Time

Week 9

AEM 2850 / 5850 : R for Business Analytics

Cornell Dyson

Fall 2025

Acknowledgements: Andrew Heiss

Announcements

Prelim 1 regrade requests due this Thursday, October 23

Group projects will be due Friday, November 14 (not Nov 7!)

- **Instructions will be posted on canvas tomorrow**
- **We will post group member updates by tomorrow** (as needed)
- We will set up group-specific workspaces on posit cloud by Thursday
  - Note: we have encountered some issues with overwriting work in the past
  - Save your work early and often to avoid overwriting each other's work
  - We recommend backing up your individual code whenever you stop work
  - Use a collaborative editing platform like google docs for your code
- Make a plan and start early!

Questions before we get started?

Plan for this week

Tuesday

- Prologue
- Visualizing time: the basics
  - Visualizing amounts over time
  - Dates and times in R
- Start example-09

Thursday

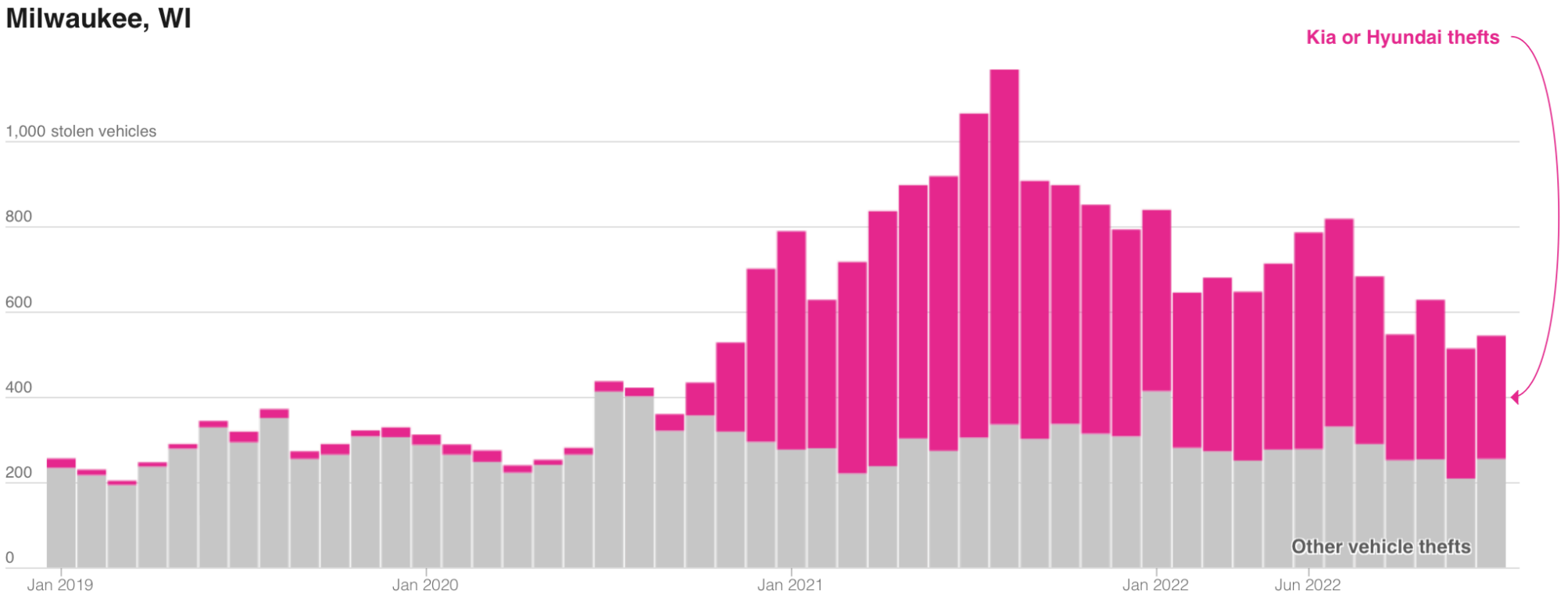
- Visualizing time: leveling up
  - Faceting plots
  - Animating plots
- Finish example-09

Reference slides at end of deck:

- Visualizing distributions over time
- Starting, ending, and decomposing time

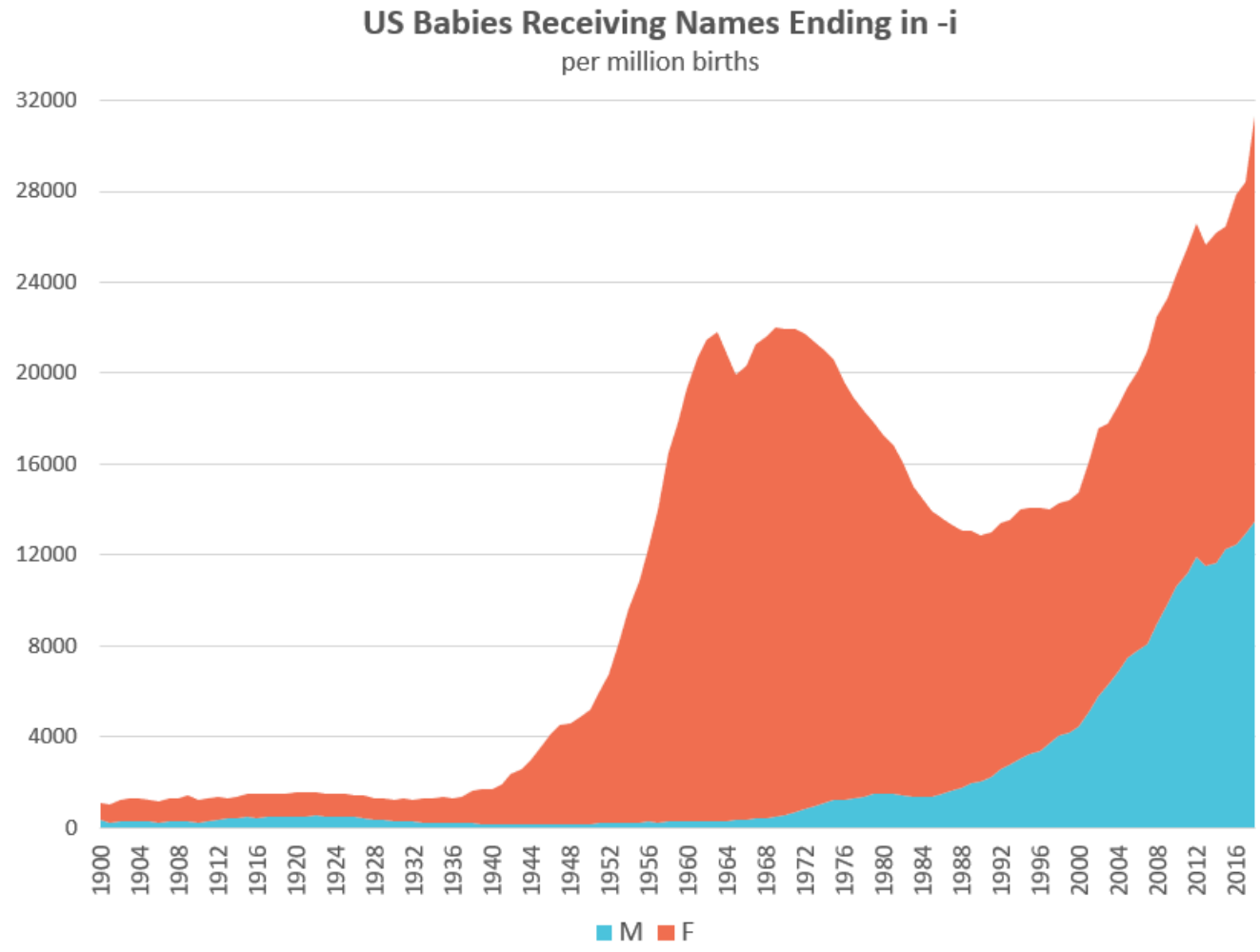
Prologue

Car thefts are rising. Is TikTok to blame?



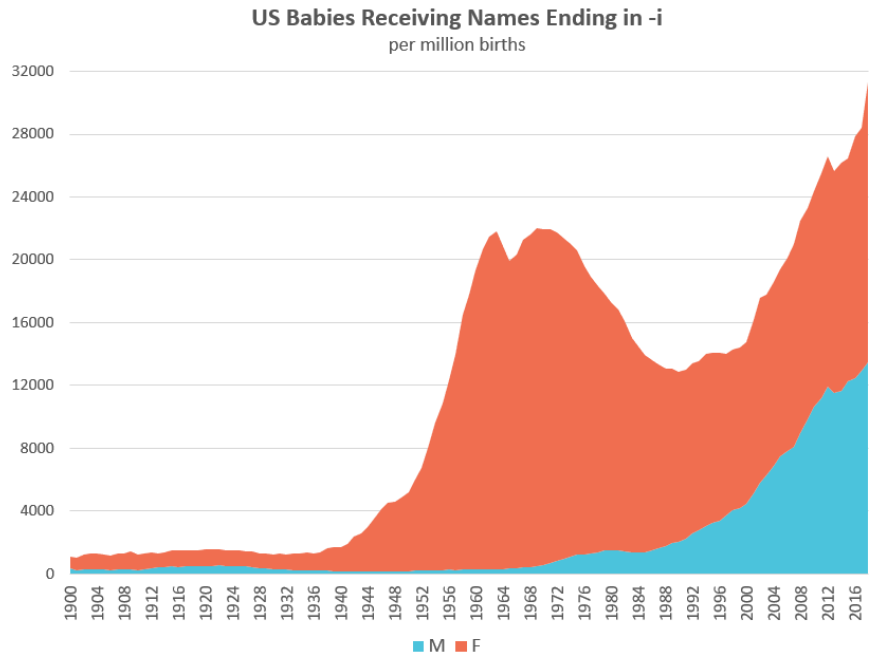
What's good about this graph? What could be better?

What's good? What could be better?

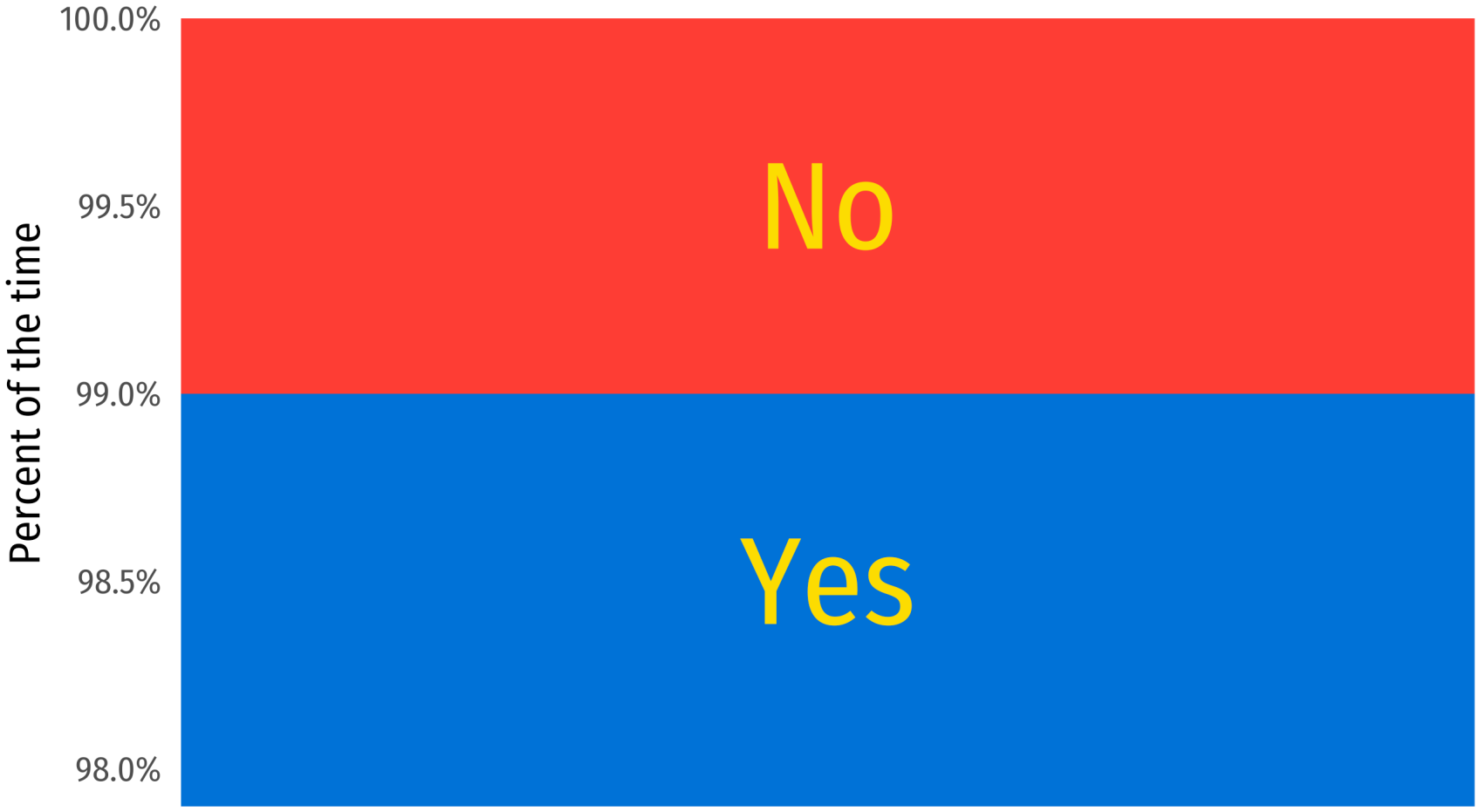


What's good? What could be better?

- Stacked curves makes it hard to see trends for females
- Legend could be bigger and use "Male" and "Female"
- x-axis labeling: use labels every 10 or 20 years (not 4!)
- y-axis labeling:
  - multiples of 4 are not intuitive
  - %s would be easier to interpret



Is truncating the y-axis misleading?



Don't be too extreme!

When discussing amounts, we said to never truncate the y-axis

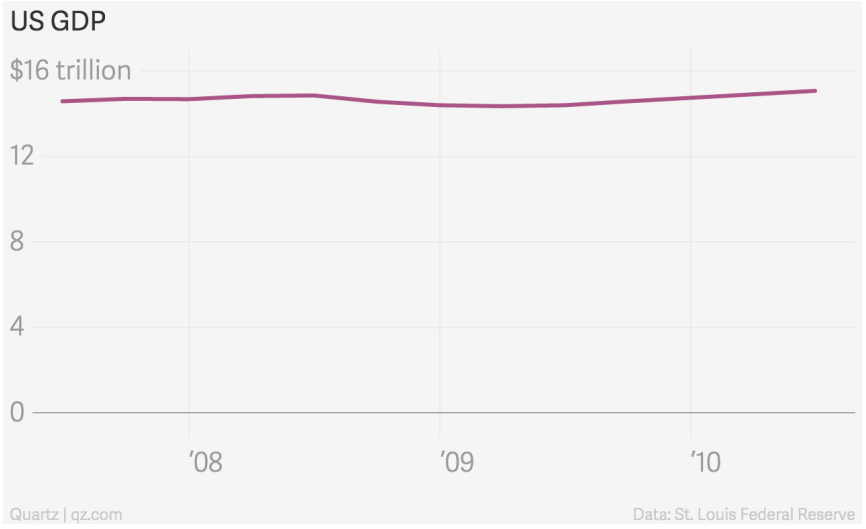
But it is actually more legal to truncate the y-axis than you might think

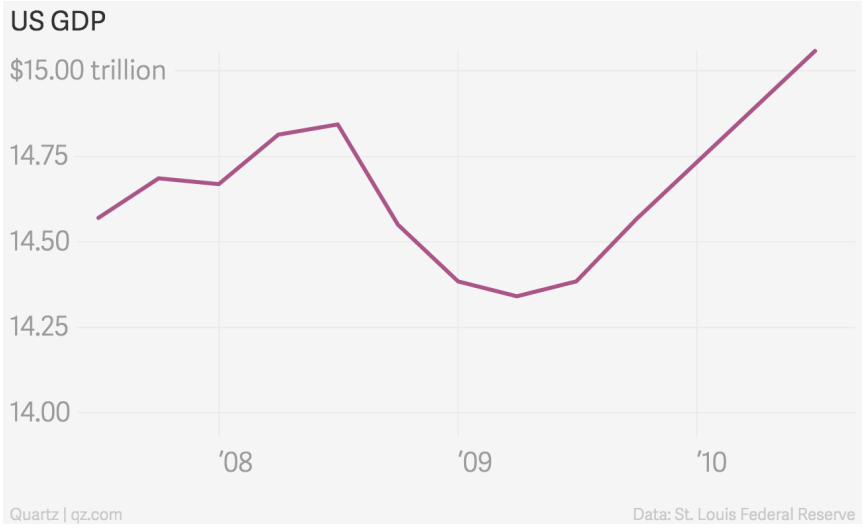
When do you think it is okay to truncate?

- When small movements matter
- When the scale itself is distorted
- When zero values are impossible

When is it okay to truncate?

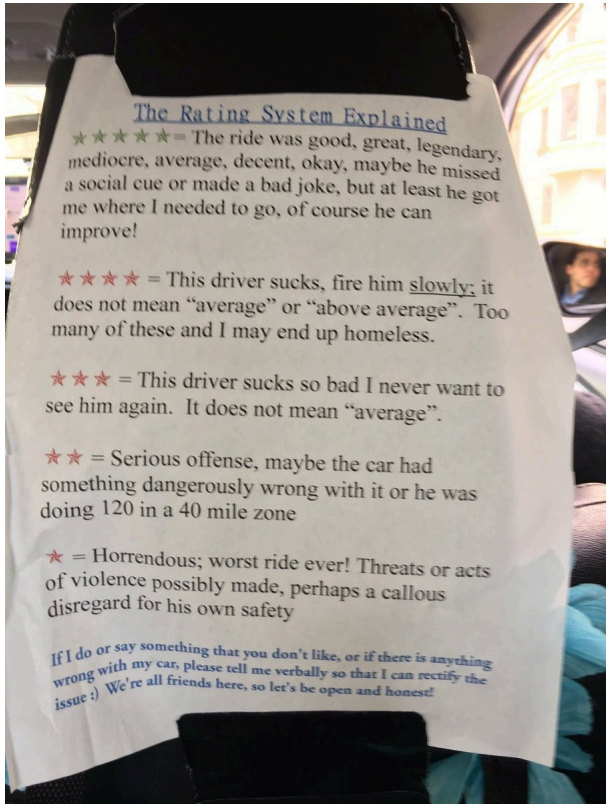
When small movements matter

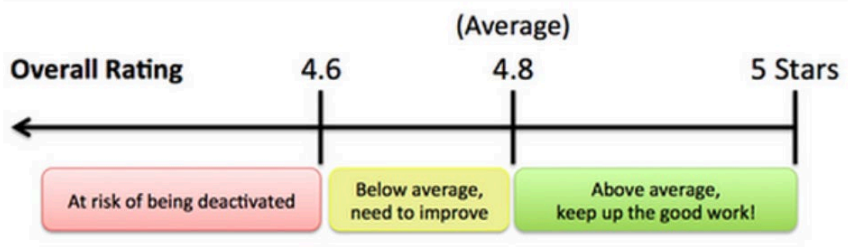




When is it okay to truncate?

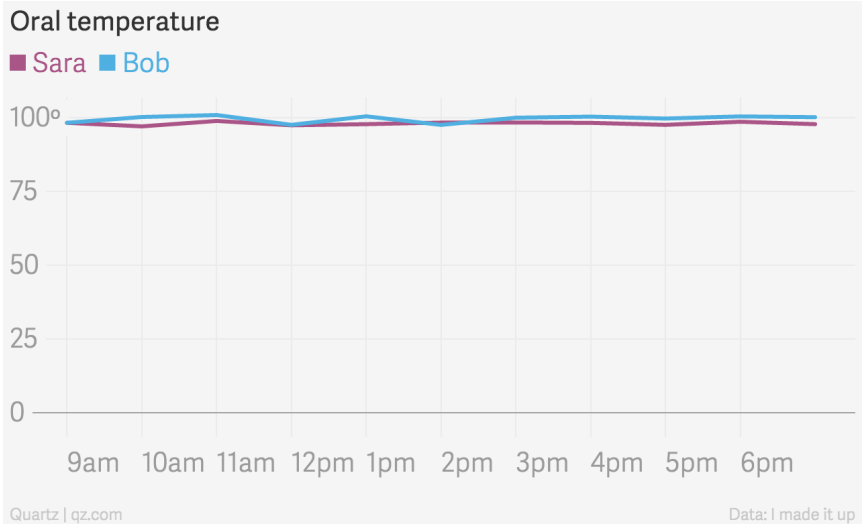
When the scale itself is distorted

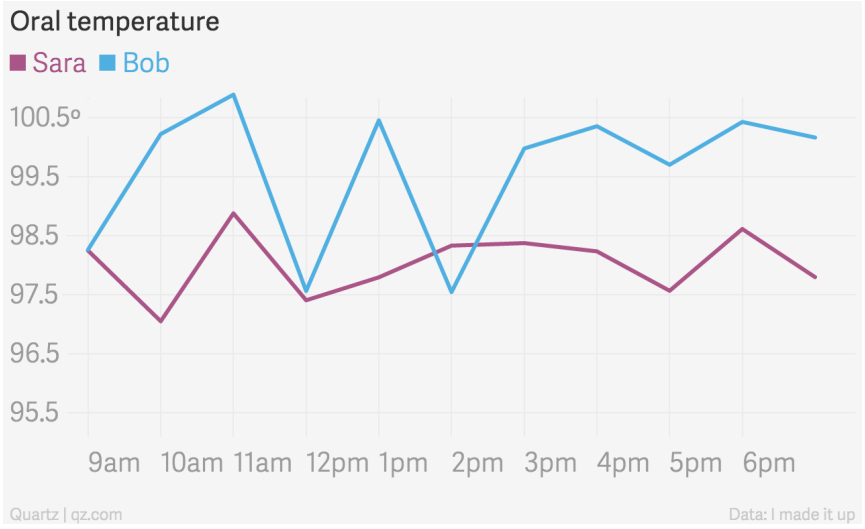




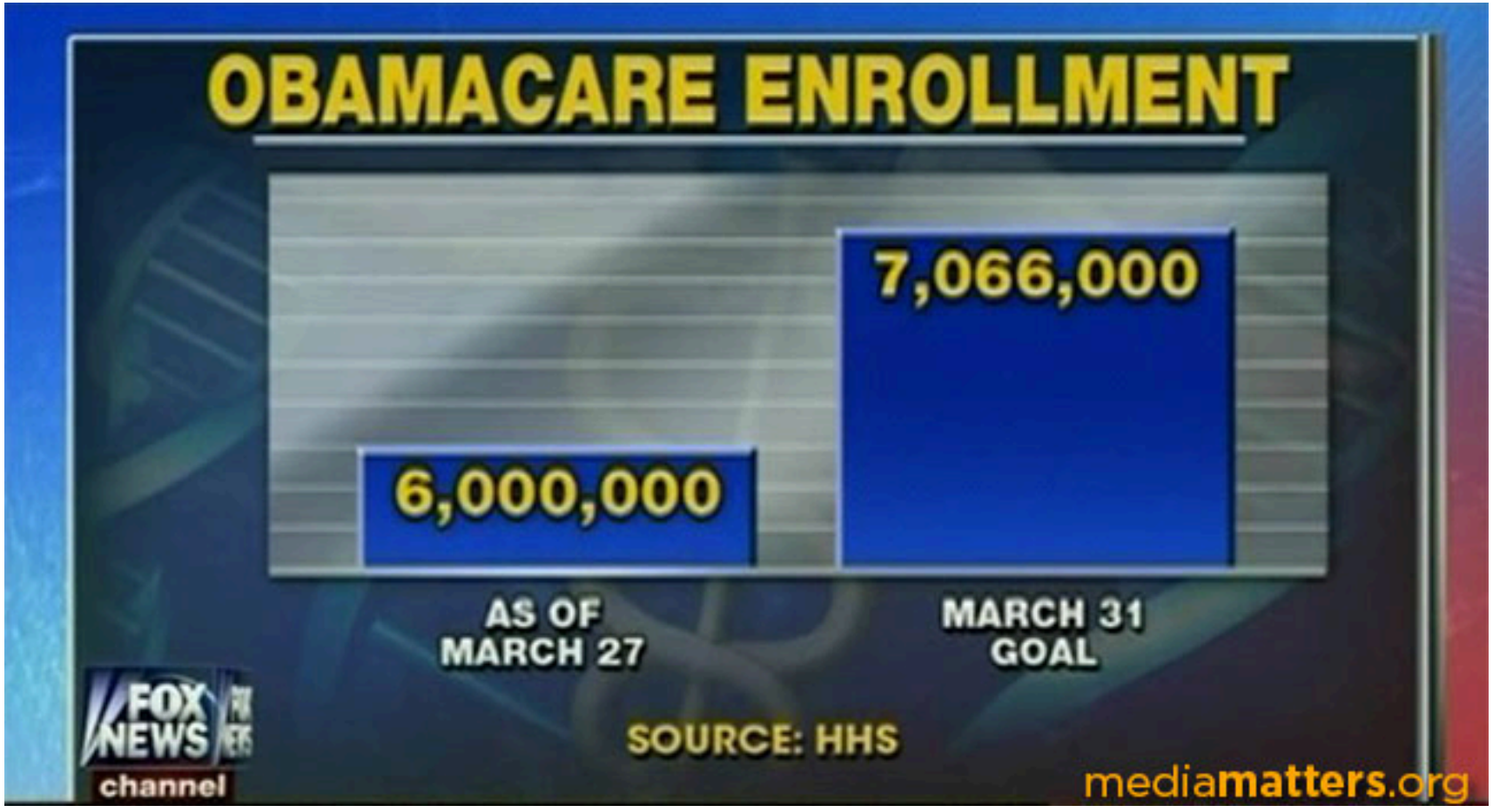
When is it okay to truncate?

When zero values are impossible





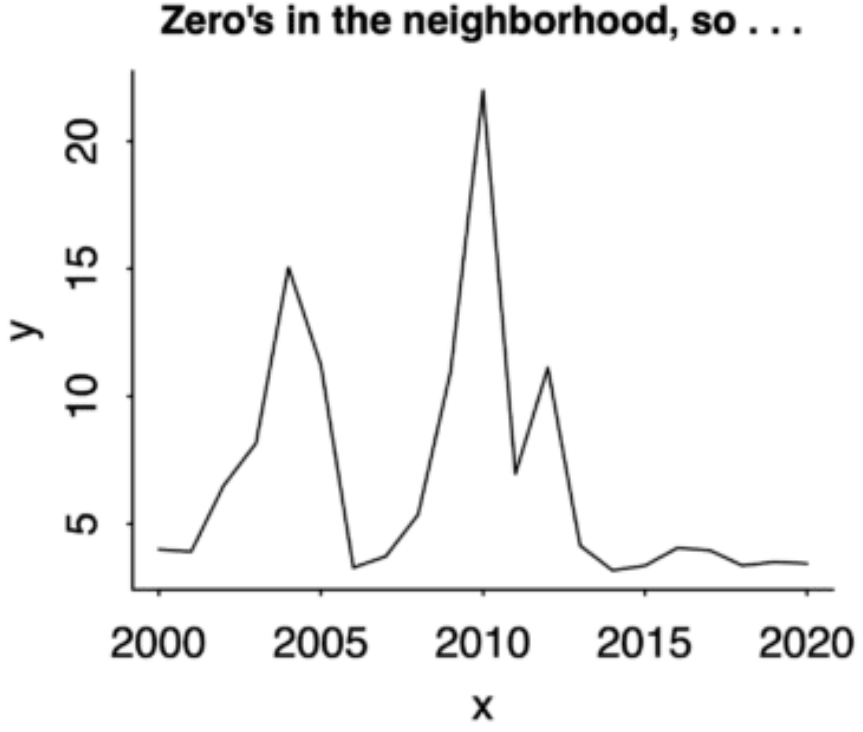
Never on bar charts!

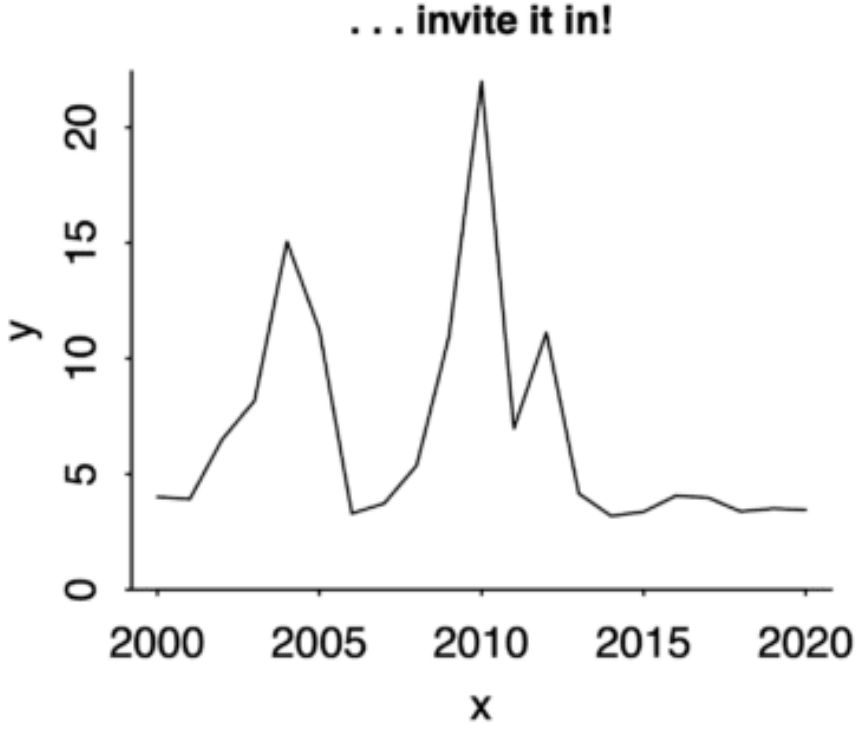


Zero is okay, too

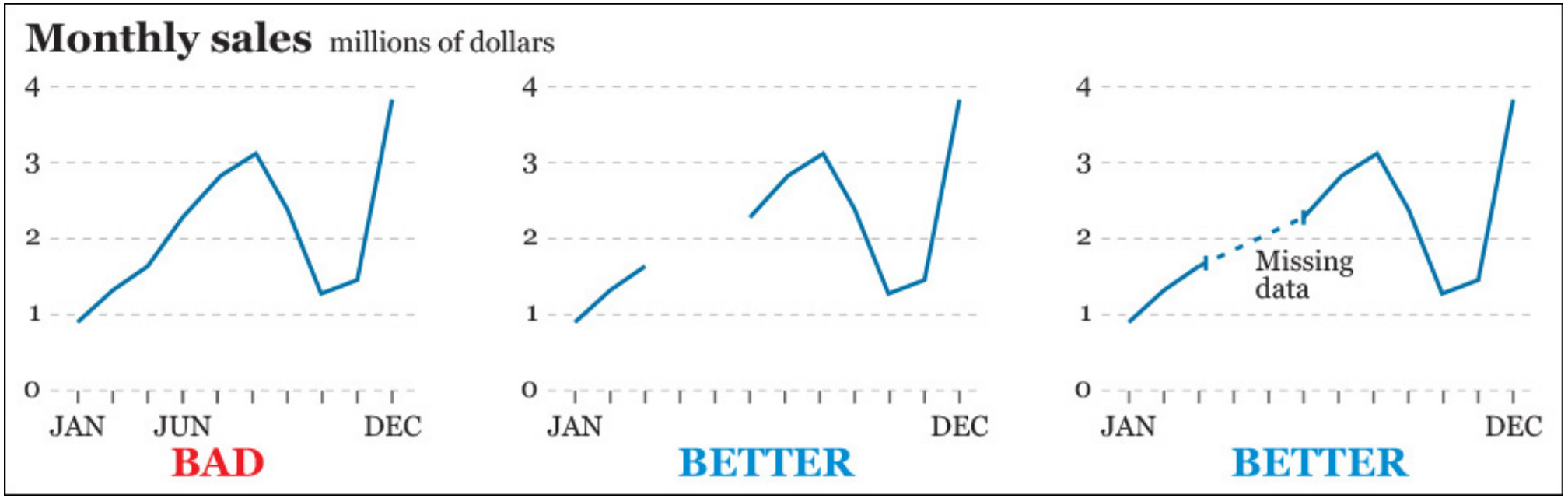
Just because you don't *have to* start at 0 doesn't mean you should *never* start at 0

Andrew Gelman's heuristic: **"If zero is in the neighborhood, invite it in."**

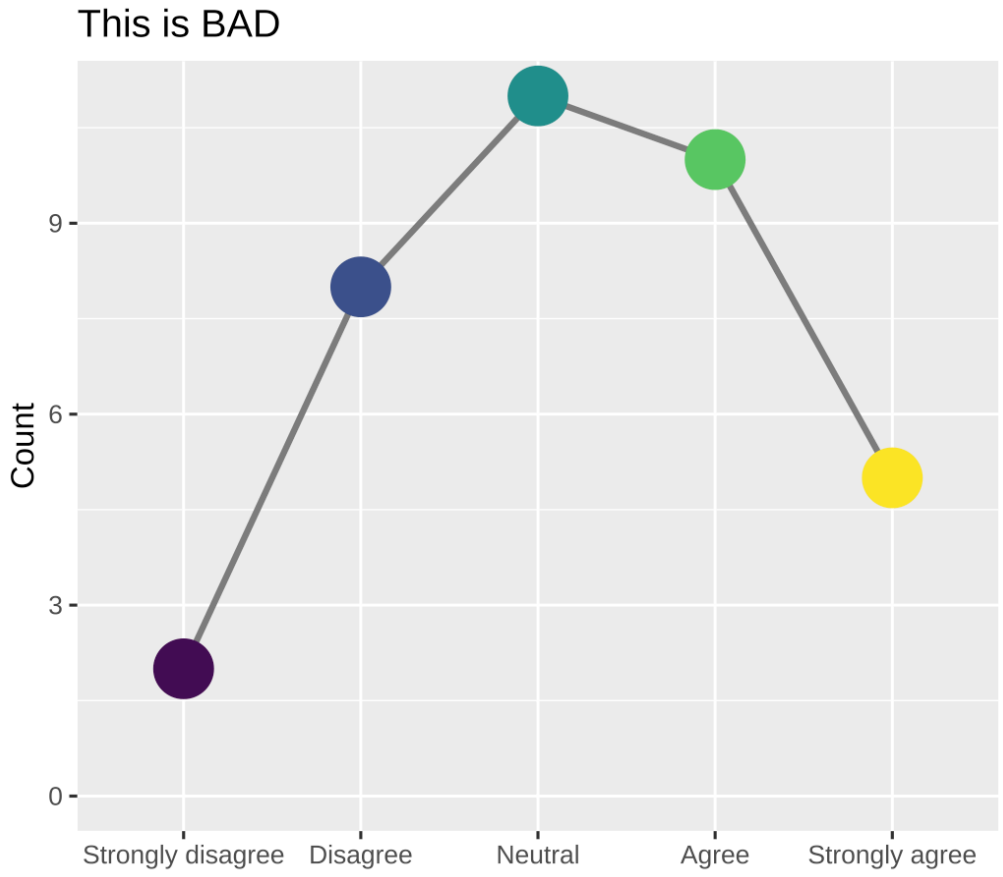


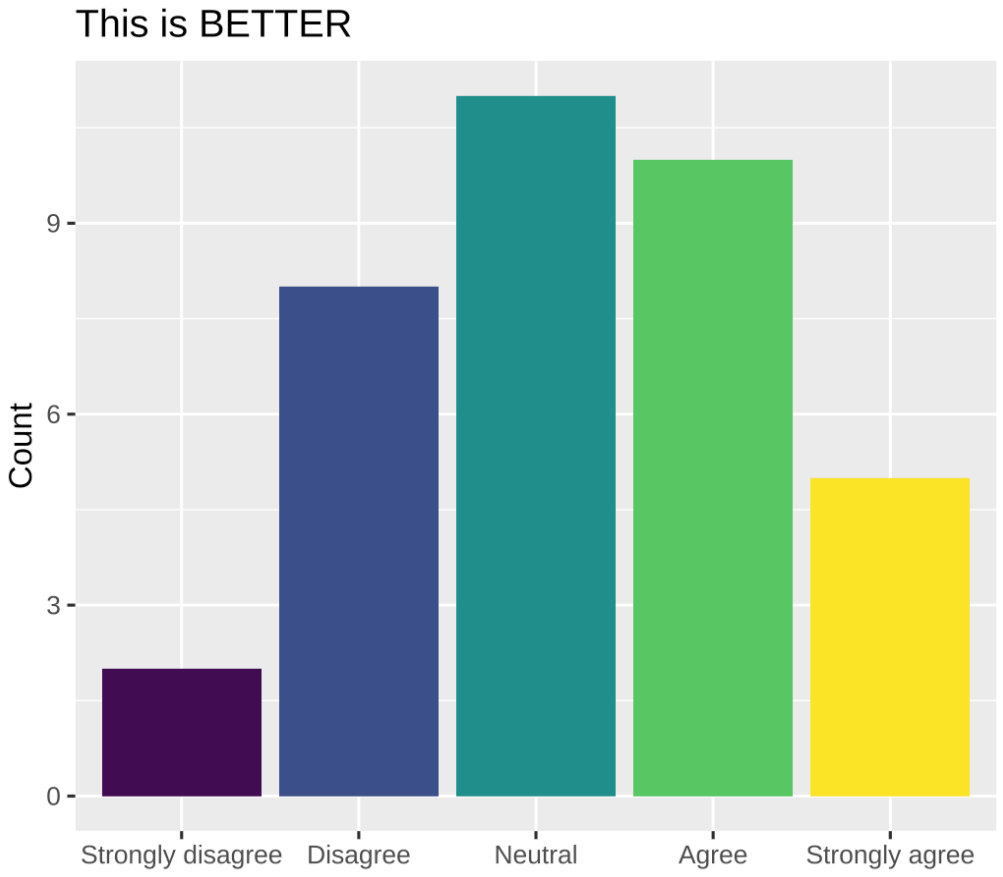


Keep date scales consistent, flag missing data



Aside: don't impute across categories





Visualizing time: the basics

Visualizing amounts over time

Time is just a variable that can be mapped to an aesthetic

Can be used as `x`, `y`, `color`, `fill`, `facet`, and even animation

Can use all sorts of `geom`s: lines, columns, points, heatmaps, densities, maps, etc.

In general, follow reading conventions to show time progression:

→ & ↓

Visualizing amounts over time using ggplot

Let's use GameStop share prices to make some plots:

```
gme_prices
```

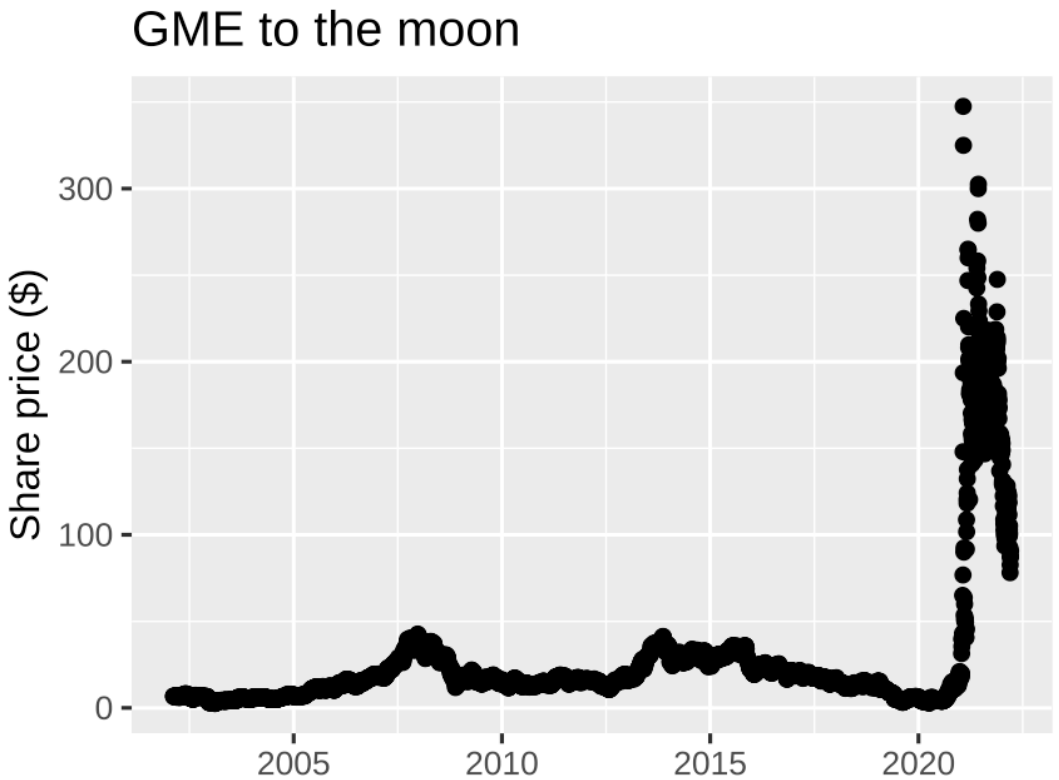
```
## # A tibble: 5,060 × 8
##    symbol date        open  high   low close   volume adjusted
##    <chr>  <date>     <dbl> <dbl> <dbl> <dbl>    <dbl>    <dbl>
##  1 GME    2002-02-13  9.62 10.1   9.52 10.0  19054000     6.77
##  2 GME    2002-02-14 10.2  10.2   9.93 10     2755400     6.73
##  3 GME    2002-02-15 10    10.0   9.85  9.95  2097400     6.70
##  4 GME    2002-02-19  9.9   9.9   9.38  9.55  1852600     6.43
##  5 GME    2002-02-20  9.6   9.88  9.52  9.88  1723200     6.65
##  6 GME    2002-02-21  9.84  9.93  9.75  9.85  1744200     6.63
##  7 GME    2002-02-22  9.93  9.93  9.6   9.68   881400     6.51
##  8 GME    2002-02-25  9.65  9.82  9.54  9.75   863400     6.56
##  9 GME    2002-02-26  9.7   9.85  9.54  9.75   690400     6.56
## 10 GME    2002-02-27  9.68  9.68  9.5   9.57  1022800     6.45
## # ℹ 5,050 more rows
```

How would you use these data to plot `adjusted` prices over time?

Time on x-axis + `geom_point()`

```
gme_prices |>
  ggplot(aes(x = date, y = adjusted)) +
  geom_point() +
  labs(
    x = NULL,
    y = "Share price ($)",
    title = "GME to the moon"
  )
```

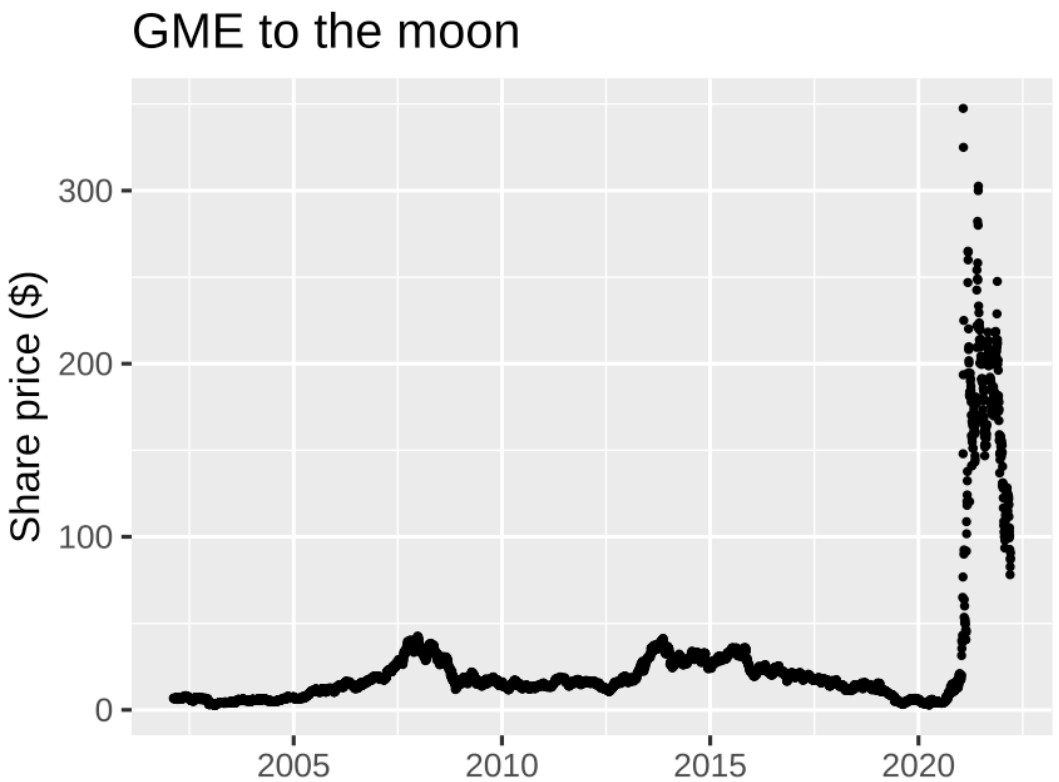
How could you reduce point overlap?



Time on x-axis + `geom_point()`

```
gme_prices |>
  ggplot(aes(x = date, y = adjusted)) +
  geom_point(size = 0.5) +
  labs(
    x = NULL,
    y = "Share price ($)",
    title = "GME to the moon"
  )
```

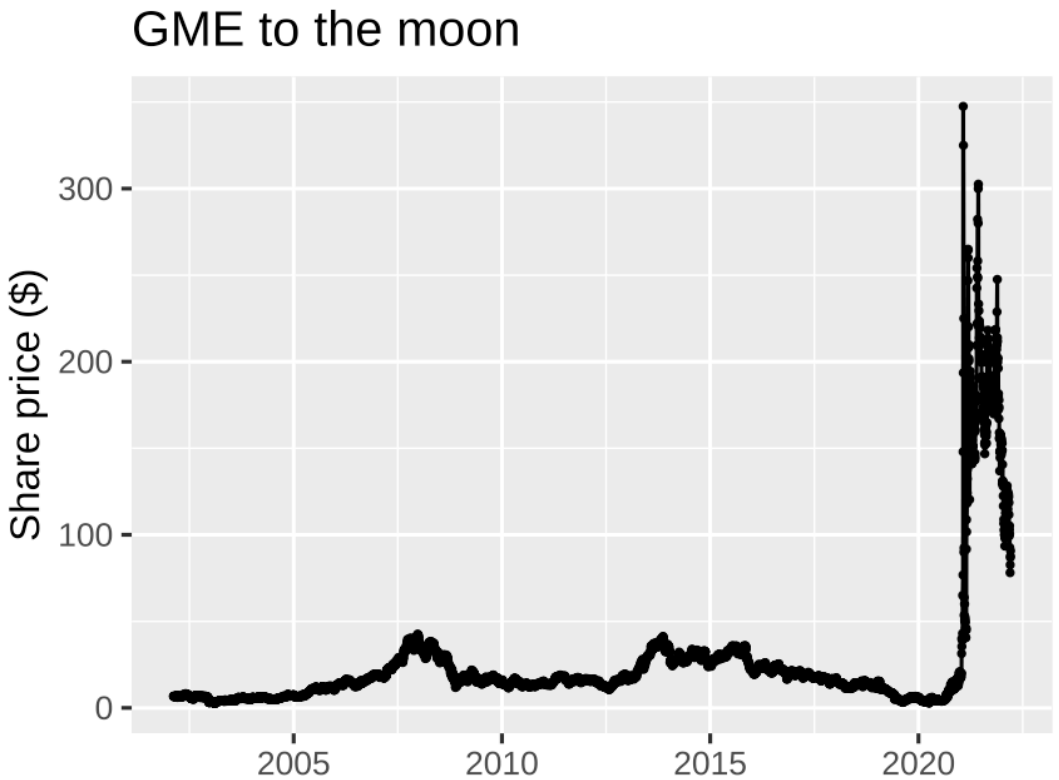
How could you add a line to this?



Time on x-axis + `geom_point()`

```
gme_prices |>
  ggplot(aes(x = date, y = adjusted)) +
  geom_point(size = 0.5) +
  geom_line() +
  labs(
    x = NULL,
    y = "Share price ($)",
    title = "GME to the moon"
  )
```

Which is better, points or lines?



Time on x-axis: points vs lines

Which is better?

It depends!

Points emphasize observations

Lines emphasize trends

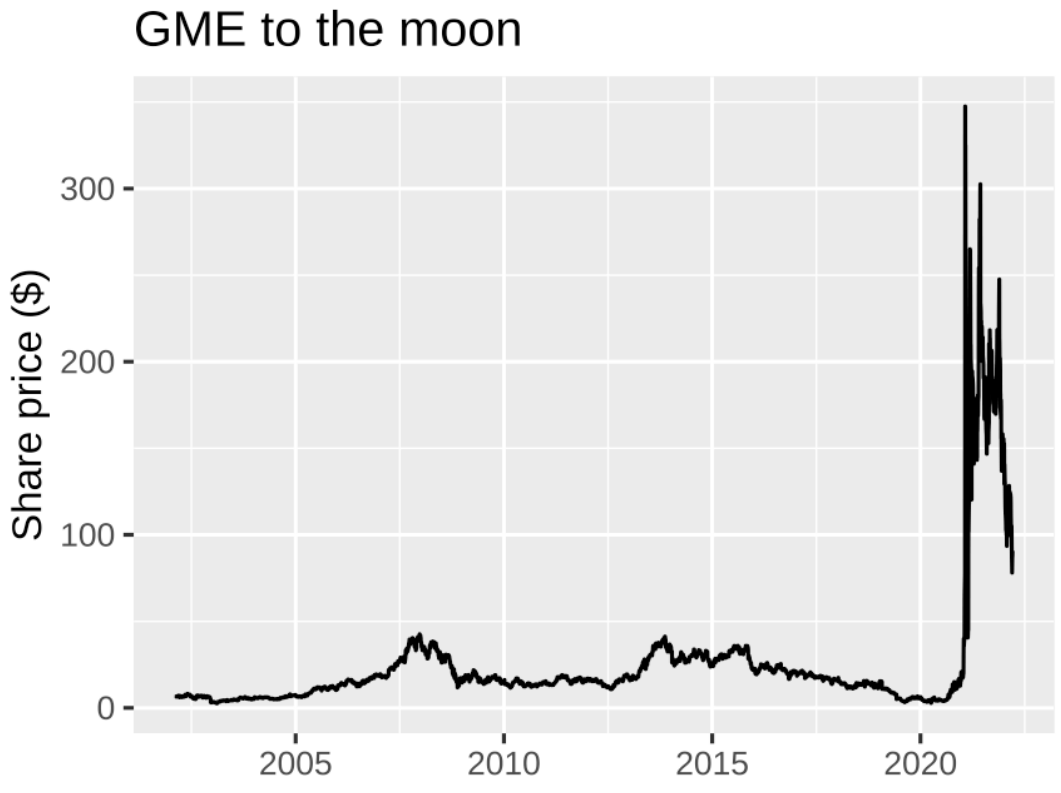
Using lines for time series data is common since data are evenly spaced and usually complete

But lines are effectively made-up data, so be careful how you use them!

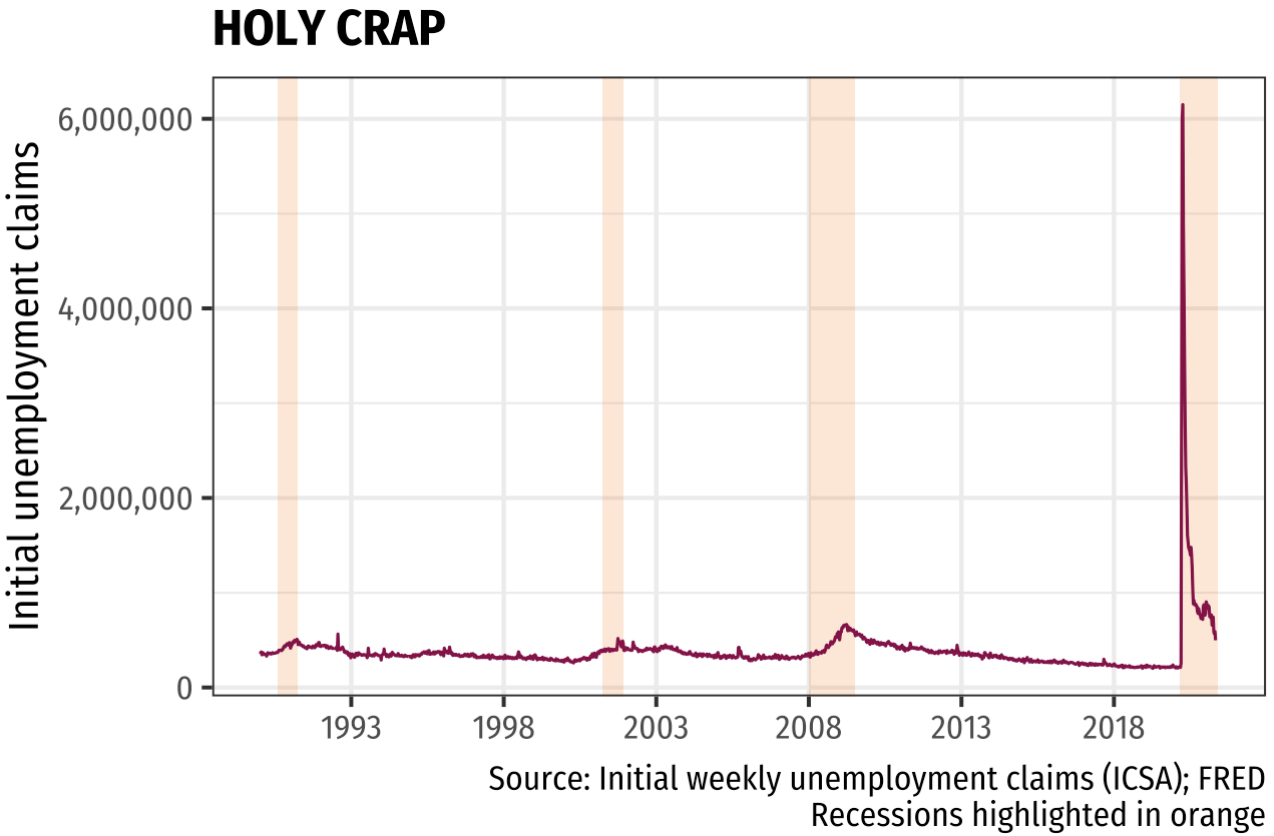
Time on x-axis + `geom_line()`

The GameStop share price plot is clearer without points:

```
gme_prices |>
  ggplot(aes(x = date, y = adjusted)) +
  geom_line() +
  labs(
    x = NULL,
    y = "Share price ($)",
    title = "GME to the moon"
  )
```



Line plots don't have to be boring



Dates and times in R

Are dates and times special?

We just made some plots without really thinking about data types

We can map "dates" to x regardless of whether they are `date` or `numeric` objects

When do we need to pay attention to the details?

1. Converting from strings to dates
2. Getting components of date-time data
3. Computing time spans

Dates and times in R

R has "native" classes for storing calendar dates and times

- For background, see `?Dates` and `?DateTimeClasses`

The `lubridate` package offers convenient tools for working with dates and times

- Loaded automatically as part of the core tidyverse since tidyverse 2.0.0
- See Chapter 17 of R4DS (2e)

Converting from strings to dates

Use `lubridate::ymd` to parse dates with **y**ear, **m**onth, and **d**ay components

```
svb_failure <- ymd("2023-03-10")
svb_failure
```

```
## [1] "2023-03-10"
```

```
class("2023-03-10")
```

```
## [1] "character"
```

```
class(svb_failure)
```

```
## [1] "Date"
```

Converting from strings to dates

`lubridate` offers many other functions for parsing dates

For example, instead of `ymd` we could use `mdy`:

```
signature_failure <- mdy("March 12, 2023")
```

```
signature_failure
```

```
## [1] "2023-03-12"
```

```
class(signature_failure)
```

```
## [1] "Date"
```

Getting components of date-time data

What year did Silicon Valley Bank fail?

```
year(svb_failure)
```

```
## [1] 2023
```

What day of the week was that?

```
wday(svb_failure)
```

```
## [1] 6
```

```
wday(svb_failure, label = TRUE)
```

```
## [1] Fri
## Levels: Sun < Mon < Tue < Wed < Thu < Fri < Sat
```

What month was that?

```
month(svb_failure)
```

```
## [1] 3
```

```
month(svb_failure, label = TRUE)
```

```
## [1] Mar
## 12 Levels: Jan < Feb < Mar < Apr < May < Jun < Jul
```

Computing time spans

How many days passed between the failures of SVB and Signature?

```
signature_failure - svb_failure
```

```
## Time difference of 2 days
```

How many days are left until classes end?

```
ymd(20251204) - today()
```

```
## Time difference of 44 days
```

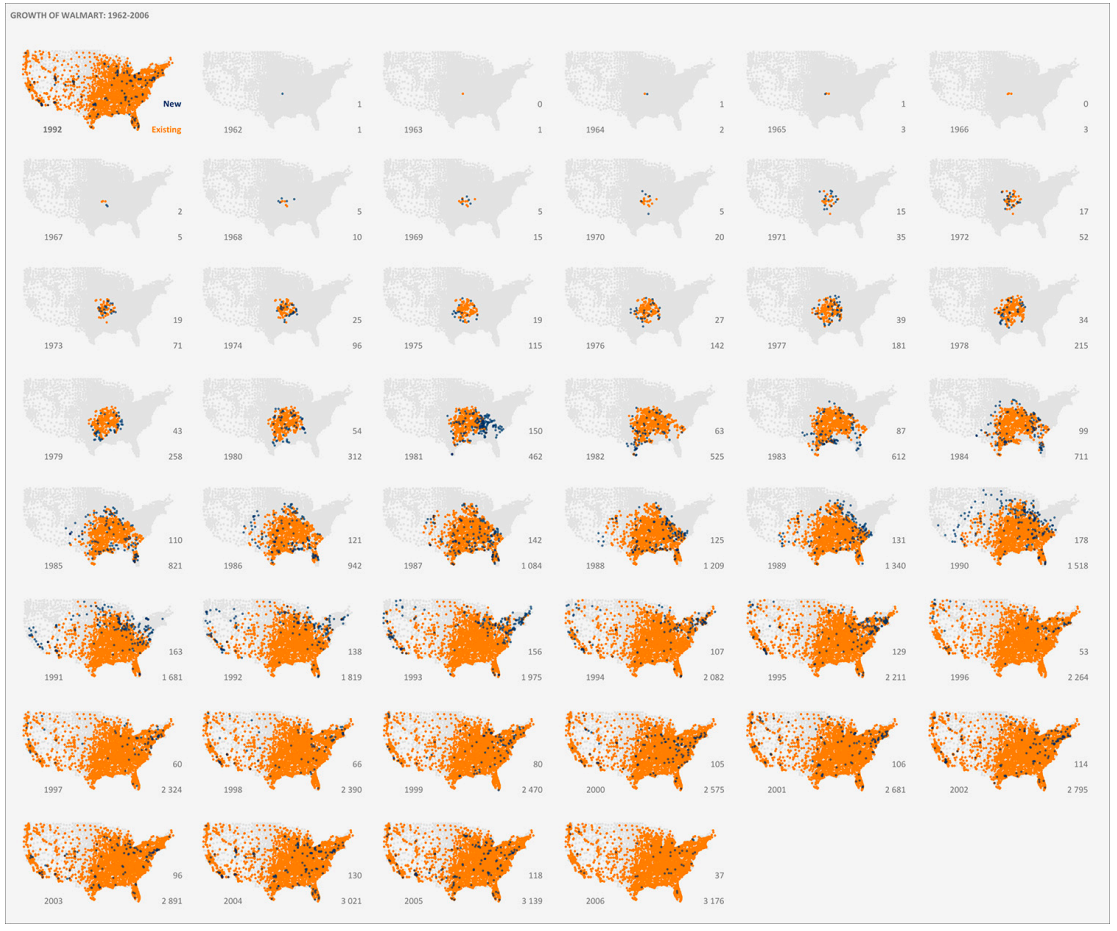
These are **durations**

See Chapter 17 of R4DS (2e) for functions to compute **periods** and **intervals**

example-09:
time-practice.R

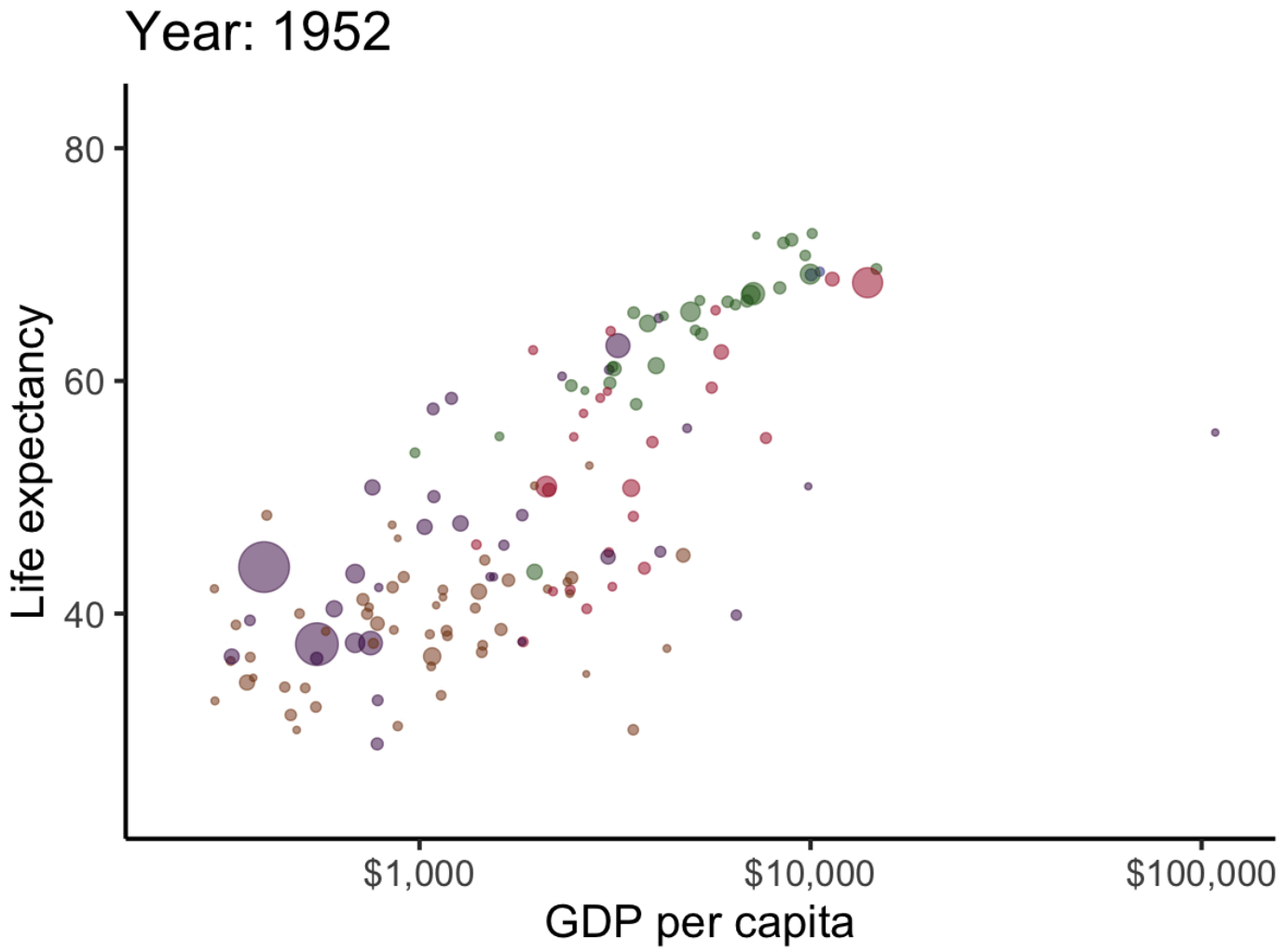
Visualizing time: leveling up

Faceting plots over time



Map of the spread of Walmart by Jorge Camões

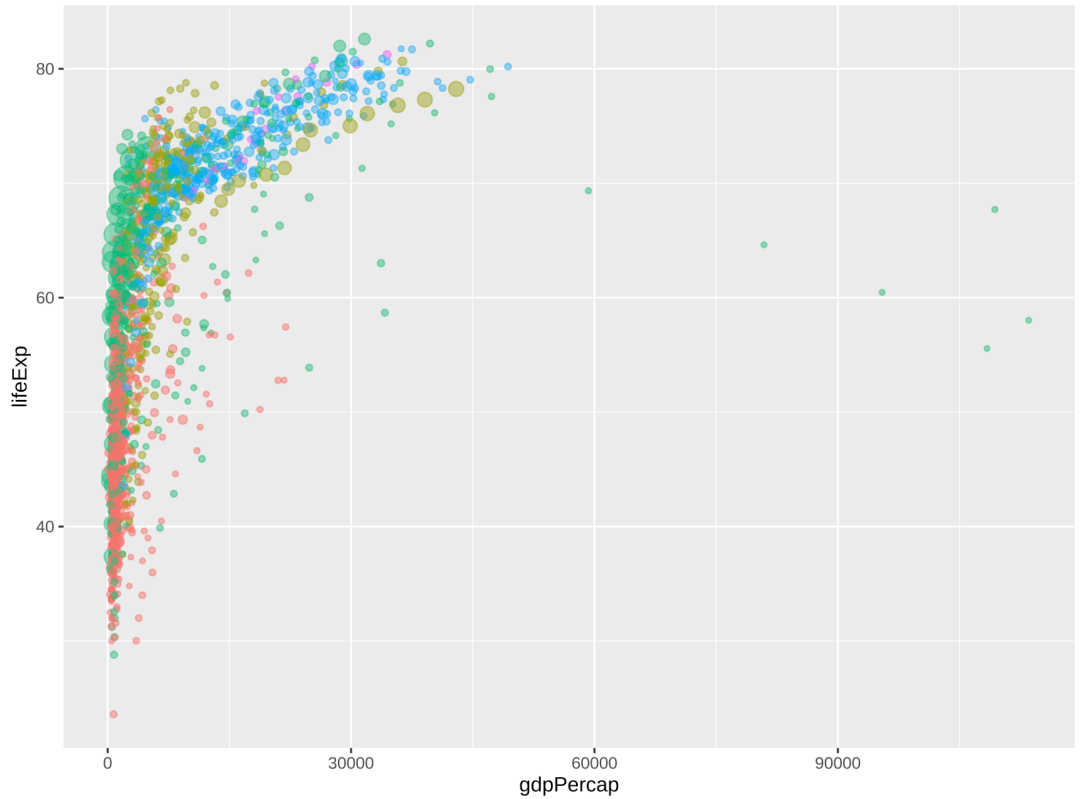
Animating plots over time



How can we make this animation in R?

First, how would we make a static plot of the data?

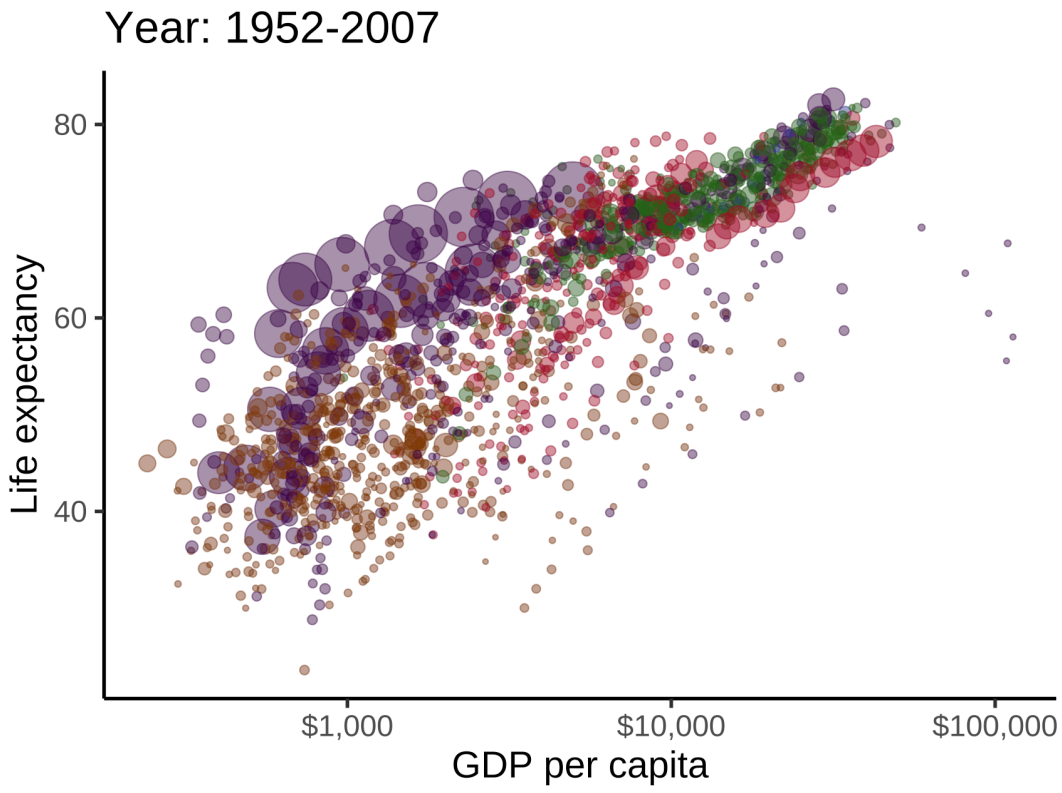
```
library(gapminder) # gapminder data

gapminder |>
  ggplot(aes(x = gdpPercap, y = lifeExp,
             size = pop, color = continent)) +
  geom_point(alpha = 0.5, show.legend = FALSE)
```



How can we make this animation in R?

Let's clean this up a bit to match the gapminder formatting:

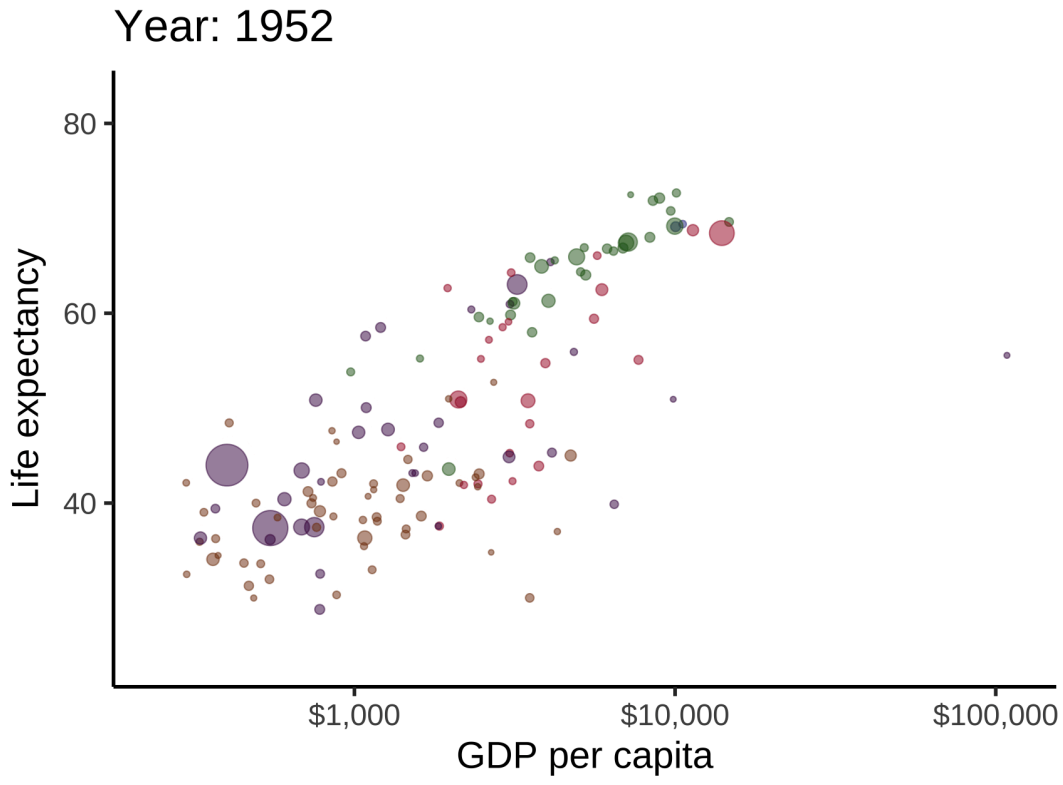
```
library(gapminder) # gapminder data

gapminder |>
  ggplot(aes(x = gdpPercap, y = lifeExp,
             size = pop, color = continent)) +
  geom_point(alpha = 0.5, show.legend = FALSE) +
  scale_color_manual(
    values = gapminder::continent_colors
  ) +
  scale_x_log10(labels = label_dollar()) +
  scale_size_continuous(range = c(1, 15)) +
  theme_classic(base_size = 20) +
  labs(x = "GDP per capita",
       y = "Life expectancy",
       title = "Year: 1952-2007")
```



Converting a static plot to an animation

Second, let's use `gganimate` to visualize changes over time

```
library(gganimate) # plot animation package

gapminder |>
  ggplot(aes(x = gdpPercap, y = lifeExp,
             size = pop, color = continent)) +
  geom_point(alpha = 0.5, show.legend = FALSE) +
  scale_color_manual(
    values = gapminder::continent_colors
  ) +
  scale_x_log10(labels = label_dollar()) +
  scale_size_continuous(range = c(1, 15)) +
  theme_classic(base_size = 20) +
  labs(x = "GDP per capita",
       y = "Life expectancy",
       title = "Year: {frame_time}") +
  transition_time(year) +
  ease_aes('linear') # default progression
```



Saving gifs

Third, use `anim_save()` to write a `.gif` that you can text to your mom

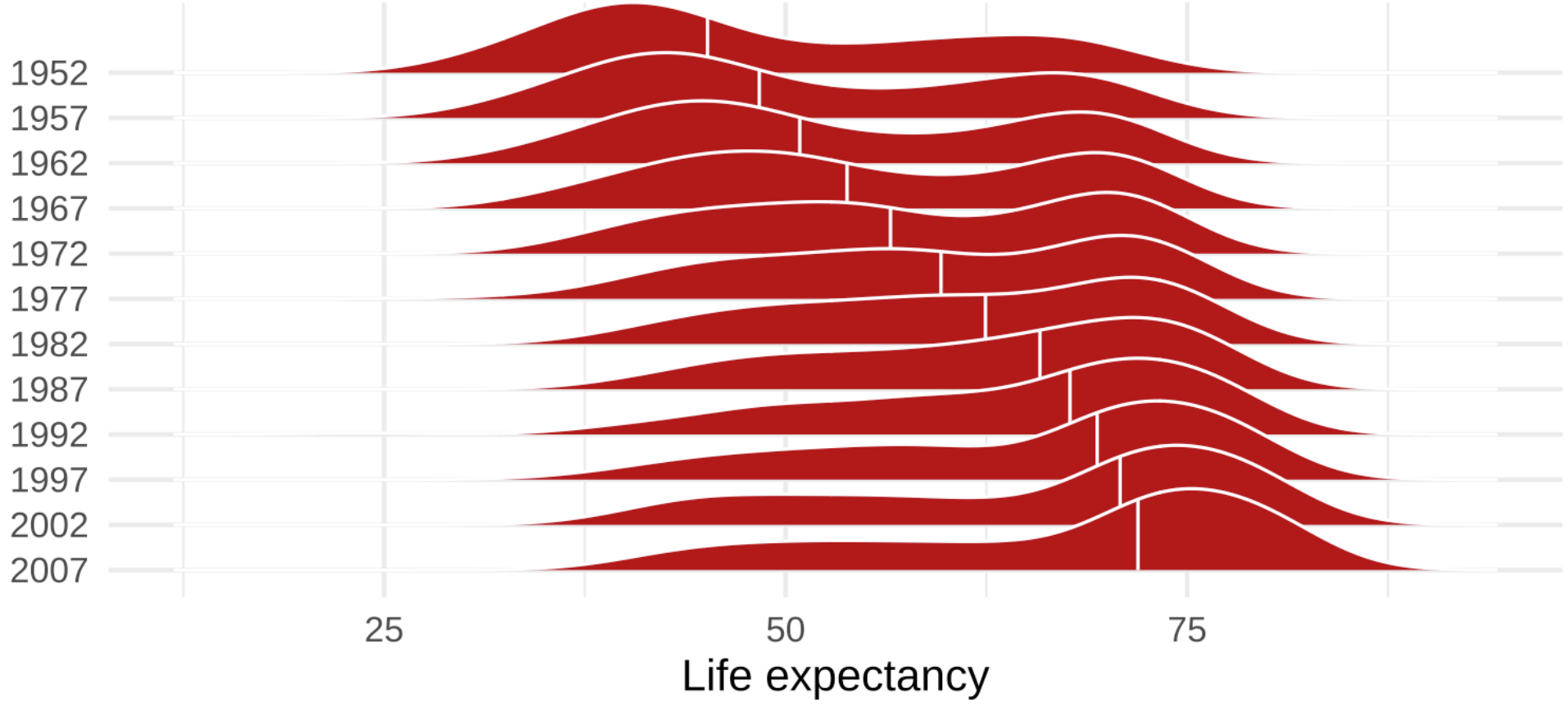
```
library(gganimate)
library(gifski)

my_gapminder_animation <- gapminder |>
  ggplot(aes(x = gdpPercap, y = lifeExp, size = pop, color = continent)) +
  geom_point(alpha = 0.5, show.legend = FALSE) +
  scale_color_manual(values = gapminder::continent_colors) +
  scale_x_log10(labels = label_dollar()) +
  scale_size_continuous(range = c(1, 15)) +
  theme_classic(base_size = 20) +
  labs(x = "GDP per capita",
       y = "Life expectancy",
       title = "Year: {frame_time}") +
  transition_time(year) +
  ease_aes('linear')

animate(my_gapminder_animation, renderer = gifski_renderer())
anim_save("content/slides/img/09/my-gapminder-animation.gif")
```

example-09:
time-practice.R

Visualizing distributions over time (for reference)

Visualizing distributions over time

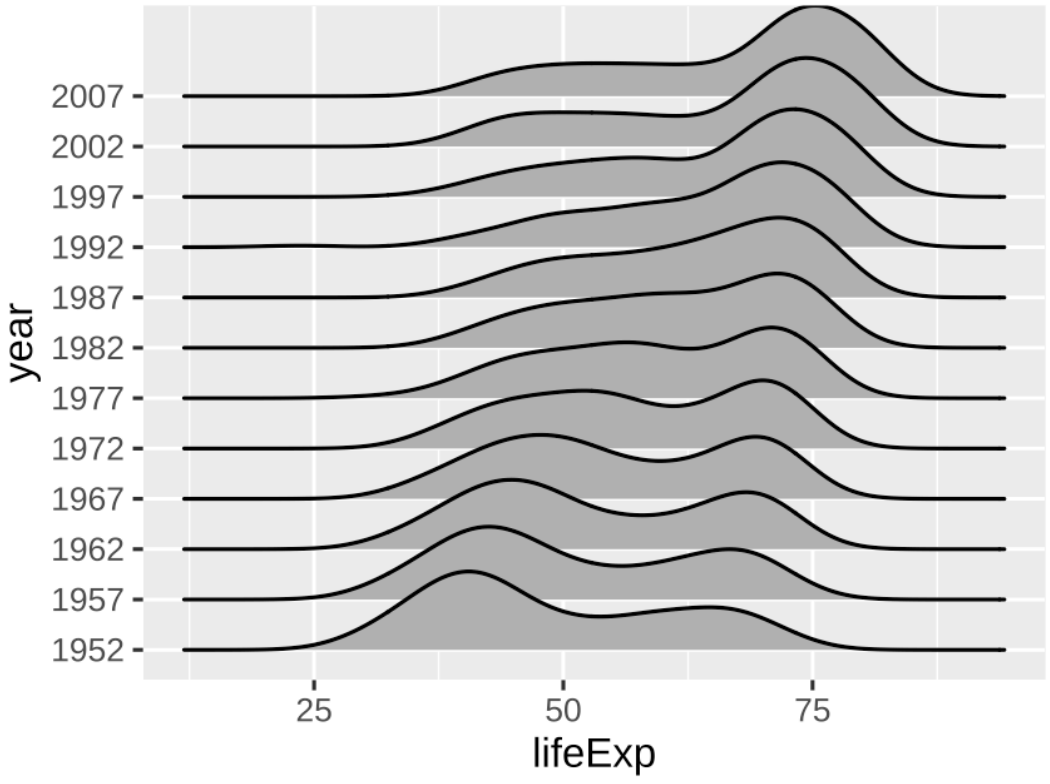
We can also visualize how distributions evolve. How could you make this plot?



Let's start simple

```
library(ggridges) # for geom_density_ridges()

gapminder |>
  mutate(year = as_factor(year)) |>
  ggplot(aes(
    x = lifeExp,
    y = year # map time to y
  )) +
  geom_density_ridges() # add geom
```

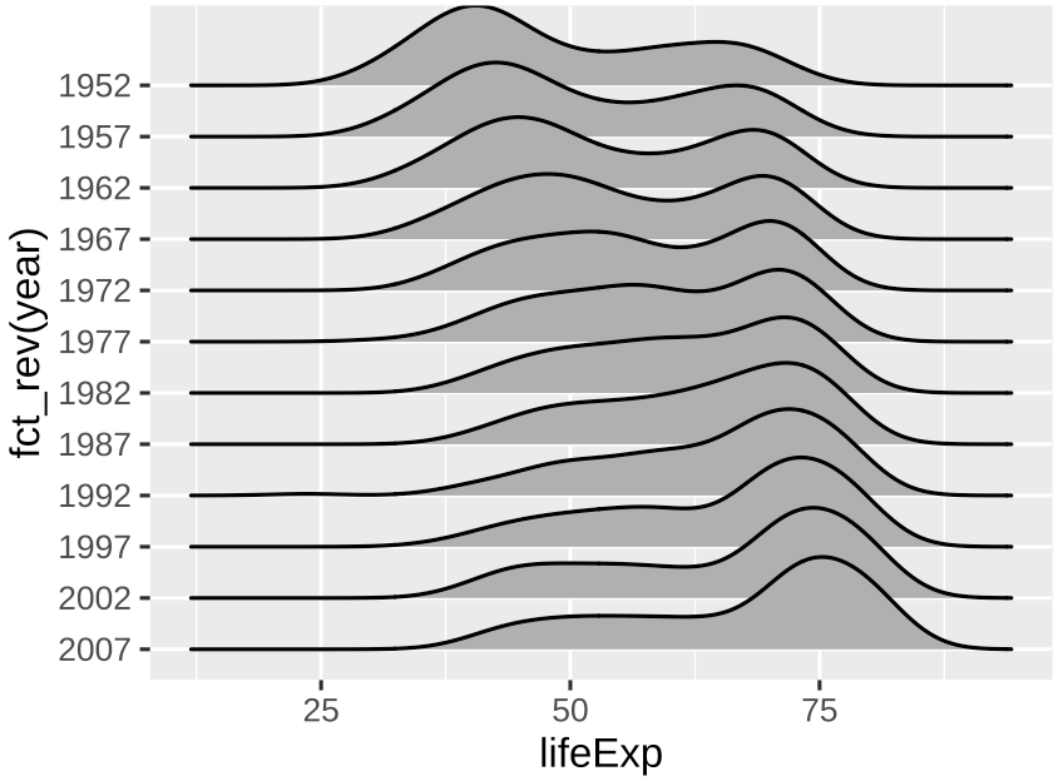
How could we modify this to follow ↓ convention for time?



Follow ↓ convention for time

```
gapminder |>
  mutate(year = as_factor(year)) |>
  ggplot(aes(
    x = lifeExp,
    y = fct_rev(year) # reverse year
  )) +
  geom_density_ridges()
```

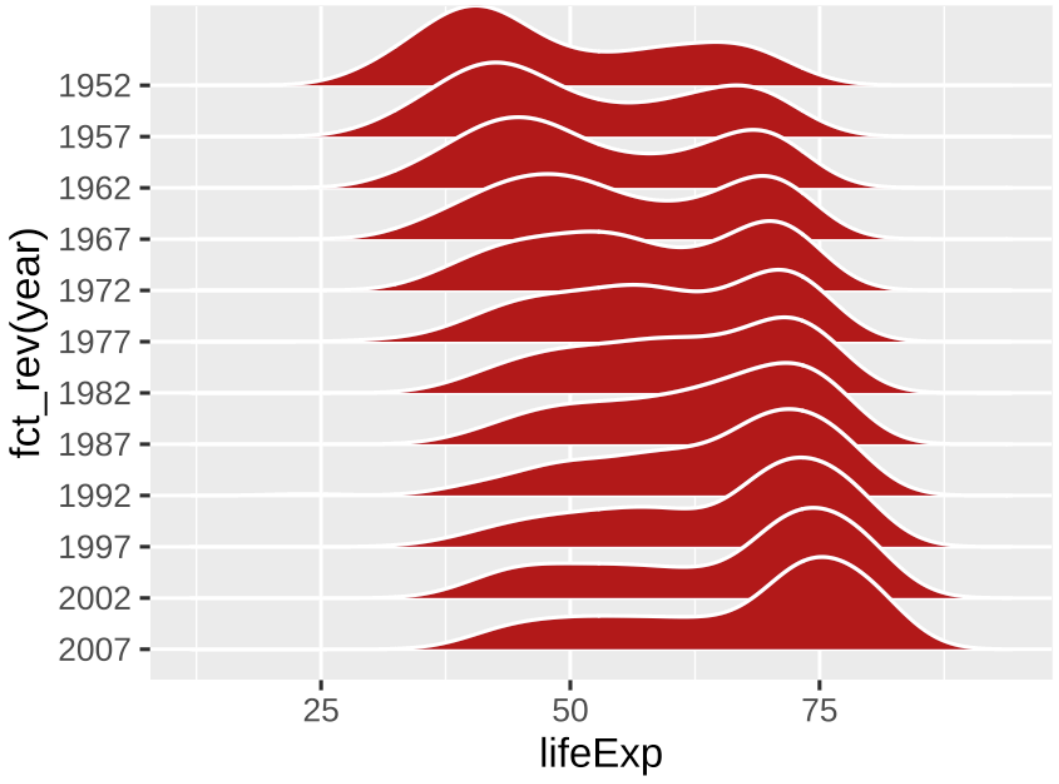
How could we make the lines white?



Add color to help visually separate densities

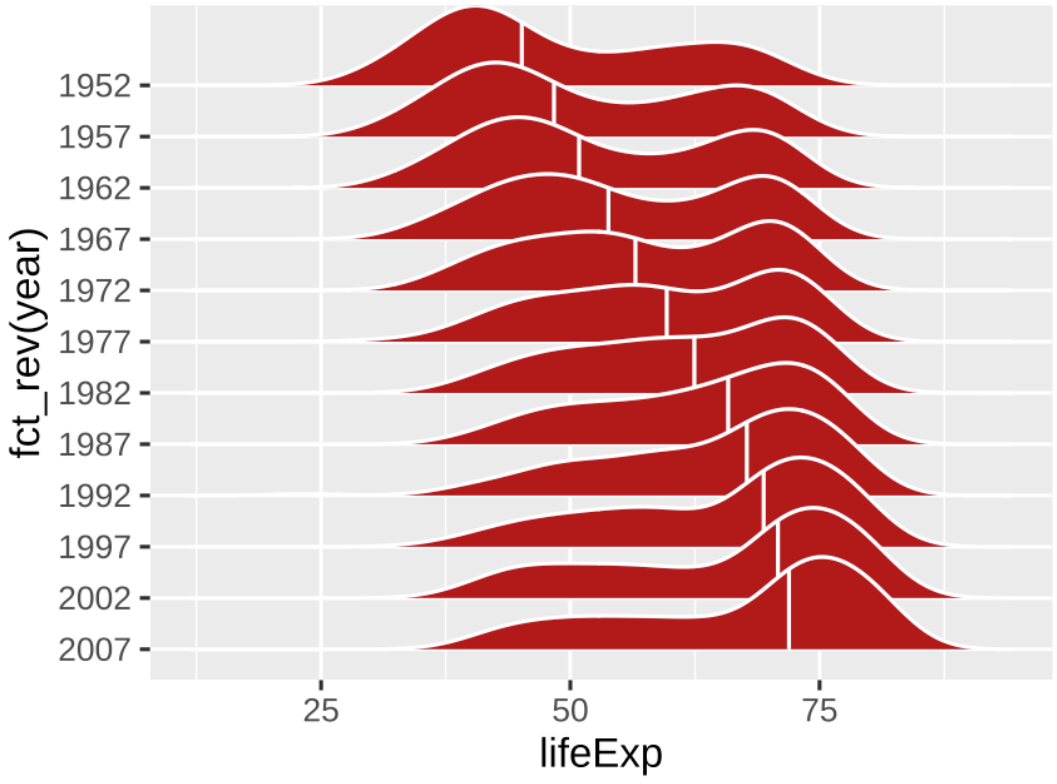
```
gapminder |>
  mutate(year = as_factor(year)) |>
  ggplot(aes(
    x = lifeExp,
    y = fct_rev(year)
  )) +
  geom_density_ridges(
    color = "white",  # separate densities
    fill = "#B31B1B"
  )
```

How could we add a line at the median of each density?



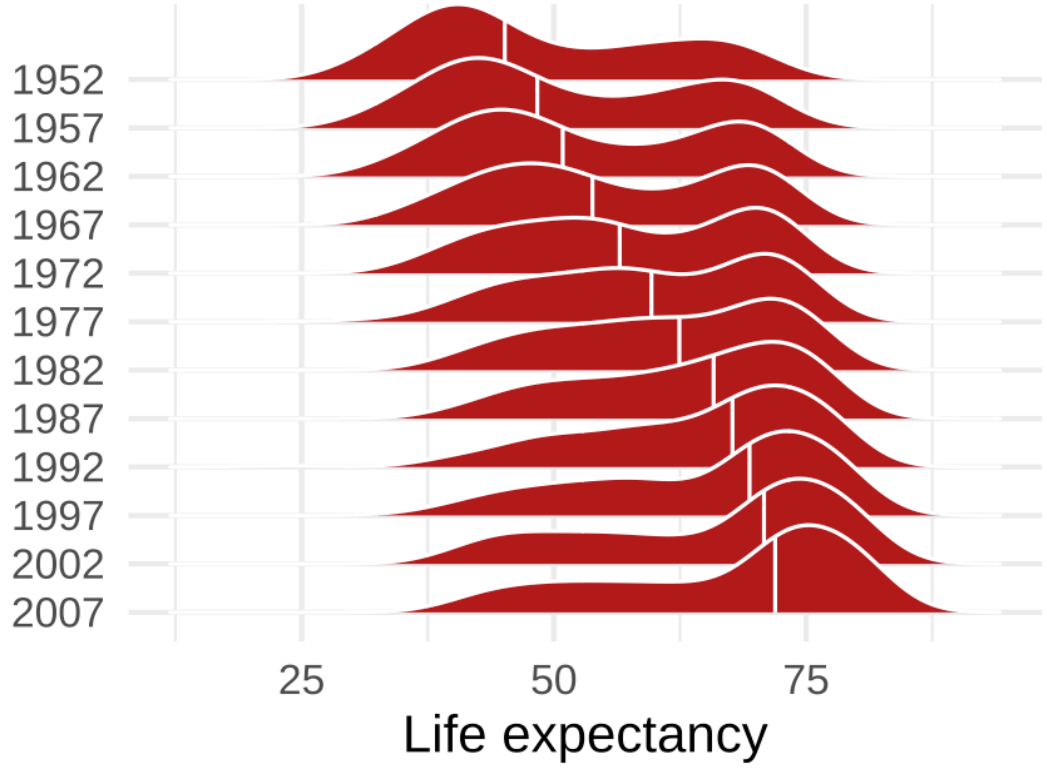
Add a line at the median of each density

```
gapminder |>
  mutate(year = as_factor(year)) |>
  ggplot(aes(
    x = lifeExp,
    y = fct_rev(year)
  )) +
  geom_density_ridges(
    color = "white",
    fill = "#B31B1B",
    quantile_lines = TRUE, # add lines
    quantiles = 2          # median
  )
```



Finish by cleaning the plot up

```
gapminder |>
  mutate(year = as_factor(year)) |>
  ggplot(aes(
    x = lifeExp,
    y = fct_rev(year)
  )) +
  geom_density_ridges(
    color = "white",
    fill = "#B31B1B",
    quantile_lines = TRUE,
    quantiles = 2
  ) +
  labs(
    x = "Life expectancy",# modify label
    y = NULL              # omit labels
  ) +
  theme_minimal(          # cleaner theme
    base_size = 14        # larger font
  )
```



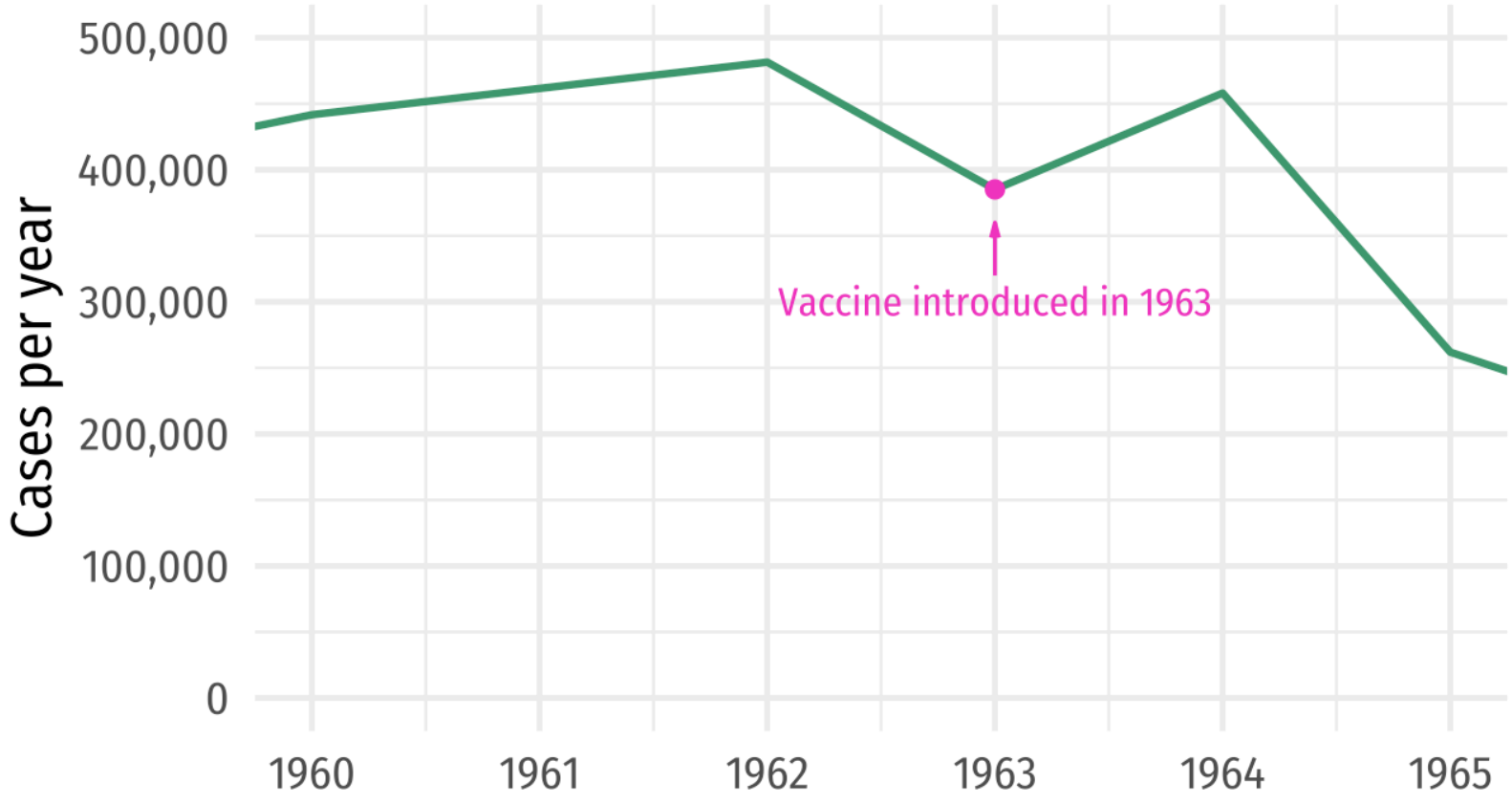
Starting, ending, and decomposing time (for reference)

You have to start (and end) somewhere

You always have to choose start and end points

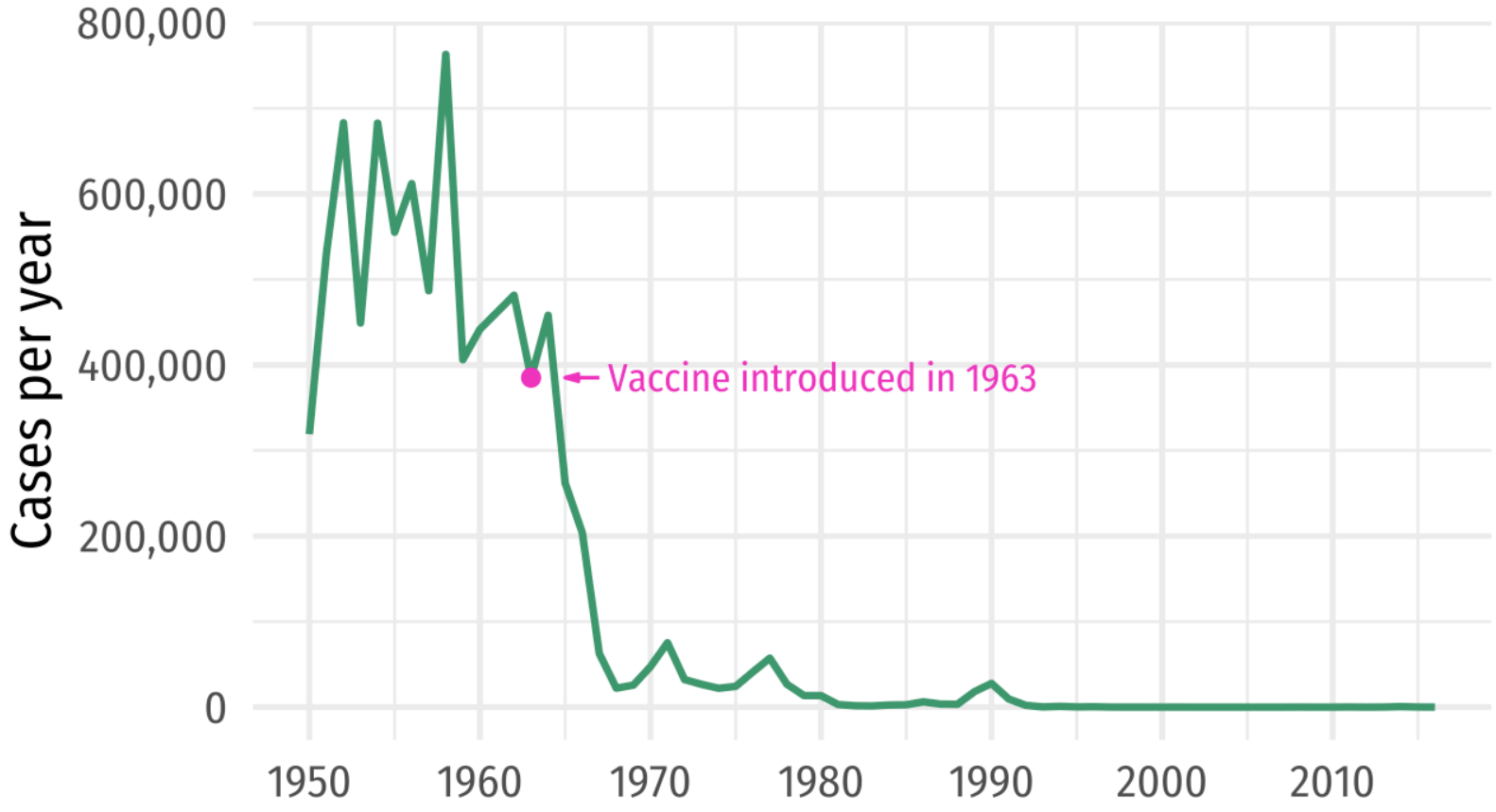
Start and end at reasonable times that help maintain the context of your story

Measles vaccine was pretty effective



Source: CDC, Epidemiology and Prevention of Vaccine-Preventable Diseases, 13th Edition

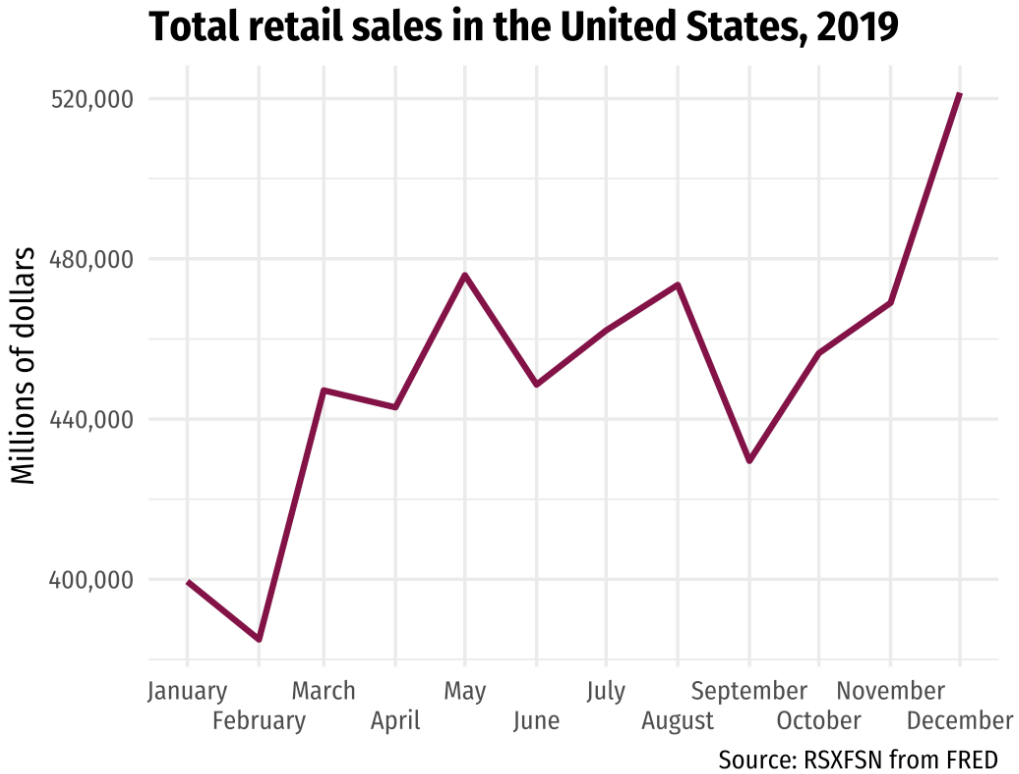
Measles vaccine was *incredible*!

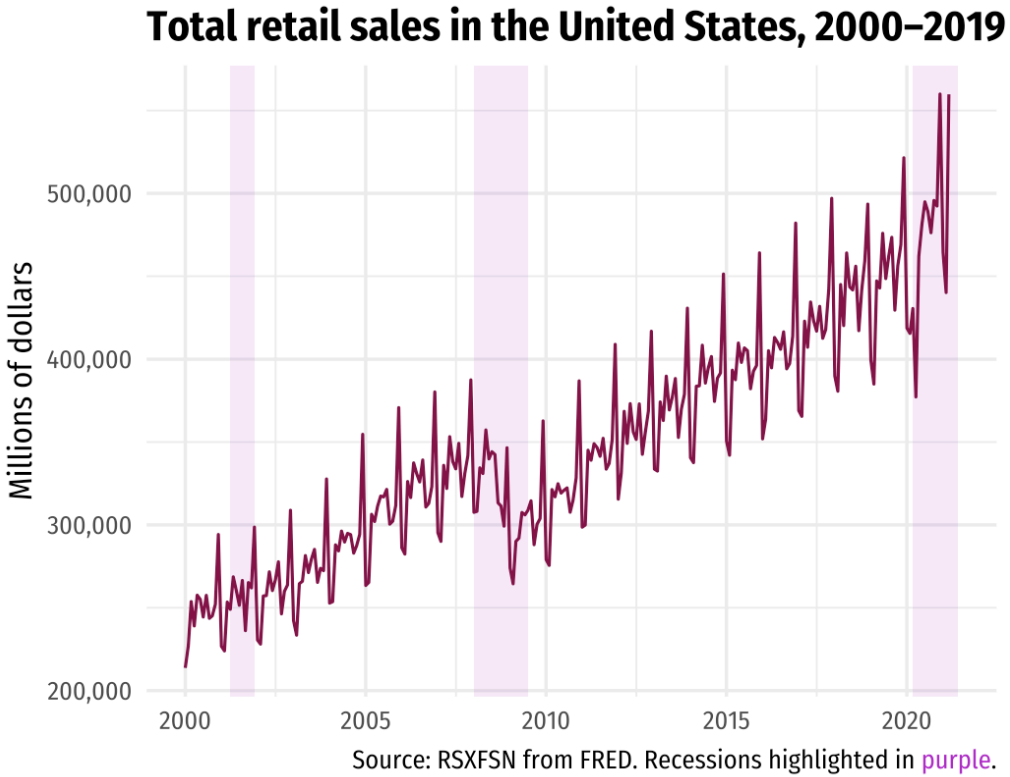


Source: CDC, Epidemiology and Prevention of Vaccine-Preventable Diseases, 13th Edition

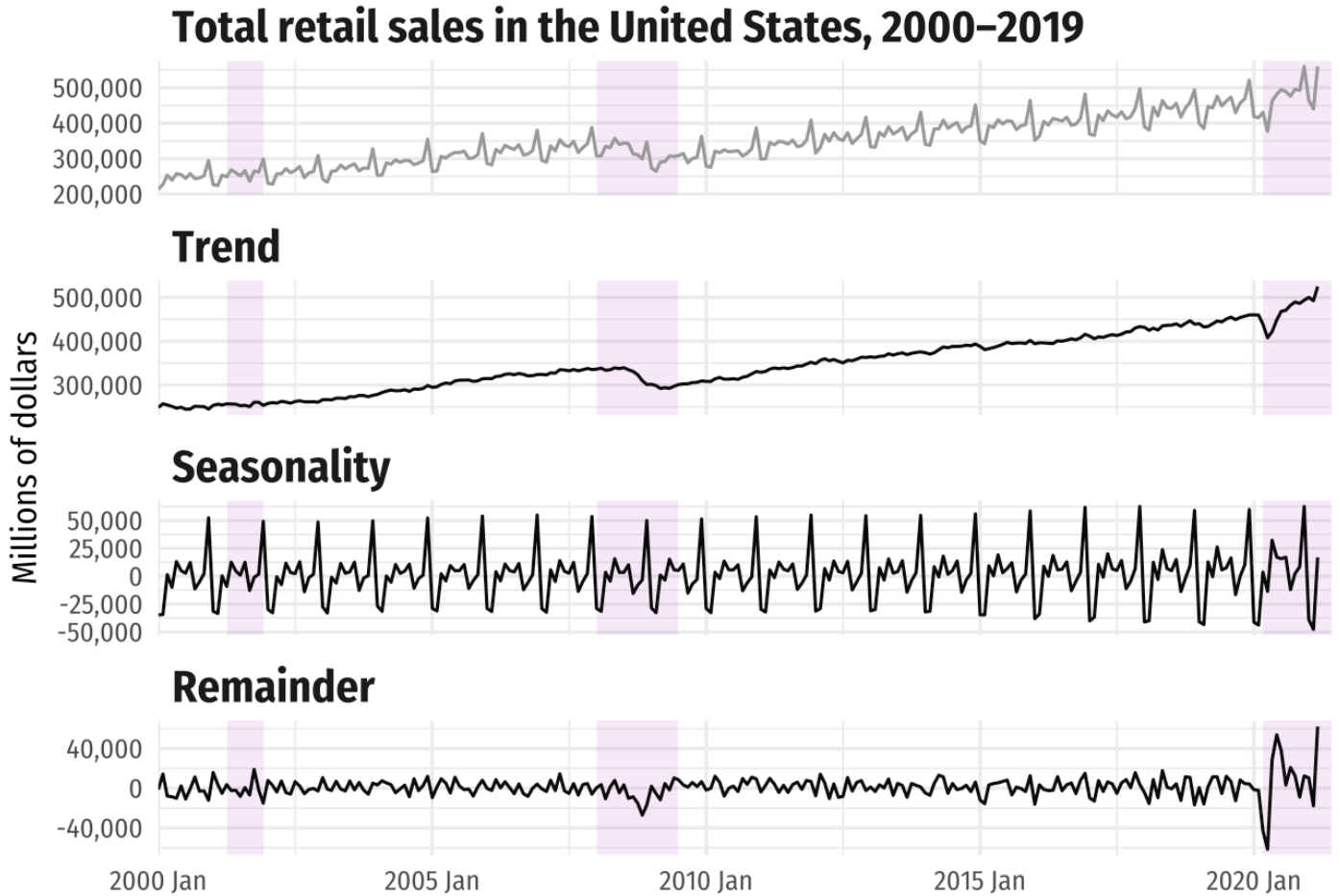
Seasonality

Don't mistake seasonality for actual trends





Seasonal adjustment



Source: RSXFSN from FRED. Recessions highlighted in purple.

Birthday decomposition