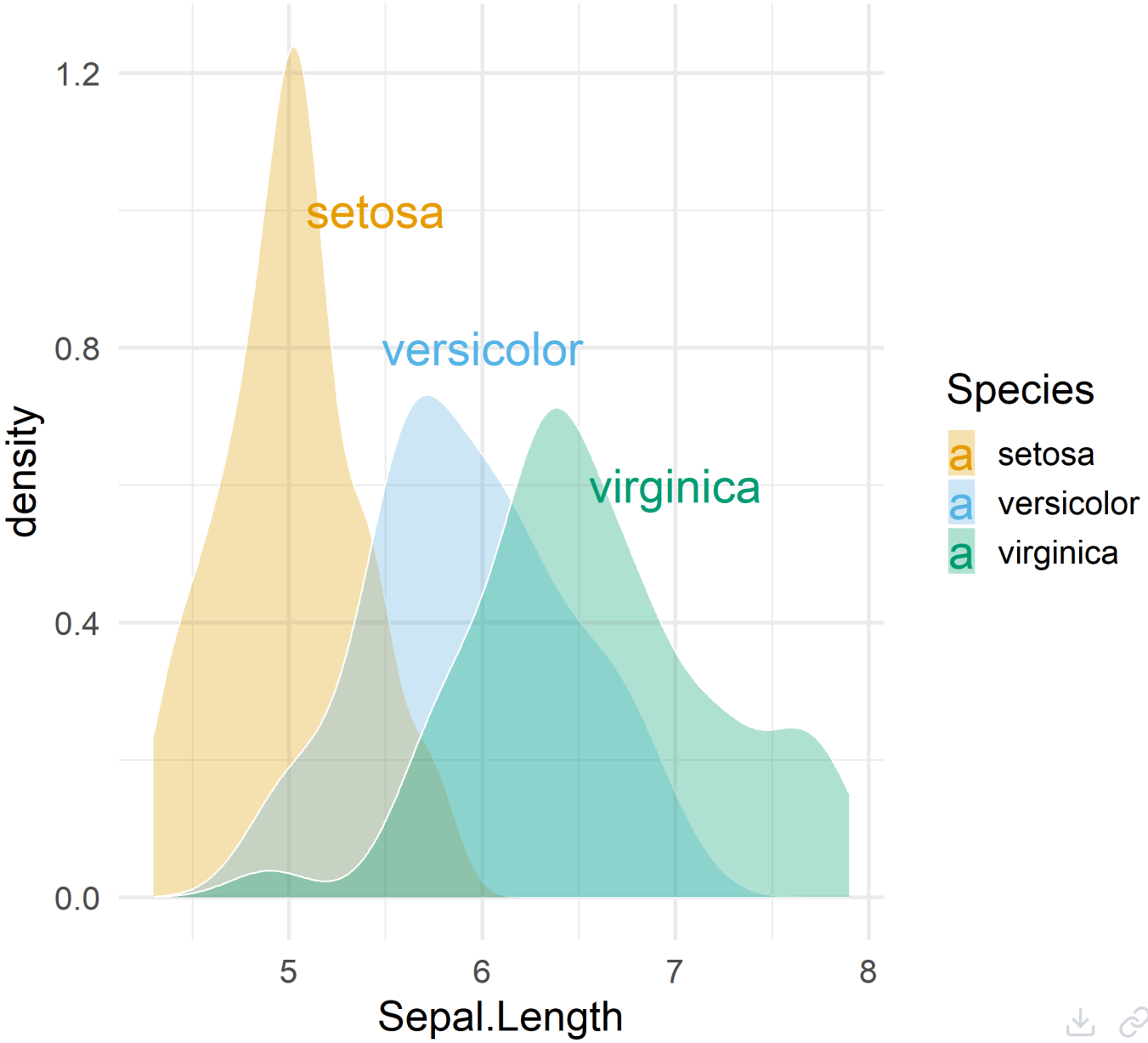
Labeling

One method

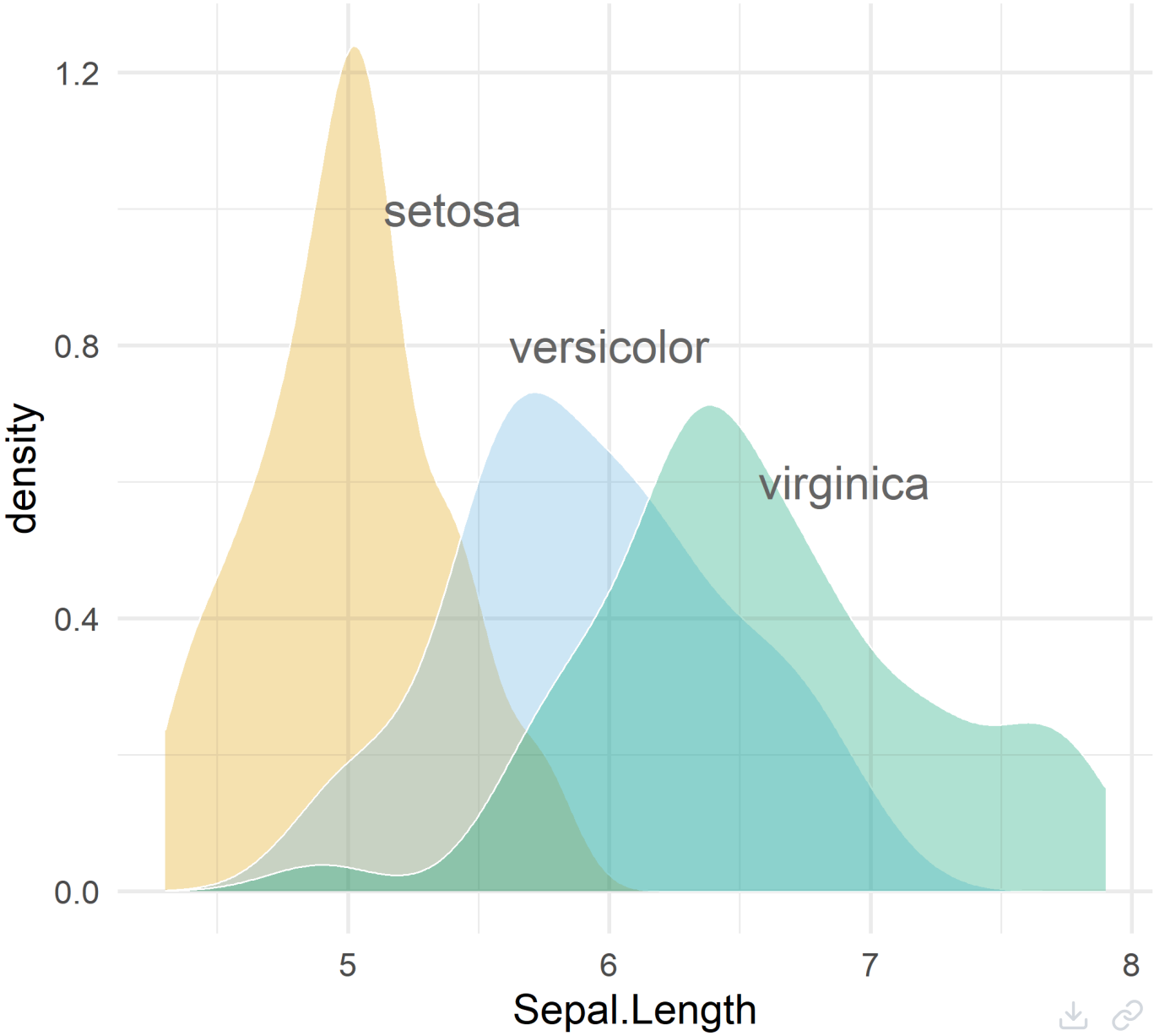
```
label_locs <- tibble(Sepal.Length = c(5.45, 6, 7),
                     density = c(1, 0.8, 0.6),
                     Species = c("setosa", "versicolor", "virgin

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


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

```
ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.3,
               color = "white") +
  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
setosa
versicolor
virginica
5
6
7
8
Sepal.Length
Species
a setosa
a versicolor
a virginica

```
label_locs <- tibble(Sepal.Length = c(5.4, 6, 6.9),
                     density = c(1, 0.75, 0.6),
                     Species = c("setosa", "versicolor", "virgin

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

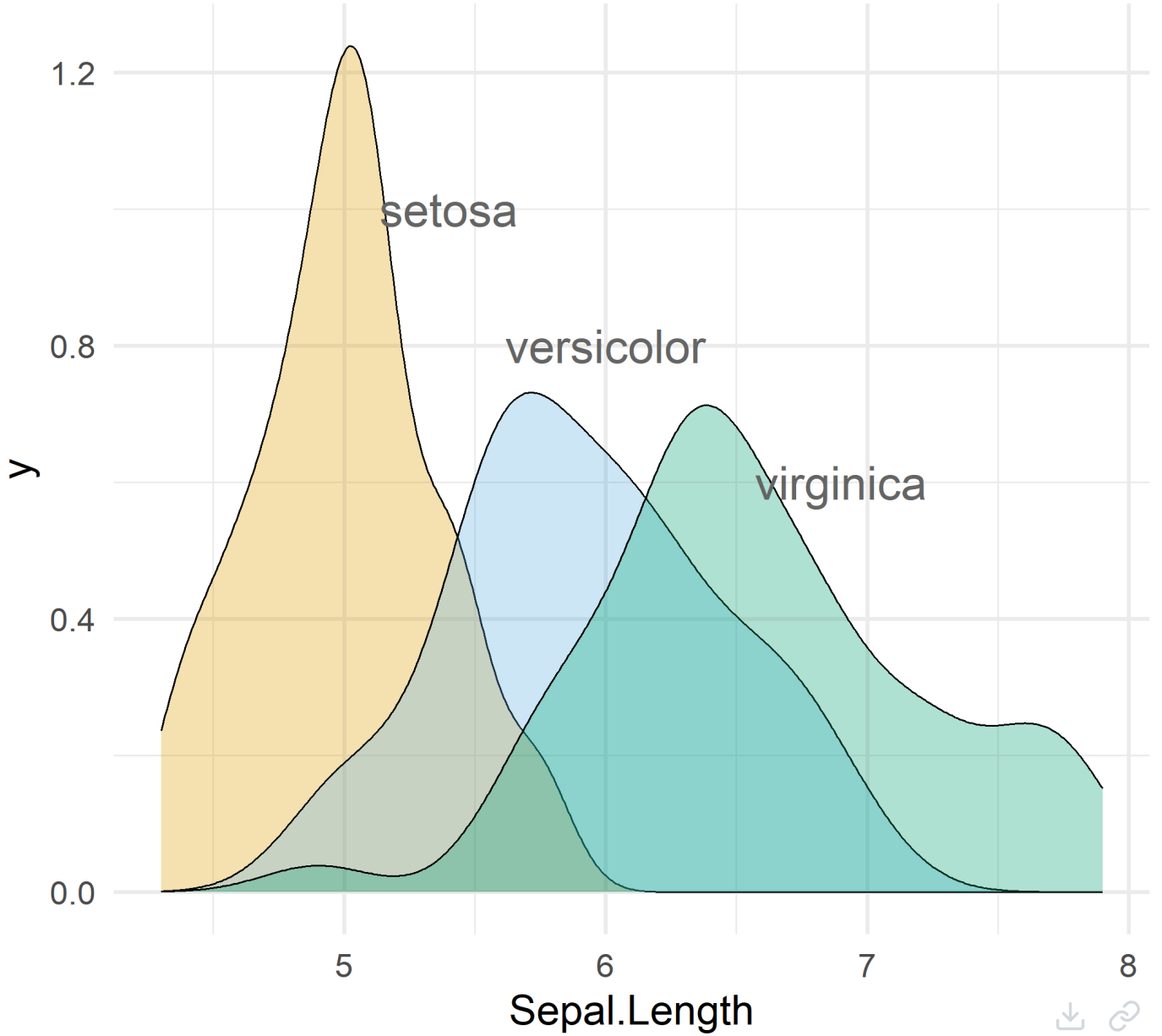

1.2
0.8
0.4
0.0
density
setosa
versicolor
virginica
5
6
7
8
Sepal.Length

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 = 5.45, y = 1, color = "g
  annotate("text", label = "versicolor", x = 6, y = 0.8, color =
  annotate("text", label = "virginica", x = 7, y = 0.6, color =
  guides(fill = "none")
```


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

Practice

Use the diamonds dataset again.

- Compute the mean carat size for each color
- Create a bar chart
- Include labels for each bar rounding the actual value to two decimals
- Make any other modifications you'd like

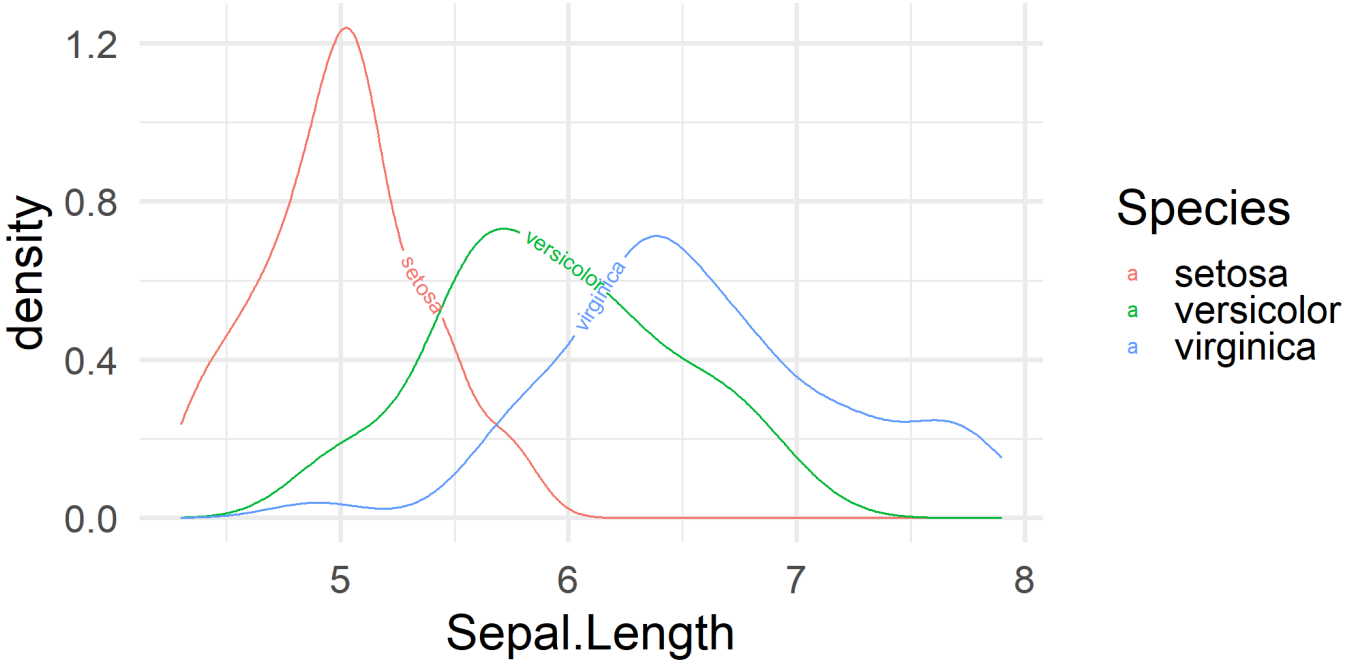
05:00

{geomtextpath

Not super new package, but looks really cool; still I haven't used much

{geomtextpath}

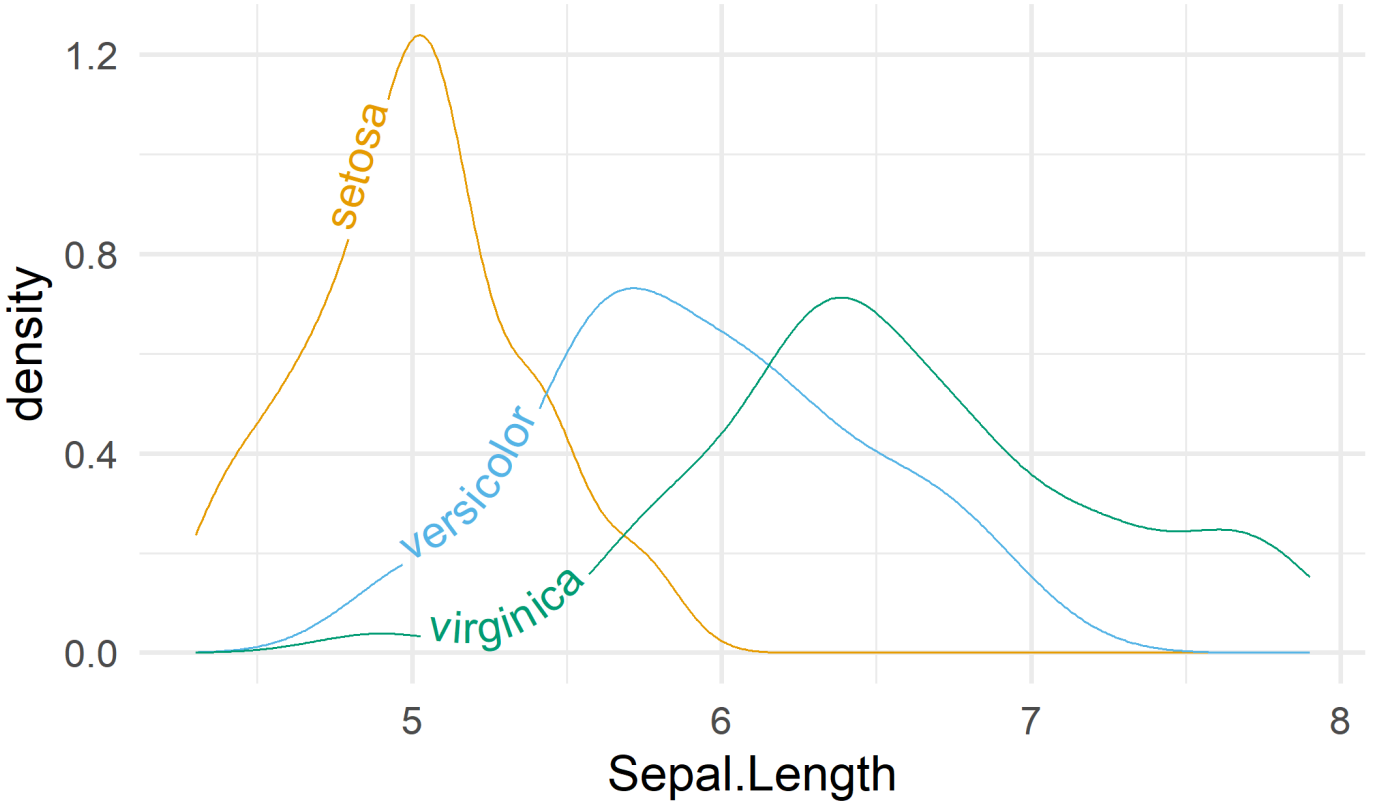
```
#install.packages("geomtextpath")
library(geomtextpath)

ggplot(iris, aes(Sepal.Length)) +
  geom_textdensity(aes(color = Species, label = Species))
```



Slight modifications

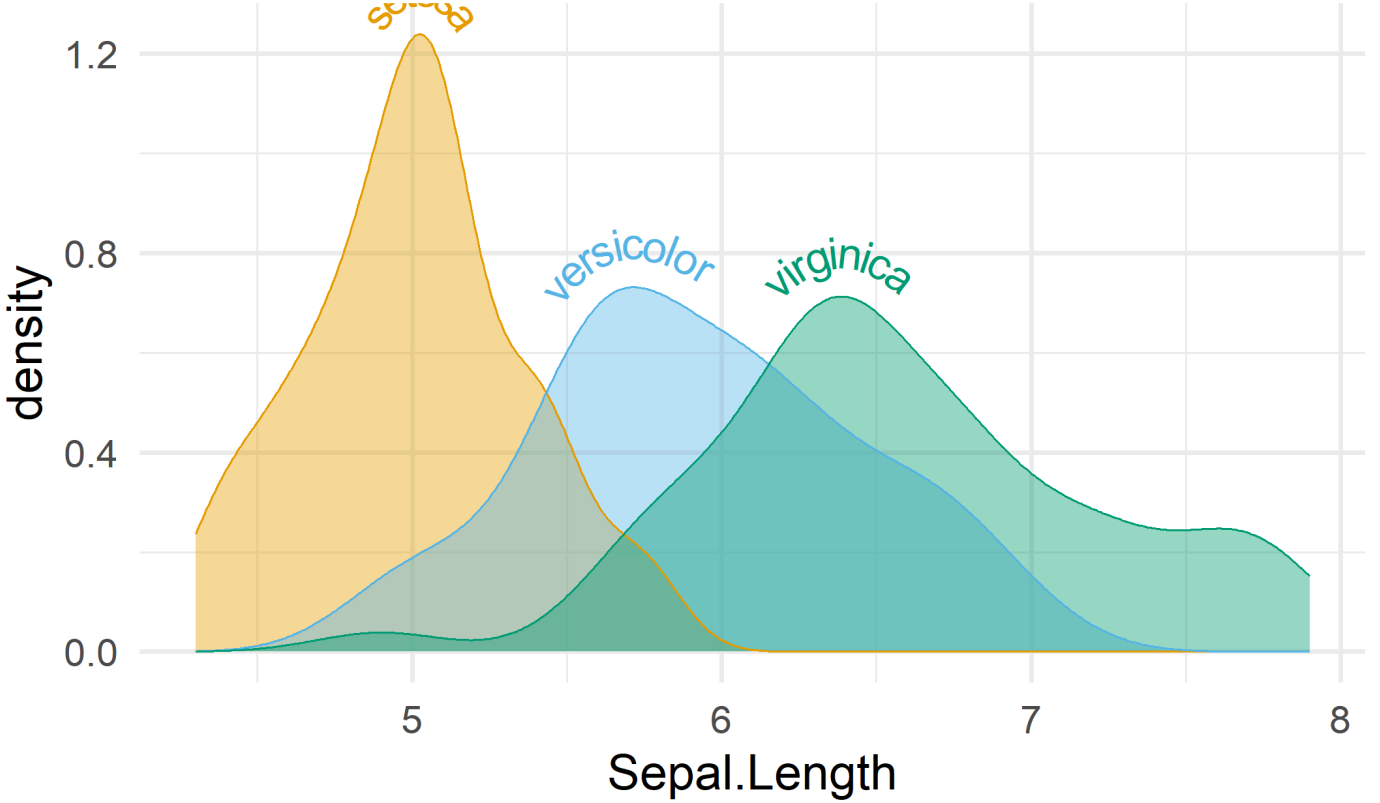
```
ggplot(iris, aes(Sepal.Length)) +
  geom_textdensity(
    aes(color = Species, label = Species),
    hjust = 0.2,
    vjust = 0.3,
    size = 8 # bigger than you'll probs need
  ) +
  scale_color_OkabeIto() +
  theme(legend.position = "none")
```

Note I couldn't get `aes(fill = Species)` to work.


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

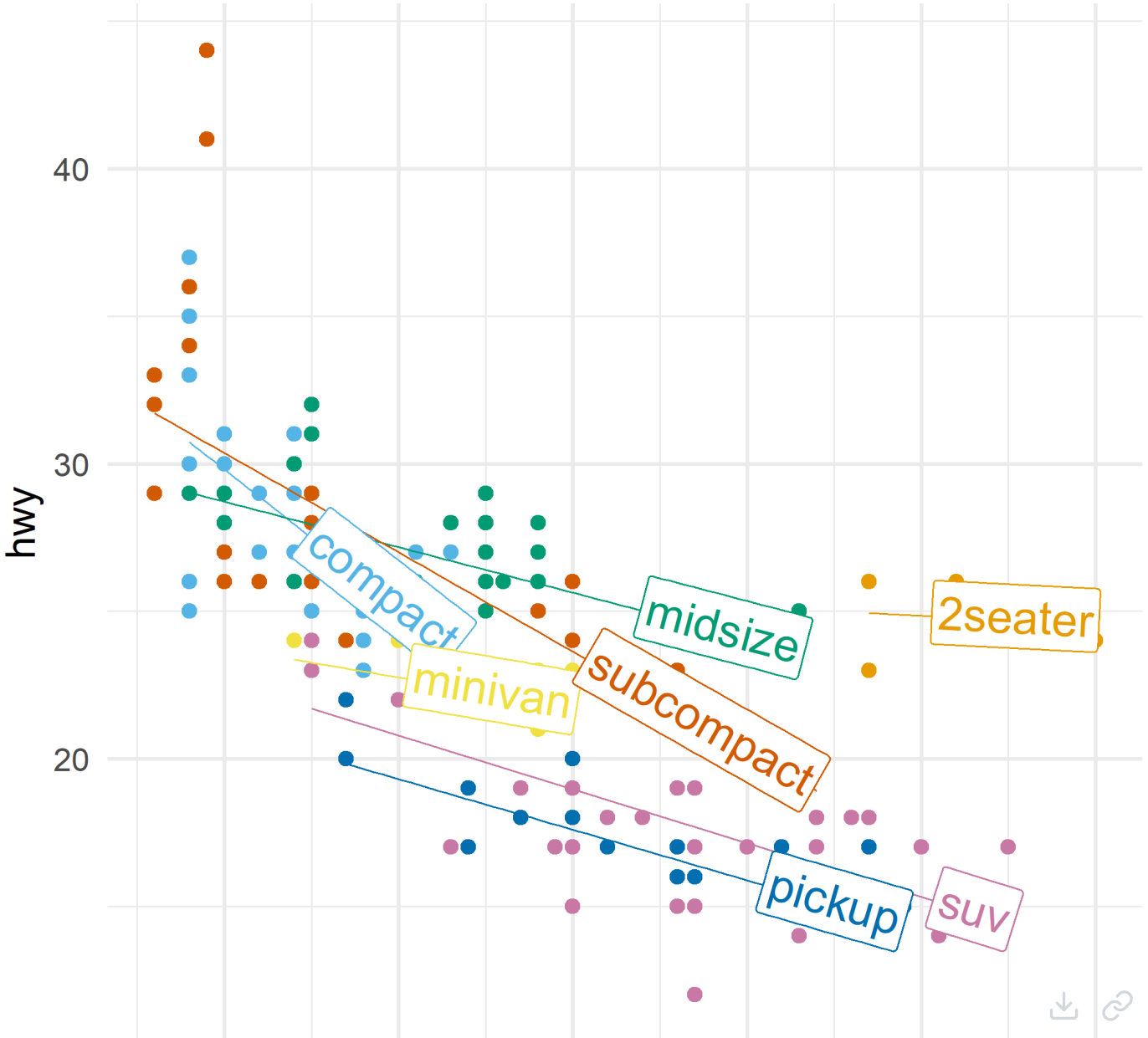
A workaround

```
ggplot(iris, aes(Sepal.Length)) +
  geom_density(aes(fill = Species), alpha = 0.4) +
  geom_textdensity(
    aes(color = Species, label = Species),
    hjust = "ymax",
    vjust = -0.3,
    size = 8
  ) +
  scale_color_OkabeIto() +
  scale_fill_OkabeIto() +
  theme(legend.position = "none")
```


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

Smooths

```
ggplot(mpg, aes(displ, hwy, color = class)) +
  geom_point() +
  geom_labelsmooth(
    aes(label = class),
    method = "lm",
    hjust = "xmax"
  ) +
  scale_color_OkabeIto() +
  guides(color = "none")
```

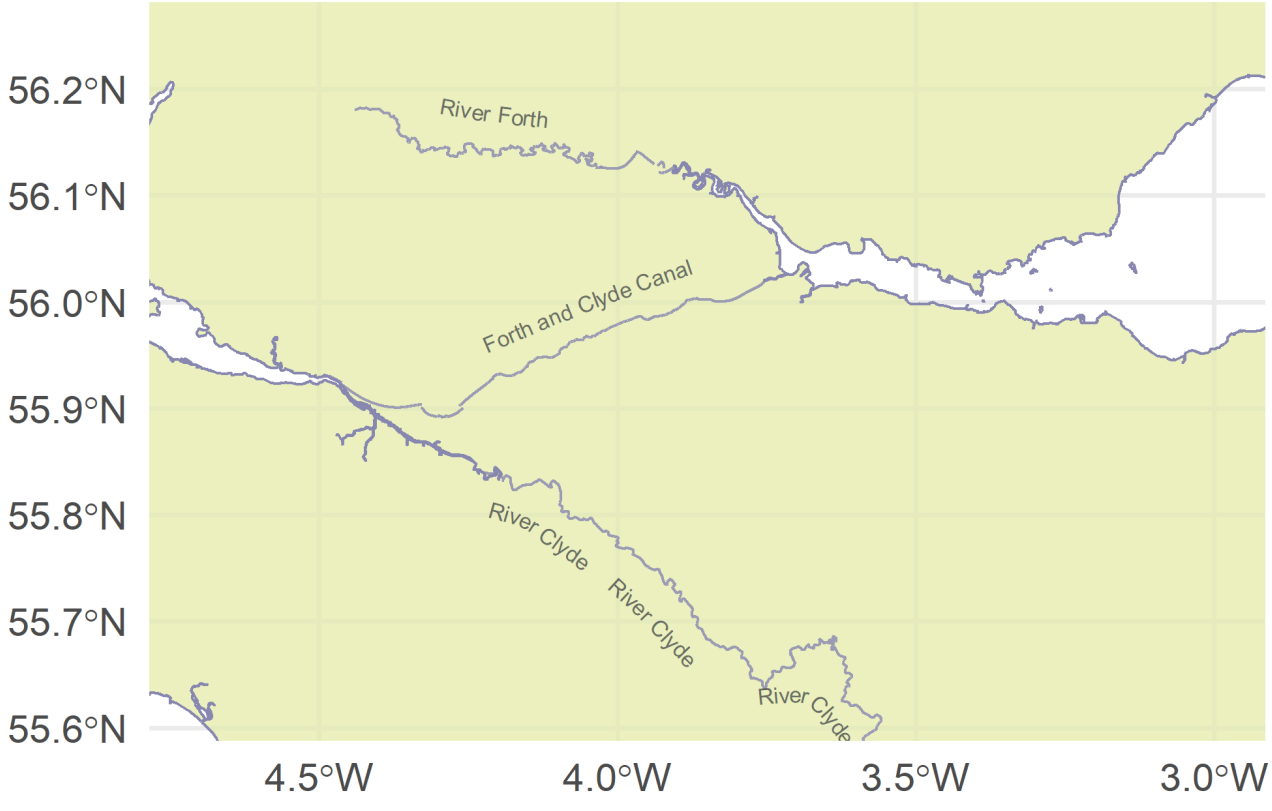

hwy
40
30
20
compact
midsize
2seater
minivan
subcompact
pickup
suv

Drop-in replacements

ggplot geom	Text equivalent	Label equivalent
`geom_path`	`geom_textpath`	`geom_labelpath`
`geom_segment`	`geom_textsegment`	`geom_labelsegment`
`geom_line`	`geom_textline`	`geom_labelline`
`geom_abline`	`geom_textabline`	`geom_labelabline`
`geom_hline`	`geom_texthline`	`geom_labelhline`
`geom_vline`	`geom_textvline`	`geom_labelvline`
`geom_curve`	`geom_textcurve`	`geom_labelcurve`
`geom_density`	`geom_textdensity`	`geom_labeldensity`
`geom_smooth`	`geom_textsmooth`	`geom_labelsmooth`

Even works w/Maps!

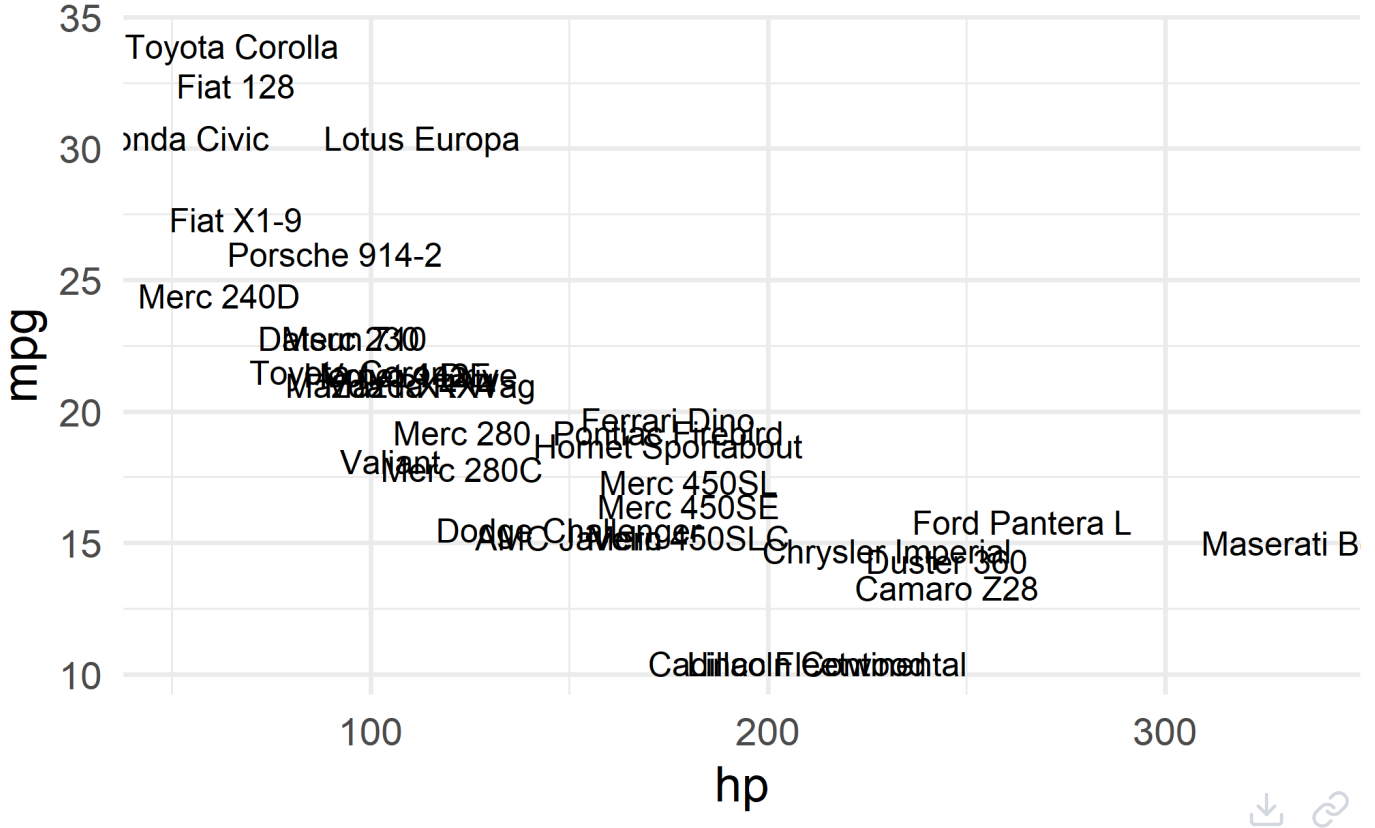
```
library(sf)

ggplot() +
  geom_textsf(
    data = waterways,
    aes(label = name),
    text_smoothing = 65,
    linecolour = "#8888B3",
    vjust = -0.8,
    fill = "#E6F0B3",
    alpha = 0.8,
    size = 4
  ) +
  theme(panel.grid = element_line()) +
  lims(x = c(-4.7, -3), y = c(55.62, 56.25))
```


56.2°N
56.1°N
56.0°N
55.9°N
55.8°N
55.7°N
55.6°N
4.5°W
4.0°W
3.5°W
3.0°W
River Forth
Forth and Clyde Canal
River Clyde
River Clyde
River Clyde

ggrepel

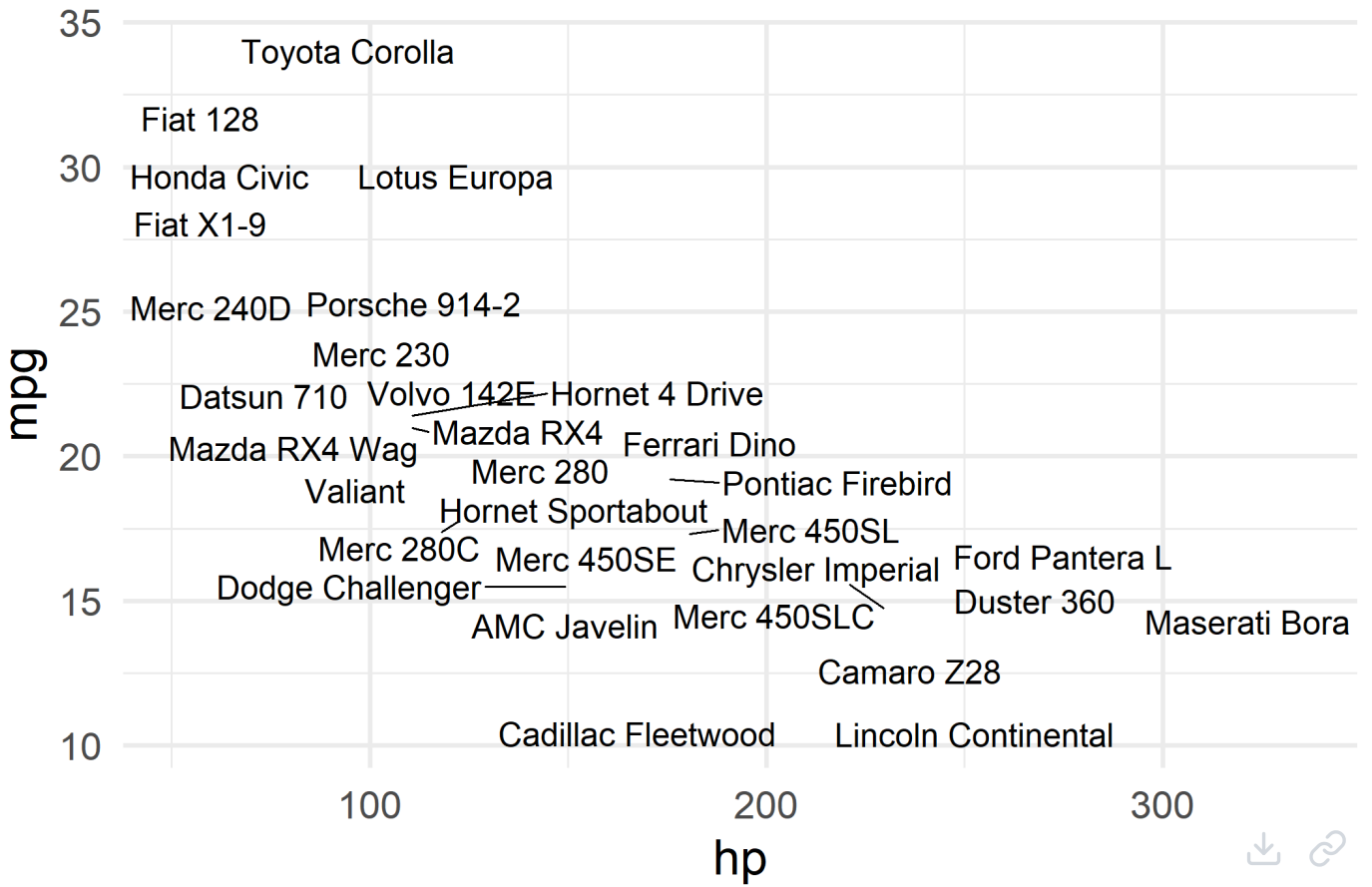
Plot text directly

```
cars <- rownames_to_column(mtcars)

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



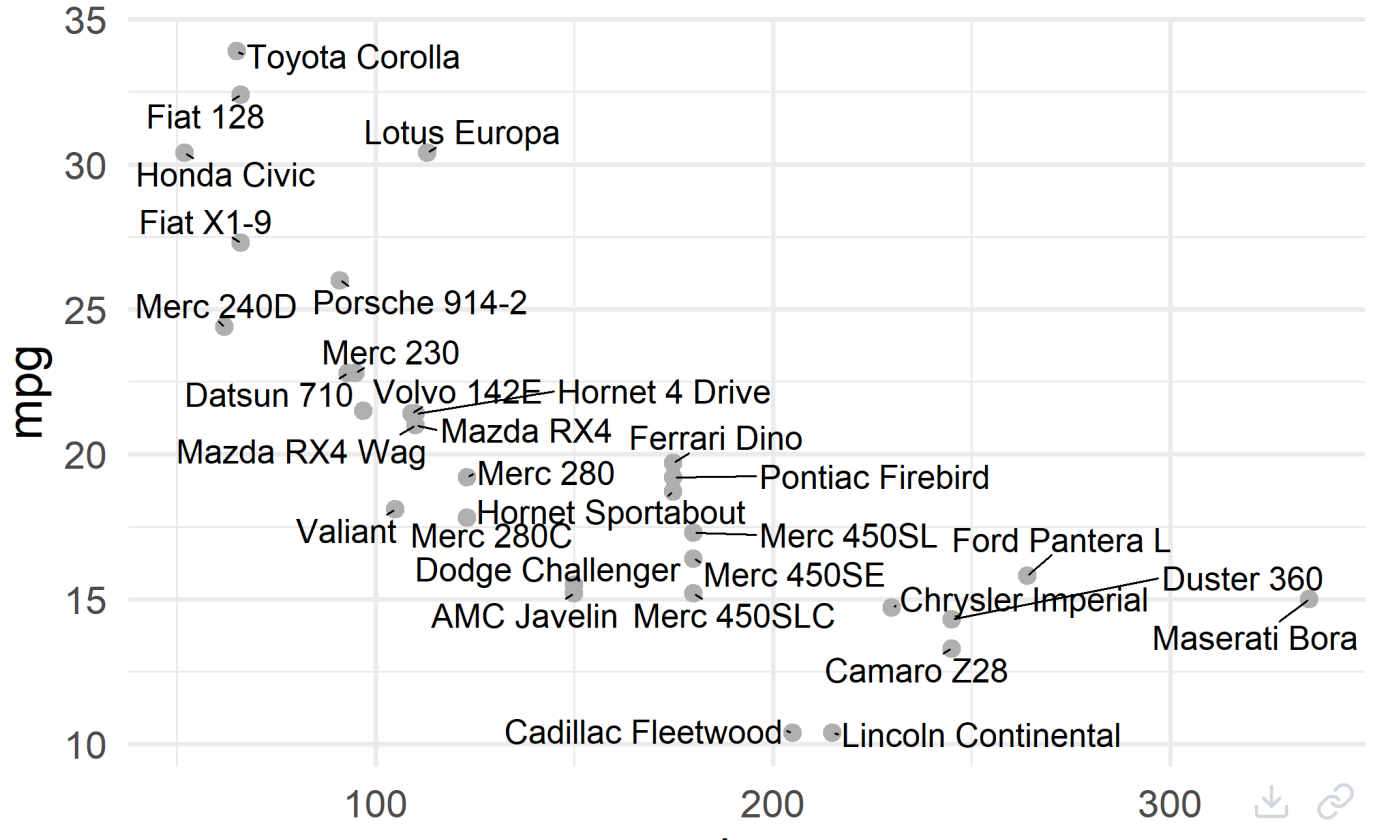
Repel text

```
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

Please follow along

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

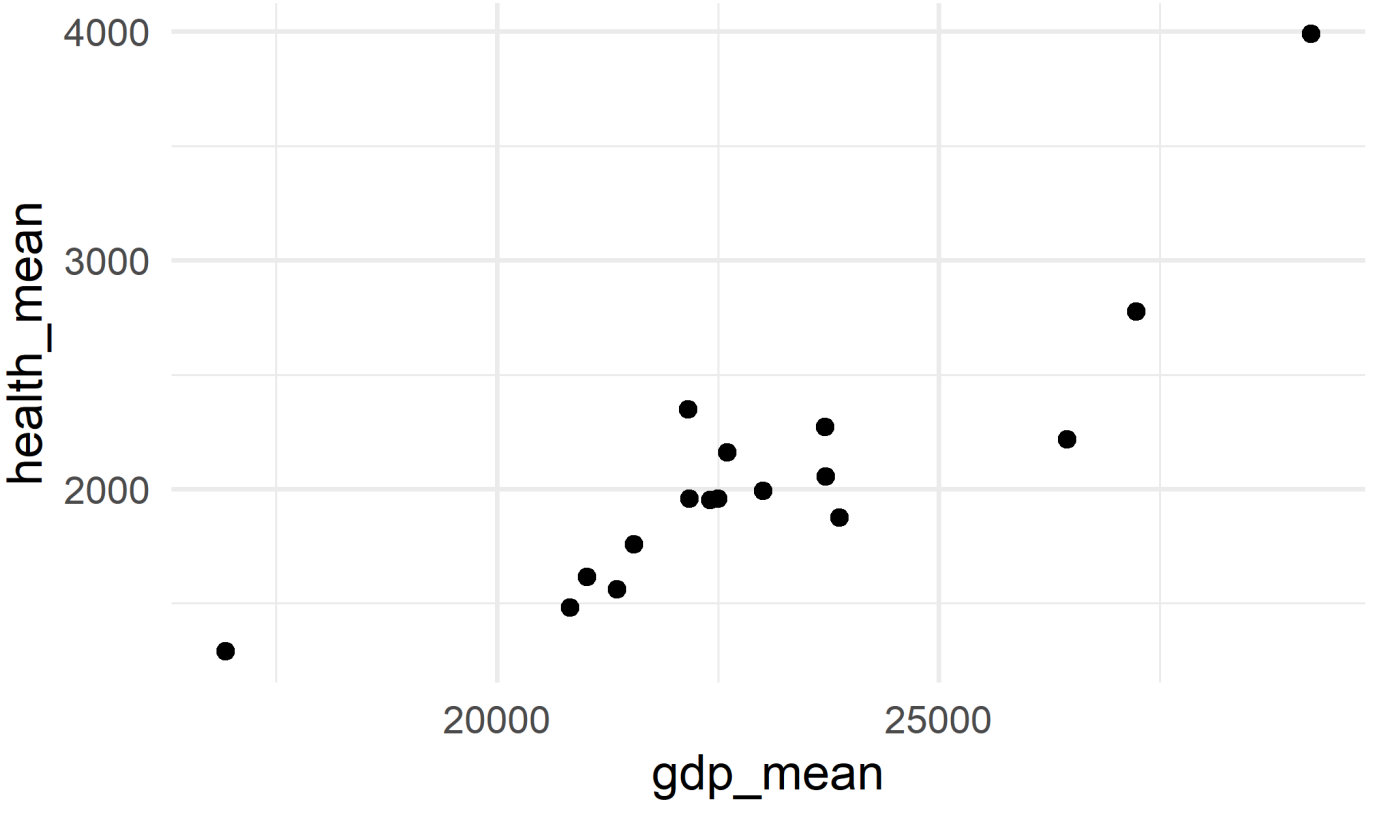
```
by_country <- organdata %>%
  group_by(consent_law, country) %>%
  summarize(donors_mean= mean(donors, na.rm = TRUE),
            donors_sd = sd(donors, na.rm = TRUE),
            gdp_mean = mean(gdp, na.rm = TRUE),
            health_mean = mean(health, na.rm = TRUE),
            roads_mean = mean(roads,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

```
by_country
```

```
## # A tibble: 17 × 8
## # Groups:   consent_law [2]
##   consent_law country     donors_mean donors_sd gdp_mean health_mean roads_mean
##   <chr>       <chr>             <dbl>     <dbl>    <dbl>       <dbl>      <dbl>
## 1 Informed    Australia          10.6     1.14    22179.       1958.      105.
## 2 Informed    Canada             14.0     0.751   23711.       2272.      109.
## 3 Informed    Denmark            13.1     1.47    23722.       2054.      102.
## 4 Informed    Germany            13.0     0.611   22163.       2349.      113.
## 5 Informed    Ireland            19.8     2.48    20824.       1480.      118.
## 6 Informed    Netherlands        13.7     1.55    23013.       1993.       76.1
## # ℹ 11 more rows
```

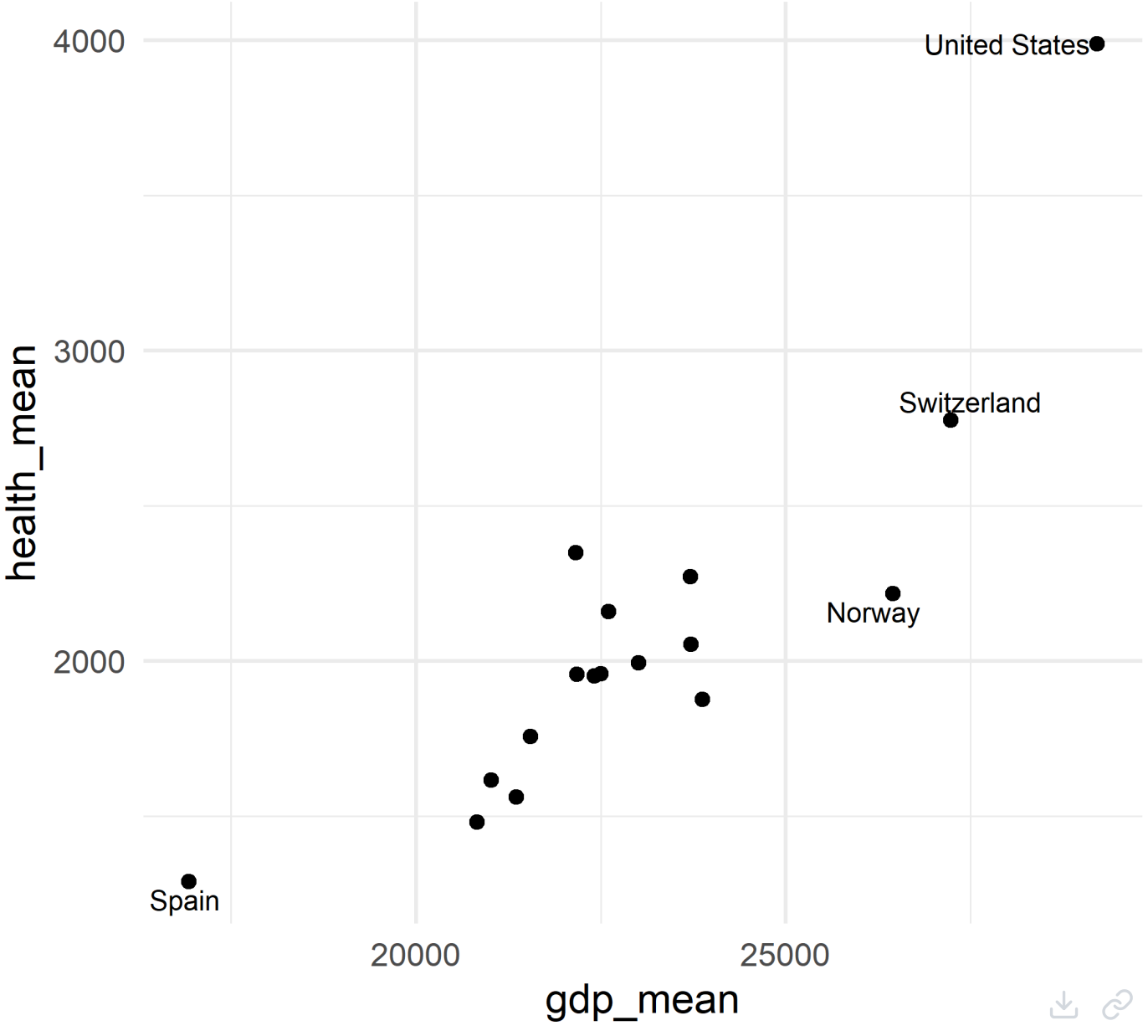
Scatterplot

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



Outliers

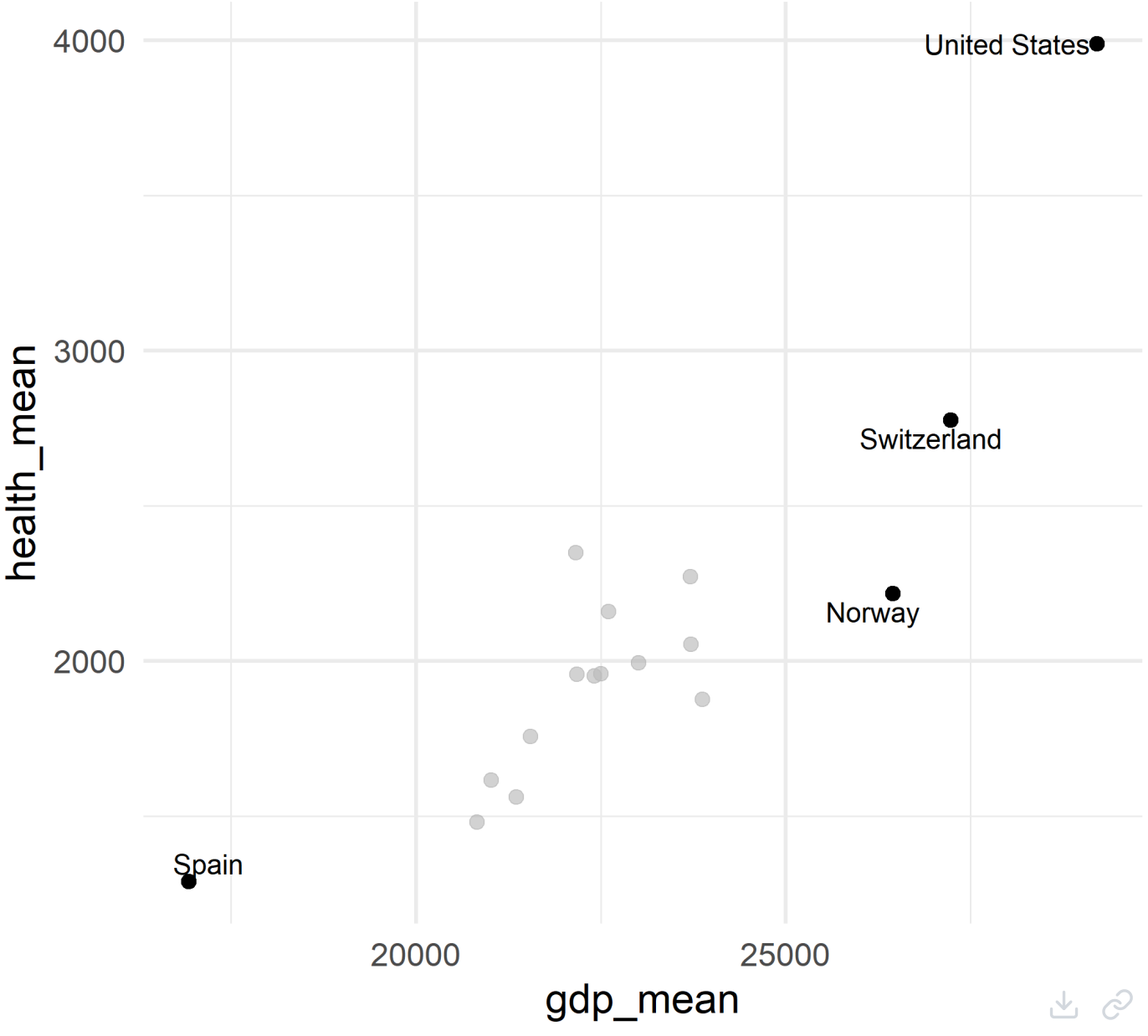
```
ggplot(by_country, aes(gdp_mean, health_mean)) +
  geom_point() +
  geom_text_repel(data = filter(by_country,
                                gdp_mean > 25000 |
                                gdp_mean < 20000),
                  aes(label = country))
```


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

Combine with highlighting

```
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

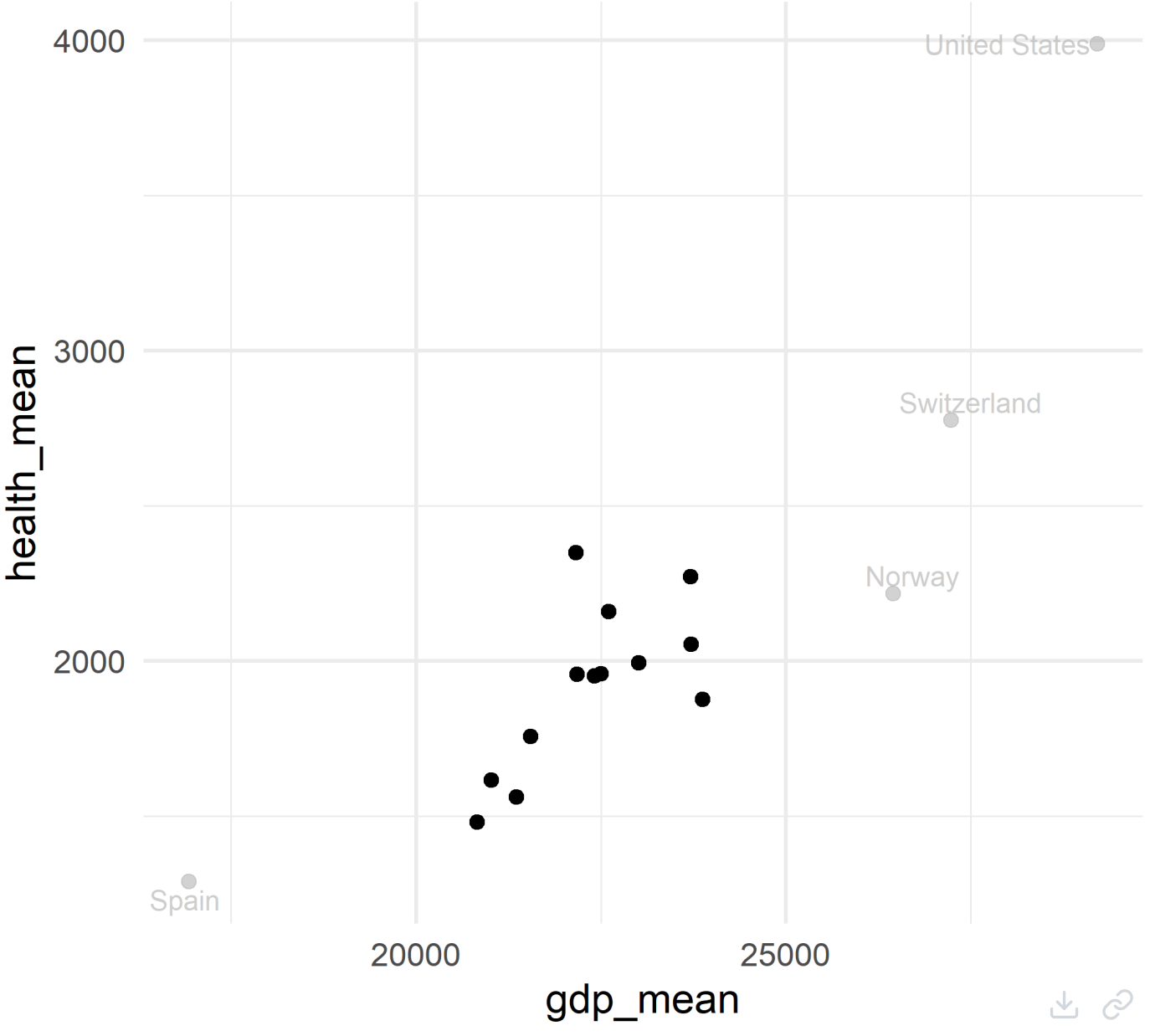

4000
3000
2000
health_mean
United States
Switzerland
Norway
Spain
20000
25000
gdp_mean

Combine with highlighting

Switch to make outliers grayed out and labeled

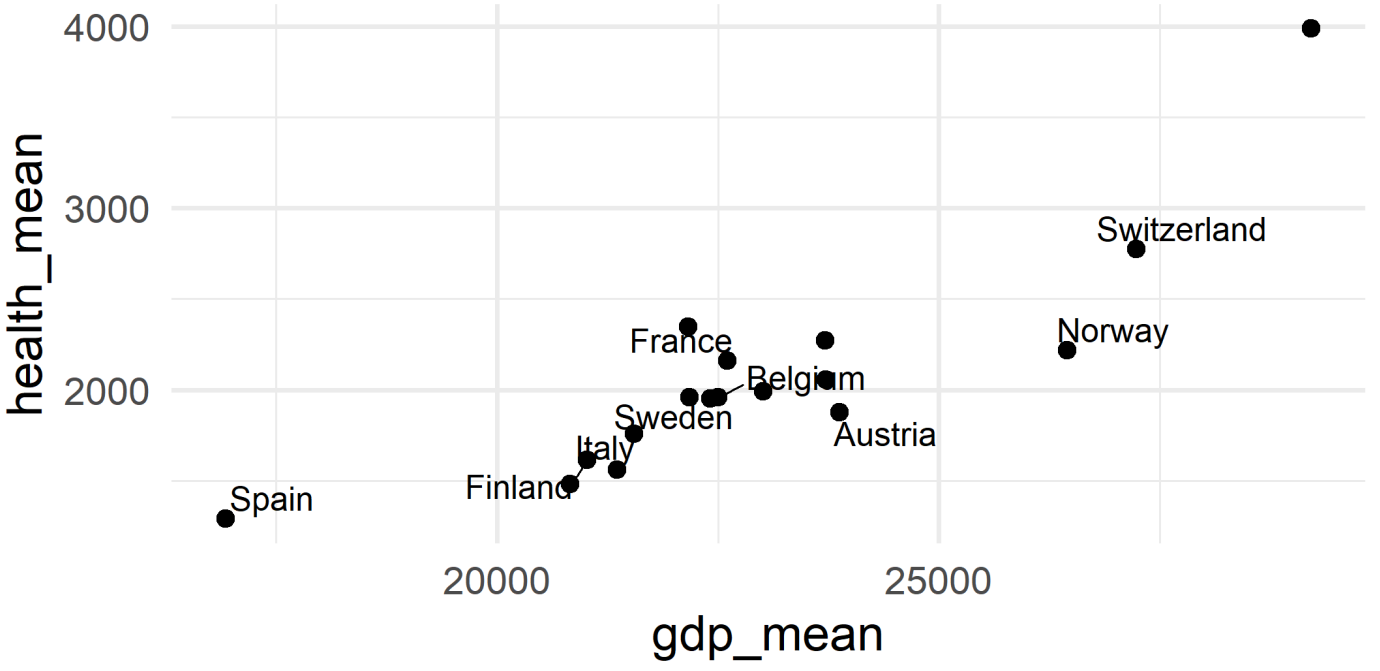
```
ggplot(by_country, aes(gdp_mean, health_mean)) +
  geom_point() +
  gghighlight(gdp_mean > 20000 & gdp_mean < 25000 ) +
  geom_text_repel(data = filter(by_country,
                                gdp_mean > 25000 |
                                gdp_mean < 20000),
                  aes(label = country),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

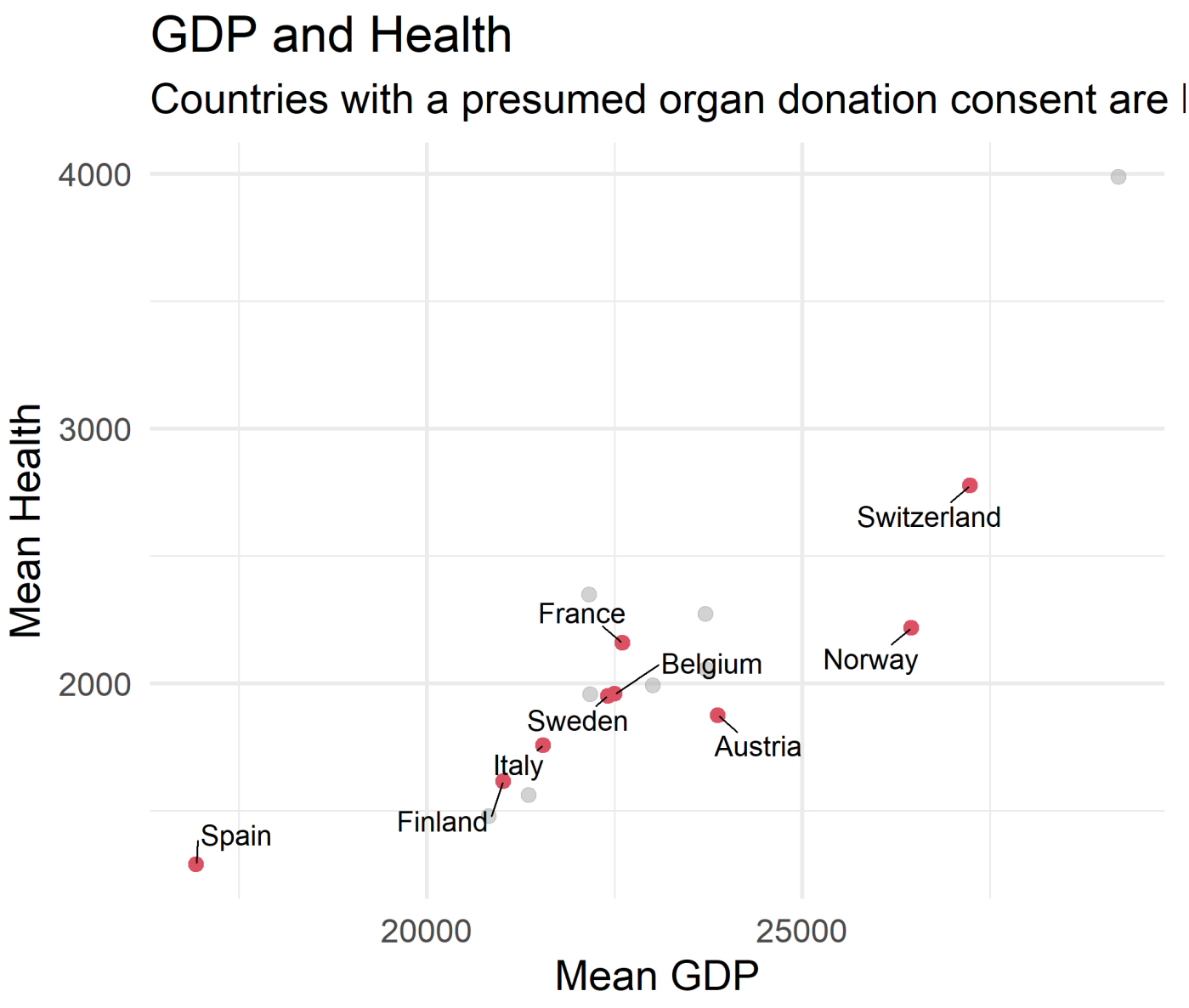
By group

```
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 = 0.75) +
  labs(title = "GDP and Health",
         subtitle = "Countries with a presumed organ donation co
         caption = "Data from the General Social Science Survey,
         x = "Mean GDP",
         y = "Mean Health")
```



om the General Social Science Survey, Distributed through the socviz R package

Practice

Use the mpg dataset

- Group by manufacturer
- Compute the mean highway `hwy` and mean `displ`
- Plot the relation between these means. Plot points and label the manufacturer of each point.

ggforce

Please follow along

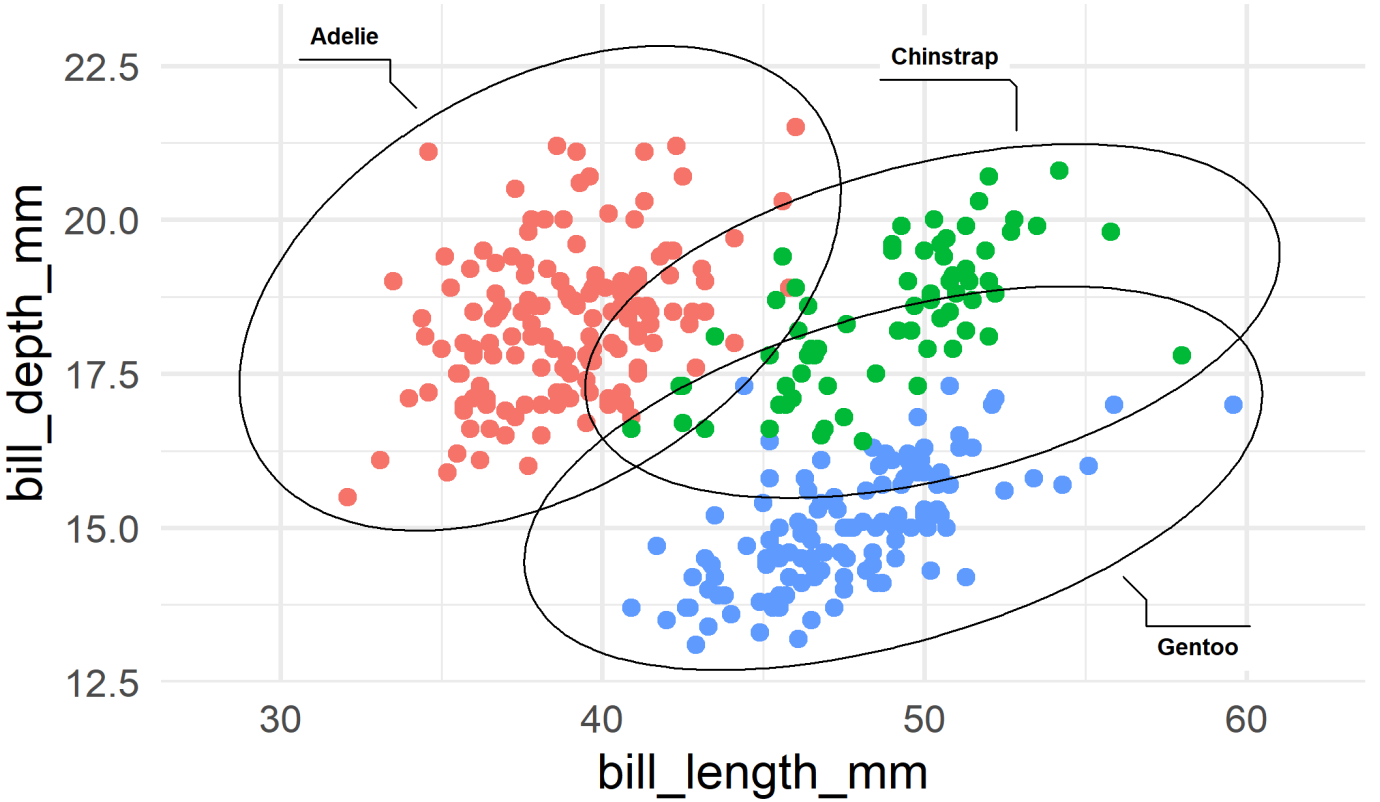
Annotating groups of points

Consider using any of the following from **ggforce** to annotate specific points

- `geom_mark_rect()`
- `geom_mark_circle()`
- `geom_mark_ellipse()`
- `geom_mark_hull()`

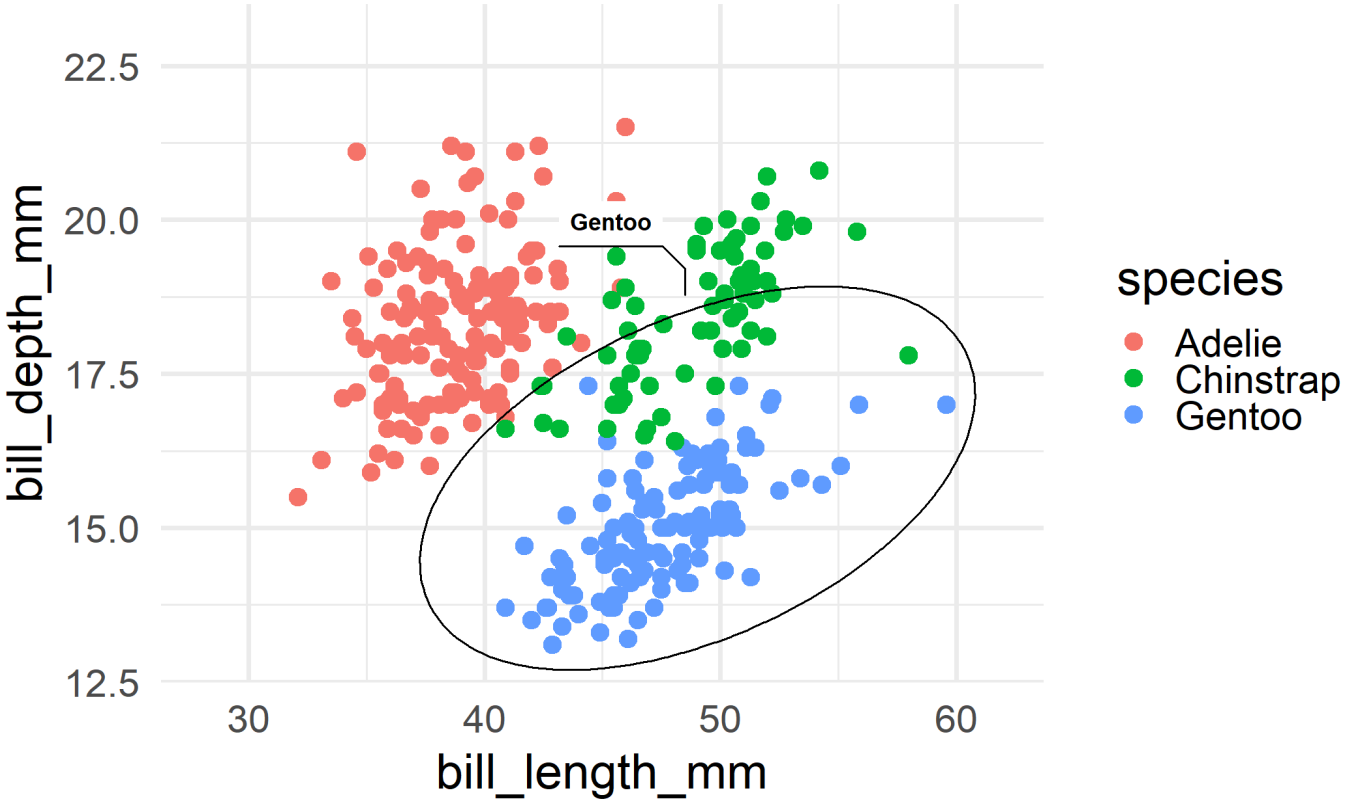
Examples

```
library(palmerpenguins)
library(ggforce)

penguins %>%
  drop_na() %>% # Can't take missing data
ggplot(aes(bill_length_mm, bill_depth_mm)) +
  geom_mark_ellipse(aes(group = species, label = species)) +
  geom_point(aes(color = species)) +
  coord_cartesian(xlim = c(28, 62), ylim = c(13, 23)) +
  guides(color = "none")
```


Adelie
Chinstrap
Gentoo
22.5
20.0
17.5
15.0
12.5
bill_depth_mm
30
40
50
60
bill_length_mm

Limit to a single group

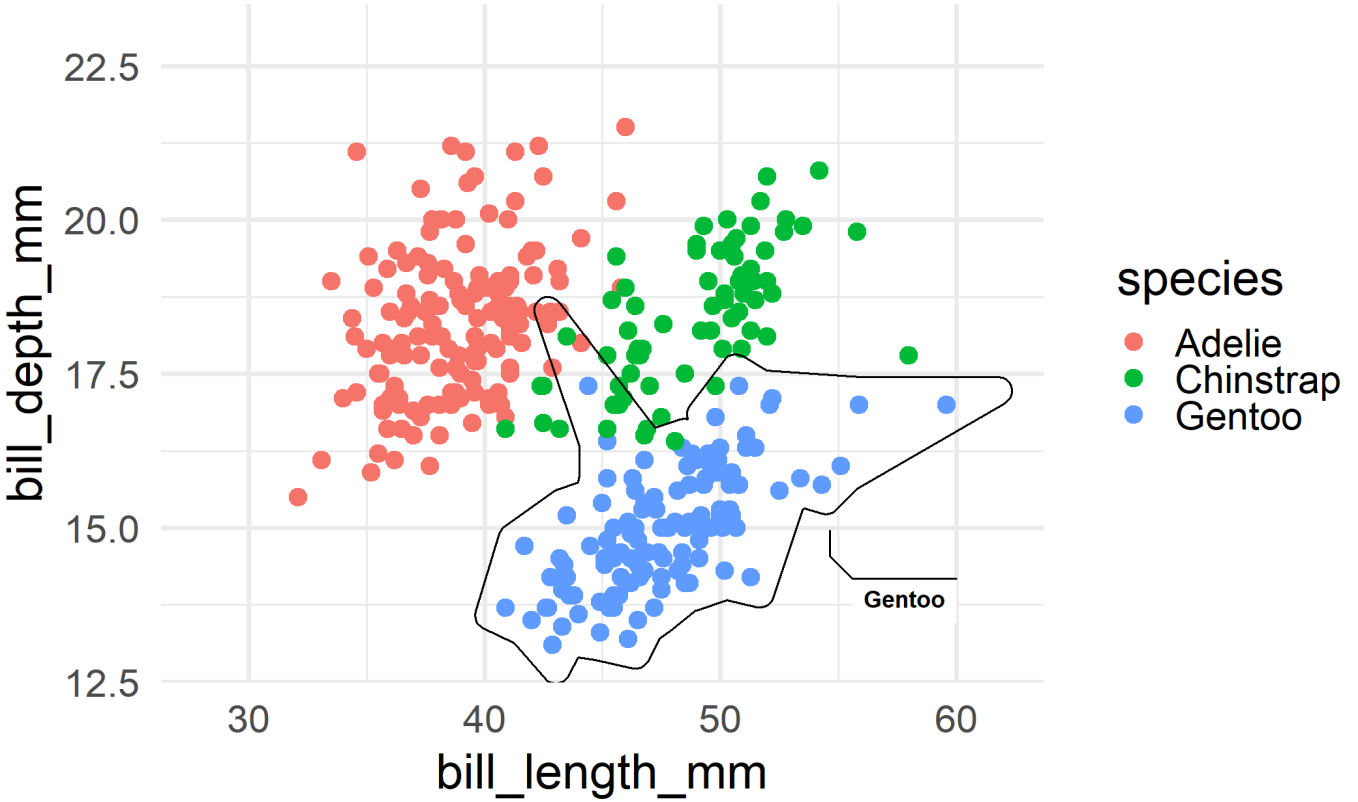
```
penguins %>%
  drop_na() %>%
ggplot(aes(bill_length_mm, bill_depth_mm)) +
  geom_point(aes(color = species)) +
  geom_mark_ellipse(aes(group = species, label = species),
                    data = filter(drop_na(penguins),
                                  species == "Gentoo")) +
  coord_cartesian(xlim = c(28, 62), ylim = c(13, 23))
```


22.5
20.0
17.5
15.0
12.5
bill_depth_mm
30
40
50
60
bill_length_mm
Gentoo
species
Adelie
Chinstrap
Gentoo

Switch to hull

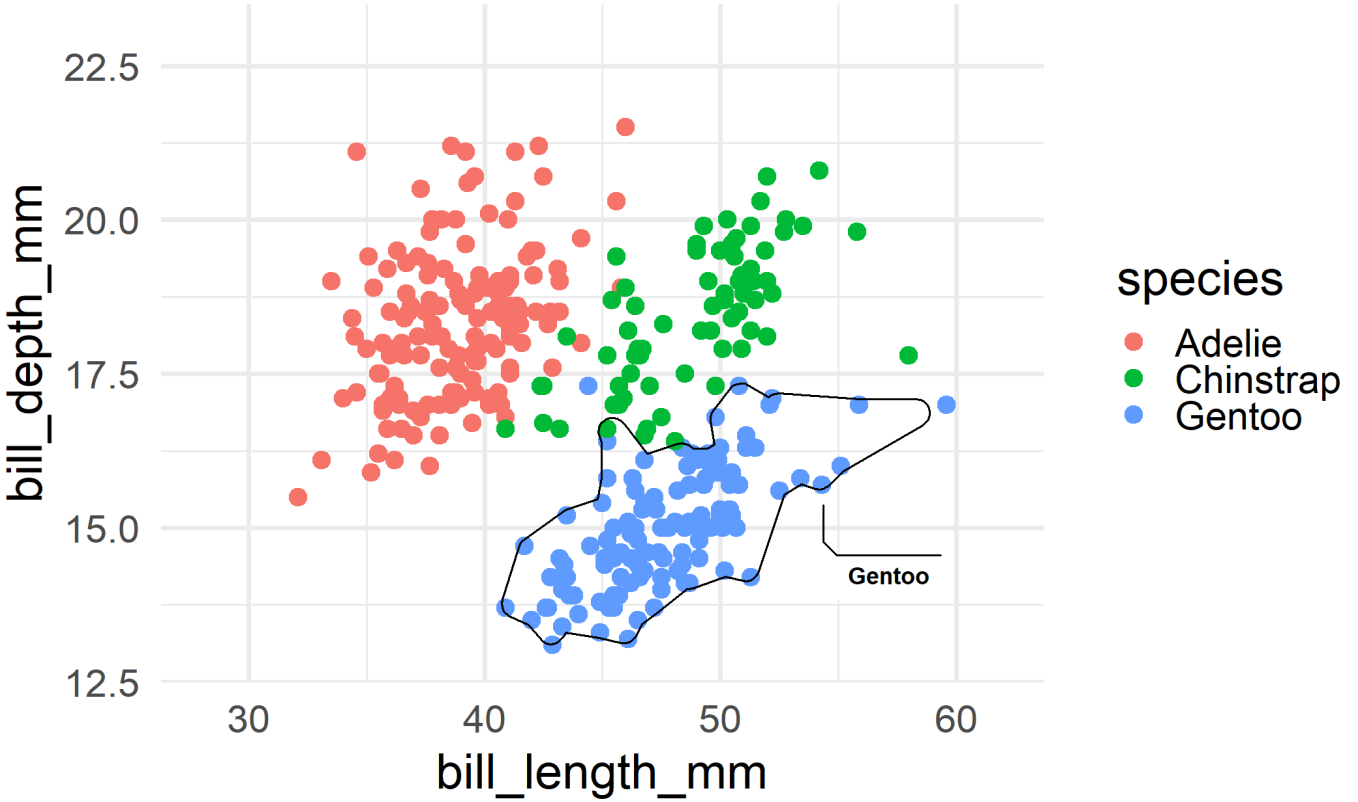
Note - requires the **concaveman** package be installed

```
penguins %>%
  drop_na() %>%
ggplot(aes(bill_length_mm, bill_depth_mm)) +
  geom_point(aes(color = species)) +
  geom_mark_hull(aes(group = species, label = species),
                 data = filter(drop_na(penguins),
                               species == "Gentoo")) +
  coord_cartesian(xlim = c(28, 62), ylim = c(13, 23))
```


22.5
20.0
17.5
15.0
12.5
bill_depth_mm
30
40
50
60
bill_length_mm
Gentoo
species
Adelie
Chinstrap
Gentoo

Change expand

```
penguins %>%
  drop_na() %>%
ggplot(aes(bill_length_mm, bill_depth_mm)) +
  geom_point(aes(color = species)) +
  geom_mark_hull(aes(group = species, label = species),
                 expand = unit(1, "mm"),
                 data = filter(drop_na(penguins),
                               species == "Gentoo")) +
  coord_cartesian(xlim = c(28, 62), ylim = c(13, 23))
```


22.5
20.0
17.5
15.0
12.5
bill_depth_mm
30
40
50
60
bill_length_mm
Gentoo
species
Adelie
Chinstrap
Gentoo

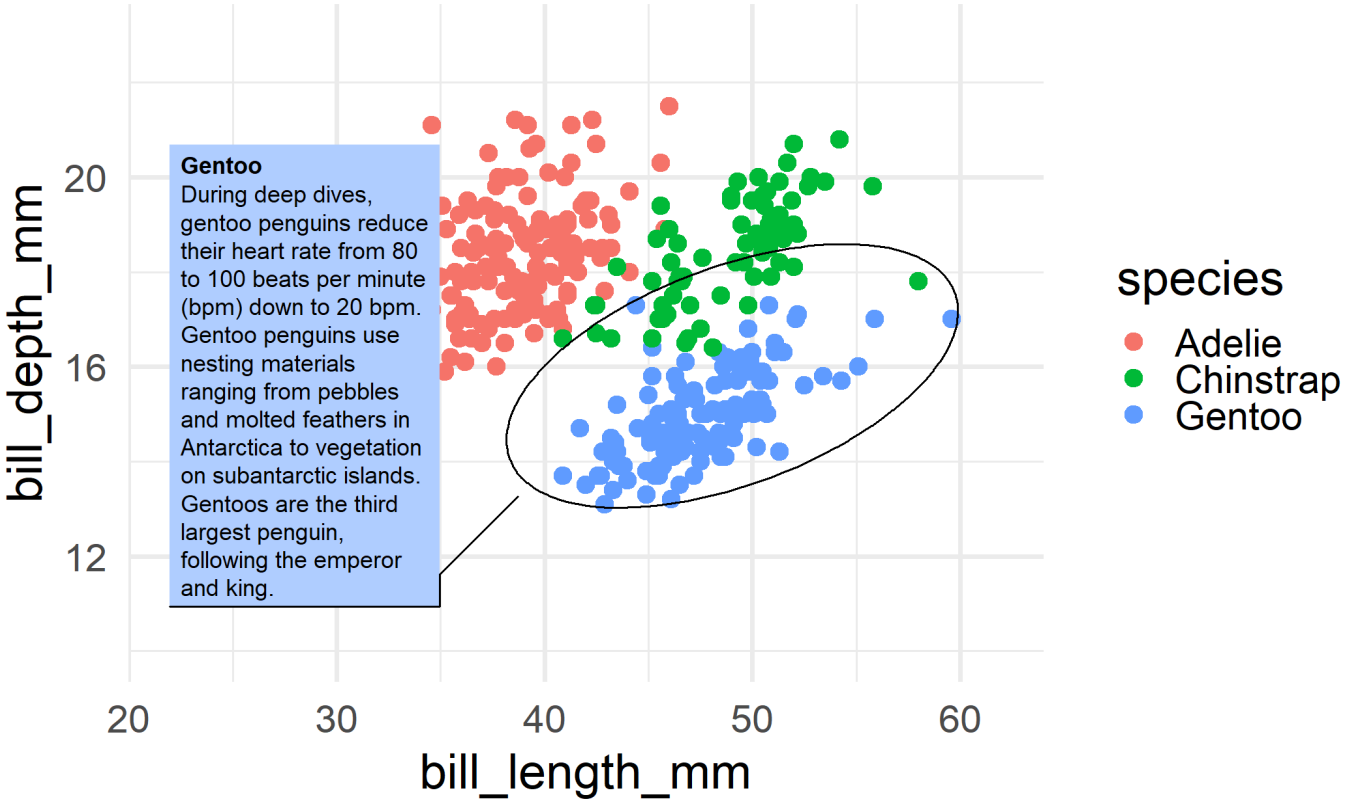
More in-depth annotations

First create a description

```
penguins <- penguins %>%
  mutate(desc = ifelse(species != "Gentoo", "", "During deep dive
```

Now add as a description

```
penguins %>%
  drop_na() %>%
ggplot(aes(bill_length_mm, bill_depth_mm)) +
  geom_point(aes(color = species)) +
  geom_mark_ellipse(aes(group = species,
                        label = species,
                        description = desc),
                    data = filter(drop_na(penguins),
                                  species == "Gentoo"),
                    label.fill = "#b3cfff") +
  coord_cartesian(xlim = c(28, 62), ylim = c(13, 23))
```

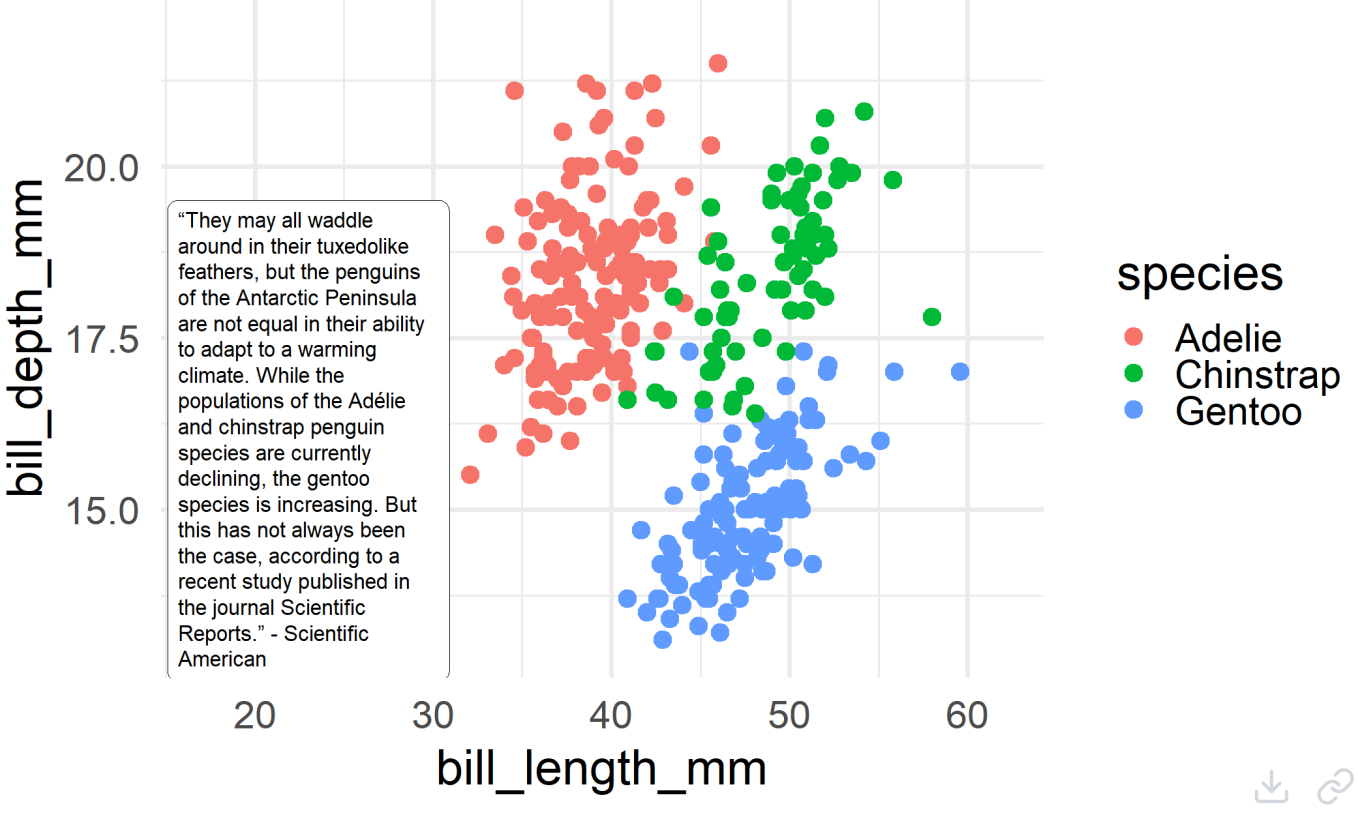

Gentoo
During deep dives, gentoo penguins reduce their heart rate from 80 to 100 beats per minute (bpm) down to 20 bpm. Gentoo penguins use nesting materials ranging from pebbles and molted feathers in Antarctica to vegetation on subantarctic islands. Gentoos are the third largest penguin, following the emperor and king.
bill_depth_mm
20
16
12
20
30
40
50
60
bill_length_mm
species
Adelie
Chinstrap
Gentoo

Similar

We can also just add a textbox through **{ggtext}**

```
txtbox <- tibble(
  bill_length_mm = 23,
  bill_depth_mm = 16,
  lab = '"They may all waddle around in their tuxedolike feathers
)
```

```
penguins %>%
  drop_na() %>%
ggplot(aes(bill_length_mm, bill_depth_mm)) +
  geom_point(aes(color = species)) +
  ggtext::geom_textbox(aes(label = lab),
                       data = txtbox) +
  coord_cartesian(xlim = c(17, 62), ylim = c(13, 22))
```



Last bit

The **ggforce** package is well worth exploring more.

See here for a nice walkthrough that has good data viz and uses some of the **ggforce** functions (as well as illustrating a few other cool packages)

Saving plots

And potentially making additional edits

Raster/Vector

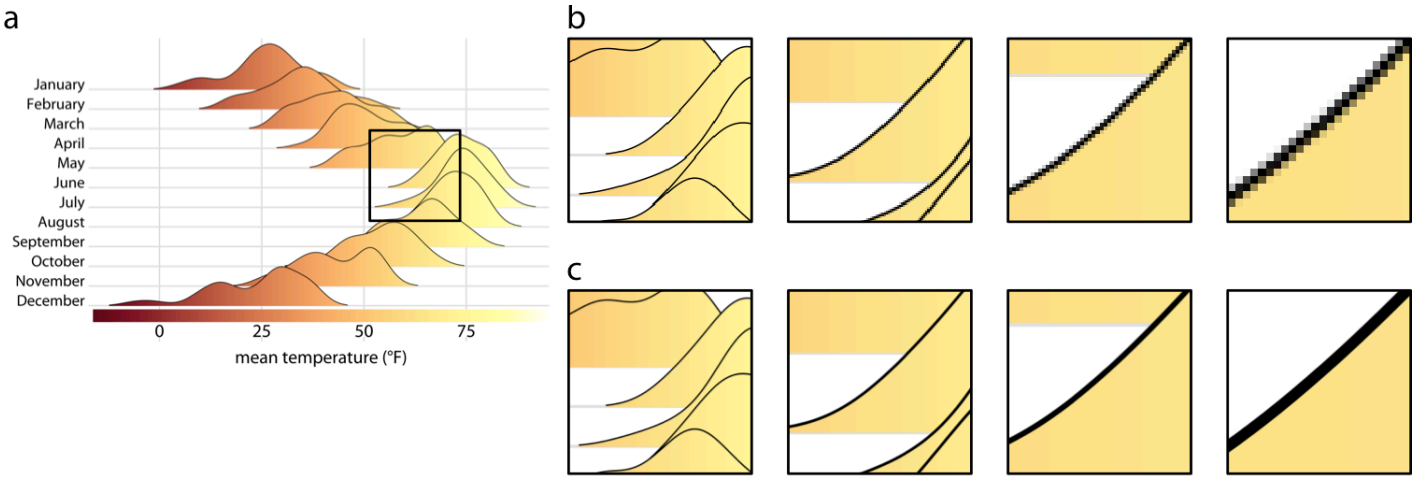
We'll talk about this again when discussing maps, but it relates to saving as well.

Raster (also called bitmap) stores images as a grid of points (pixels)

Vector store instructions for how the figure should be drawn. Image is redrawn as it is printed/displayed on screen

see Wilke, Chapter 27 for more information.

Differences



Vector graphics: pdf, eps, svg

Raster graphics: png, jpeg, tiff, gif

Image from Wilke

Downsides to vector graphics

- Possible differences in appearance between displays (programs, computers, etc.)
- Very large/complex figures can balloon to giant file sizes and be slow to render

Lossy/Lossless compression

- Lossless - guarantees image is, pixel for pixel, identical to original
    - png and tiff use lossless compression
- Lossy - accepts some minor image artifacts to reduce size
    - jpeg

Practical advice

Export to PDF

If that won't work (web), use png

Practice

- Create a plot
- Save it as a PDF with `ggsave()`

Note, the first argument to `ggsave()` is the `path`, so you could do something like

```
ggsave(here::here("myplot.pdf"))
```

You can also specify the width/height.

By default, it will save the last plot you produced, but you can also specify it with `plot =` argument, where you pass an object that has the plot

Modifications

I rarely do this, but if I do, I tend to use Inkscape, which is free.

```
ggplot(mpg, aes(displ, hwy)) +
  geom_point(color = "gray80") +
  geom_point(color = "#FD7A43", data = filter(mpg, cyl == 4))

ggsave("~/Desktop/example-plot.pdf", width = 6.5, height = 6.5)
```

Compound figures

Please follow along

Options

My favorite: {patchwork}

- {cowplot}
- {ggpubr}
- {gridExtra}

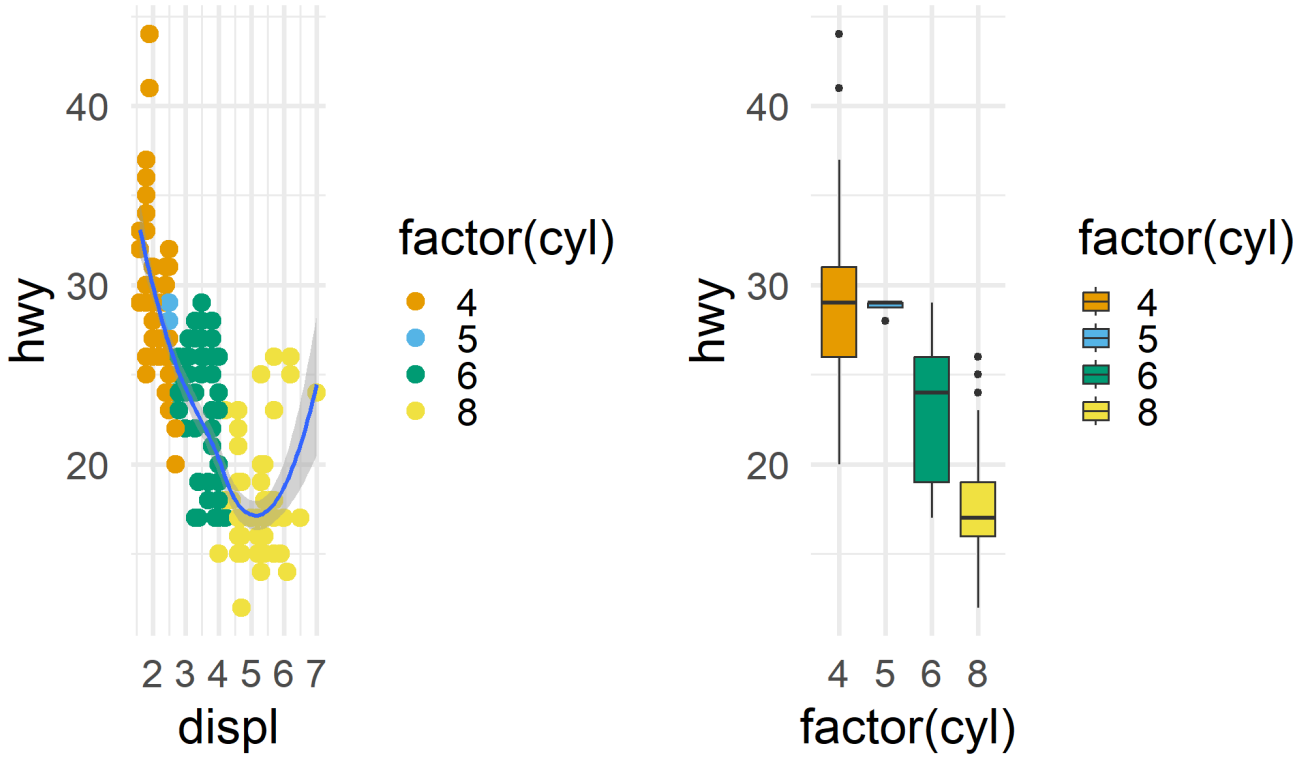
Example

- First, create two plots

```
p1 <- ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(color = factor(cyl))) +
  geom_smooth() +
  scale_color_OkabeIto()

p2 <- ggplot(mpg, aes(factor(cyl), hwy)) +
  geom_boxplot(aes(fill = factor(cyl))) +
  scale_fill_OkabeIto()
```

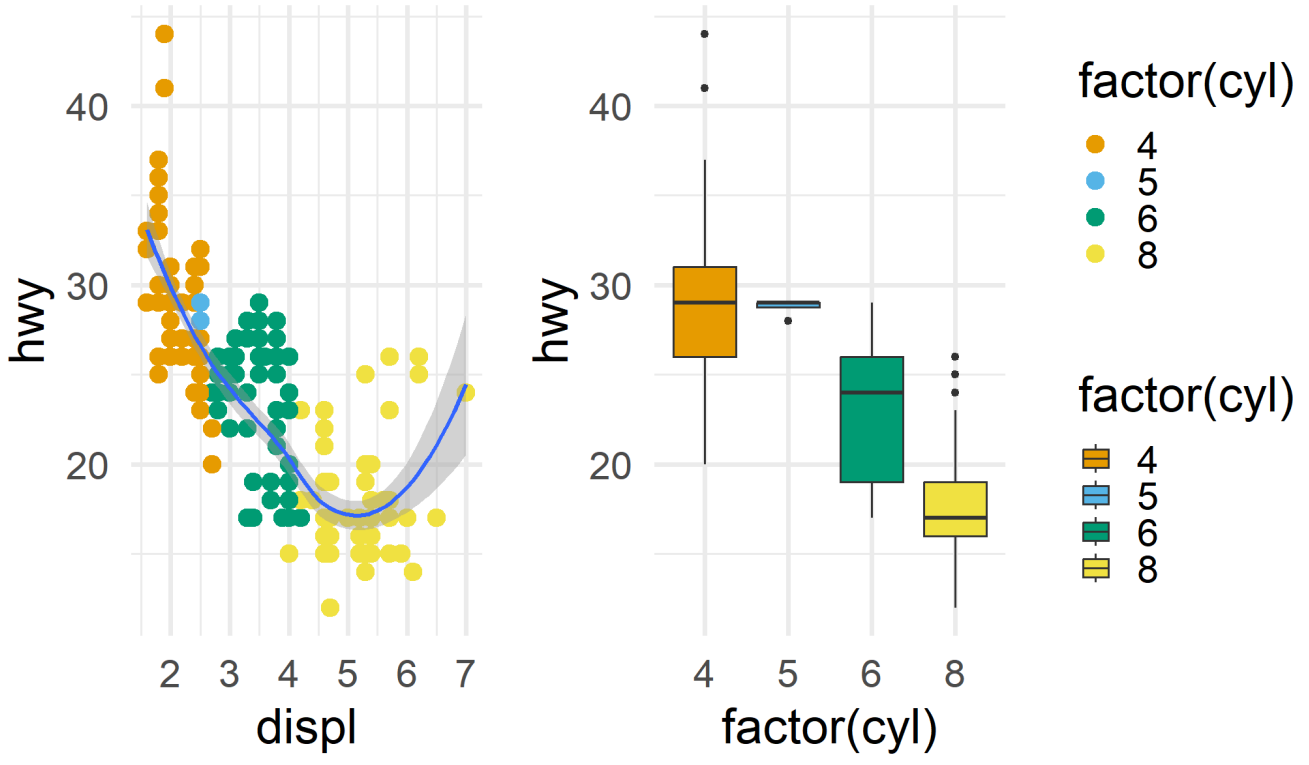
Side by side

```
library(patchwork)
p1 + p2
```



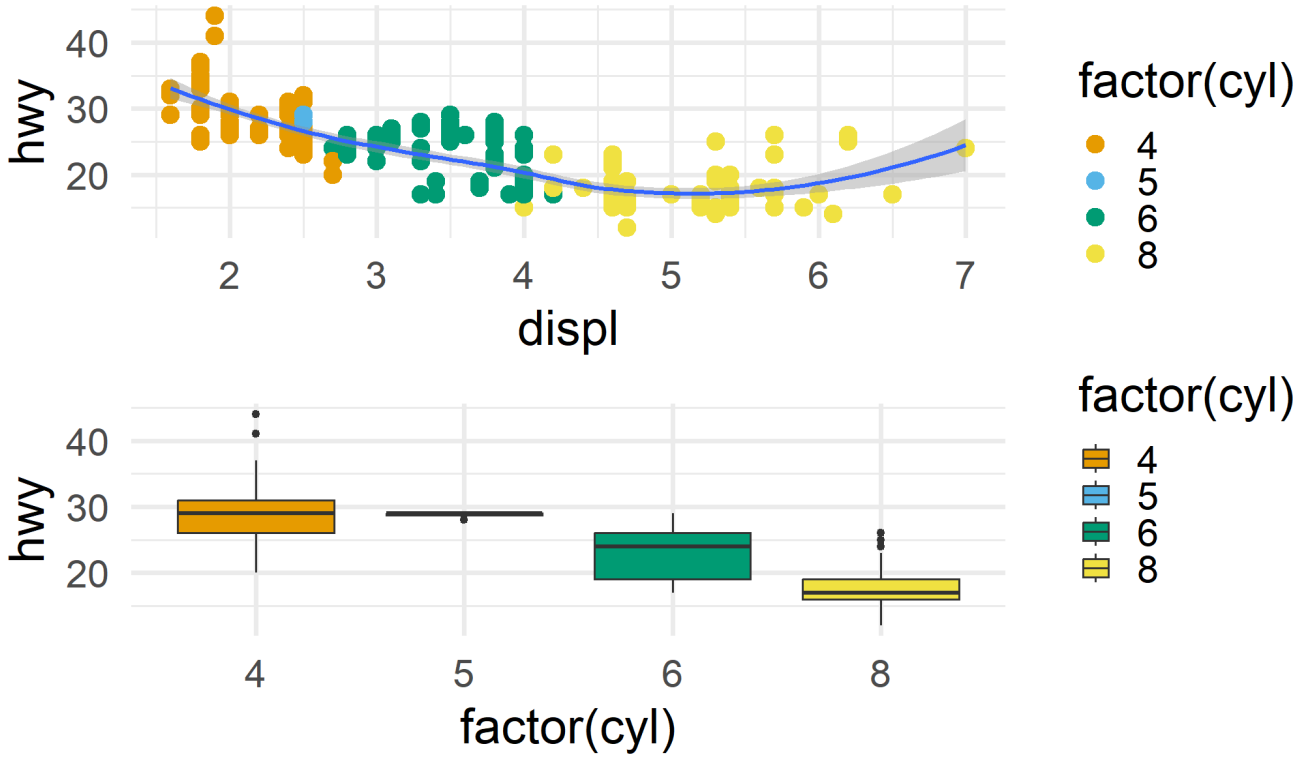
Collect legends

```
p1 + p2 + plot_layout(guides = "collect")
```



Stack vertically

```
p1 / p2 + plot_layout(guides = "collect")
```

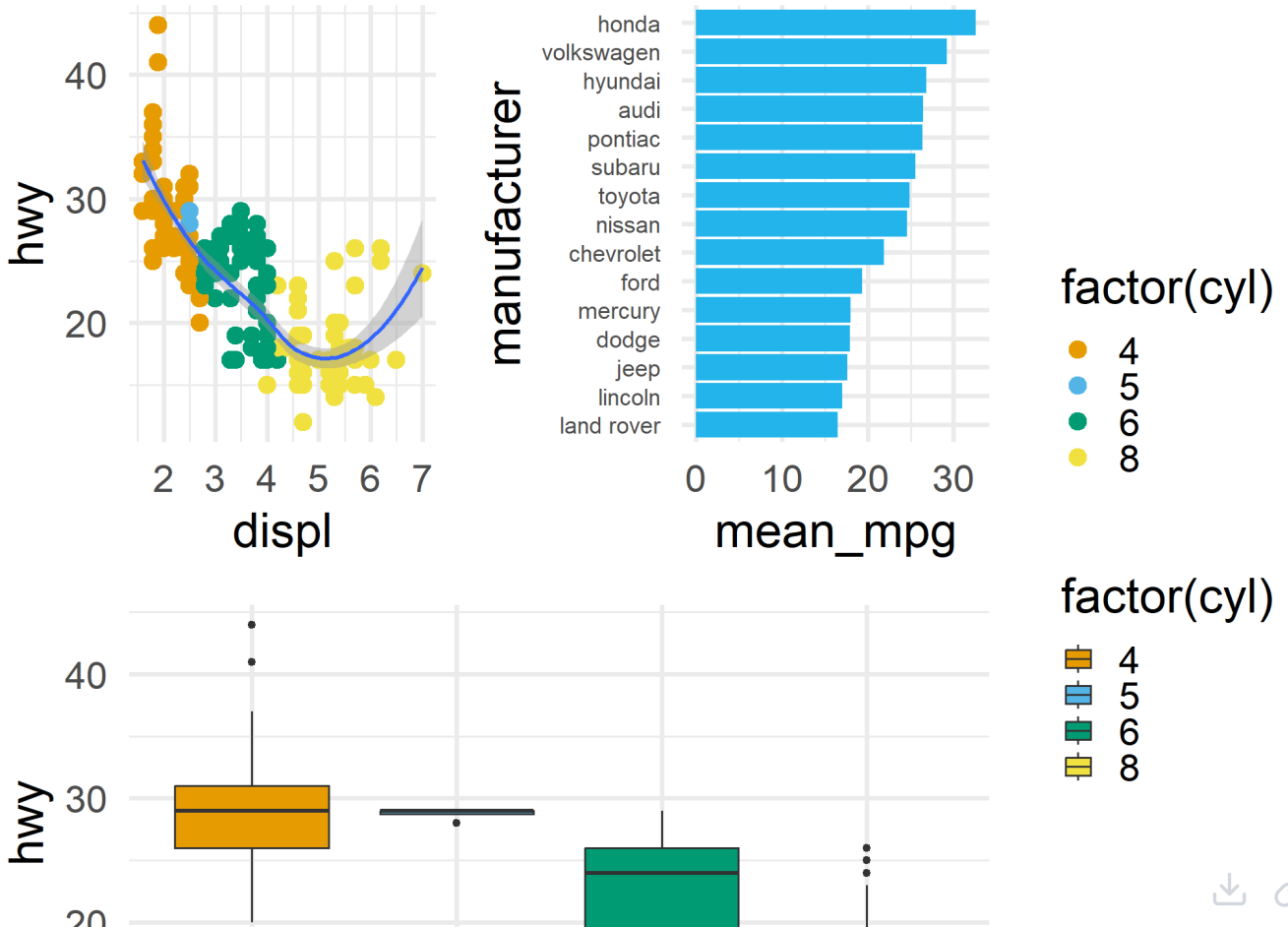


Add a third plot

```
p3 <- mpg %>%
  group_by(manufacturer) %>%
  summarize(mean_mpg = mean(hwy, na.rm = TRUE)) %>%
  mutate(manufacturer = fct_reorder(manufacturer, mean_mpg)) %>%
  ggplot(aes(mean_mpg, manufacturer)) +
    geom_col(fill = "#25B6EE") +
    theme(axis.text.y = element_text(size = 12))
```

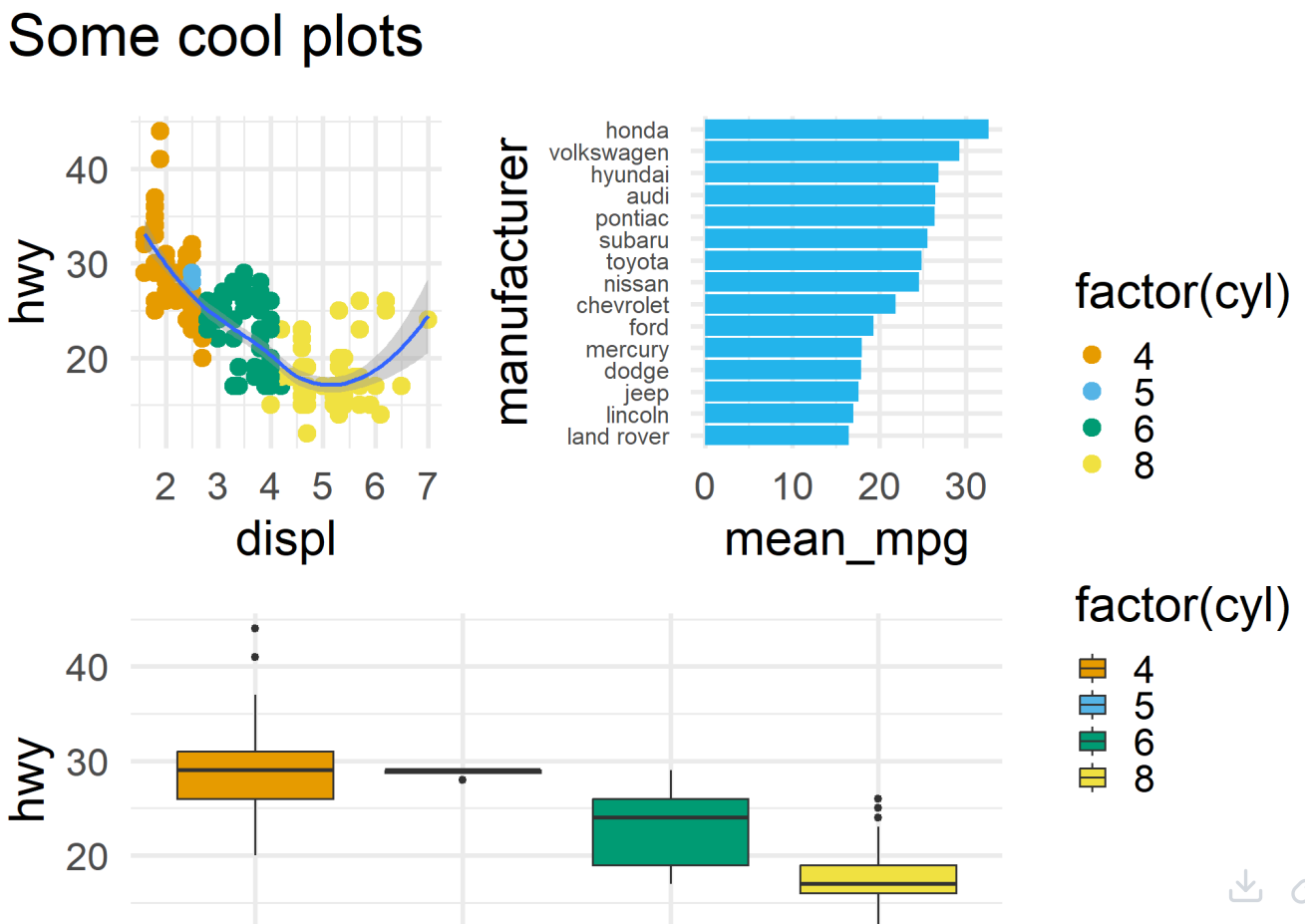
Put box plot on bottom

```
(p1 + p3) / p2 + plot_layout(guides = "collect")
```



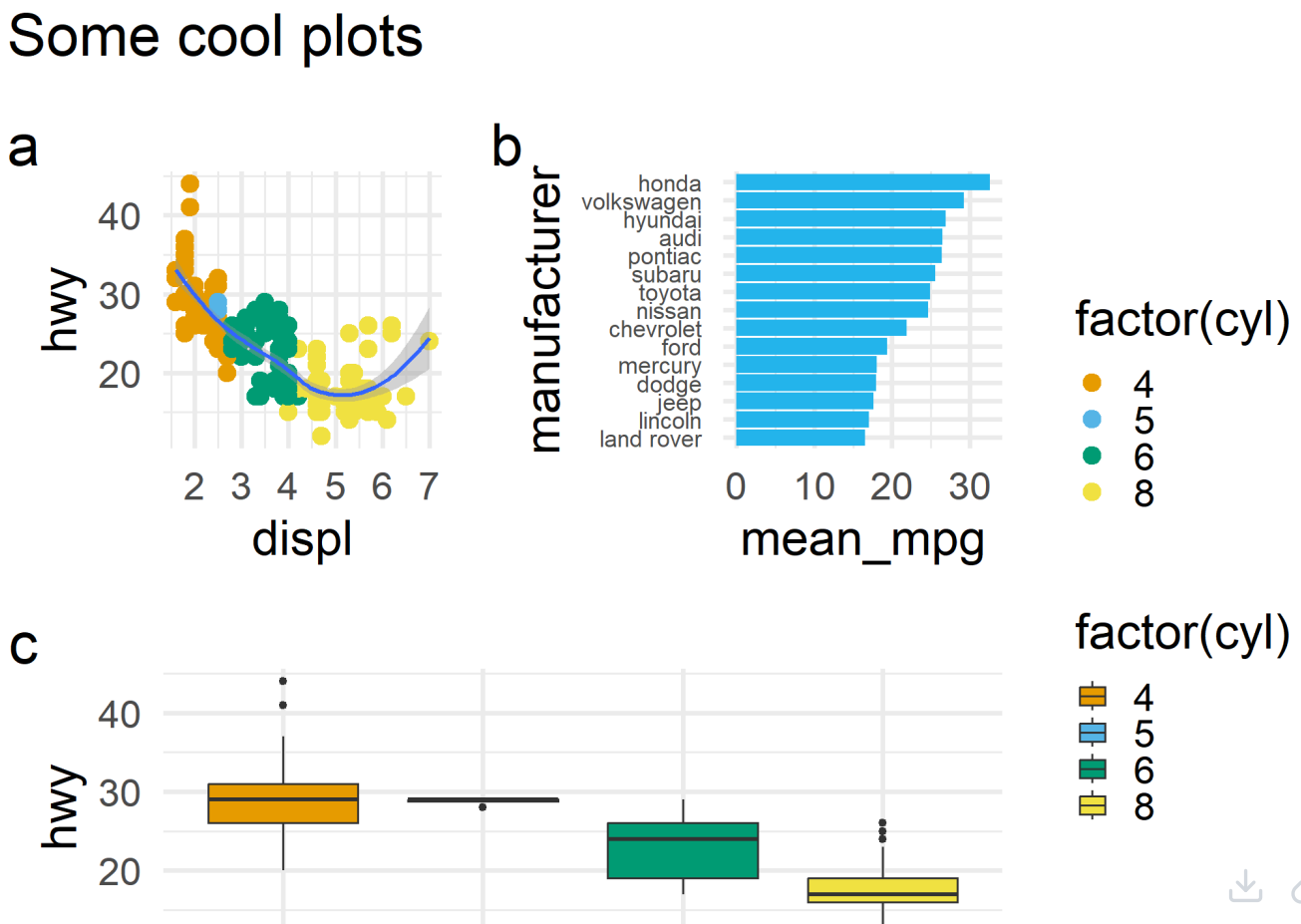
Overall title

```
(p1 + p3) / p2 + plot_layout(guides = "collect") +
  plot_annotation("Some cool plots")
```



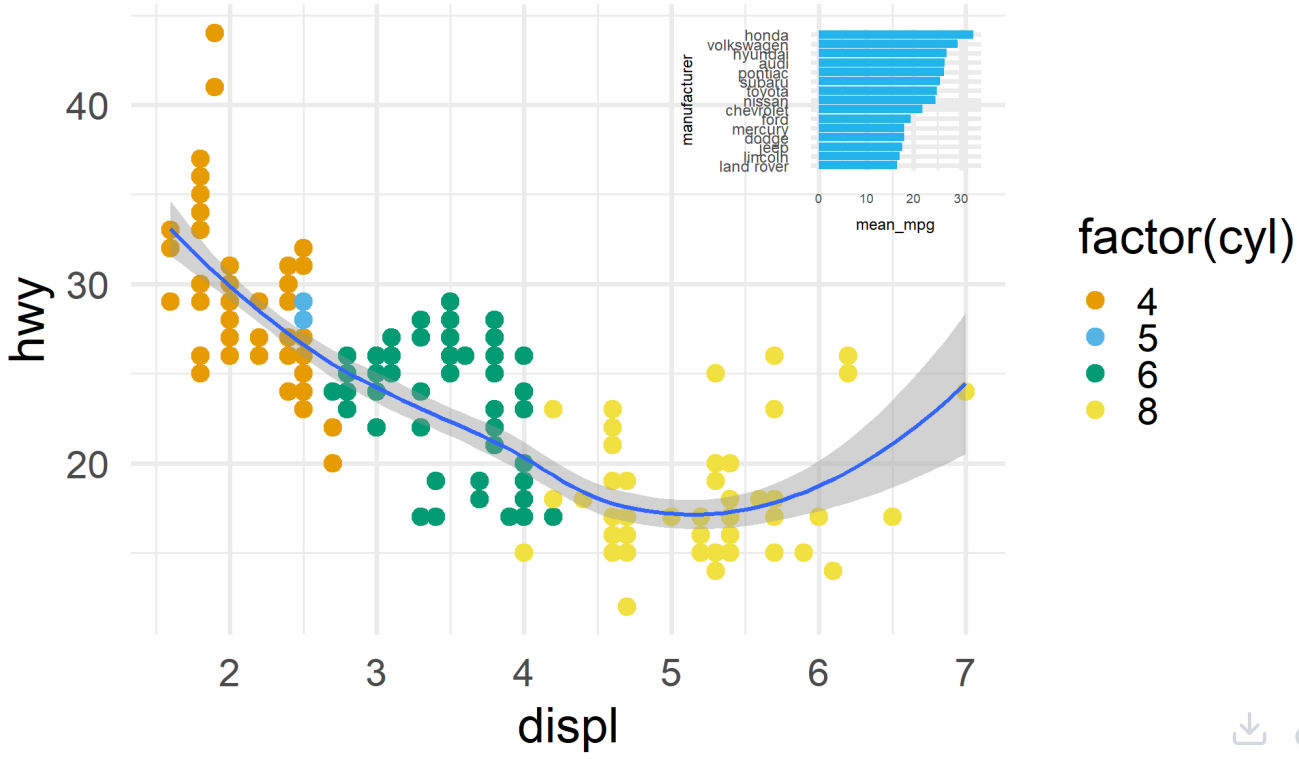
Tags

```
(p1 + p3) / p2 + plot_layout(guides = "collect") +
  plot_annotation("Some cool plots", tag_levels = "a")
```

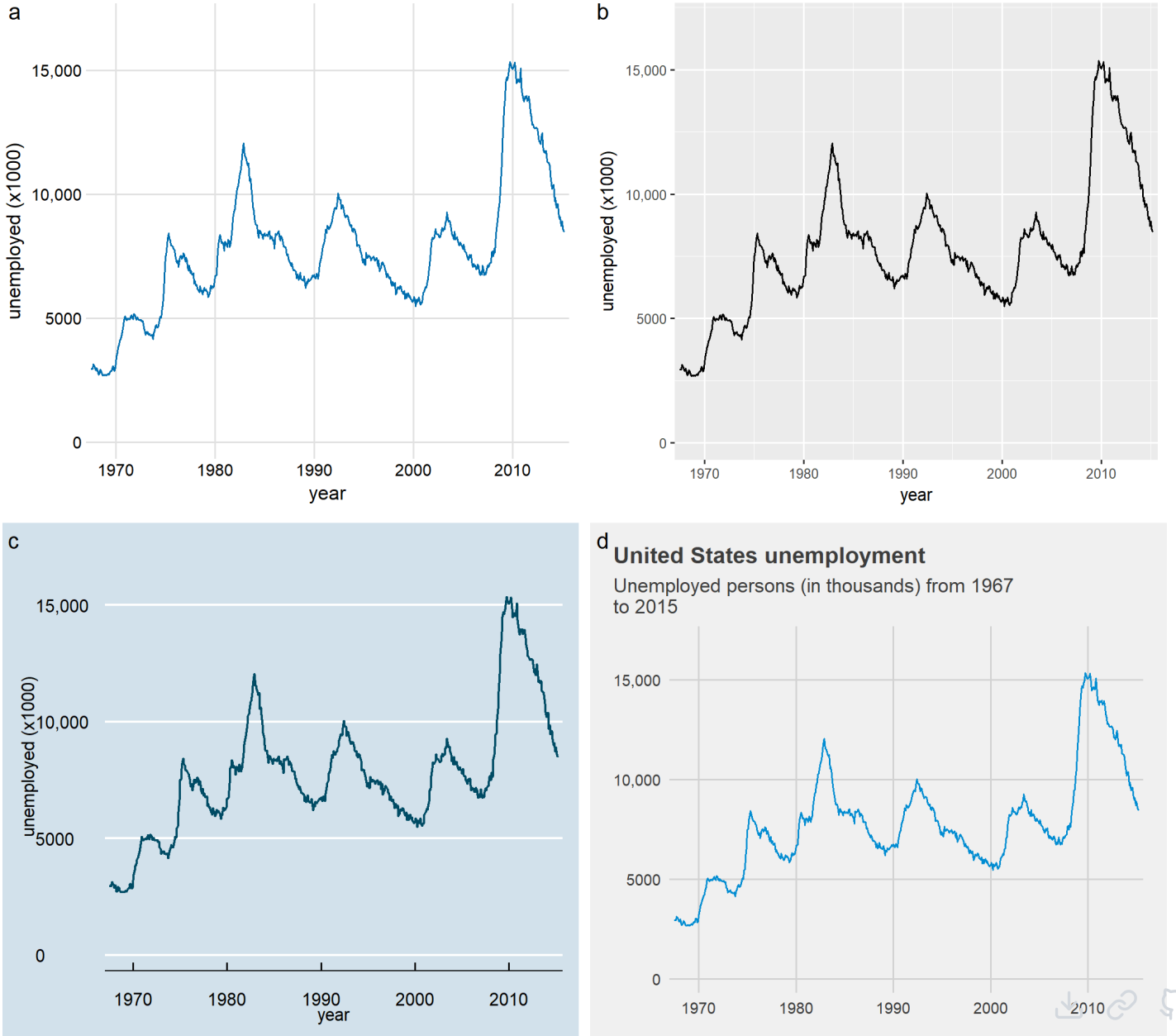


Insets

```
p3_small_txt <- p3 +
  theme(axis.text.y = element_text(size = 8),
        text = element_text(size = 8))

p1 + inset_element(p3_small_txt, 0.6, 0.6, 1, 1)
```



Themes Refresher


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

- Good 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`

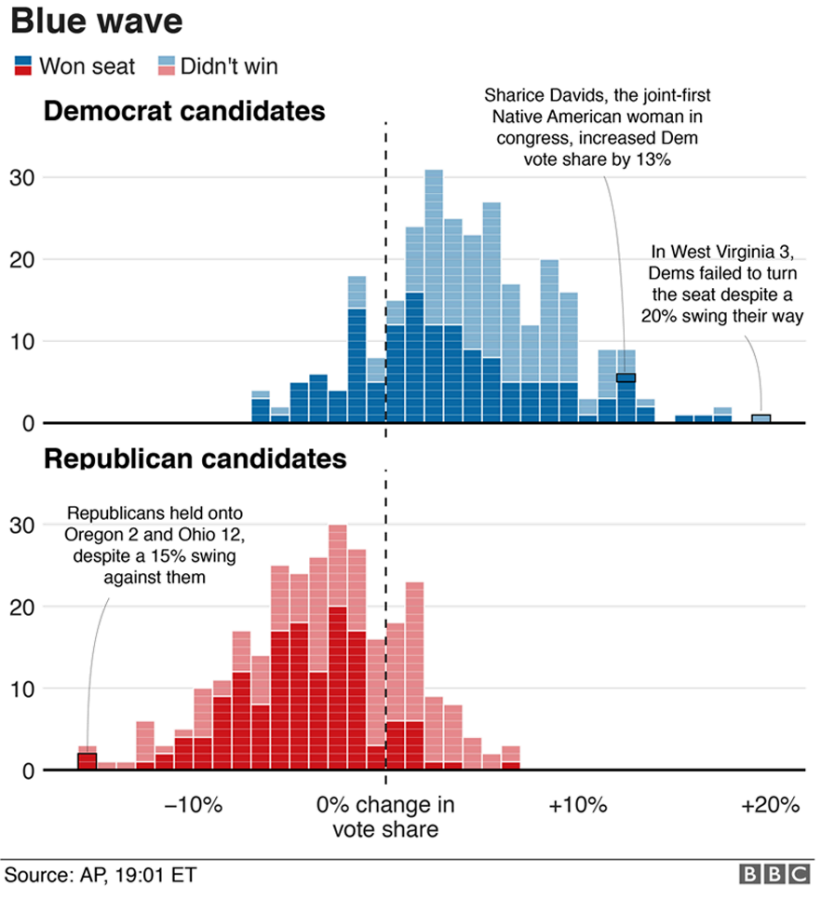
See the themes here

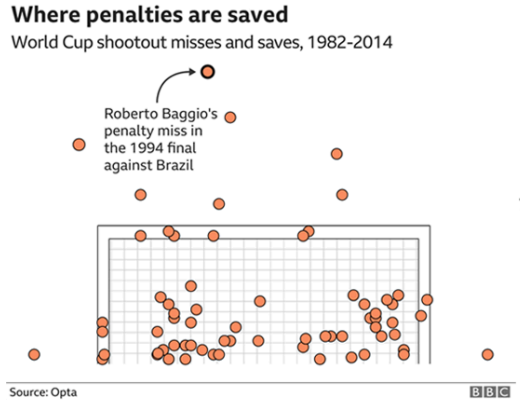
BBC

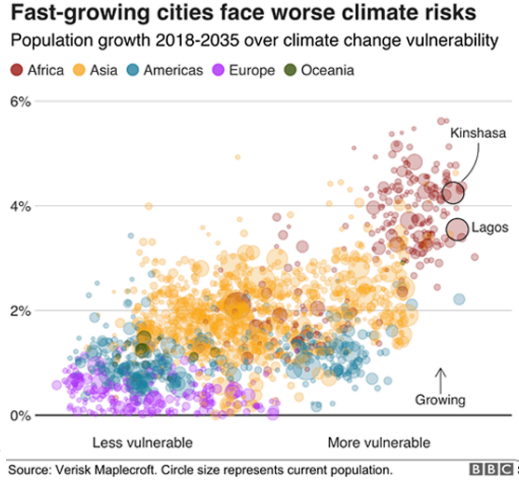
The BBC uses ggplot for most of its graphics. They've developed a package with a theme and some functions to help make it match their style more.

See the repo here

Their Journalism Cookbook is really nice too


Blue wave
Won seat
Didn't win
Democrat candidates
Sharice Davids, the joint-first Native American woman in congress, increased Dem vote share by 13%
In West Virginia 3, Dems failed to turn the seat despite a 20% swing their way
30
20
10
0
Republican candidates
Republicans held onto Oregon 2 and Ohio 12, despite a 15% swing against them
30
20
10
0
−10%
0% change in vote share
+10%
+20%
Source: AP, 19:01 ET
BBC


Where penalties are saved
World Cup shootout misses and saves, 1982-2014
Roberto Baggio's penalty miss in the 1994 final against Brazil
Source: Opta
BBC


Fast-growing cities face worse climate risks
Population growth 2018-2035 over climate change vulnerability
Africa
Asia
Americas
Europe
Oceania
6%
4%
2%
0%
Kinshasa
Lagos
Growing
Less vulnerable
More vulnerable
Source: Verisk Maplecroft. Circle size represents current population.
BBC


MPs rejected Theresa May's deal by 230 votes
For
Con (196)
202
Lab 3, Ind 3
Against
Con rebels (118)
Labour (248)
432
SNP 35, Lib Dem 11, DUP 10, Plaid 4, Green 1, Ind 5
Source: Commons Votes Services. Excludes 'tellers', the Speaker and deputies
BBC


Earnings vary across unis even within subjects
Impact on men's earnings relative to the average degree
Below average
Above average
Economics
Law
Business
Maths
Creative arts
Computing
History
Psychology
Physical Sciences
Politics
Engineering
Comms
Geography
Allied to medicine
Philosophy
Languages
Architecture
Pharmacology
Technology
Medicine
Biosciences
Sociology
Sports Sciences
Agriculture
Physics
English
Education
Social care
Northumbria
Canterbury Christchuch
Cons Dance &Drama
Cambridge
Oxford
Bristol
-50%
0%
50%
100%
Source: Institute for Fiscal Studies
BBC

Similarly, the Urban Institute Visual Guide

See the repo here

So, I created one!

- Based on UO's visual guide here

[demo]

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

- You can basically change your plot to look however you want through `theme`
- Generally a bit more complicated
- I've used ggplot for *years* and only really now gaining fluency with it

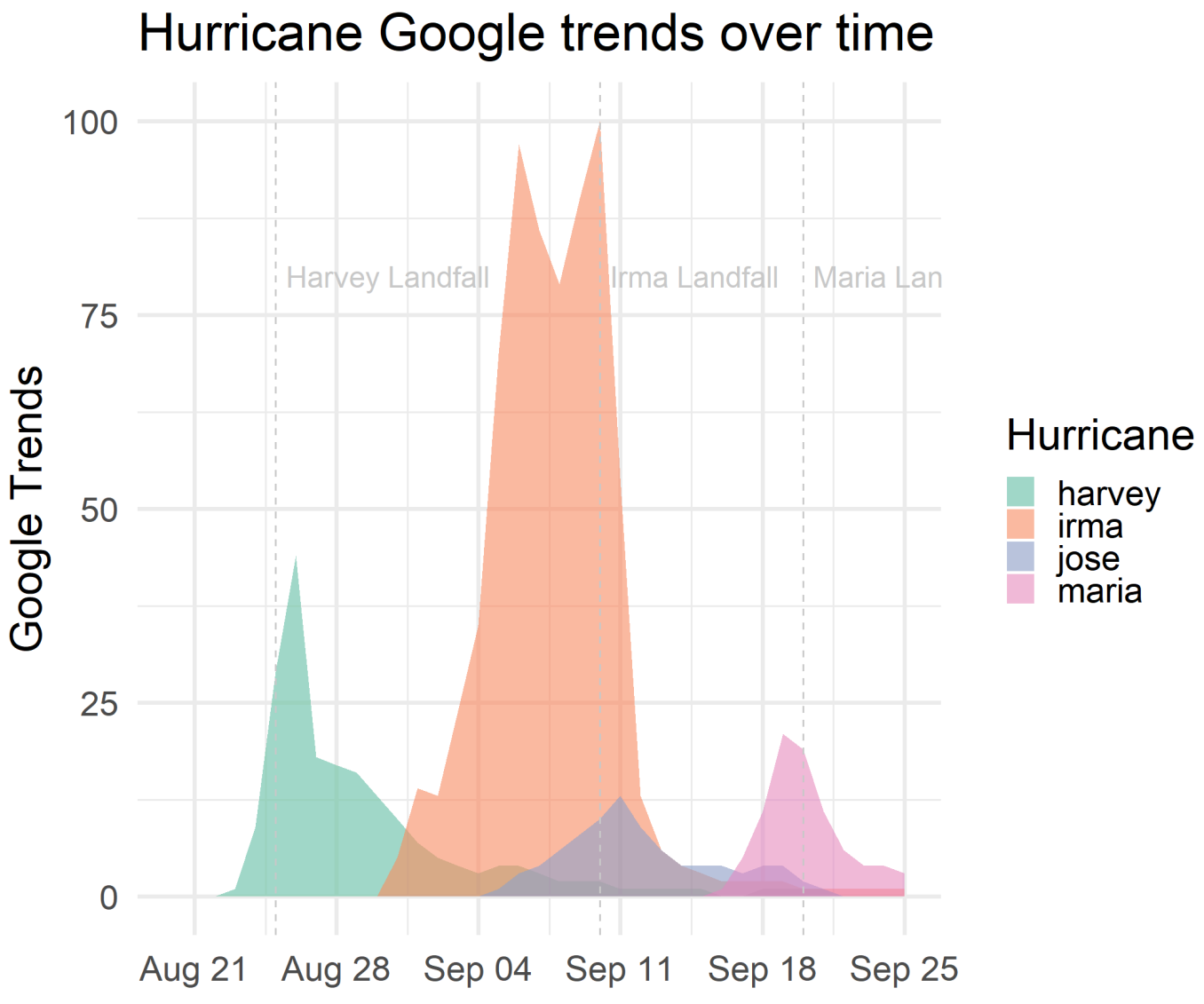
Last Quick example

The *google_trends* dataset comes from a fivethirtyeight story about how the media covered hurricanes and Trump.

```
library(fivethirtyeight)
g <- google_trends %>%
  pivot_longer(starts_with("hurricane"),
               names_to = "hurricane",
               values_to = "interest",
               names_pattern = "_(.+)_")

landfall <- tibble(
  date = lubridate::mdy(
    c("August 25, 2017", "September 10, 2017", "September 20, 201
  ),
  hurricane = c("Harvey Landfall", "Irma Landfall", "Maria Landfa
)
```

Let's start by visualizing the change in trends for each hurricane over time in one plot with three scales. We can map color to a discrete scale here. We can add vertical lines to show when each of them made 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
  scale_fill_brewer("Hurricane", palette = "Set2")
```



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

We can use ggthemeassist to make a whole bunch of changes!

```
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(0.2, -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")
)
```

Lab PS-3

Lab 5

Q & A with Dr. Daniel Anderson

Next time

Wrap up visualizing changes over time/distributions and Intro to Websites, Flex dashboards, CSS customizations?

Note: Change in schedule and Lab-PS3- Last Lab problem set in this class- is also posted