Visualization

Week 3: Explore, visually

- Why visualize?
- Elements of statistical graphics
- Principles of effective visualization

This week: data visualization

Objective: introduce the uses, types, anatomy, and construction of statistical graphics.

- **Why visualize?**
  - No one likes reading a table
  - Exploratory graphics: discovery
  - Presentation graphics: communication
- **Statistical graphics**
  - Elements: axes, geometric objects, aesthetic attributes, and text
  - Construction: mapping data to graphical elements
  - Common statistical graphics
- **Principles of effective visualization**
  - Tips and advice

Why visualize?

- Tables are not effective ways of preventing simple stories
- Exploratory graphics: discovery
- Presentation graphics: communication

Notice your reaction

Table 1. Mean Achievement Estimates by Gender, Subject, Grade, and Year

	ELA							Math						
	2009	2010	2011	2012	2013	2014	2015	2009	2010	2011	2012	2013	2014	2015
							Male							
3	-0.02	-0.03	-0.03	-0.03	-0.05	-0.03	-0.05	0.08	0.08	0.07	0.07	0.07	0.08	0.09
4	-0.03	-0.03	-0.04	-0.05	-0.06	-0.05	-0.07	0.06	0.06	0.05	0.05	0.05	0.07	0.06
5	-0.02	-0.05	-0.05	-0.05	-0.06	-0.06	-0.09	0.05	0.05	0.04	0.04	0.03	0.04	0.03
6	-0.03	-0.04	-0.04	-0.05	-0.06	-0.07	-0.10	0.06	0.05	0.04	0.04	0.03	0.03	0.02
7	-0.05	-0.05	-0.06	-0.06	-0.08	-0.08	-0.11	0.05	0.05	0.04	0.02	0.01	0.01	0.00
8	-0.06	-0.06	-0.06	-0.06	-0.08	-0.10	-0.11	0.07	0.05	0.05	0.02	0.01	0.01	0.00
							Female							
3	0.18	0.16	0.16	0.17	0.16	0.17	0.18	0.05	0.04	0.03	0.04	0.04	0.05	0.07
4	0.17	0.15	0.15	0.16	0.15	0.16	0.17	0.03	0.03	0.03	0.05	0.03	0.04	0.05
5	0.17	0.16	0.15	0.16	0.15	0.15	0.17	0.03	0.03	0.02	0.04	0.03	0.04	0.06
6	0.19	0.19	0.18	0.19	0.17	0.17	0.20	0.07	0.06	0.06	0.05	0.06	0.07	0.08
7	0.20	0.20	0.20	0.20	0.18	0.18	0.22	0.08	0.07	0.06	0.06	0.06	0.07	0.06
8	0.22	0.21	0.21	0.20	0.19	0.19	0.23	0.08	0.07	0.06	0.05	0.06	0.07	0.09
							Male-Female							
3	-0.19	-0.19	-0.20	-0.20	-0.21	-0.20	-0.22	0.03	0.04	0.03	0.03	0.03	0.03	0.02
4	-0.20	-0.18	-0.20	-0.21	-0.21	-0.21	-0.24	0.03	0.03	0.02	0.01	0.02	0.03	0.01
5	-0.19	-0.21	-0.20	-0.20	-0.21	-0.21	-0.26	0.02	0.02	0.01	0.00	0.00	-0.01	-0.03
6	-0.22	-0.22	-0.22	-0.23	-0.24	-0.25	-0.29	-0.01	-0.01	-0.02	-0.01	-0.03	-0.05	-0.06
7	-0.25	-0.25	-0.26	-0.26	-0.26	-0.26	-0.33	-0.03	-0.02	-0.03	-0.04	-0.04	-0.06	-0.06
8	-0.27	-0.27	-0.27	-0.27	-0.27	-0.29	-0.35	-0.01	-0.02	-0.02	-0.03	-0.05	-0.06	-0.10

Notes: Table is based on the mean achievement estimates, standardized to the National NAEP distribution within subject, grade and year. To account for the fact that the data are unbalanced (not all districts have estimates in each grade, year, and subject), we obtain the estimated average test score means for each subject, grade, and year by gender from a model regressing the average test scores in a gender-subject-grade-year-district cell on a set of district and gender-subject-grade-year fixed effects. The averages reported in each cell in Table 1 are the estimated coefficients from the gender-subject-grade-year dummy variables in this model. Note that these averages are not weighted by sample size, and thus reflect the mean test score for the average district in each subject, grade, year, and gender.

(There's a reaason that tables are usually put in appendices.)

Types of graphics

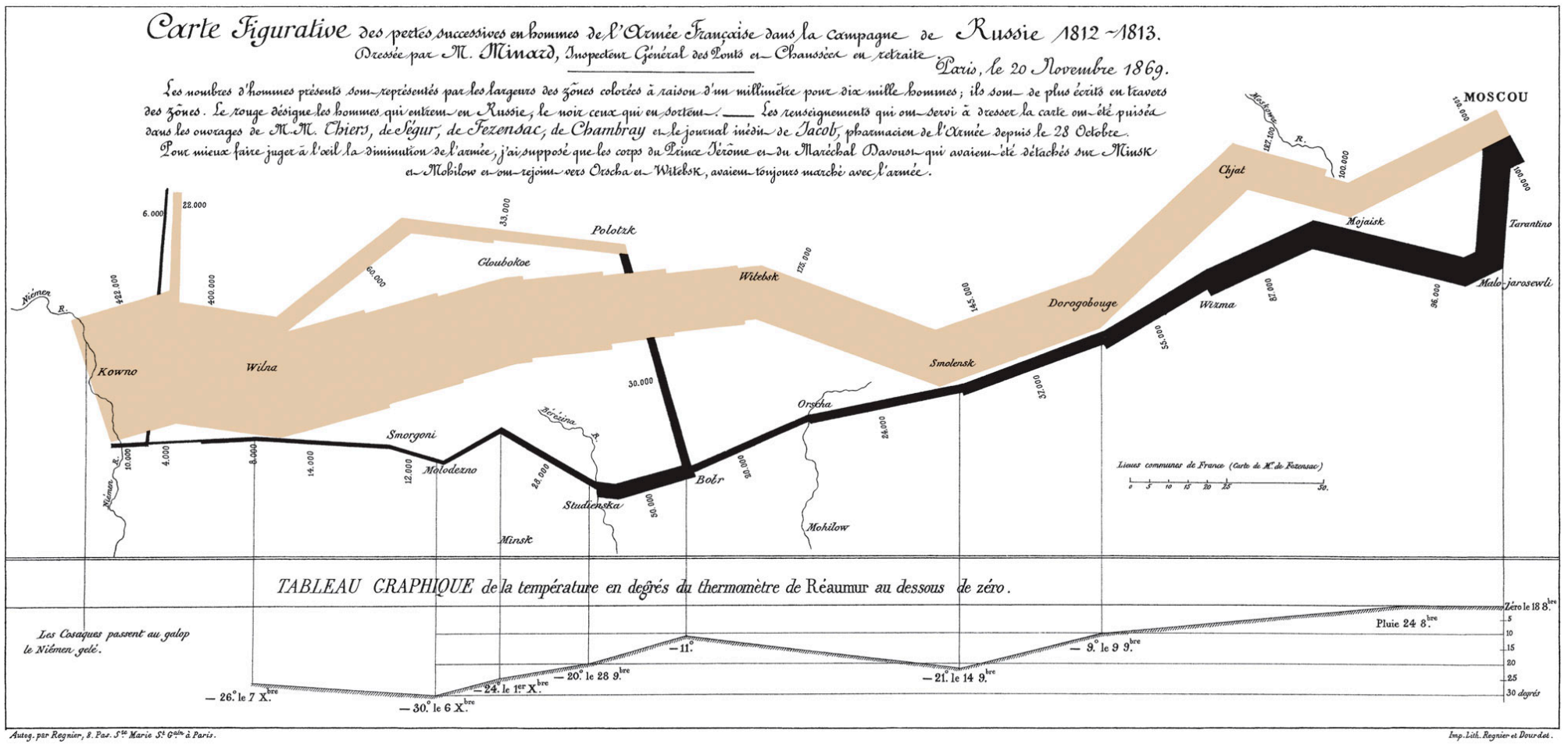
There is a broad distinction between:

- **exploratory graphics**, which are intended to be seen only by analysts; and
- **presentation graphics**, which are intended to be seen by an audience.

Types of Graphics

- Exploratory graphics are made quickly in large volumes, and usually not formatted too carefully. Think of them like the pages of a sketchbook.
- Presentation graphics are made with more attention to detail.
- The two are not mutually exclusive: an especially helpful exploratory graphic is often worth developing as a presentation graphic to help an audience understand ‘what the data look like’.

Infographics



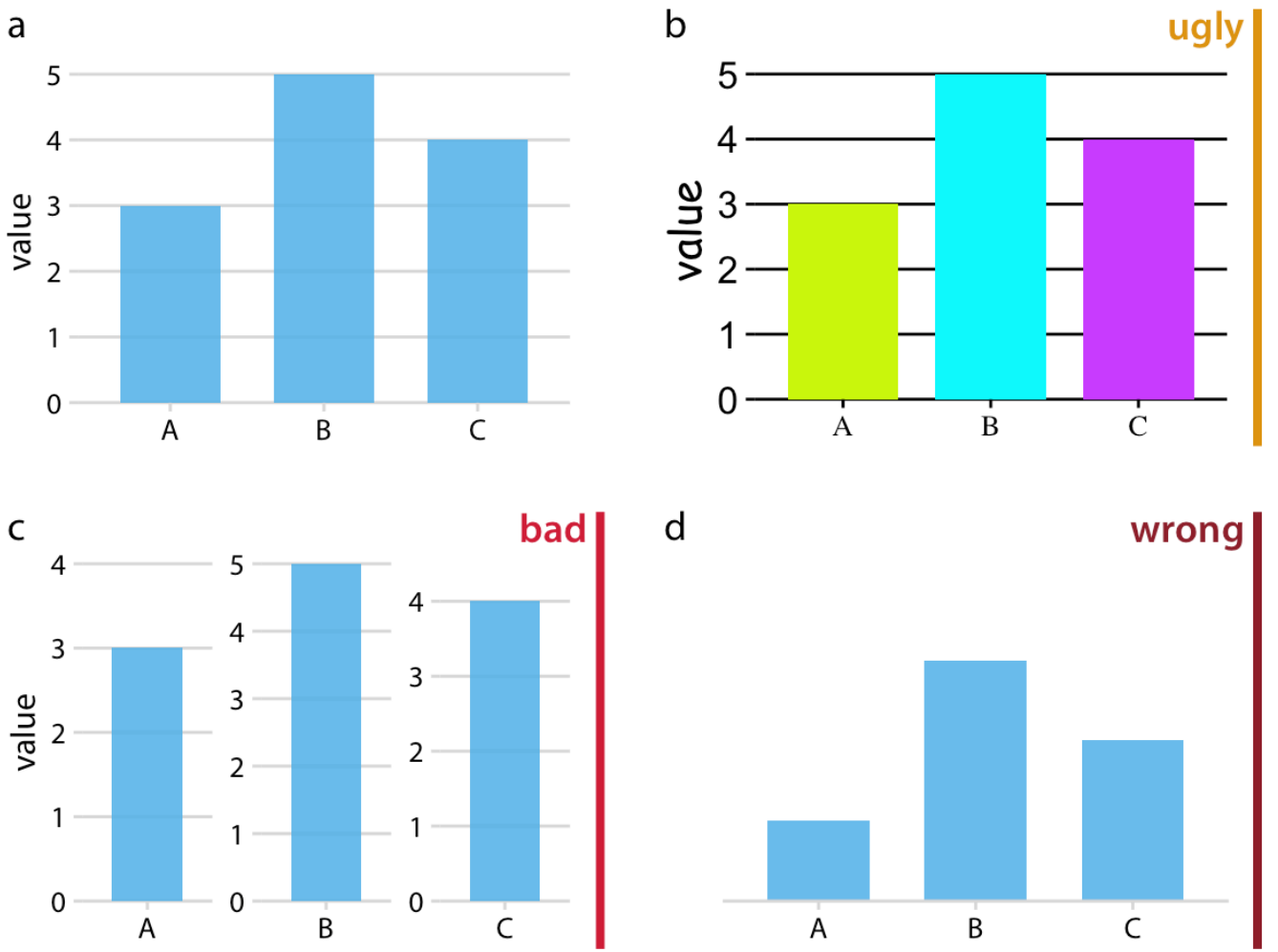
Infographics

- Usually much more complex, meant to engage with for a long time
- Dense with information presented in multiple ways
- Works of art

Common statistical graphics and their uses

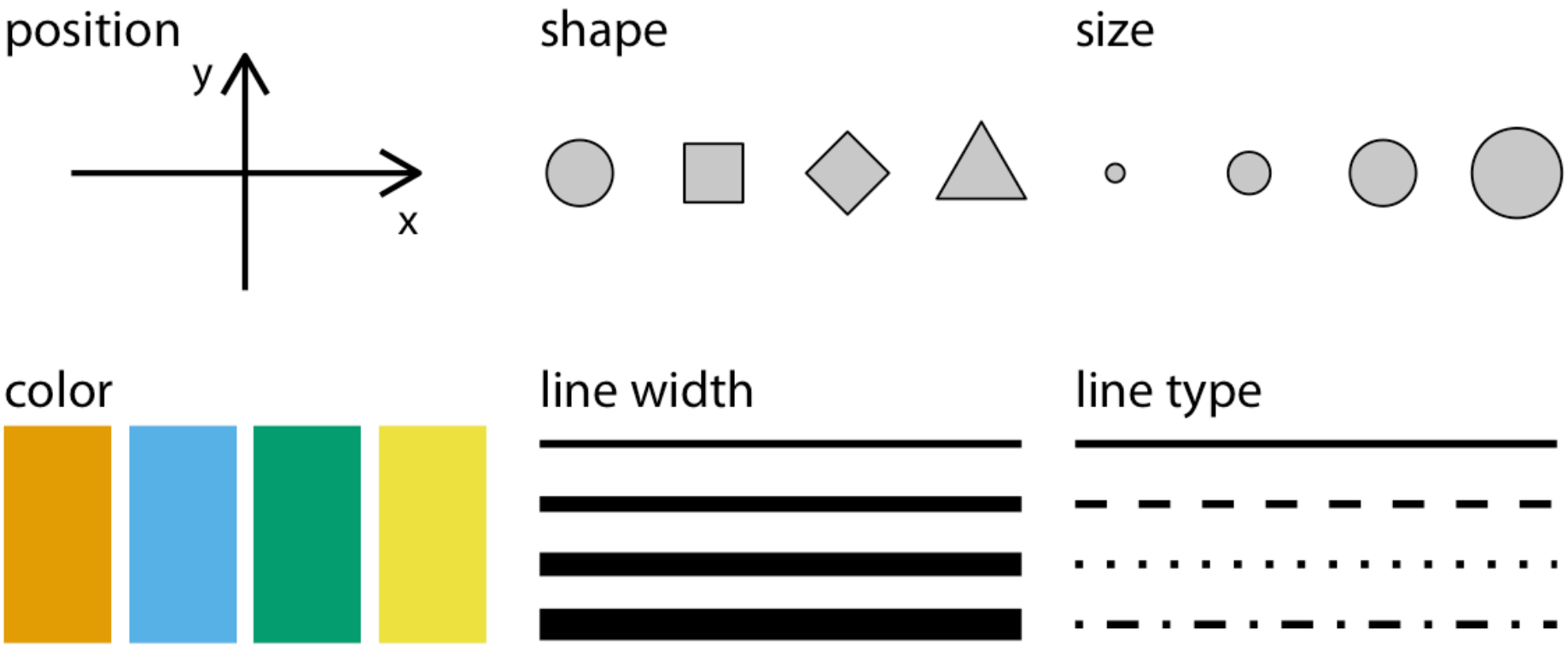
Some examples:

The Good, The Bad and The Ugly



Course reference: The Fundamentals of Data Visualization

Aesthetics



Aesthetic attributes

Aesthetic attributes (or ‘aesthetics’ for short) will mean qualities of geometric objects, like color.

Primary aesthetics in graphics are:

- Shape (for points)
- Color (outline color and fill colro)
- Point size / linewidth
- Opacity

Text

Text is used to:

- label axes,
- label objects
- create legends
- set titles.

Creates the story – text gives a plot its meaning!

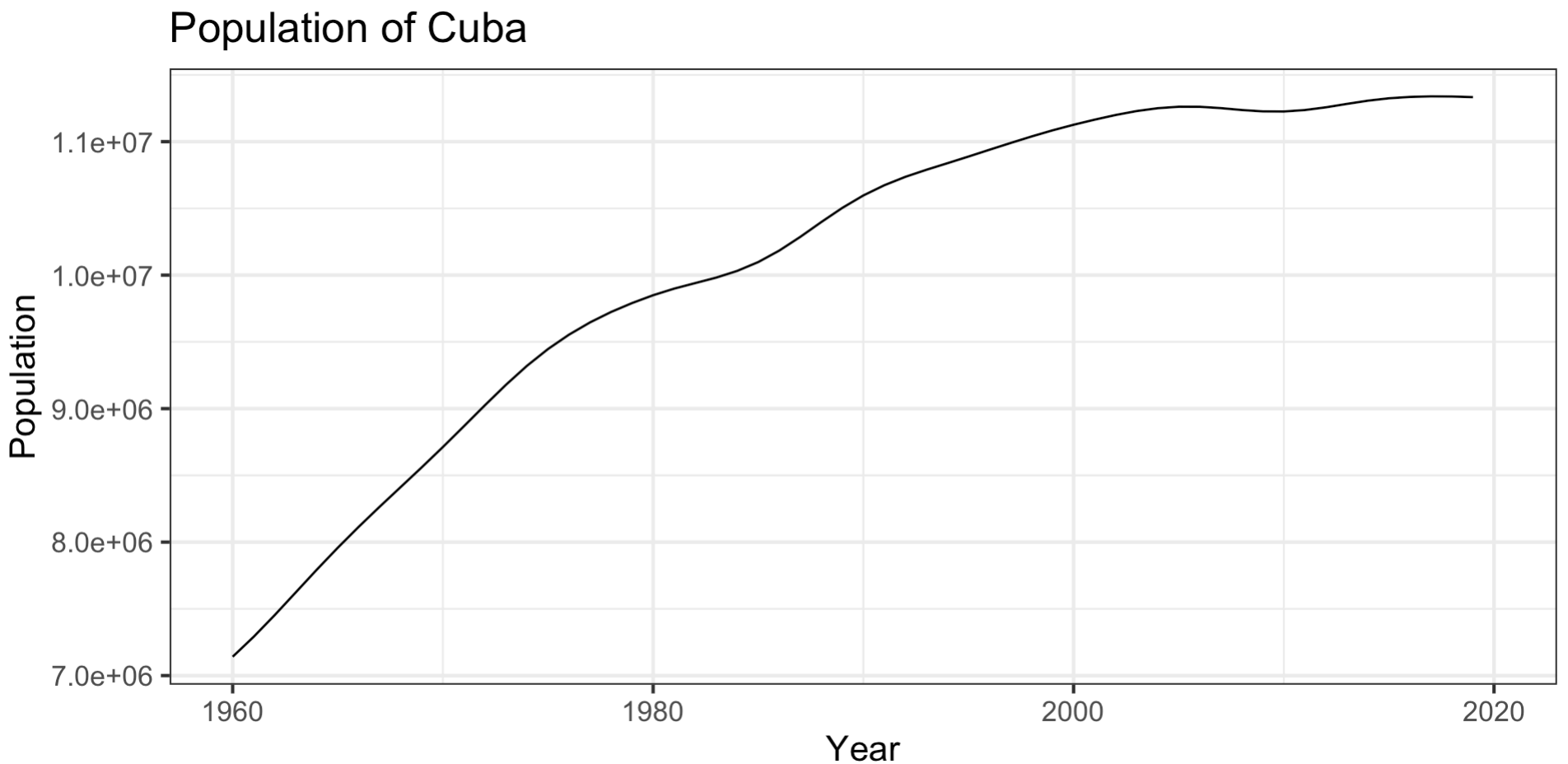
Statistical graphics are mappings

Statistical graphics are **mappings** of dataframe columns to geometric objects and aesthetic attributes. For a simple example, consider the following time series of Cuba's population by year:

```
# Create the population plot
pop_plot <- pop %>%
  filter(`Country Code` == 'CUB') %>%
  ggplot(aes(x = Year, y = Population)) +
  geom_line() +
  labs(title = "Population of Cuba") +
  theme_bw(base_size=16)

pop_plot
```

Statistical graphics are mappings



Statistical graphics are mappings

In the plot:

- population $\longrightarrow$ y-coordinate;
- year $\longrightarrow$ x-coordinate;
- the line connects the rows of the dataframe.

Mapping variables to aesthetics

Now consider aggregated populations by global region and year:

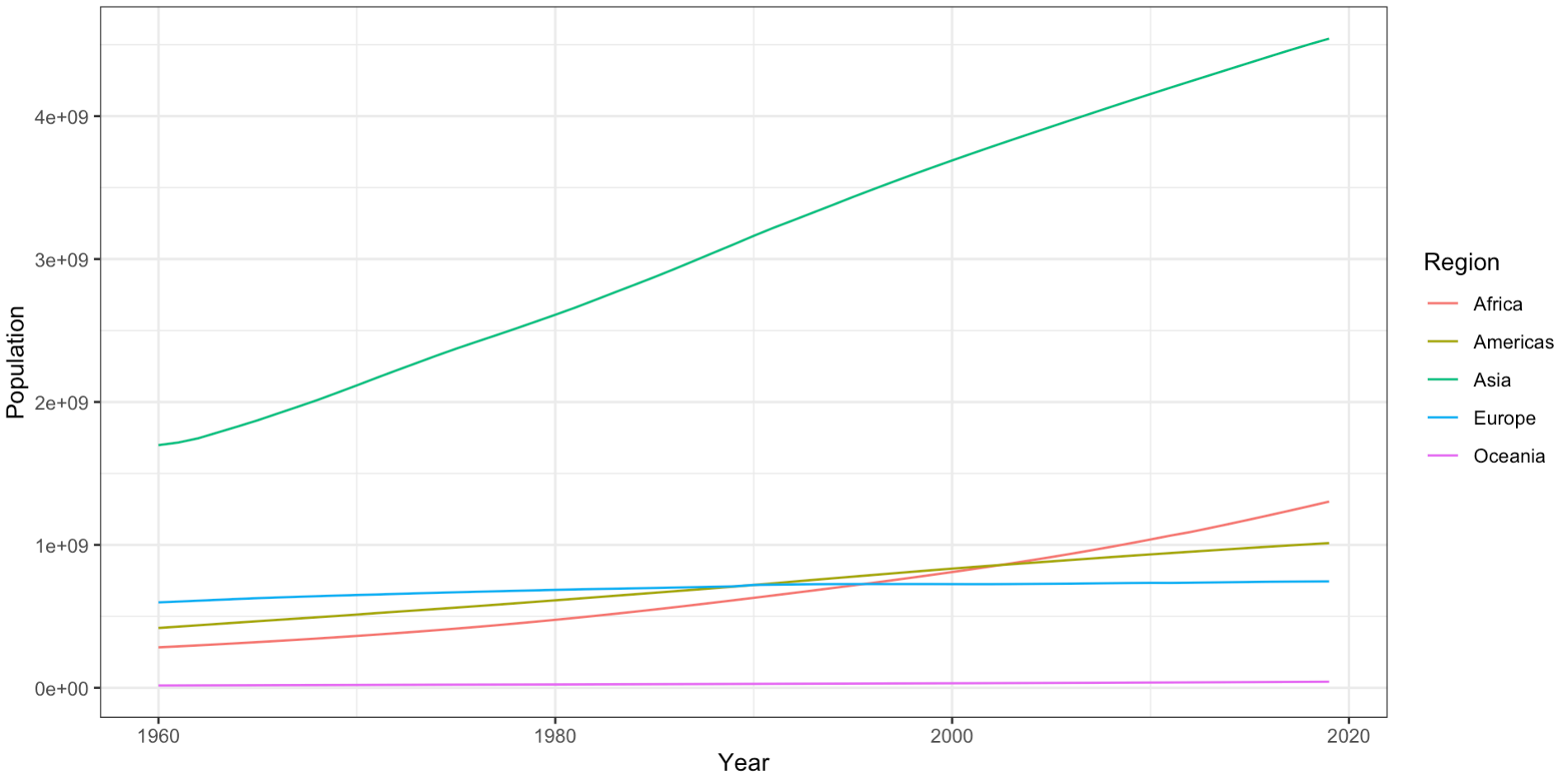
```
# A tibble: 300 × 3
   Region   Year       Population
   <fct>    <date>          <dbl>
 1 Africa   1960-01-01  282962801
 2 Americas 1960-01-01  418482836
 3 Asia     1960-01-01 1698003789
 4 Europe   1960-01-01  597599081
 5 Oceania  1960-01-01   16023128
 6 Africa   1961-01-01  289804906
 7 Americas 1961-01-01  427950859
 8 Asia     1961-01-01 1716191885
 9 Europe   1961-01-01  603375654
10 Oceania  1961-01-01   16353964
# ℹ 290 more rows
```

Mapping variables to aesthetics

```r
# Create the plot
popregion_plot <- popregion |>
  ggplot(aes(x = Year, y = Population, color = Region)) +
  geom_line(alpha = 1) +
  theme_bw()

popregion_plot
```

Mapping variables to aesthetics



Mapping variables to aesthetics

In this plot:

- population $\longrightarrow$ y
- year $\longrightarrow$ x
- region $\longrightarrow$ color

Types of Variables

Table 2.1: Types of variables encountered in typical data visualization scenarios.

Type of variable	Examples	Appropriate scale	Description
quantitative/numerical continuous	1.3, 5.7, 83, $1.5x10^{-2}$	continuous	Arbitrary numerical values. These can be integers, rational numbers, or real numbers.
quantitative/numerical discrete	1, 2, 3, 4	discrete	Numbers in discrete units. These are most commonly but not necessarily integers. For example, the numbers 0.5, 1.0, 1.5 could also be treated as discrete if intermediate values cannot exist in the given dataset.
qualitative/categorical unordered	dog, cat, fish	discrete	Categories without order. These are discrete and unique categories that have no inherent order. These variables are also called *factors*.
qualitative/categorical ordered	good, fair, poor	discrete	Categories with order. These are discrete and unique categories with an order. For example, “fair” always lies between “good” and “poor”. These variables are also called *ordered factors*.
date or time	Jan. 5 2018, 8:03am	continuous or discrete	Specific days and/or times. Also generic dates, such as July 4 or Dec. 25 (without year).
text	The quick brown fox jumps over the lazy dog.	none, or discrete	Free-form text. Can be treated as categorical if needed.

A grammar of graphics

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

A `ggplot consists of

- data (the input)
- *at least* one geom_ function
- mappings for each geom_

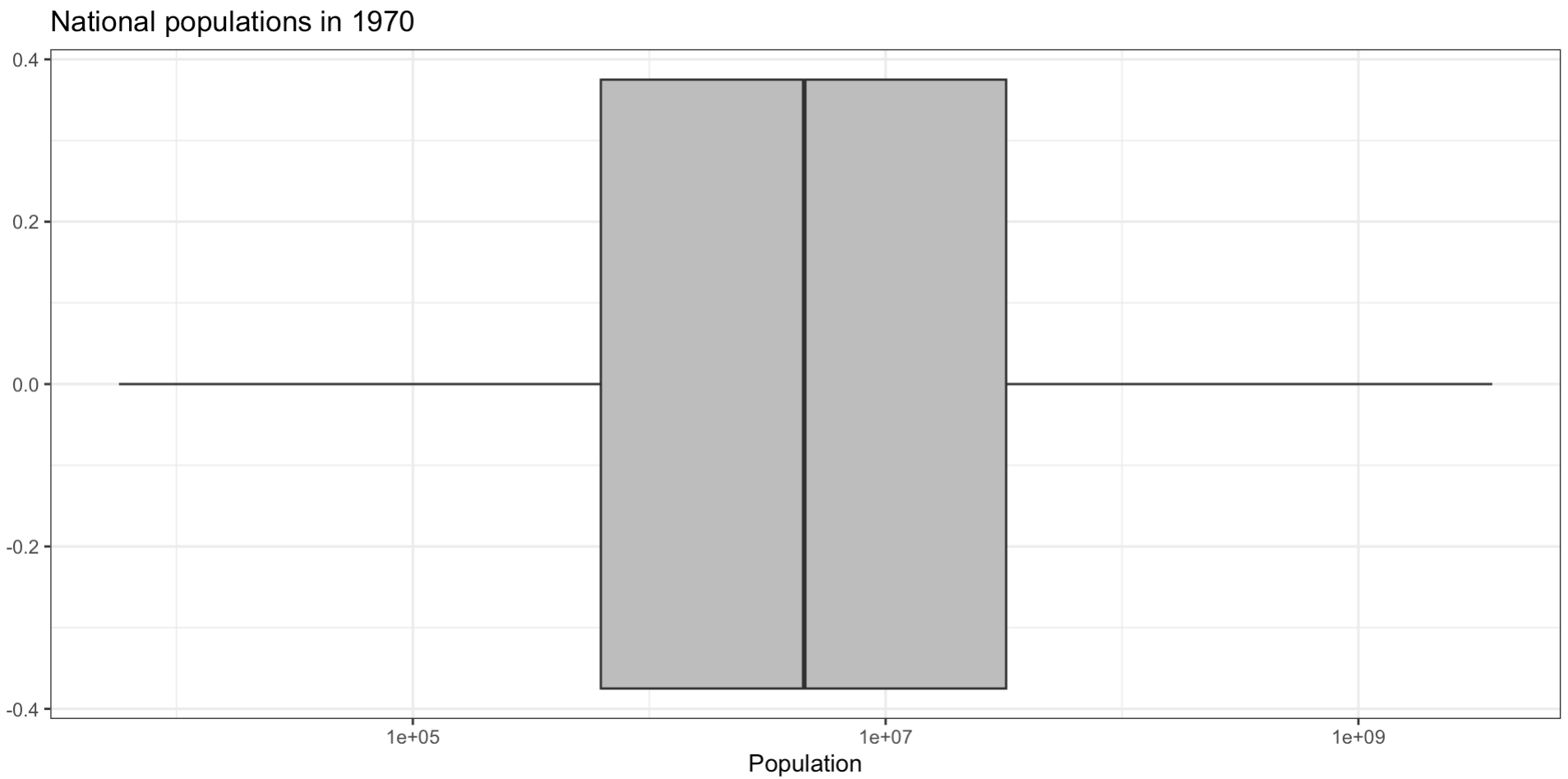
Single-variable graphics: boxplot

Single-variable graphics are used to display the distribution of values of a single variable.

Boxplots show the spread, center, and skewness of values.

```
pop_boxplot <- pop |>
  filter(Year == "1970-01-01") %>%
  ggplot(aes(x = Population)) +
  geom_boxplot(fill="gray") +
  scale_x_log10() +
  labs(title = "National populations in 1970") +
  theme_bw()
pop_boxplot
```

Single-variable graphics: boxplot



Single-variable graphics: histogram

Histograms show the relative frequencies of values of a single variable.

One can see spread, center, skewness, and outliers, but ***also shape***.

Single-variable graphics

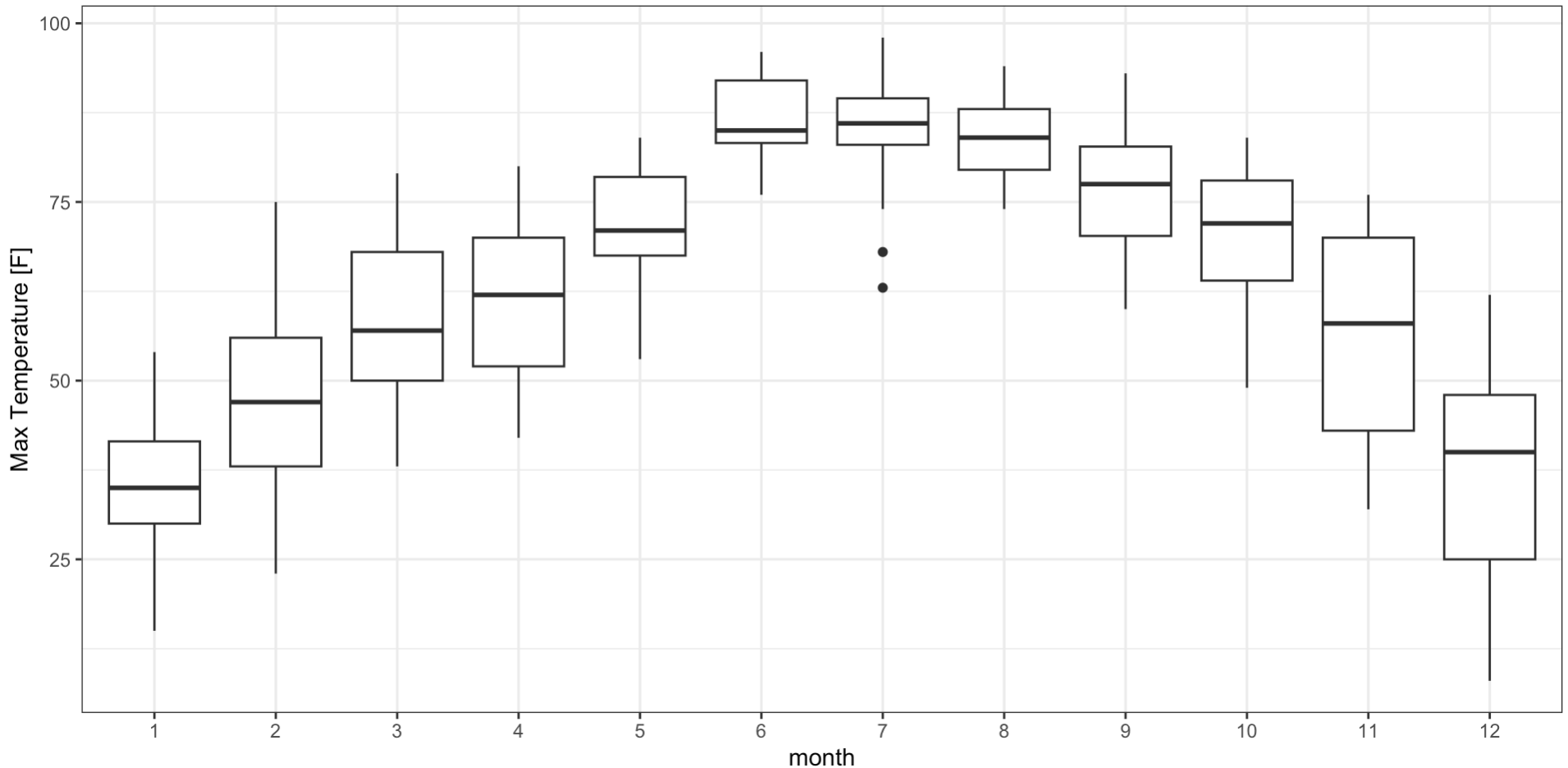
Not necessarily limited to univariate data.

For example, we could make one boxplot for each year and show the distributions of national populations for each year:

```
lincoln_weather <- ggridges::lincoln_weather

# Boxplot
temp_boxplot <- lincoln_weather |>
  mutate(date = ymd(CST), month=as.factor(month(date))) |>
  ggplot(aes(x=month, y=`Max Temperature [F]`)) +
  geom_boxplot() + theme_bw()
temp_boxplot
```

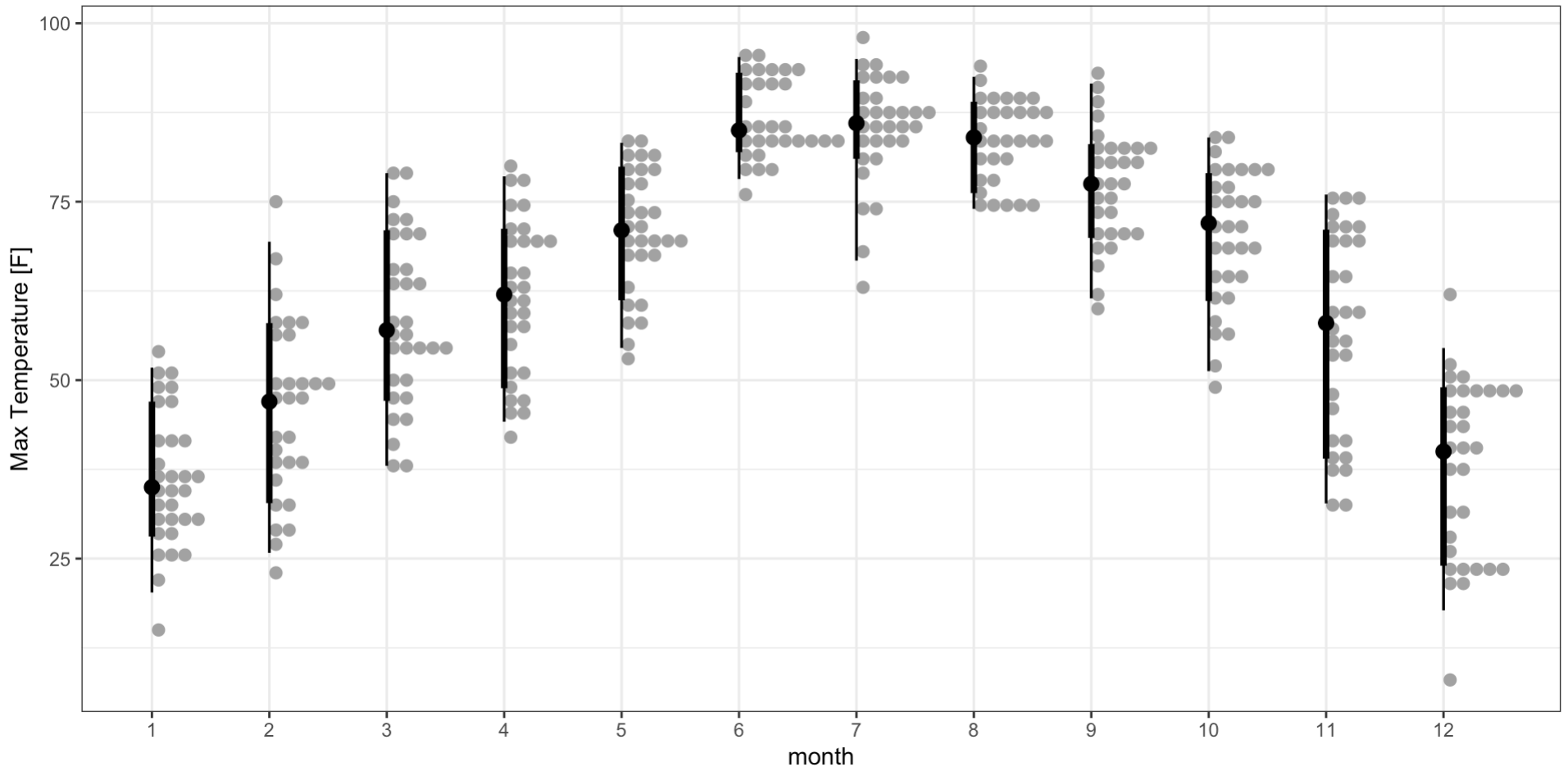
Single-variable graphics



Single-variable graphics

```
interval_plot <- lincoln_weather |>
  mutate(date = ymd(CST), month=as.factor(month(date))) |>
  ggplot(aes(x=month, y=`Max Temperature [F]`)) +
  ggdist::stat_dotsinterval() + theme_bw()
interval_plot
```

Single-variable graphics



Two-variable graphics: scatterplot

Scatterplots display relationships between two variables.

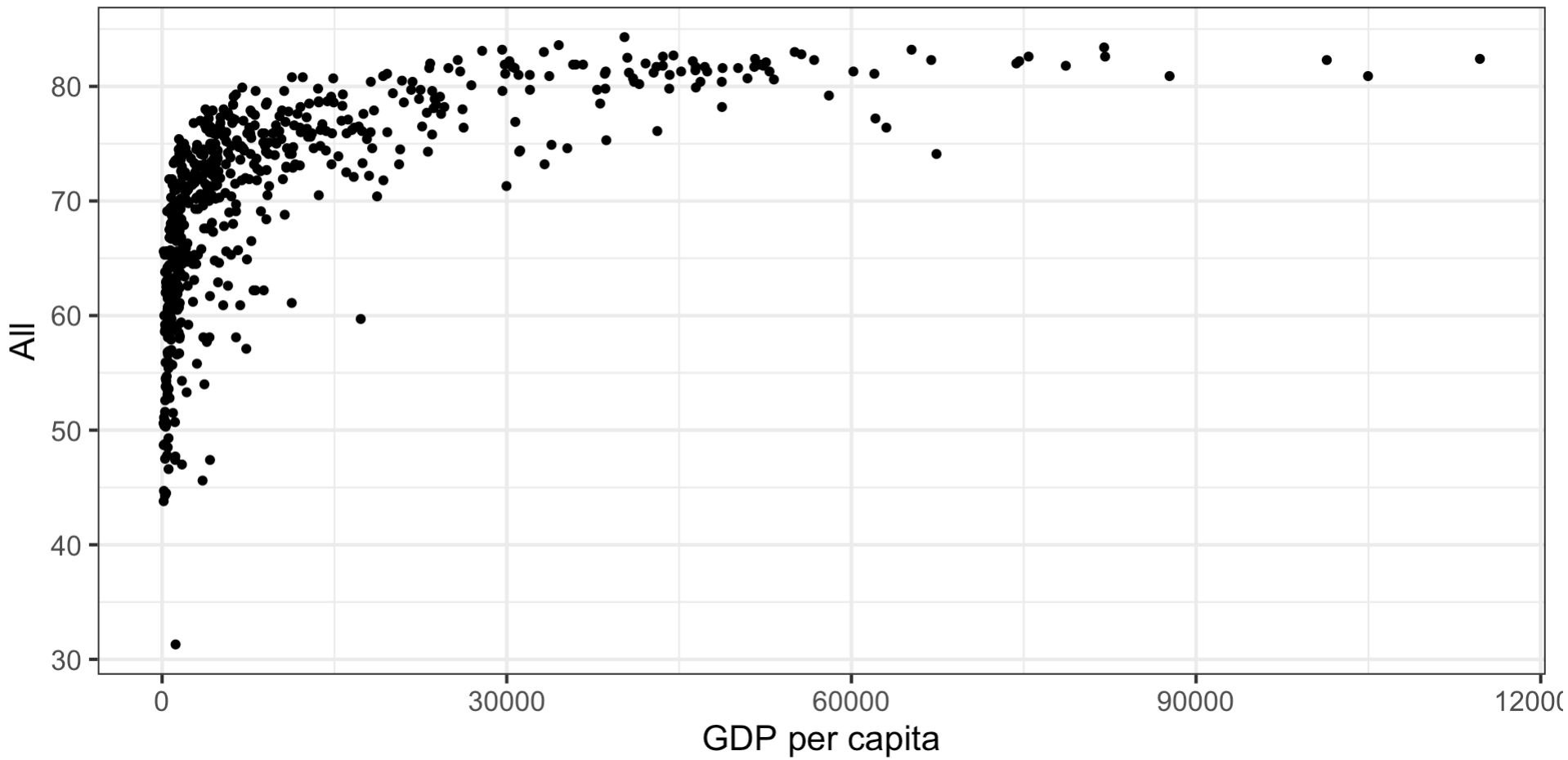
The pattern of scatter shows that life expectancy generally increases with GDP per capita.

Two-variable graphics: scatterplot

```
gdp_per_cap <- read_csv("data/lab3-data.csv")

gdp_per_cap |>
  ggplot(aes(x=`GDP per capita`, y=All)) +
  geom_point() + theme_bw(base_size=16)
```

Two-variable graphics: scatterplot



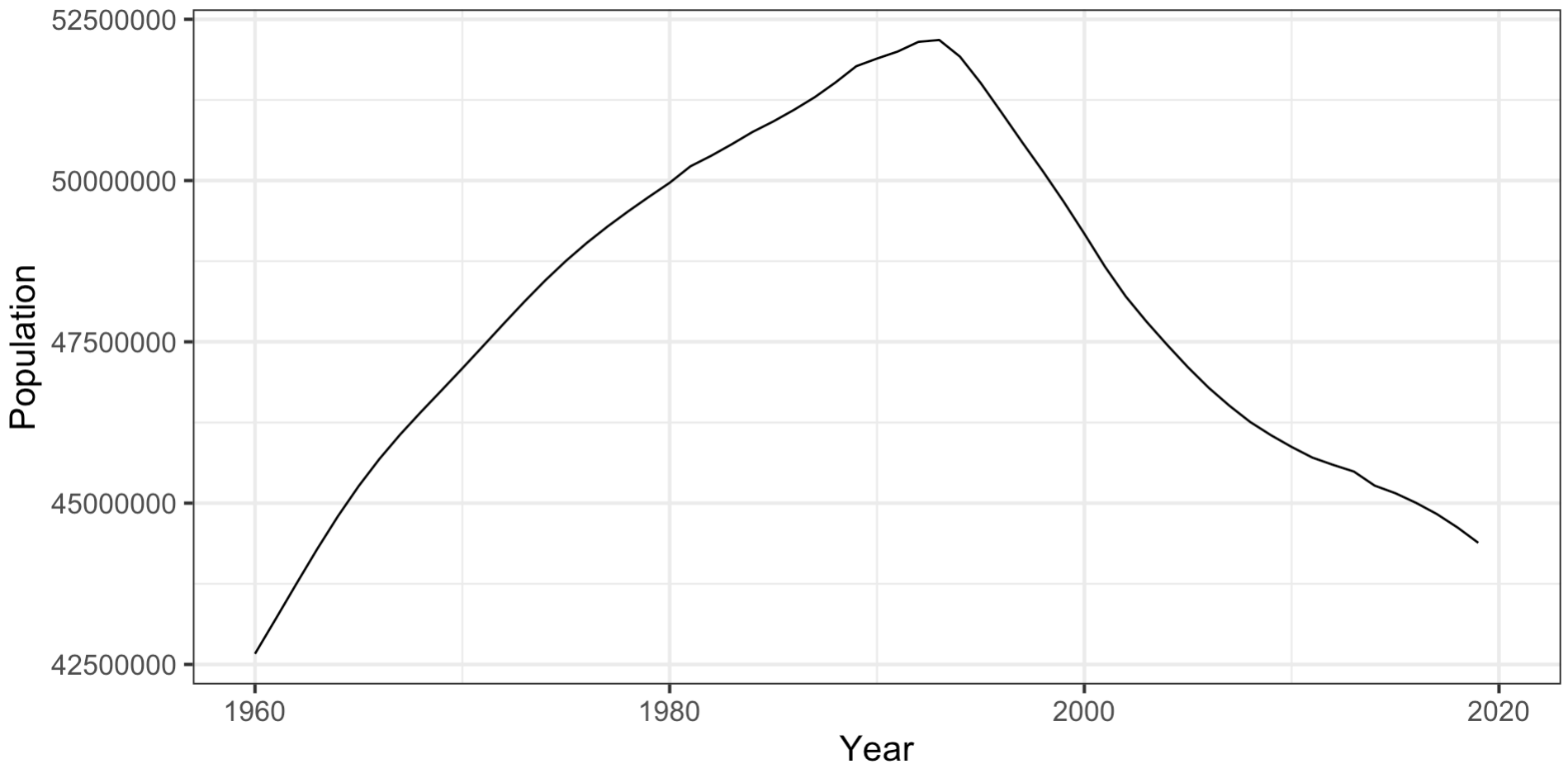
Two-variable graphics: line plot

Line plots display trajectories by connecting rows in a dataframe.

These can represent trends, time courses, or paths traveled.

```
pop |> filter(`Country Name` == "Ukraine") |>
  ggplot(aes(x=Year, y=Population)) +
  geom_line() + theme_bw(base_size=16)
```

Two-variable graphics: line plot



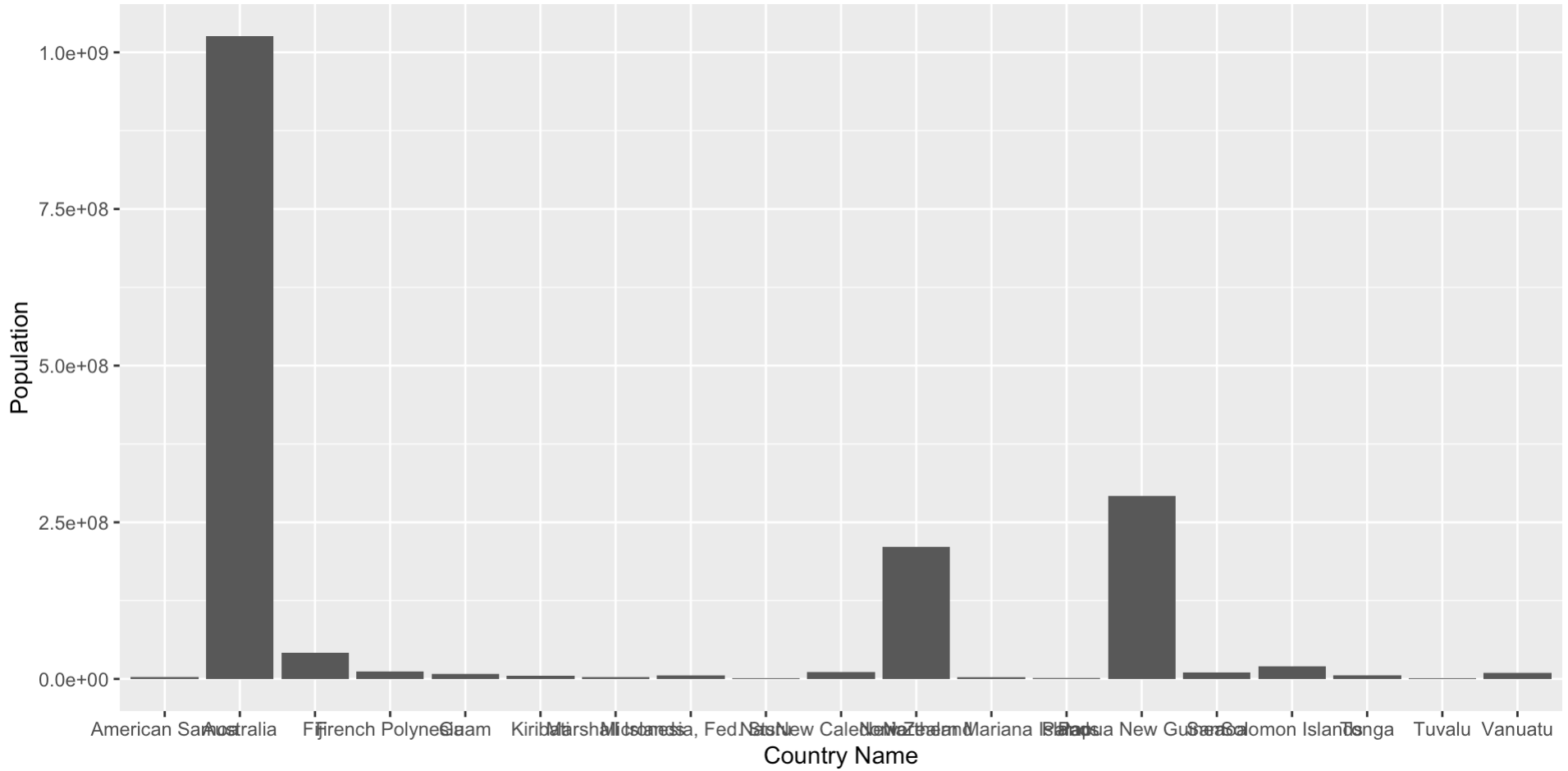
Two-variable graphics: bar plot

Bar plots usually depict the relationship between the magnitude of one variable and another.

For instance:

```
country_region <- inner_join(countryinfo, pop, by = "Country Code")
country_region |>
  filter(Region == "Oceania") |>
  ggplot(aes(y=Population, x=`Country Name`)) +
  geom_col()
```

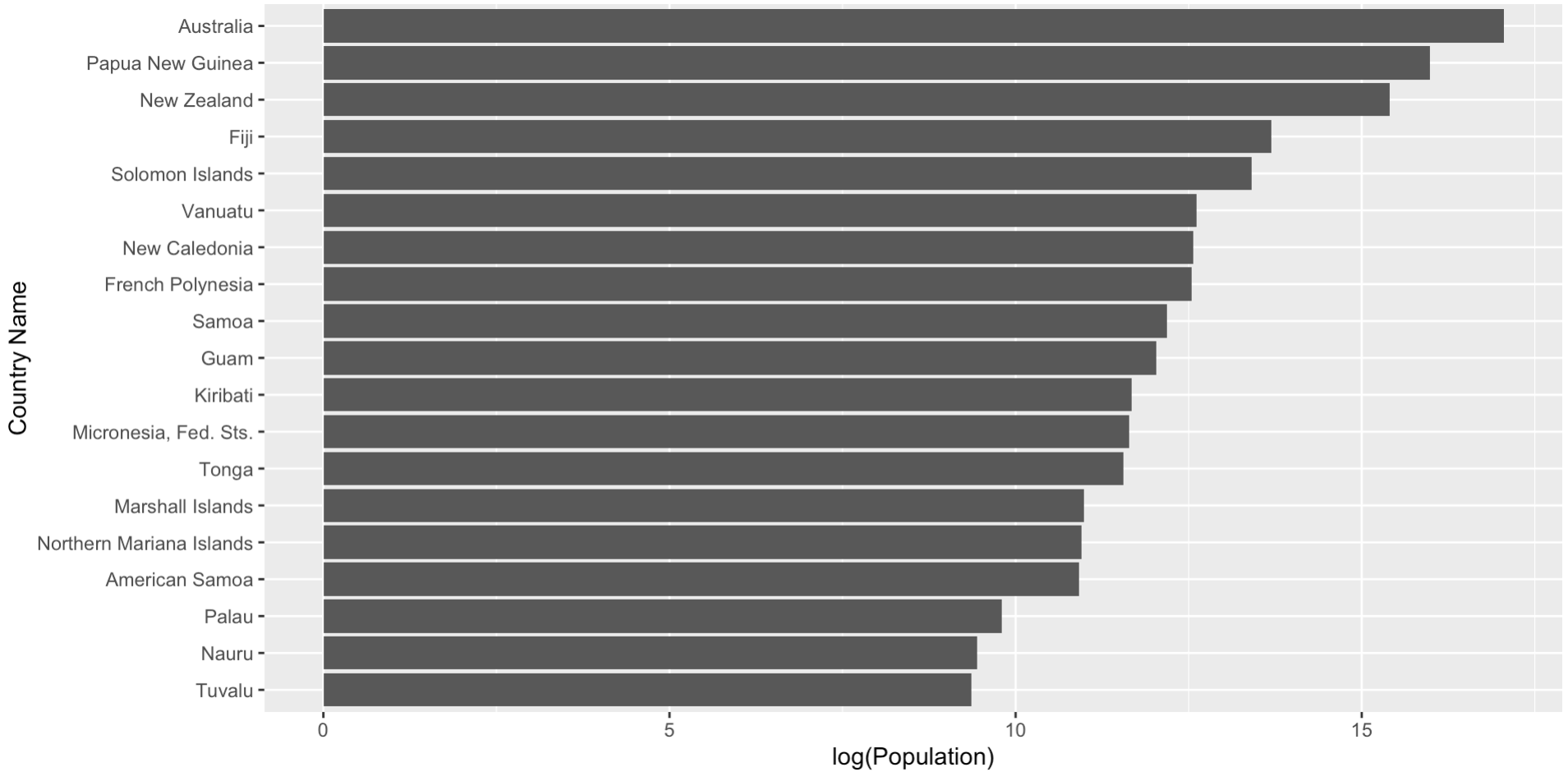
Two-variable graphics: bar plot



An improved bar plot

```
country_region |>
  filter(Region == "Oceania", Year == "2019-01-01") |>
  mutate(`Country Name` = as.factor(`Country Name`)) |>
  mutate(`Country Name` = fct_reorder(`Country Name`, Population)) |>
  ggplot(aes(y=log(Population), x=`Country Name`)) +
  geom_col() +
  coord_flip()
```

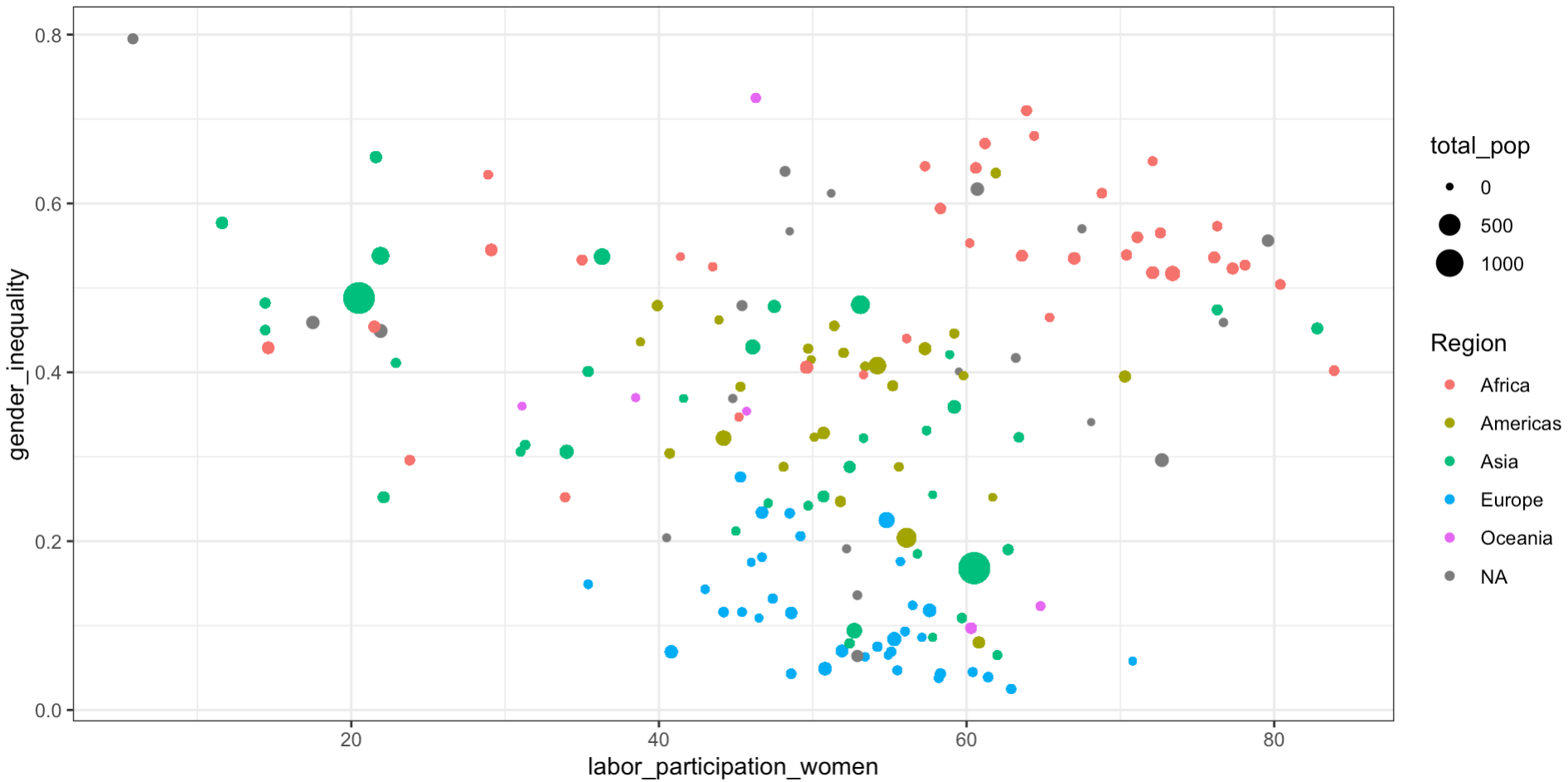
An improved bar plot



Multi-variable plots

```
country_data |> ggplot() +
  geom_point(aes(x=labor_participation_women,
                 y=gender_inequality,
                 size=total_pop,
                 col=Region)) +
  theme_bw()
```

Multi-variable plots



Multi-variable plots

In the plot:

- population $\rightarrow$ size of the point
- women labor participation $\rightarrow$ x-coordinate;
- gender inequality $\rightarrow$ y-coordinate;
- color $\rightarrow$ region

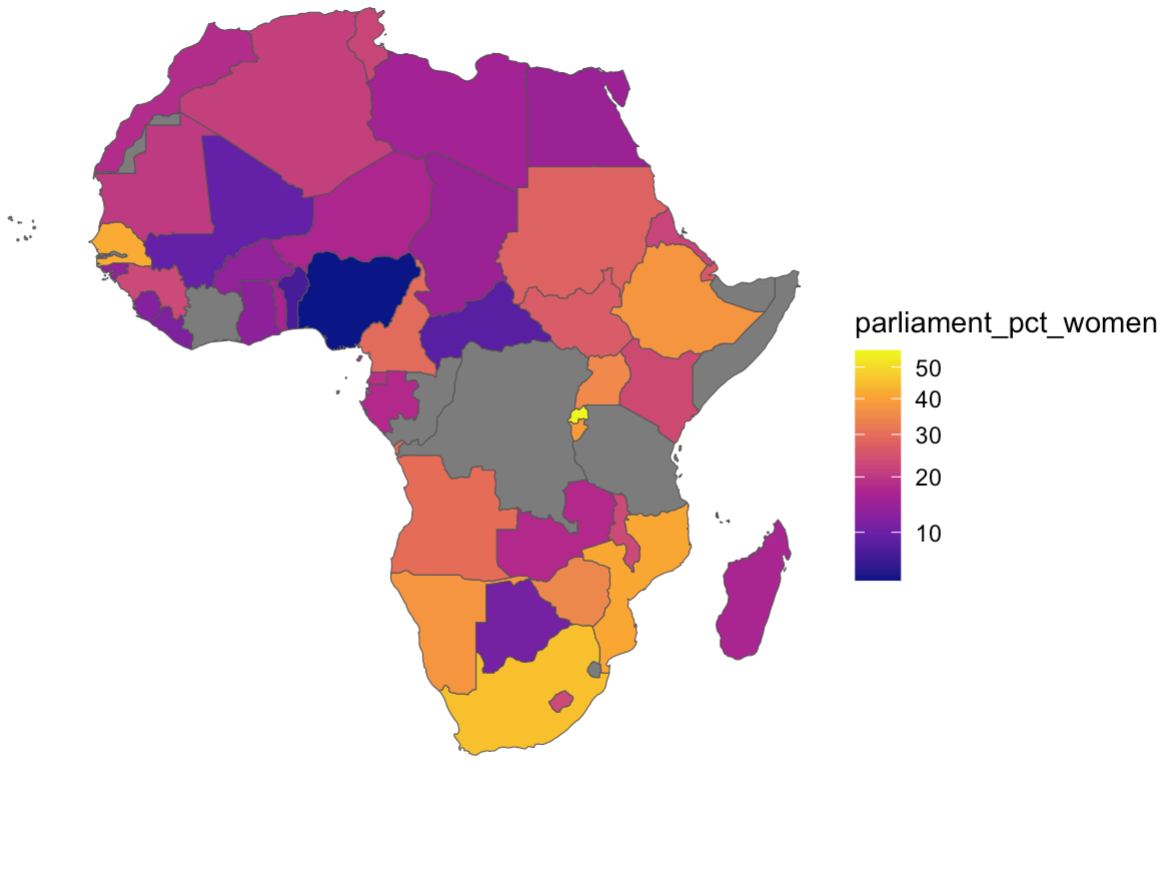
Multi-variable plots

Shape files for geopgraphy:

```
Simple feature collection with 54 features and 2 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -25.34155 ymin: -46.96289 xmax: 51.39023 ymax: 37.34038
Geodetic CRS:  WGS 84
First 10 features:
        name_en parliament_pct_women                       geometry
1      Zimbabwe                 34.6 MULTIPOLYGON (((31.28789 -2...
2        Zambia                 18.0 MULTIPOLYGON (((30.39609 -1...
3        Uganda                 34.9 MULTIPOLYGON (((33.90322 -1...
4       Tunisia                 22.6 MULTIPOLYGON (((11.50459 33...
5          Togo                 16.5 MULTIPOLYGON (((0.9004883 1...
6      Tanzania                   NA MULTIPOLYGON (((39.49648 -6...
7      Eswatini                   NA MULTIPOLYGON (((31.94824 -2...
```

```r
africa_data |>
  ggplot() + geom_sf(aes(fill = parliament_pct_women)) +
    scale_fill_viridis_c(option = "plasma", trans = "sqrt") + theme_void()
```

Multi-variable plots



Visualization for exploration

> The greatest value of a picture is when it forces us to notice what we never expected to see.
>
> — John Tukey

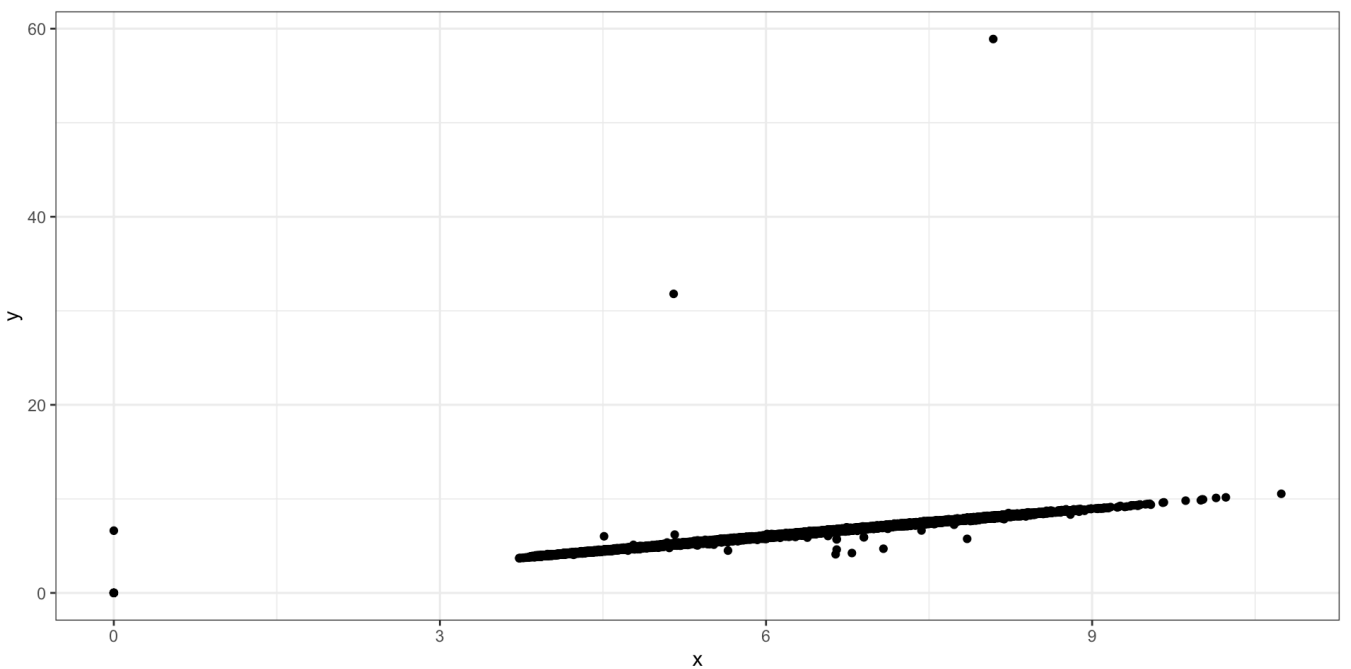
Diamonds data

```
Rows: 53,940
Columns: 10
$ carat   <dbl> 0.23, 0.21, 0.23, 0.29, 0.31, 0.24, 0.24, 0.26, 0.22, 0.23,
0.…
$ cut     <ord> Ideal, Premium, Good, Premium, Good, Very Good, Very Good,
Ver…
$ color   <ord> E, E, E, I, J, J, I, H, E, H, J, J, F, J, E, E, I, J, J, J,
I,…
$ clarity <ord> SI2, SI1, VS1, VS2, SI2, VVS2, VVS1, SI1, VS2, VS1, SI1, VS1,
…
$ depth   <dbl> 61.5, 59.8, 56.9, 62.4, 63.3, 62.8, 62.3, 61.9, 65.1, 59.4,
64…
$ table   <dbl> 55, 61, 65, 58, 58, 57, 57, 55, 61, 61, 55, 56, 61, 54, 62,
58…
```

Diamonds data

Consider the relationship between the length and width of diamonds:

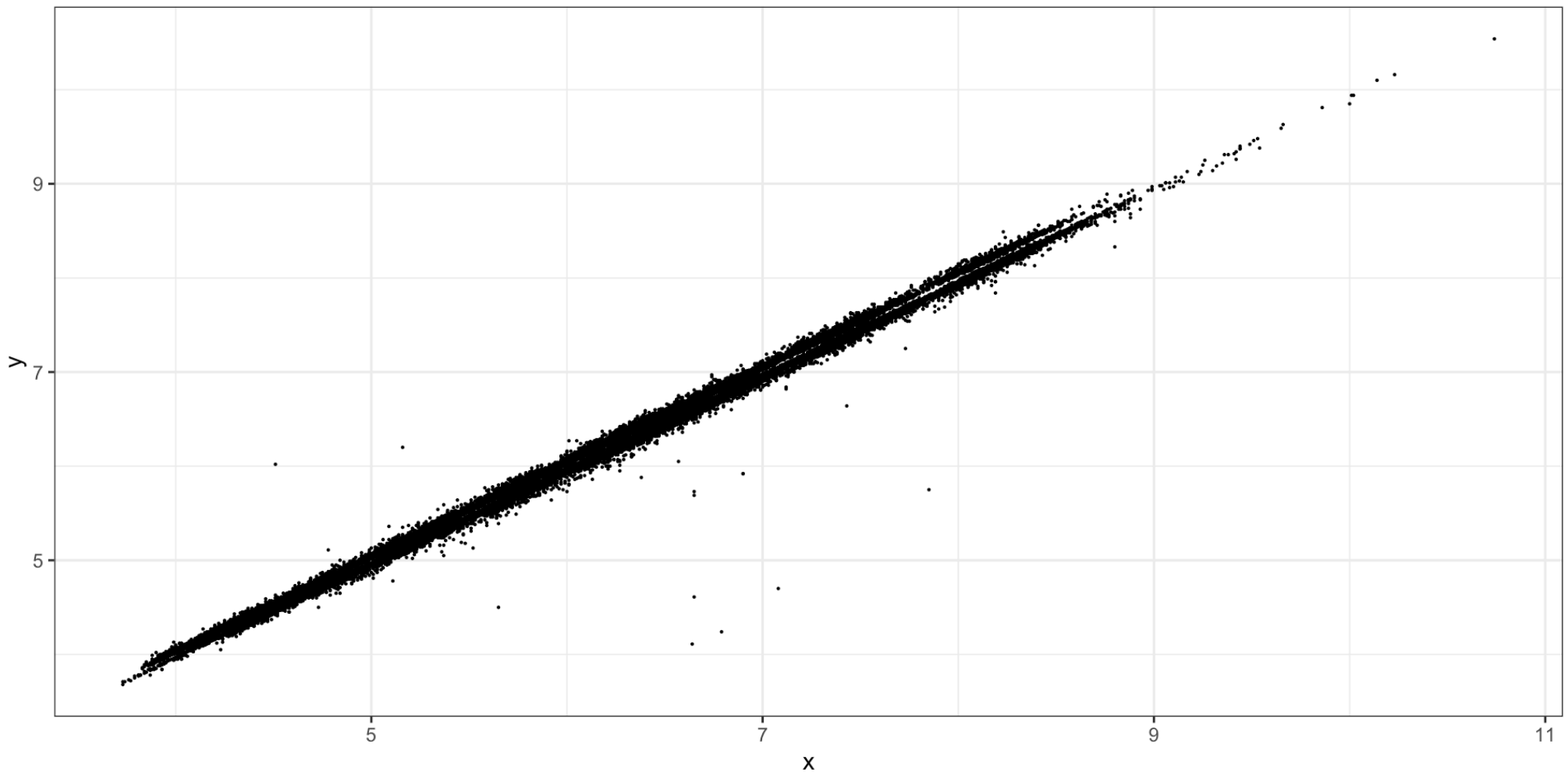
▶ Code



What do you notice? What can be done?

Outliers and artifacts

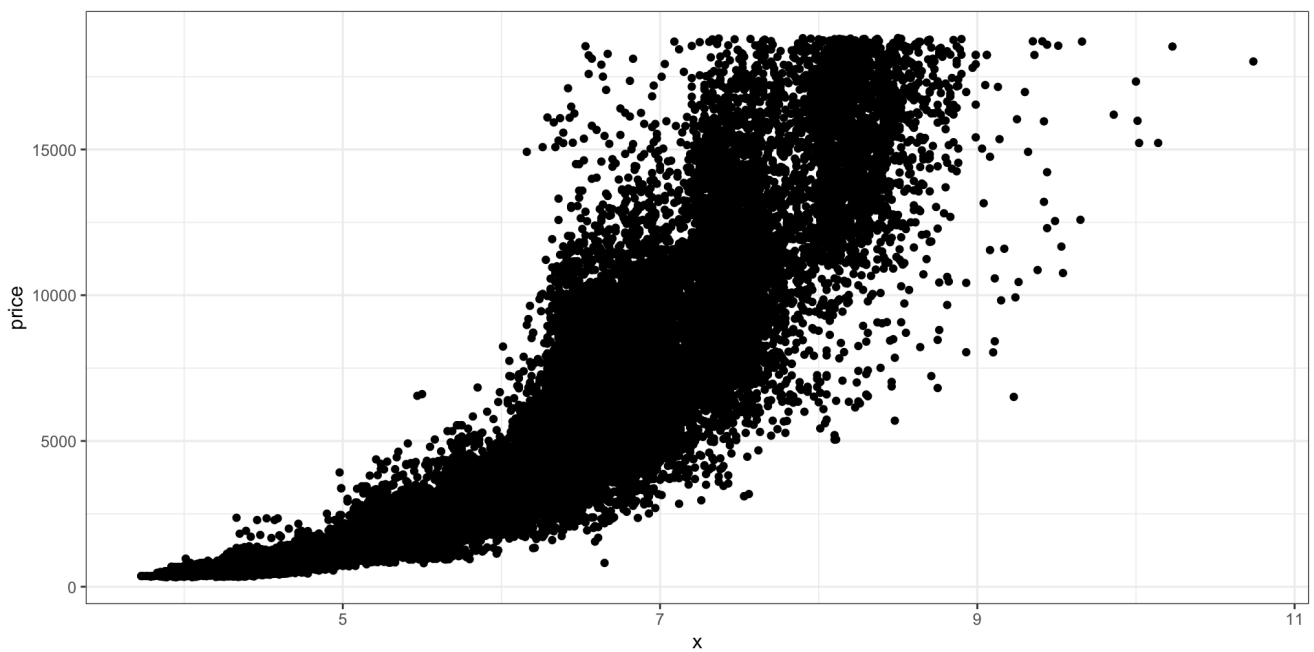
- Code



Outliers and artifacts

What about the relationship between length and price?

```
diamonds %>%
 filter(y <= 30, x >= 3) %>%
ggplot(aes(x = x, y = price)) +
  geom_point() + theme_bw()
```

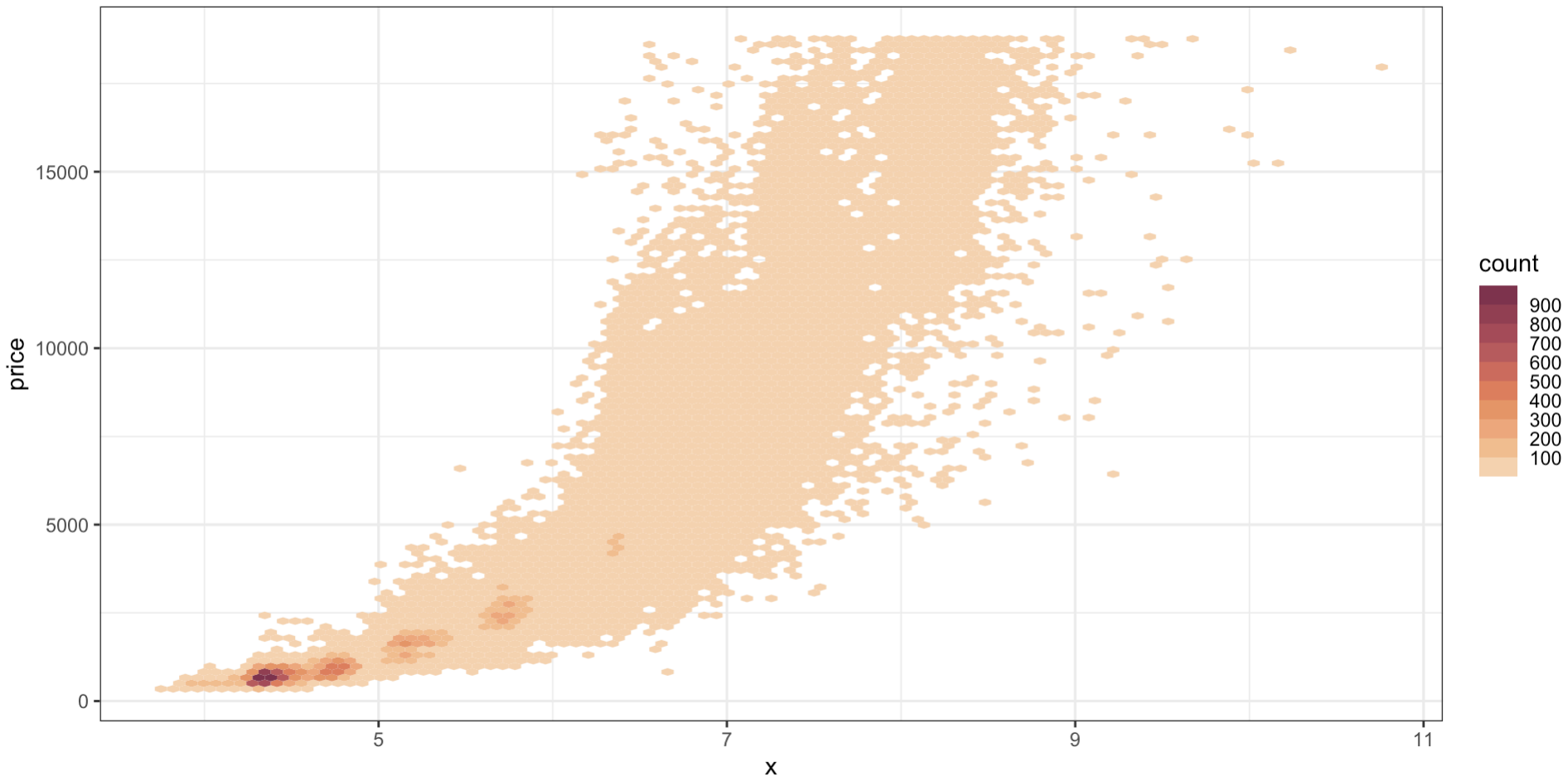


We have so many points that *overplotting* becomes an issue.

Overplotting

```
diamonds %>%
 filter(y <= 30, x >= 3) %>%
ggplot(aes(x = x, y = price)) +
  geom_hex(bins=100) + theme_bw() +
  colorspace::scale_fill_binned_sequential(palette = "BurgYl", n.breaks=10)
```

Overplotting

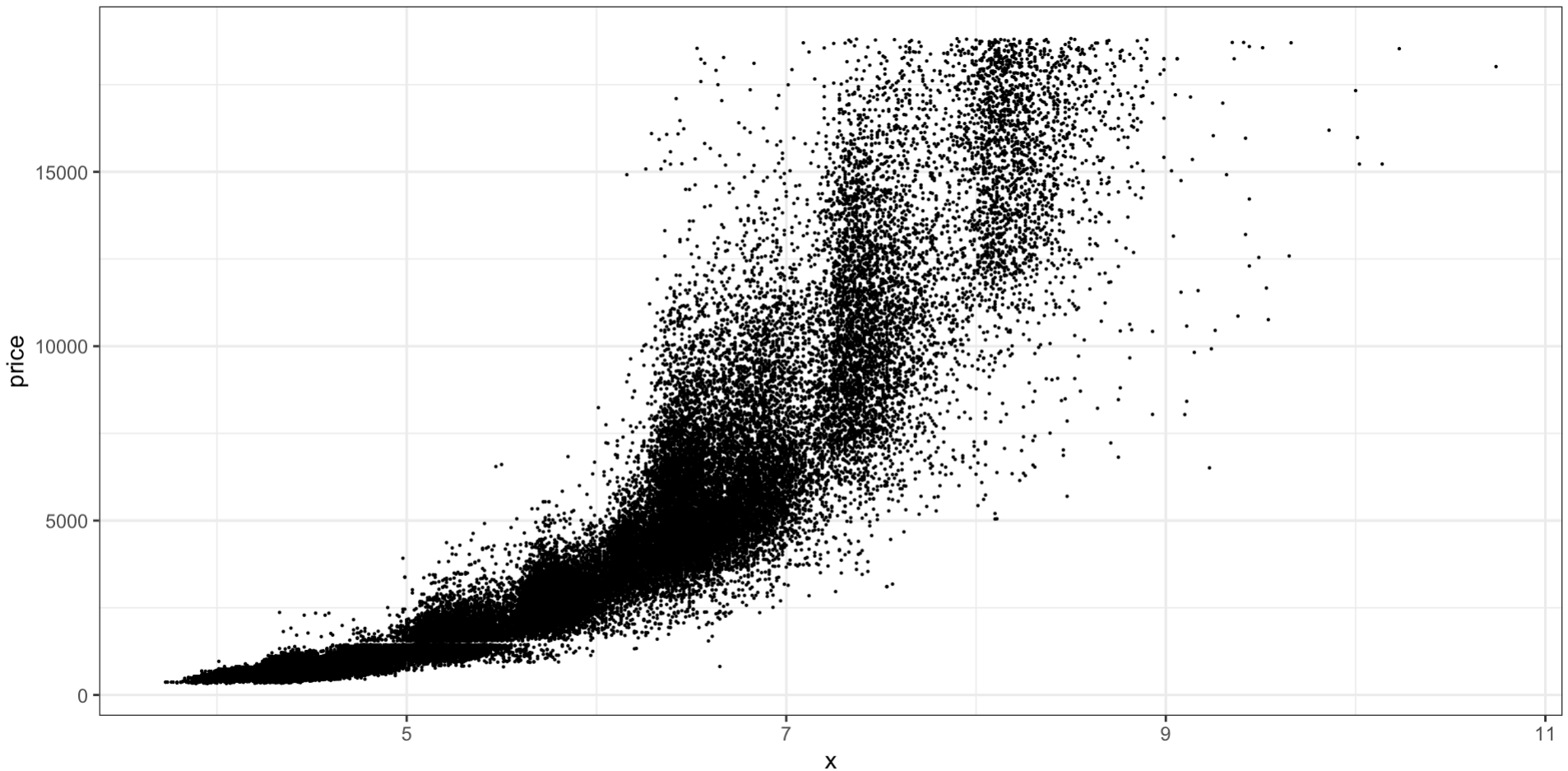


Overplotting

As an alternative, let's reduce the point size

```
diamonds %>%
 filter(y <= 30, x >= 3) %>%
ggplot(aes(x = x, y = price)) +
  geom_point(size=0.1) + theme_bw()
```

Overplotting

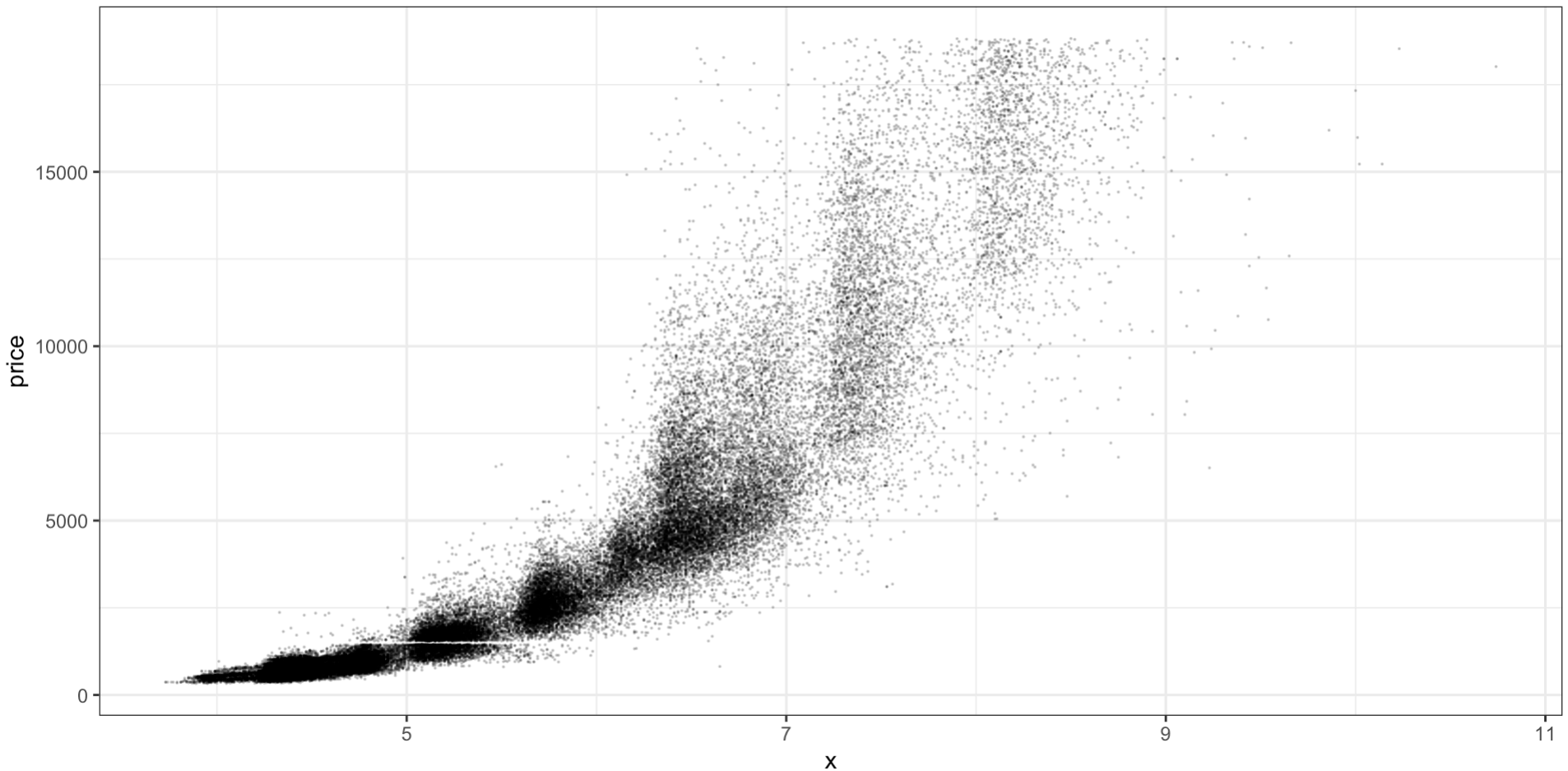


Overplotting

Let’s reduce the point size even more and add opacity:

```
diamonds %>%
 filter(y <= 30, x >= 3) %>%
ggplot(aes(x = x, y = price)) +
  geom_point(size=0.01, alpha=0.2) + theme_bw()
```

Overplotting

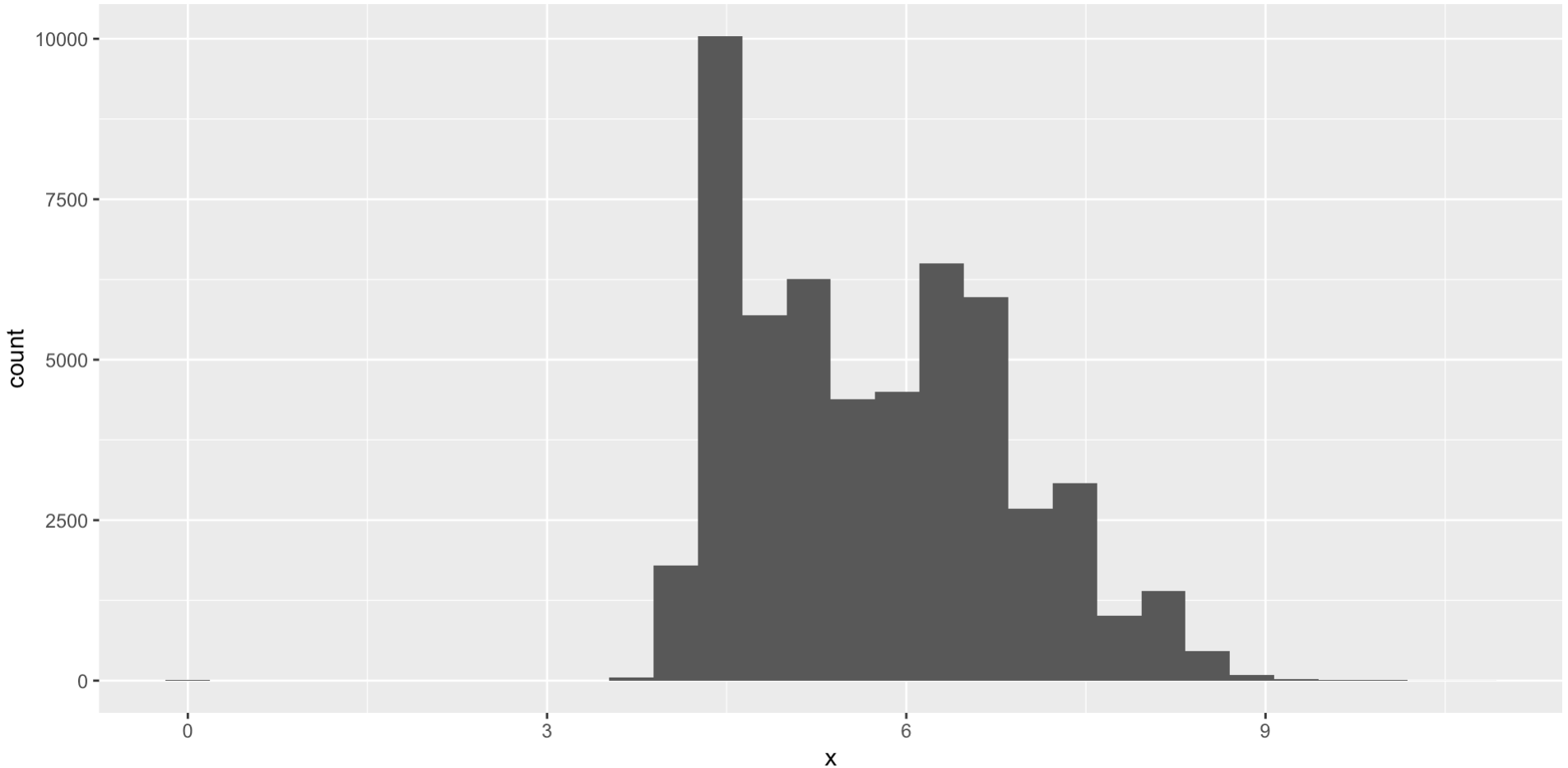


Outliers and artifacts

Let’s look at just the diamond length:

```
1 diamonds |> ggplot(aes(x=x)) + geom_histogram()
```

Outliers and artifacts

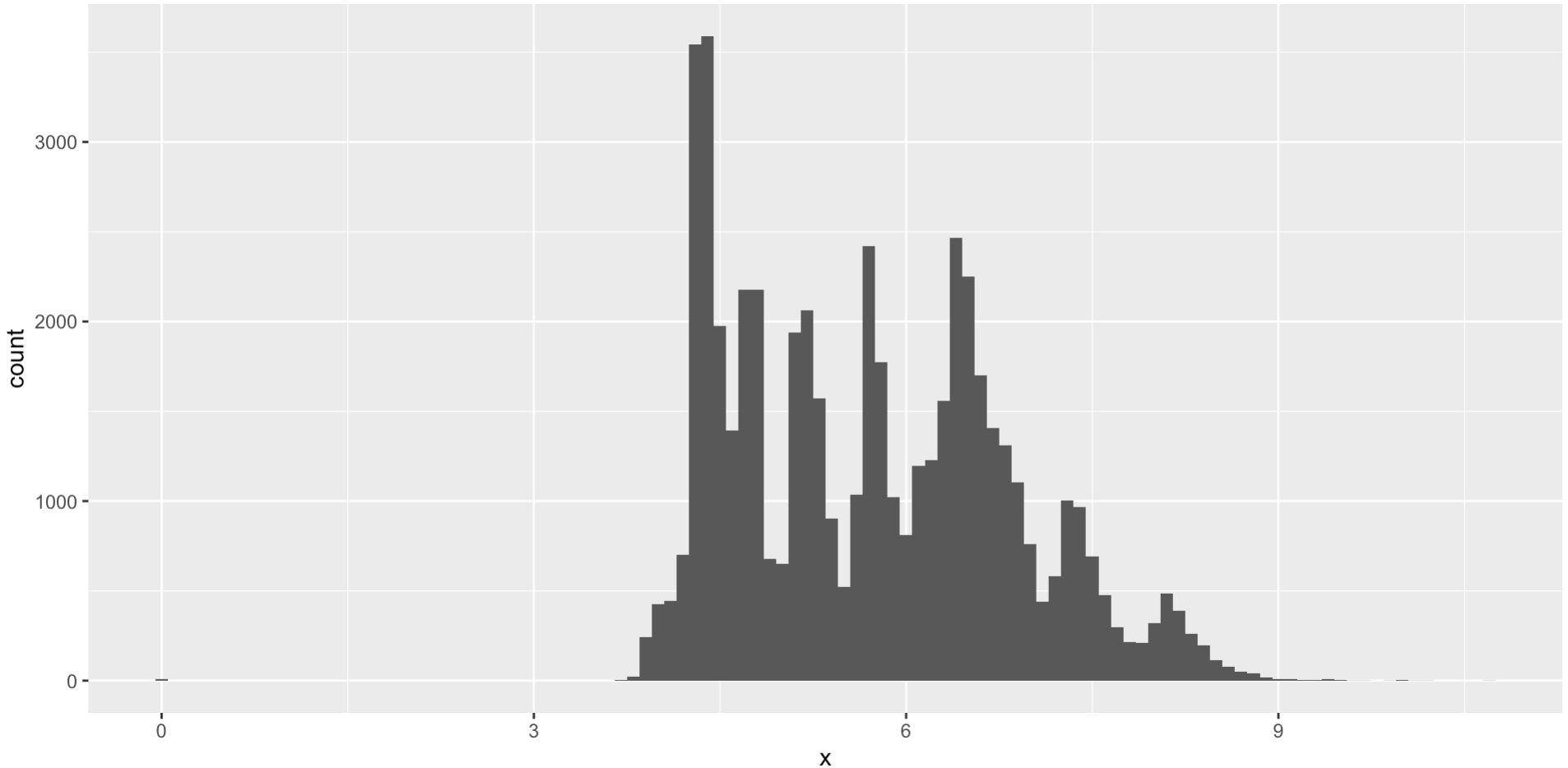


Outliers and artifacts

A smaller binwidth reveals more structure.

```
1 diamonds |> ggplot(aes(x=x)) + geom_histogram(binwidth=0.1)
```

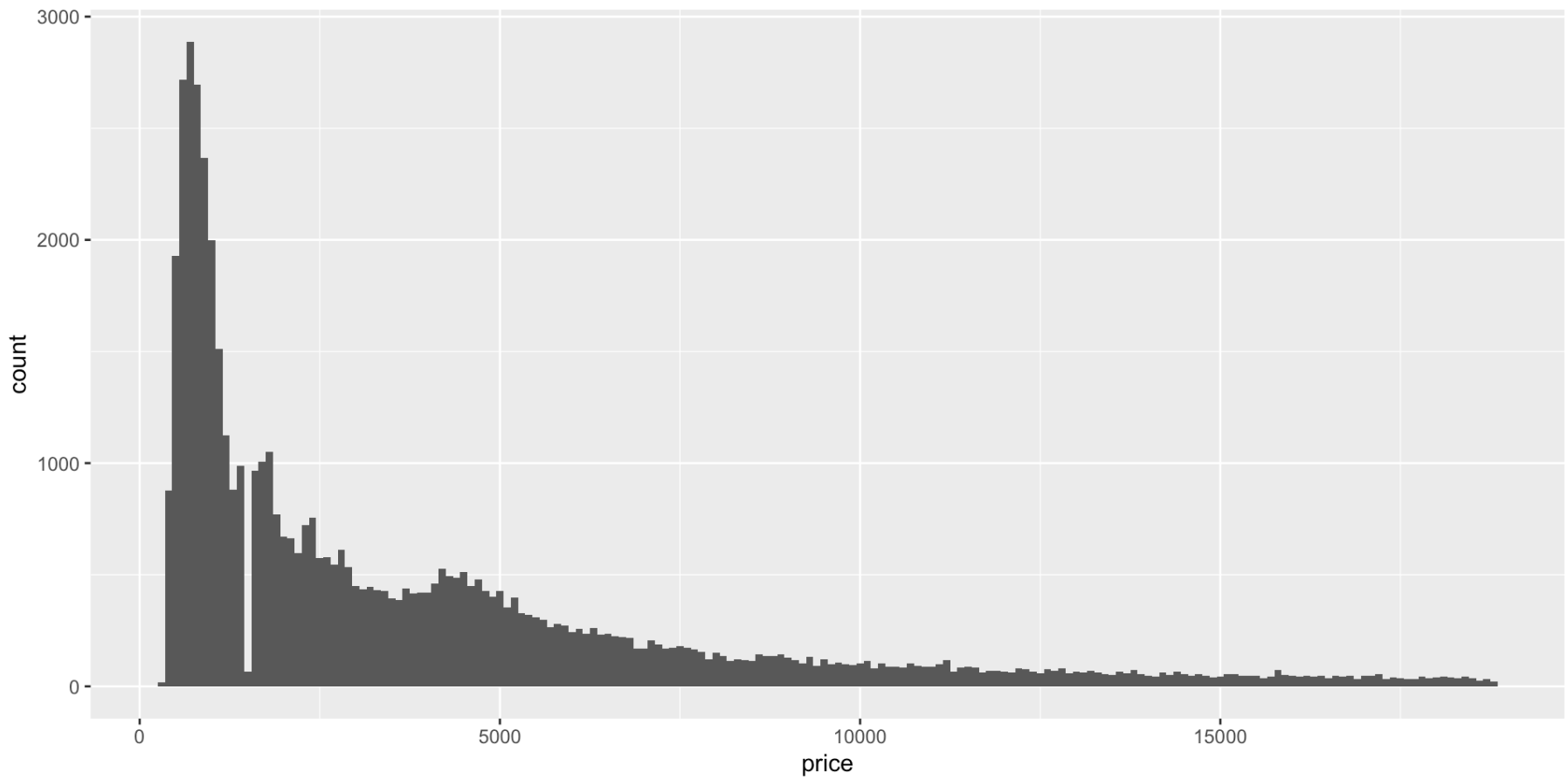
Outliers and artifacts



Outliers and artifacts

What about price?

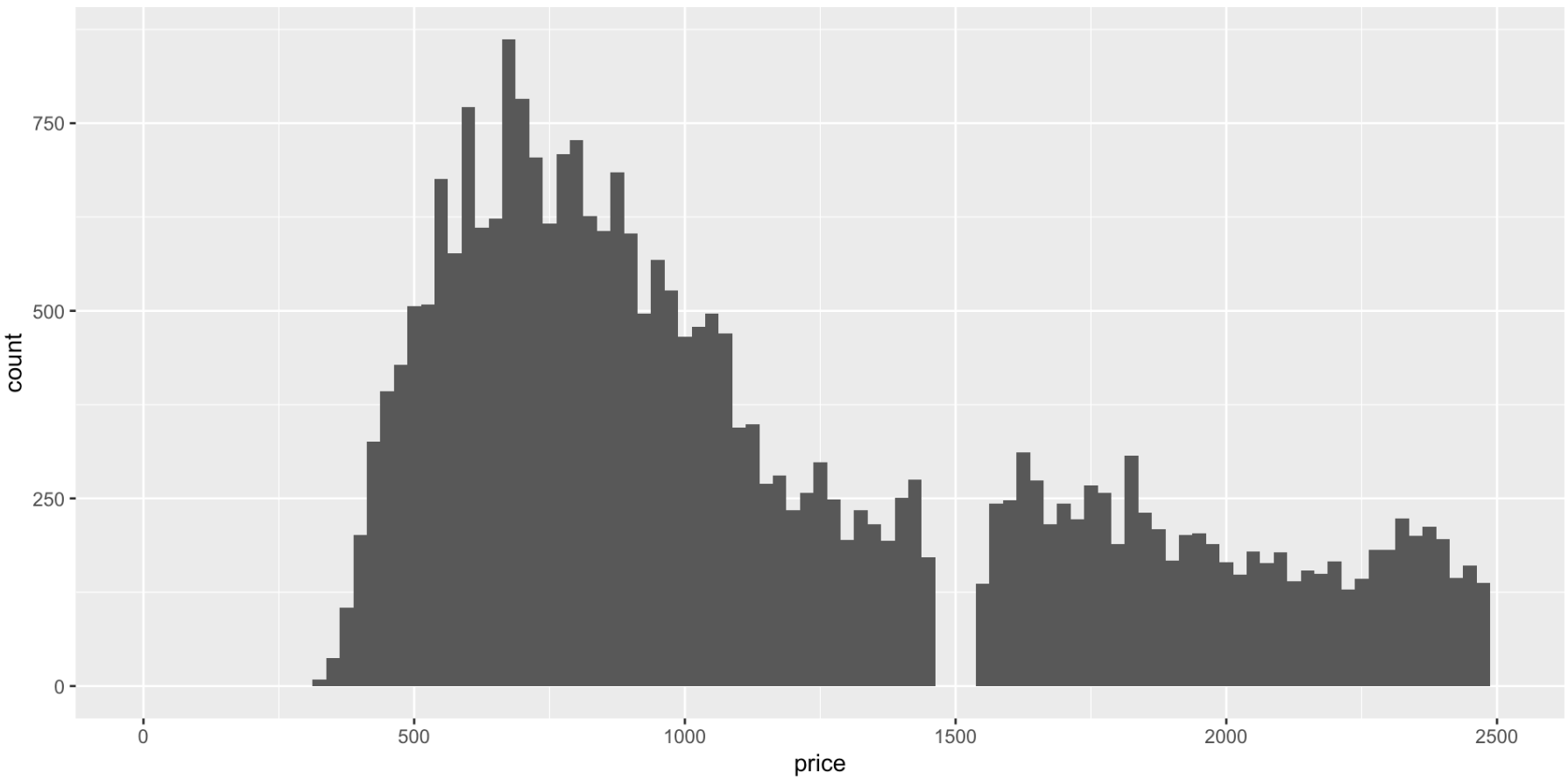
```
1 diamonds |> ggplot(aes(x=price)) +
2   geom_histogram(binwidth=100)
```



Outliers and artifacts

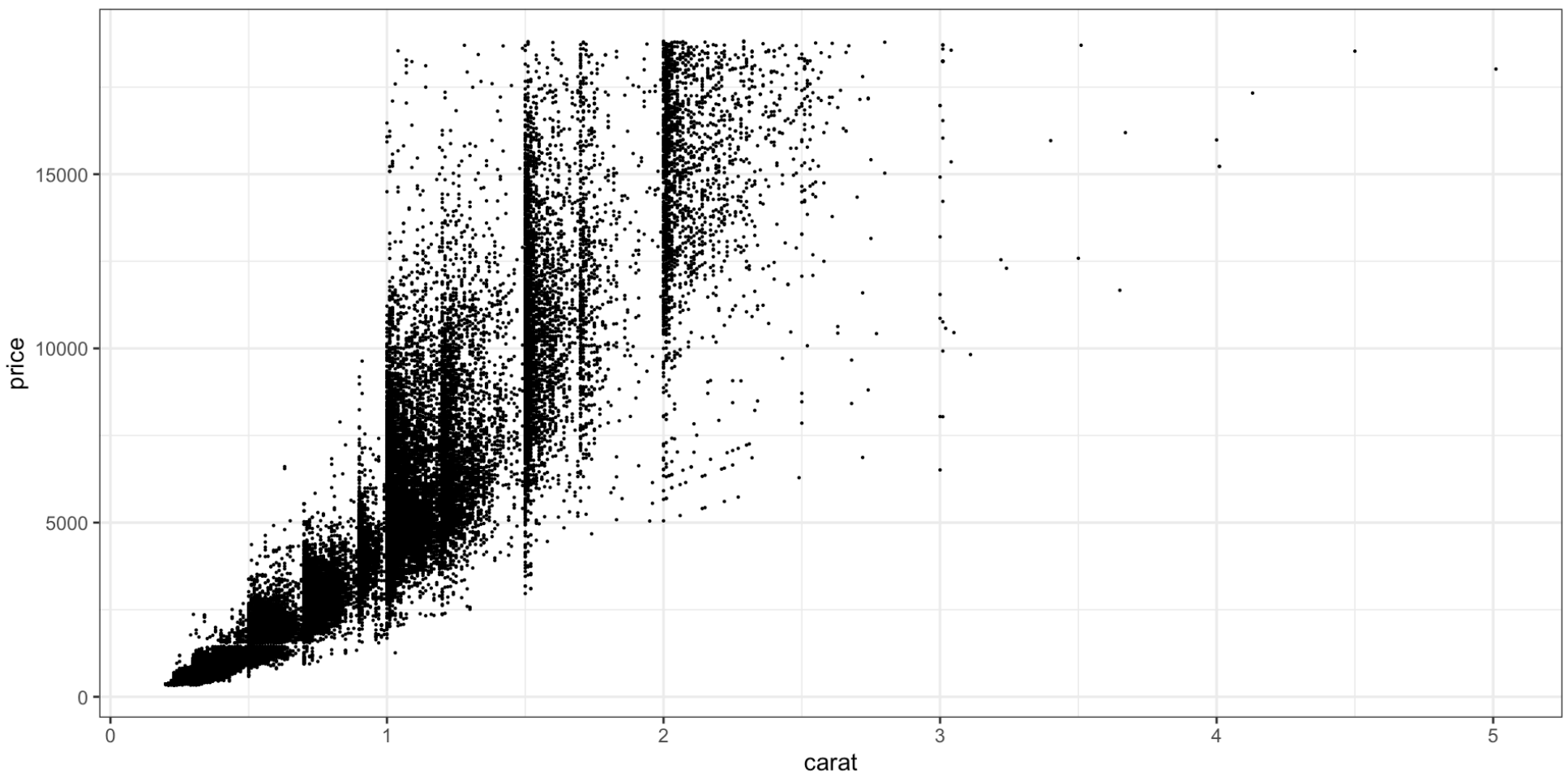
Zooming in:

```
1 diamonds |> ggplot(aes(x=price)) +
2   geom_histogram(binwidth=25) + xlim(c(0, 2500))
```



Outliers and artifacts

```
1 diamonds |> ggplot() +
2   geom_point(aes(x=carat, y=price), size=0.1) +
3   theme_bw()
```



Billboard Data

Let's return to the billboard dataset

```r
billboard_long <- billboard |>
  pivot_longer(cols = starts_with("wk"),
               names_to = "week",
               names_prefix = "wk",
               names_transform = list(week = as.integer),
               values_to = "rank")
billboard_long
```

```
# A tibble: 24,092 × 5
   artist track                   date.entered  week  rank
   <chr>  <chr>                   <date>       <int> <dbl>
 1 2 Pac  Baby Don't Cry (Keep... 2000-02-26       1    87
 2 2 Pac  Baby Don't Cry (Keep... 2000-02-26       2    82
 3 2 Pac  Baby Don't Cry (Keep... 2000-02-26       3    72
 4 2 Pac  Baby Don't Cry (Keep... 2000-02-26       4    77
 5 2 Pac  Baby Don't Cry (Keep... 2000-02-26       5    87
 6 2 Pac  Baby Don't Cry (Keep... 2000-02-26       6    94
 7 2 Pac  Baby Don't Cry (Keep... 2000-02-26       7    99
 8 2 Pac  Baby Don't Cry (Keep... 2000-02-26       8    NA
 9 2 Pac  Baby Don't Cry (Keep... 2000-02-26       9    NA
```

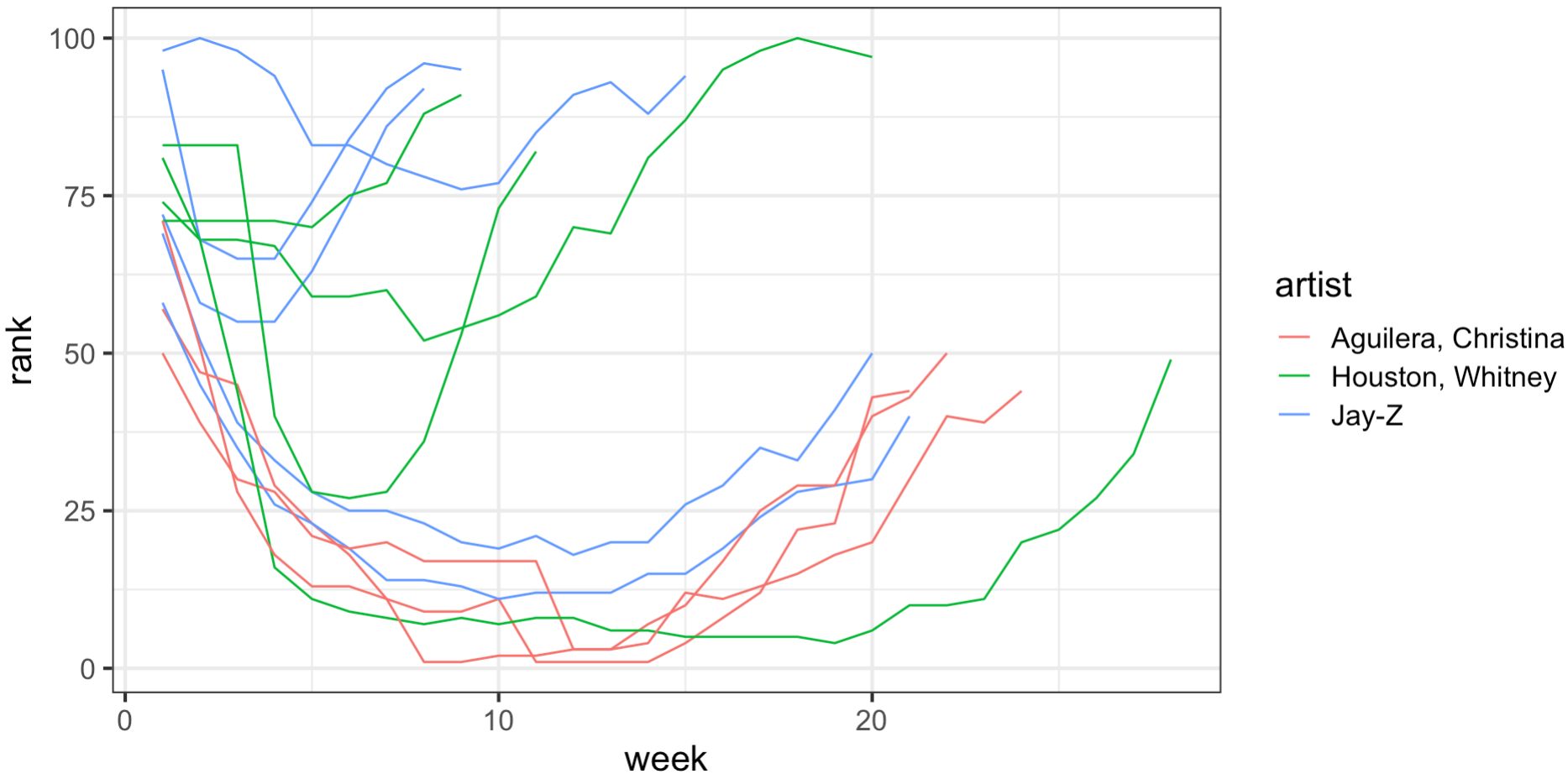
```
10 2 Pac  Baby Don't Cry (Keep... 2000-02-26      10    NA
# ℹ 24,082 more rows
```

Exploring song rank

Let's plot the change in song rank for the songs by 3 artists by week:

```
billboard_long |>
  filter(artist %in% c("Jay-Z", "Houston, Whitney", "Aguilera, Christina")) |>
  drop_na() |>
  ggplot() + geom_line(aes(x=week, y=rank, col=artist, group=track)) +
  theme_bw(base_size=16)
```

Exploring song rank

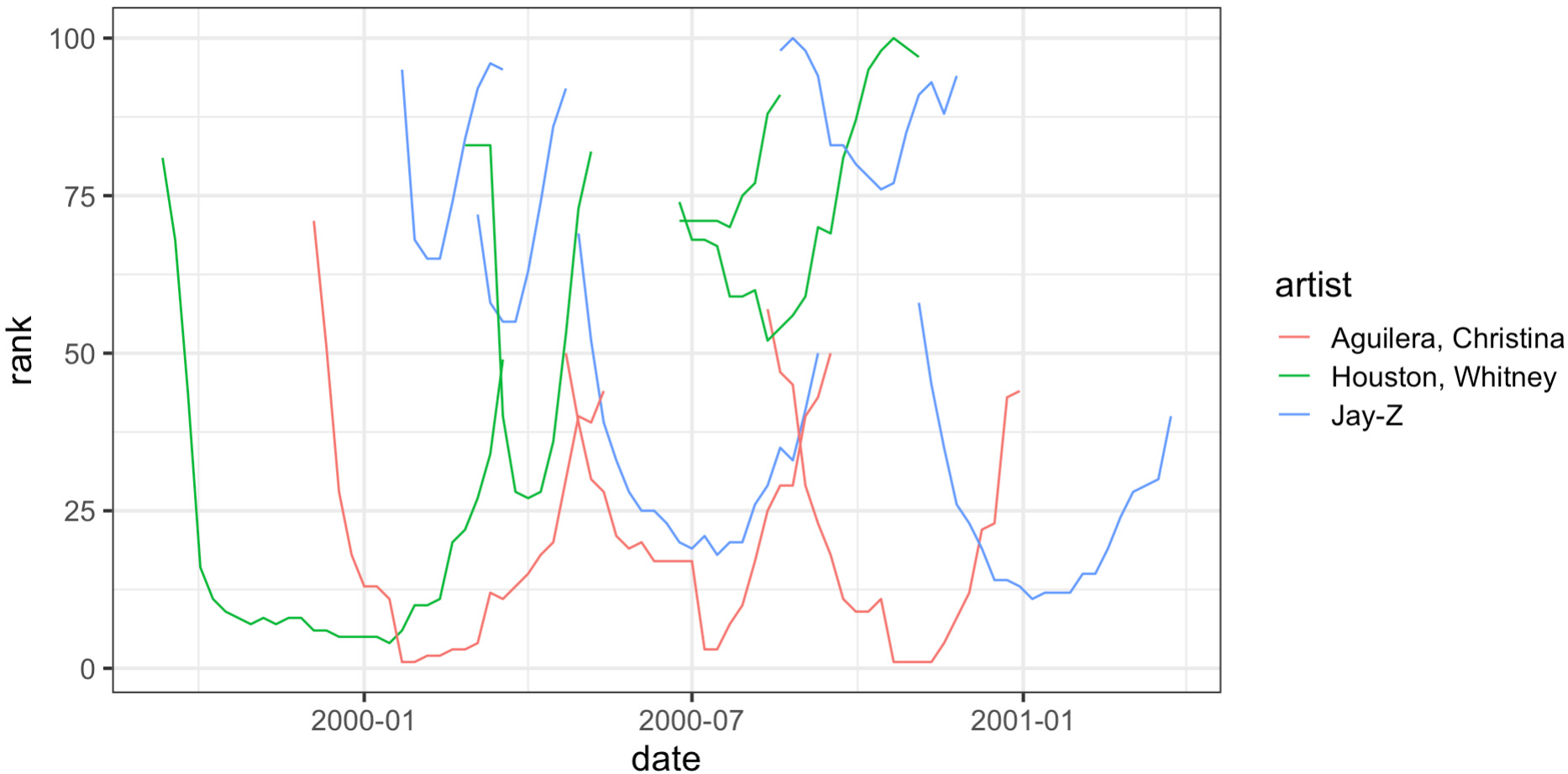


Exploring song rank

Create a new column which is the actual date the song achieved that rank:

```
billboard_long |>
  filter(artist %in% c("Jay-Z", "Houston, Whitney", "Aguilera, Christina")) |>
  drop_na() |>
  mutate(date = date.entered + weeks(week)) |>
  ggplot() + geom_line(aes(x=date, y=rank, col=artist, group=track)) +
  theme_bw(base_size=16)
```

Exploring song rank



Billboard Data

```r
billboard_long <- billboard |>
  pivot_longer(cols = starts_with("wk"),
               names_to = "week",
               names_prefix = "wk",
               names_transform = list(week = as.integer),
               values_to = "rank")
billboard_long
```

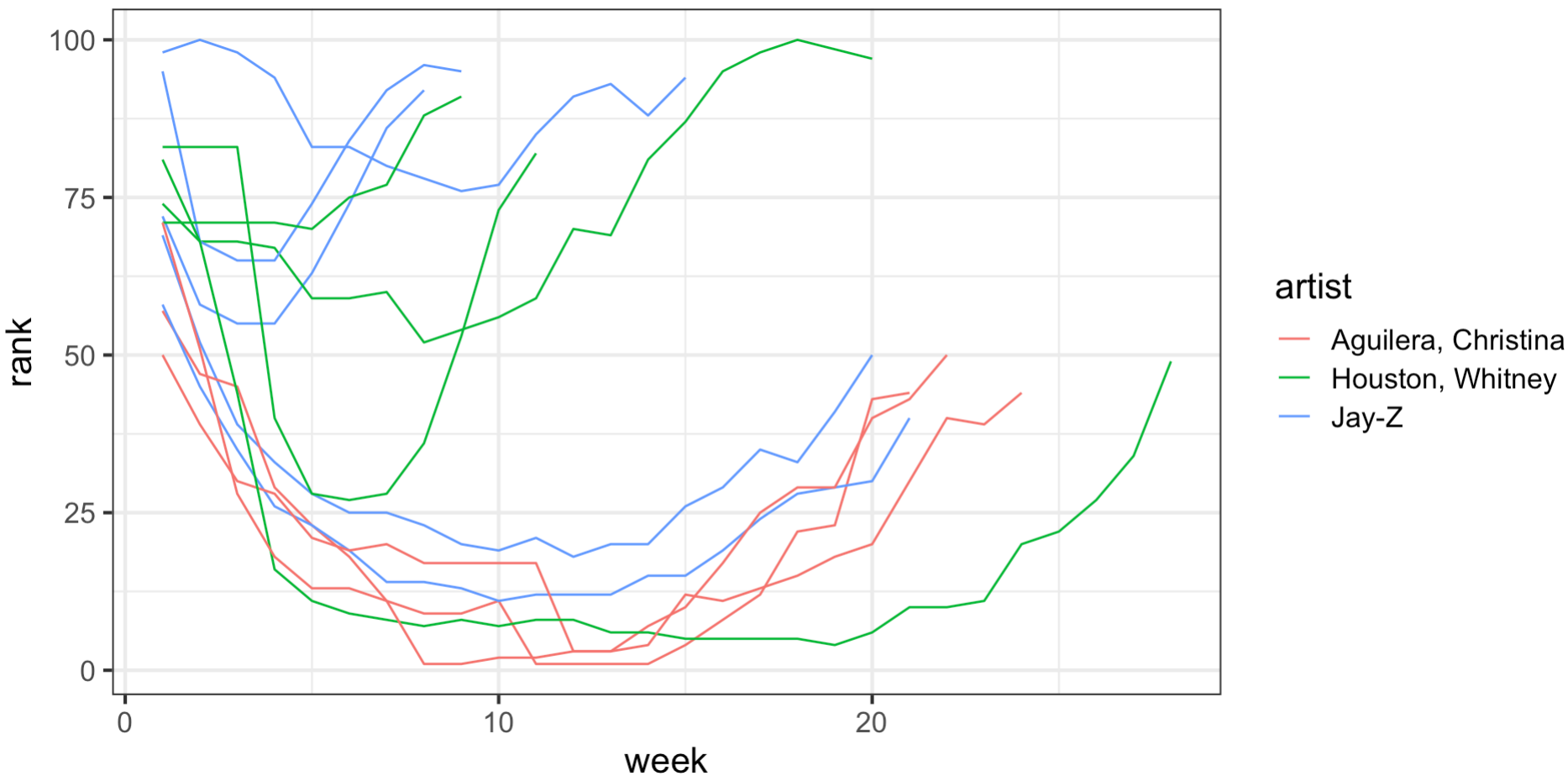
```
# A tibble: 24,092 × 5
   artist track                   date.entered  week  rank
   <chr>  <chr>                   <date>       <int> <dbl>
 1 2 Pac  Baby Don't Cry (Keep... 2000-02-26       1    87
 2 2 Pac  Baby Don't Cry (Keep... 2000-02-26       2    82
 3 2 Pac  Baby Don't Cry (Keep... 2000-02-26       3    72
 4 2 Pac  Baby Don't Cry (Keep... 2000-02-26       4    77
 5 2 Pac  Baby Don't Cry (Keep... 2000-02-26       5    87
 6 2 Pac  Baby Don't Cry (Keep... 2000-02-26       6    94
 7 2 Pac  Baby Don't Cry (Keep... 2000-02-26       7    99
 8 2 Pac  Baby Don't Cry (Keep... 2000-02-26       8    NA
 9 2 Pac  Baby Don't Cry (Keep... 2000-02-26       9    NA
10 2 Pac  Baby Don't Cry (Keep... 2000-02-26      10    NA
# ℹ 24,082 more rows
```

Exploring song rank

Let's plot the change in song rank for the songs by 3 artists by week:

```
billboard_long |>
  filter(artist %in% c("Jay-Z", "Houston, Whitney", "Aguilera, Christina")) |>
  drop_na() |>
  ggplot() + geom_line(aes(x=week, y=rank, col=artist, group=track)) +
  theme_bw(base_size=16)
```

Exploring song rank

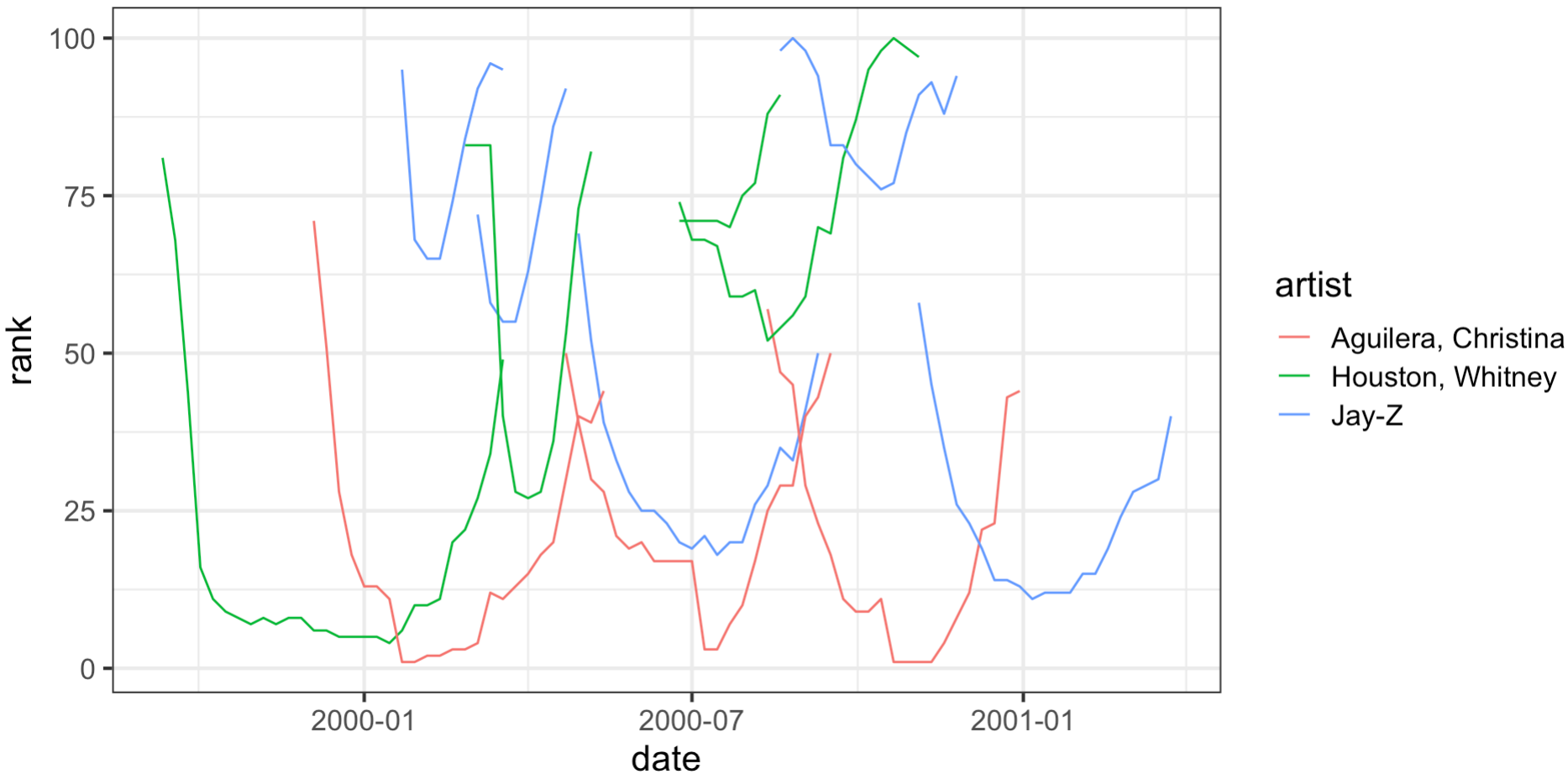


Exploring song rank

Create a new column which is the actual date the song achieved that rank:

```
billboard_long |>
  filter(artist %in% c("Jay-Z", "Houston, Whitney", "Aguilera, Christina")) |>
  drop_na() |>
  mutate(date = date.entered + weeks(week)) |>
  ggplot() + geom_line(aes(x=date, y=rank, col=artist, group=track)) +
  theme_bw(base_size=16)
```

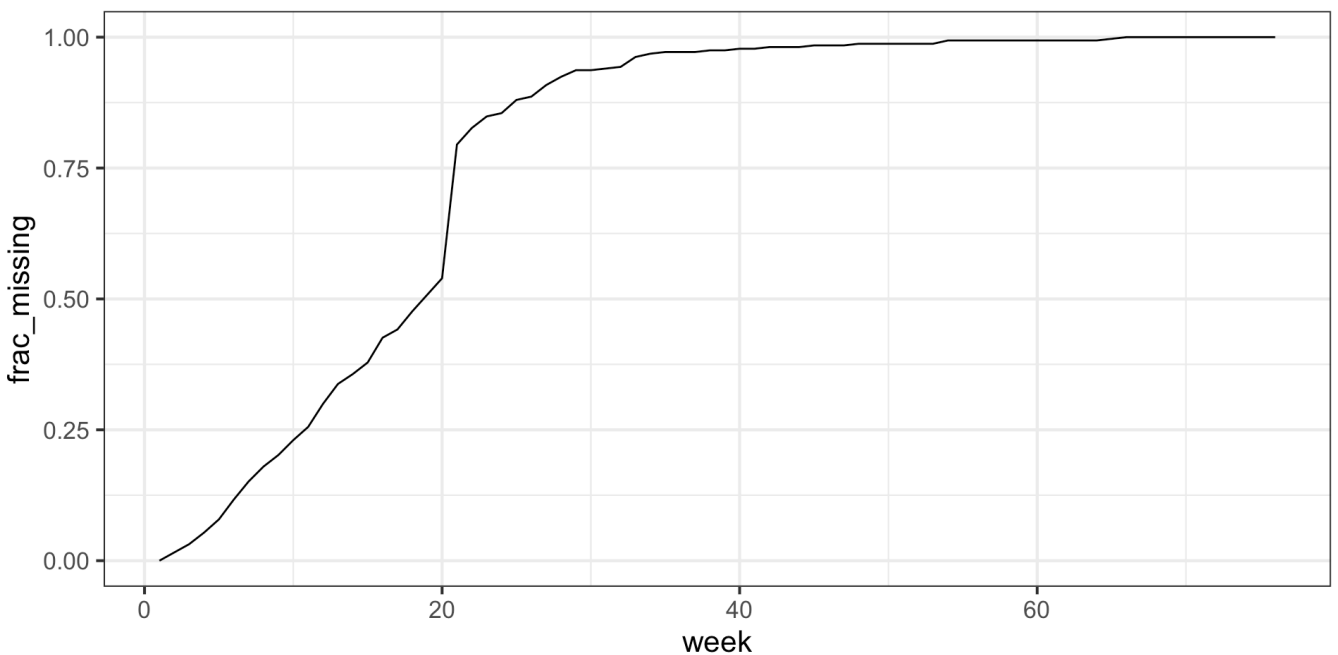
Exploring song rank



Exploring Missingness

There is a lot of missingness. What patterns can we identify?

```
#output-location: slide
billboard_long |>
  group_by(week) |>
  summarize(frac_missing = mean(is.na(rank))) |>
  ggplot() + geom_line(aes(x=week, y=frac_missing)) +
  theme_bw(base_size=16)
```

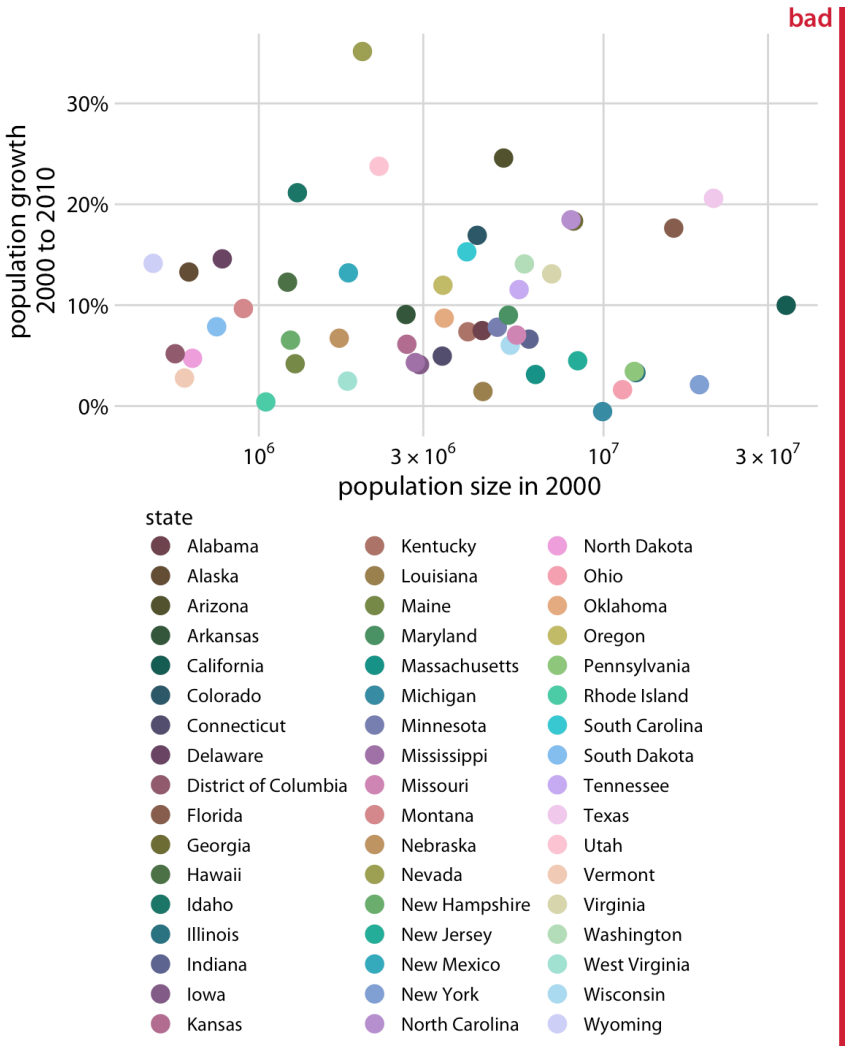


Workflow tips and advice

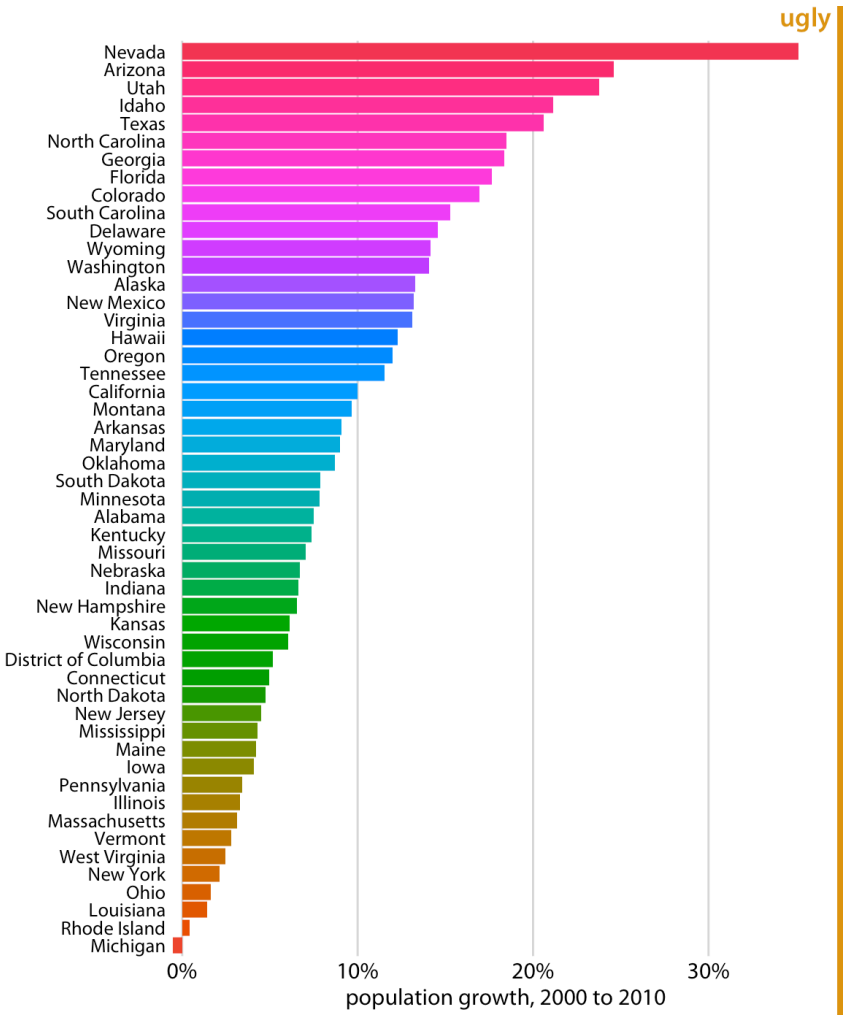
Developing graphics is an iterative trial-and-error process:

- **Hone in on an essential question or relationship.**
  - This might require making a few crude plots to help you decide where to focus.
- **Start simple.**
  - Begin with the most basic plot you can.
- **Add complexity as you go.**
  - Decide what to add or change based on *your own questions* rather than an envisioned endpoint.
- **Change one thing at a time, and keep copies.**
  - You might well need to backtrack.
- **Don’t be afraid of starting over.**
  - If an idea doesn’t pan out, no harm done!

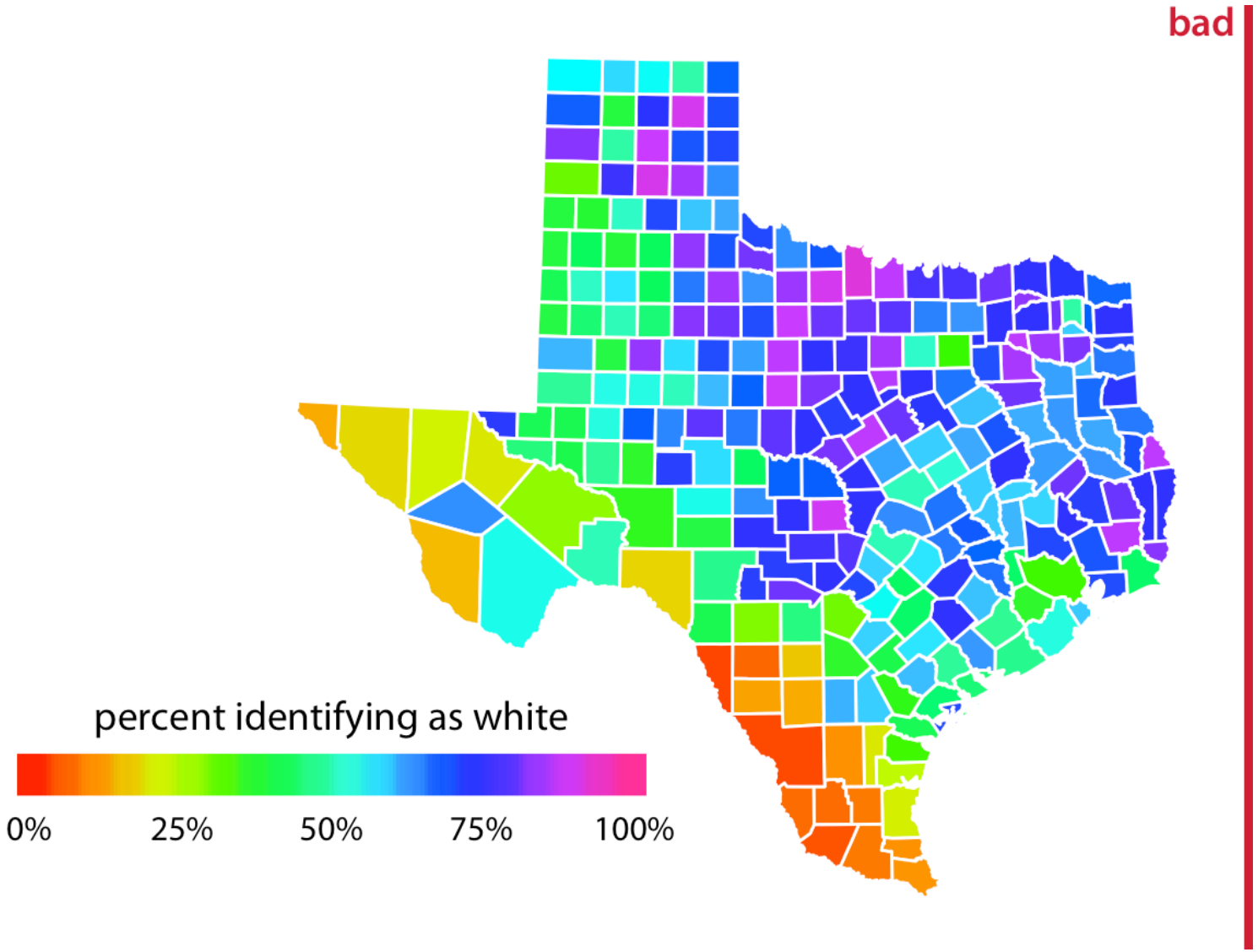
Pitfalls of Color Use



Pitfalls of Color Use



Pitfalls of Color Use



Color palettes

Choose the color palette that is most appropriate for your data:

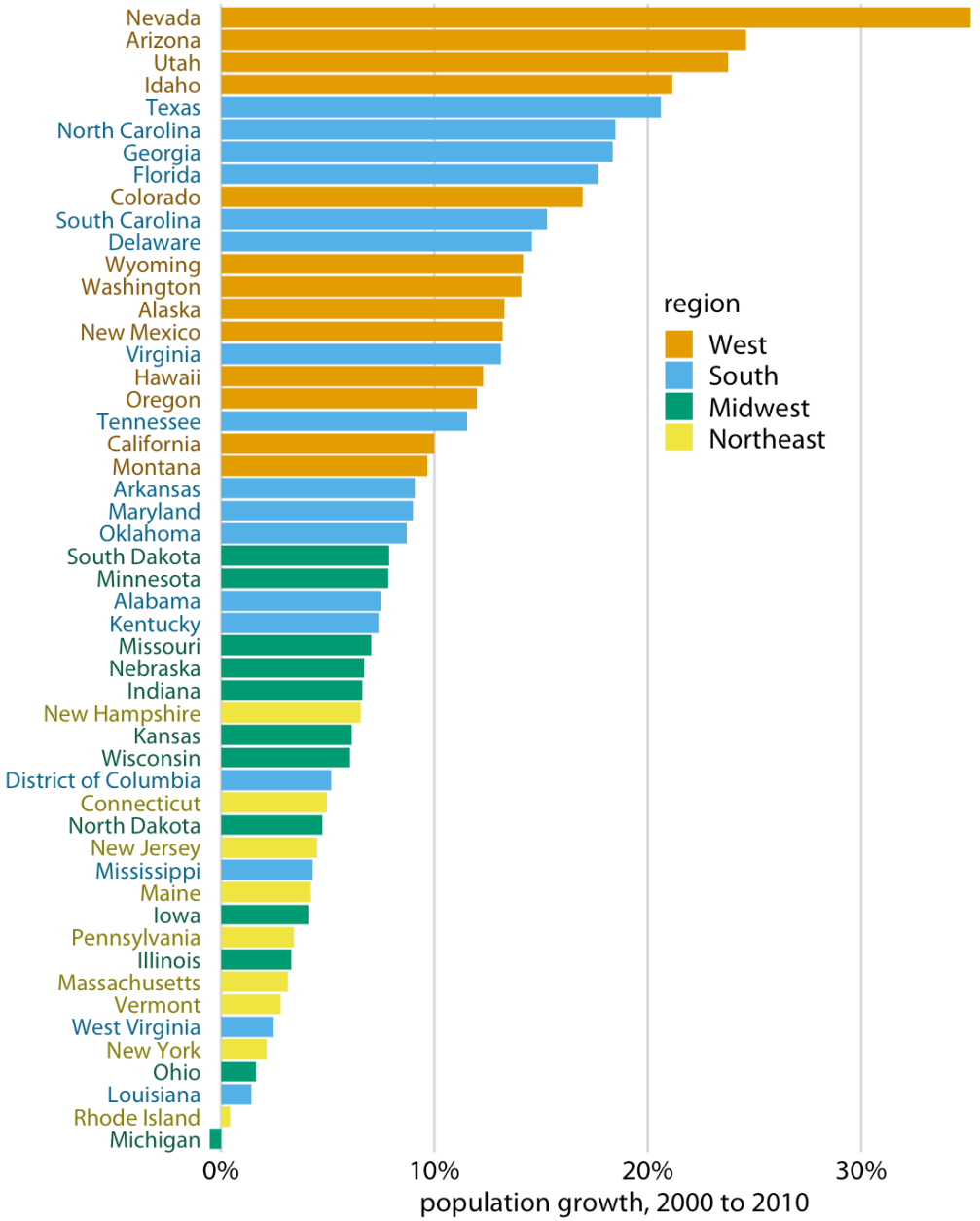
- *Qualitative:* Designed for coding categorical information, i.e., where no particular ordering of categories is available
- *Sequential:* Designed for coding ordered/numeric information, i.e., where colors go from high to low (or vice versa)
- *Diverging:* Designed for coding ordered/numeric information around a central neutral value, i.e., where colors diverge from neutral to two extremes.

Color

There are three main use cases for *color*:

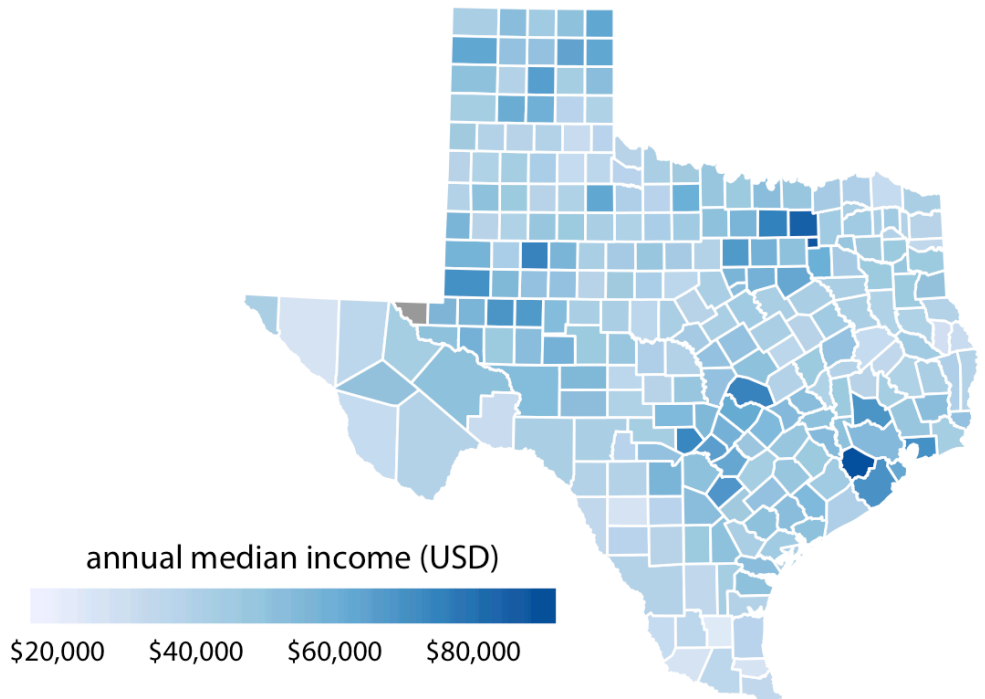
1. Color to distinguish groups of data
2. Color to represent data values
3. Color to highlight

Color for distinguishing groups



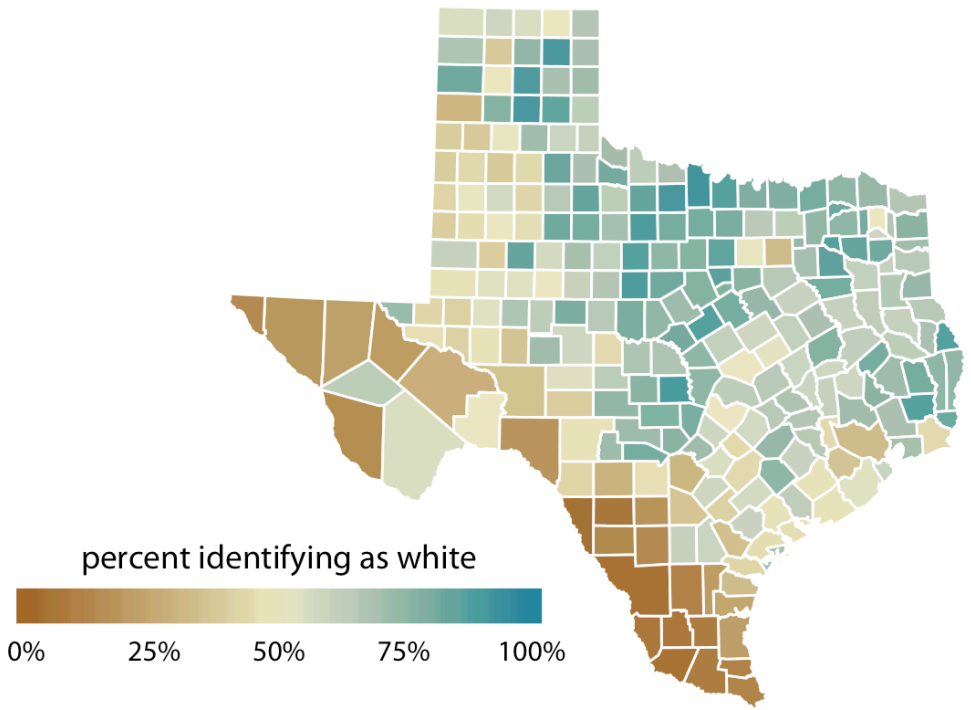
This uses a *qualitative palette* to distinguish.

Color for representing data



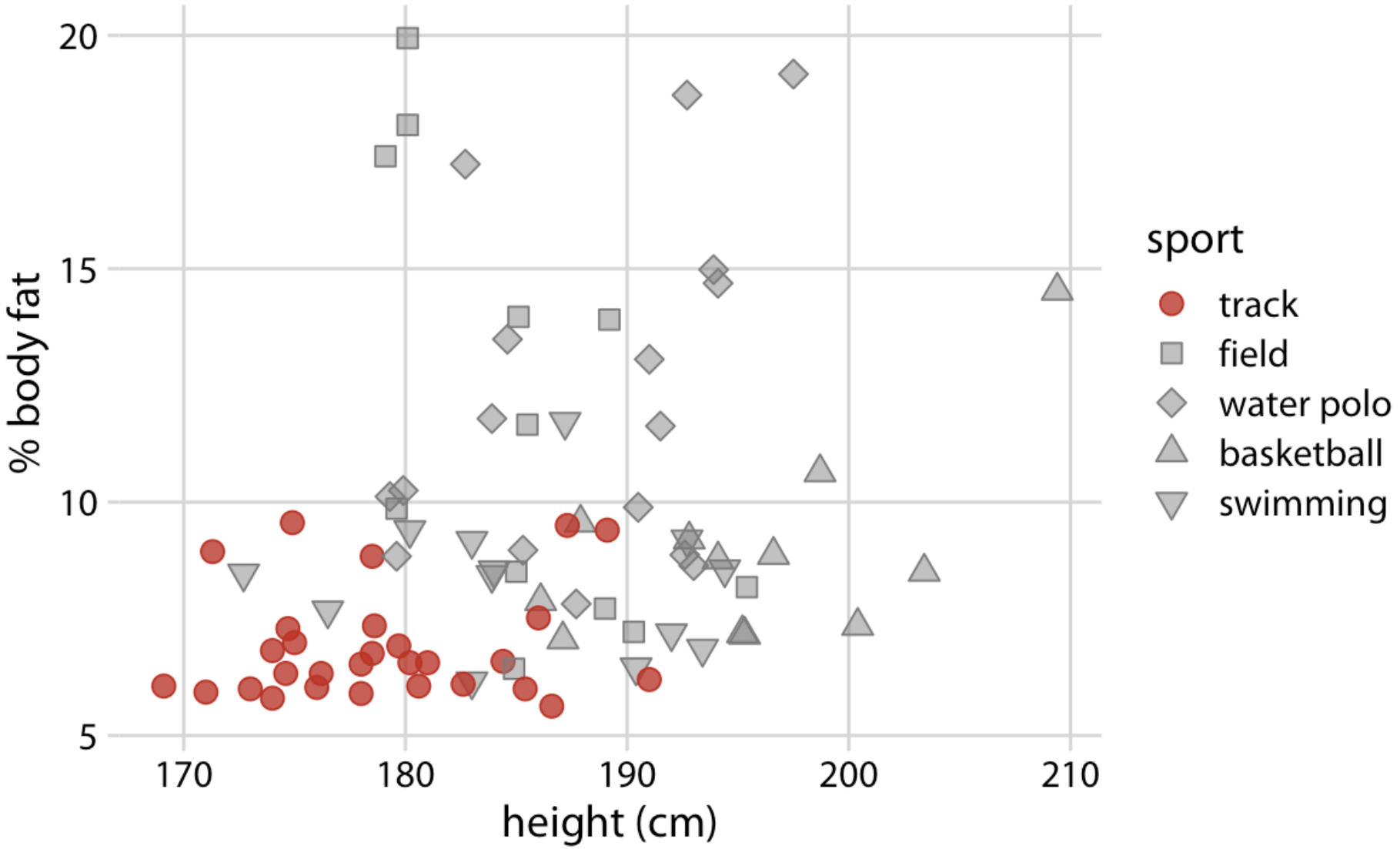
This uses a *sequentially palette* to distinguish high income (dark) from low incom (light)

Color for representing data

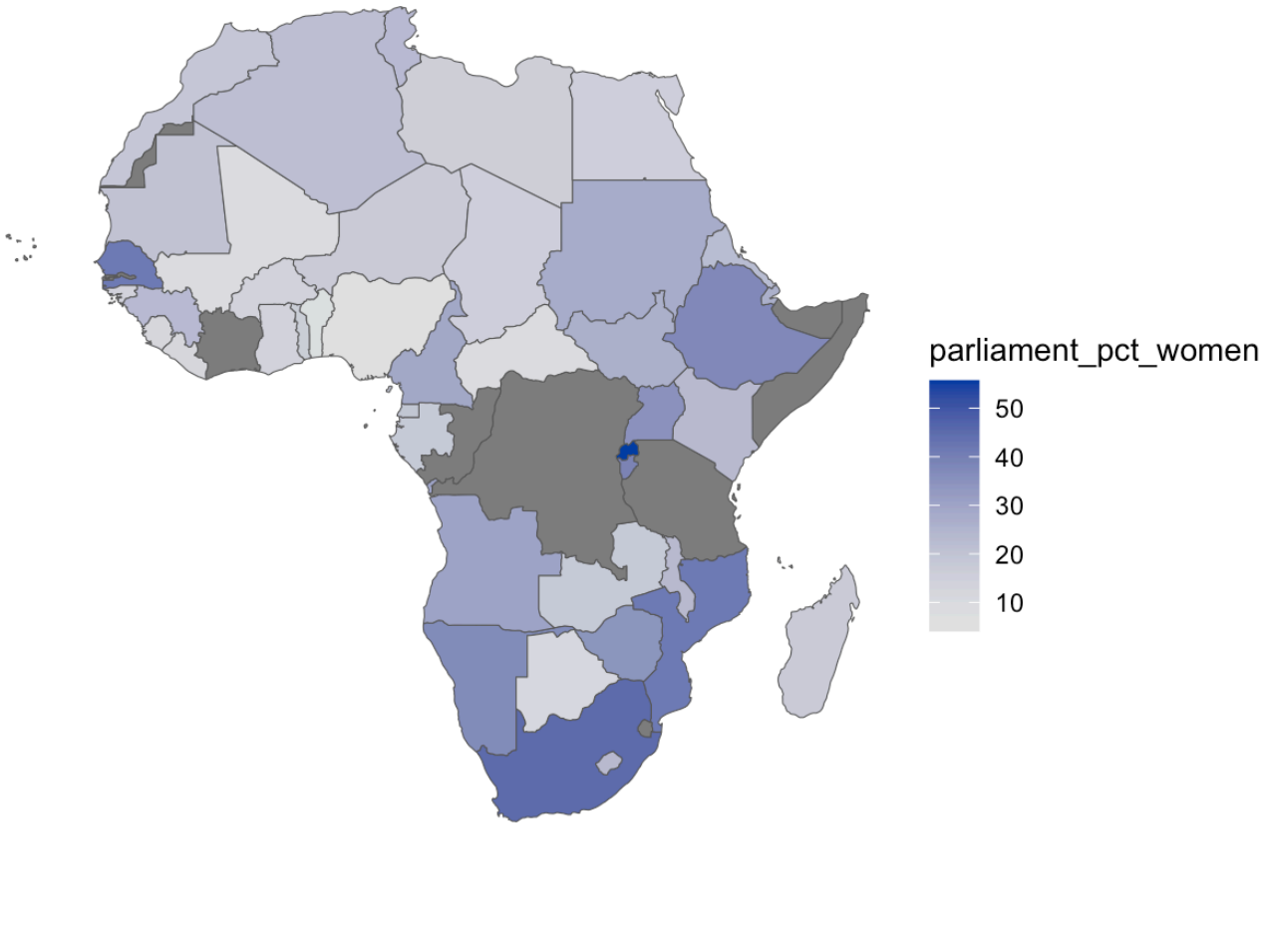


Notice how the colors *diverge* from toward two distinct colors away from the reference (50%).

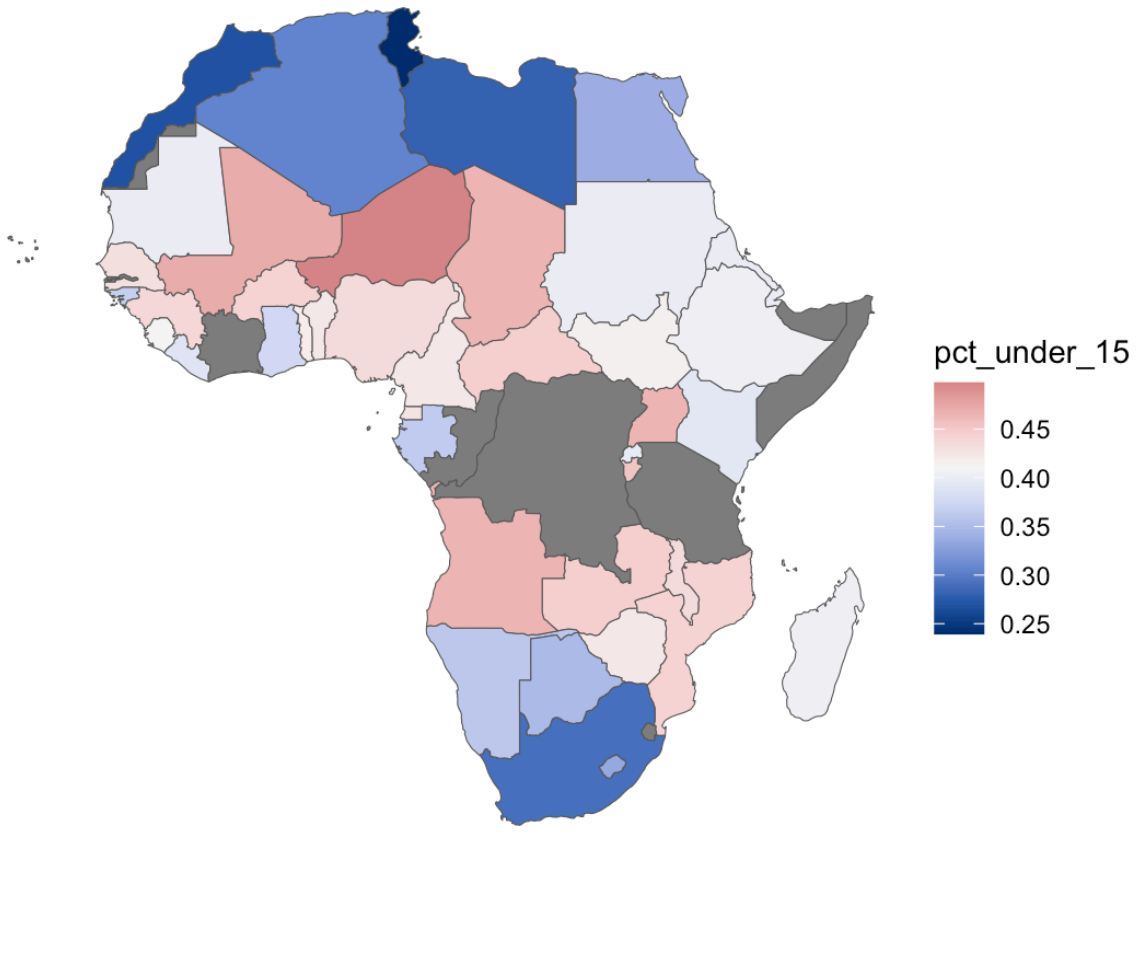
Color for highlighting



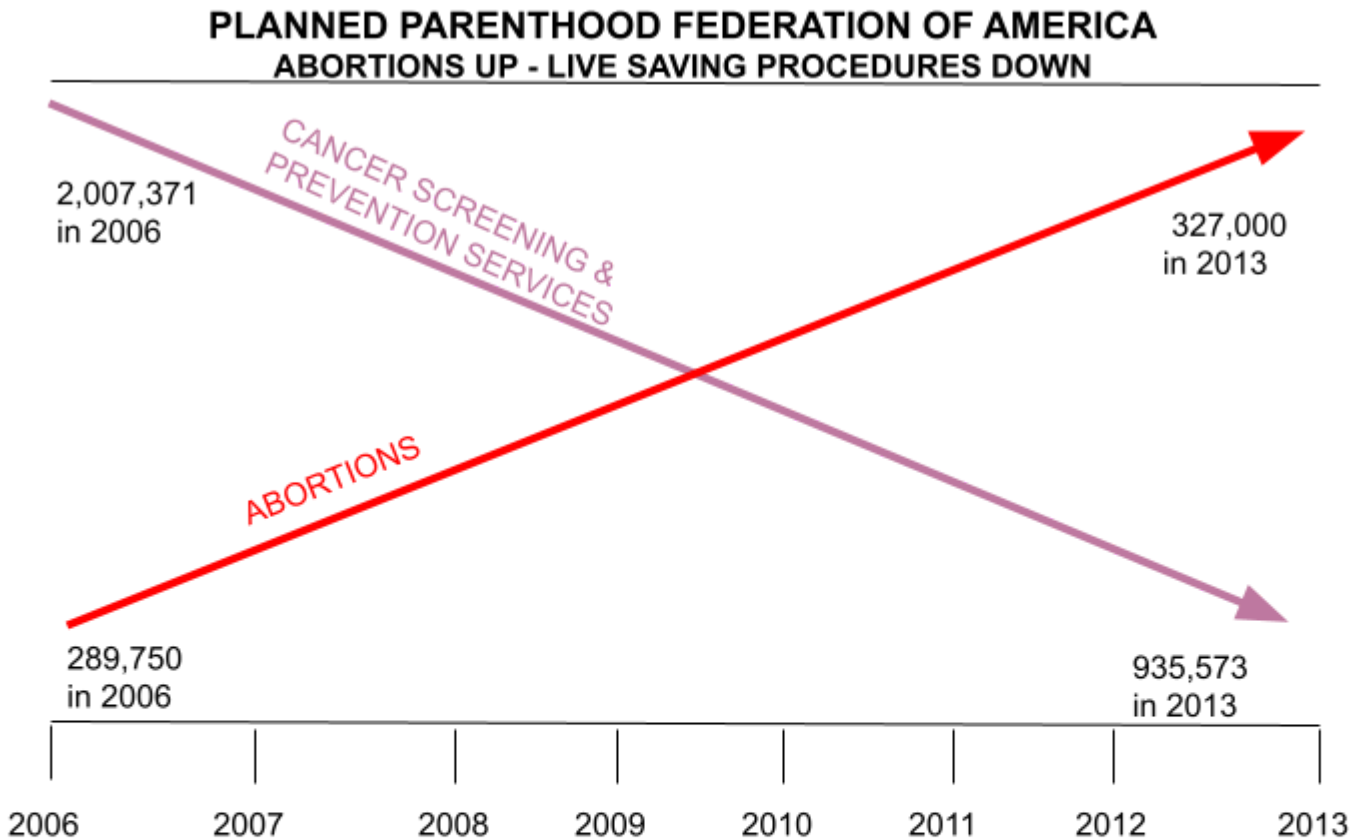
Sequential colors



Diverging colors



Planned Parenthood Example



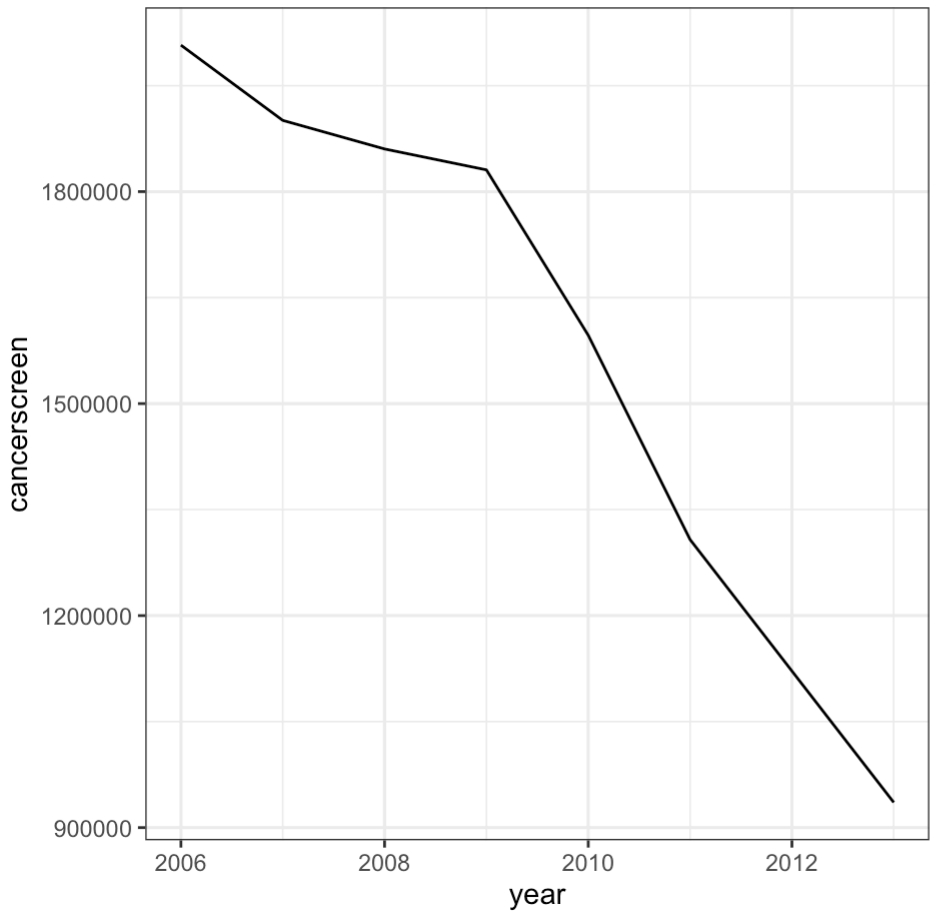
Source: American United for Life

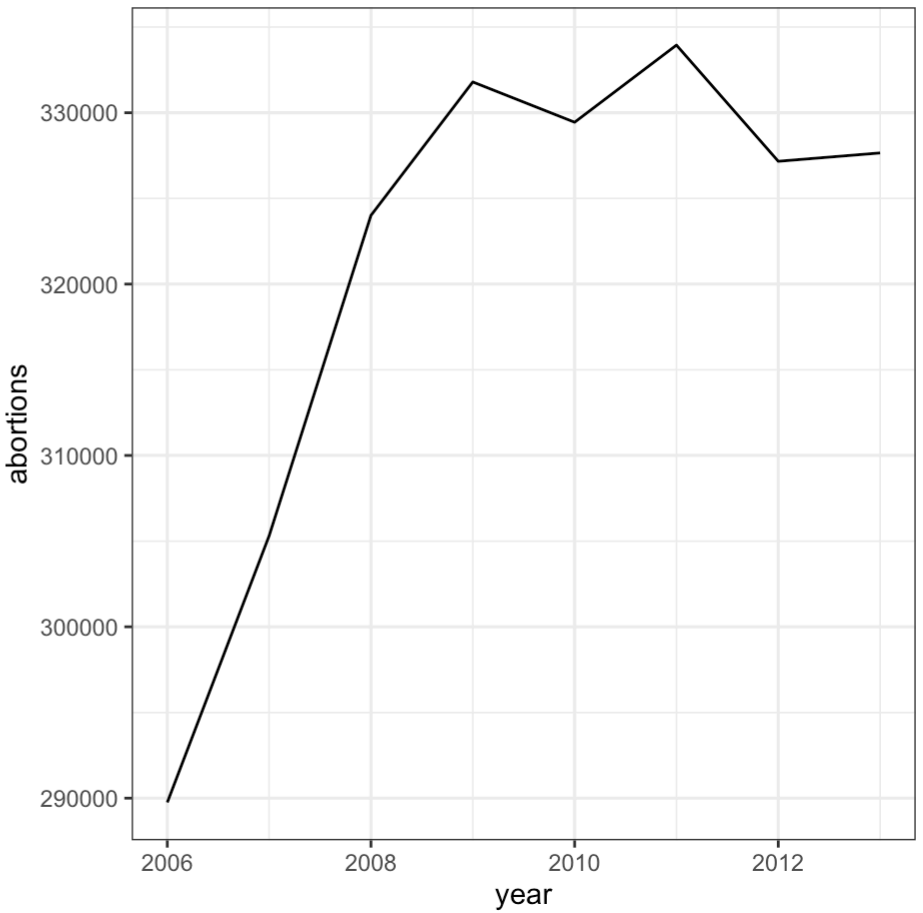
Planned parenthood services

```
cancer_plot <- pp_data |> ggplot() + geom_line(aes(x=year, y=cancerscreen))
abortion_plot <- pp_data |> ggplot() + geom_line(aes(x=year, y=abortions))

## Plot them side by side
cancer_plot + abortion_plot
```

Planned parenthood services

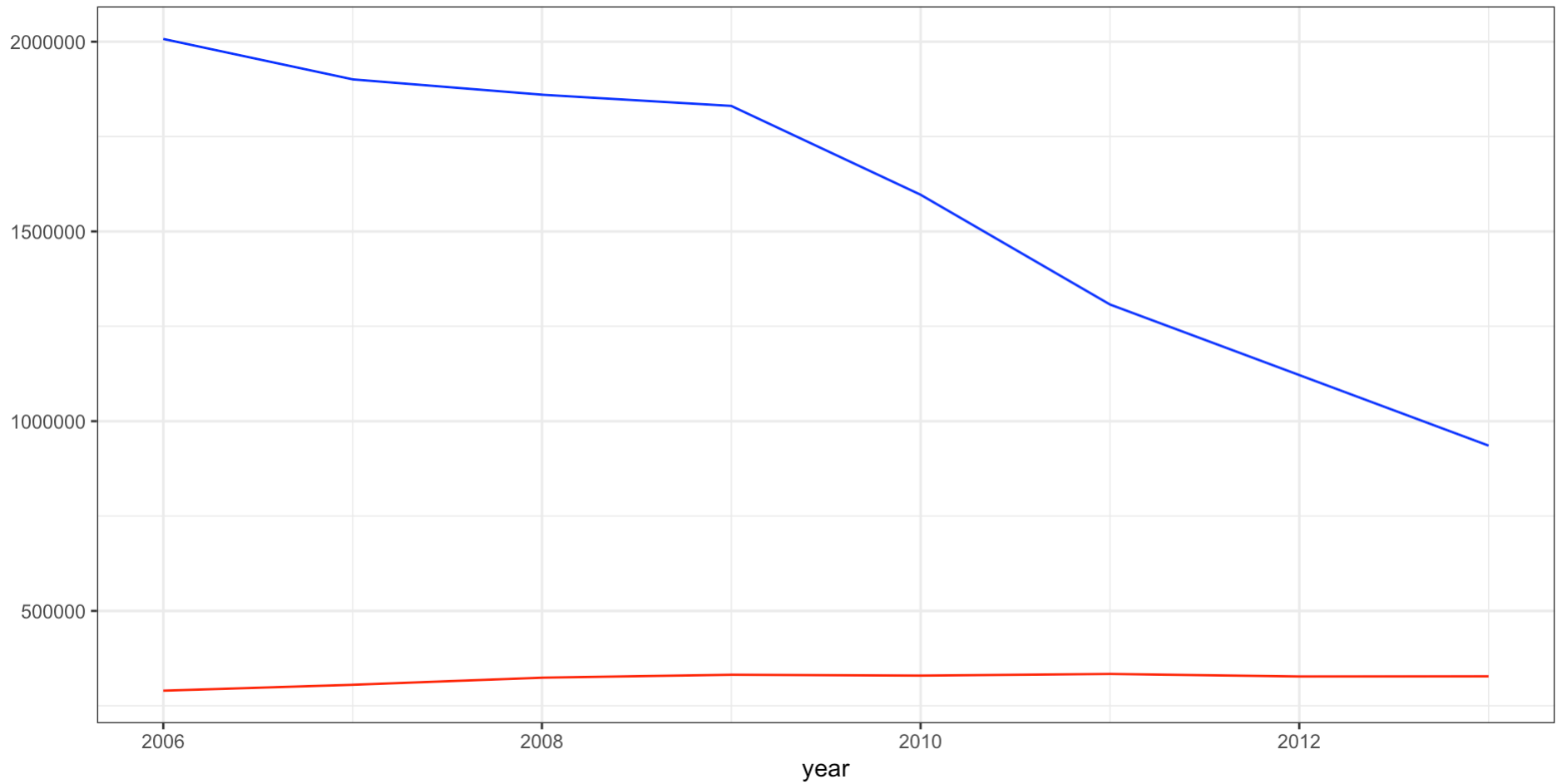




On the same plot

```
pp_data |> ggplot() +
  geom_line(aes(x=year, y=cancerscreen), col="blue") +
  geom_line(aes(x=year, y=abortions), col="red") +
  ylab("") +
  theme_bw()
```

On the same plot



Comparing data across groups

What's the best way to compare data across multiple groups?

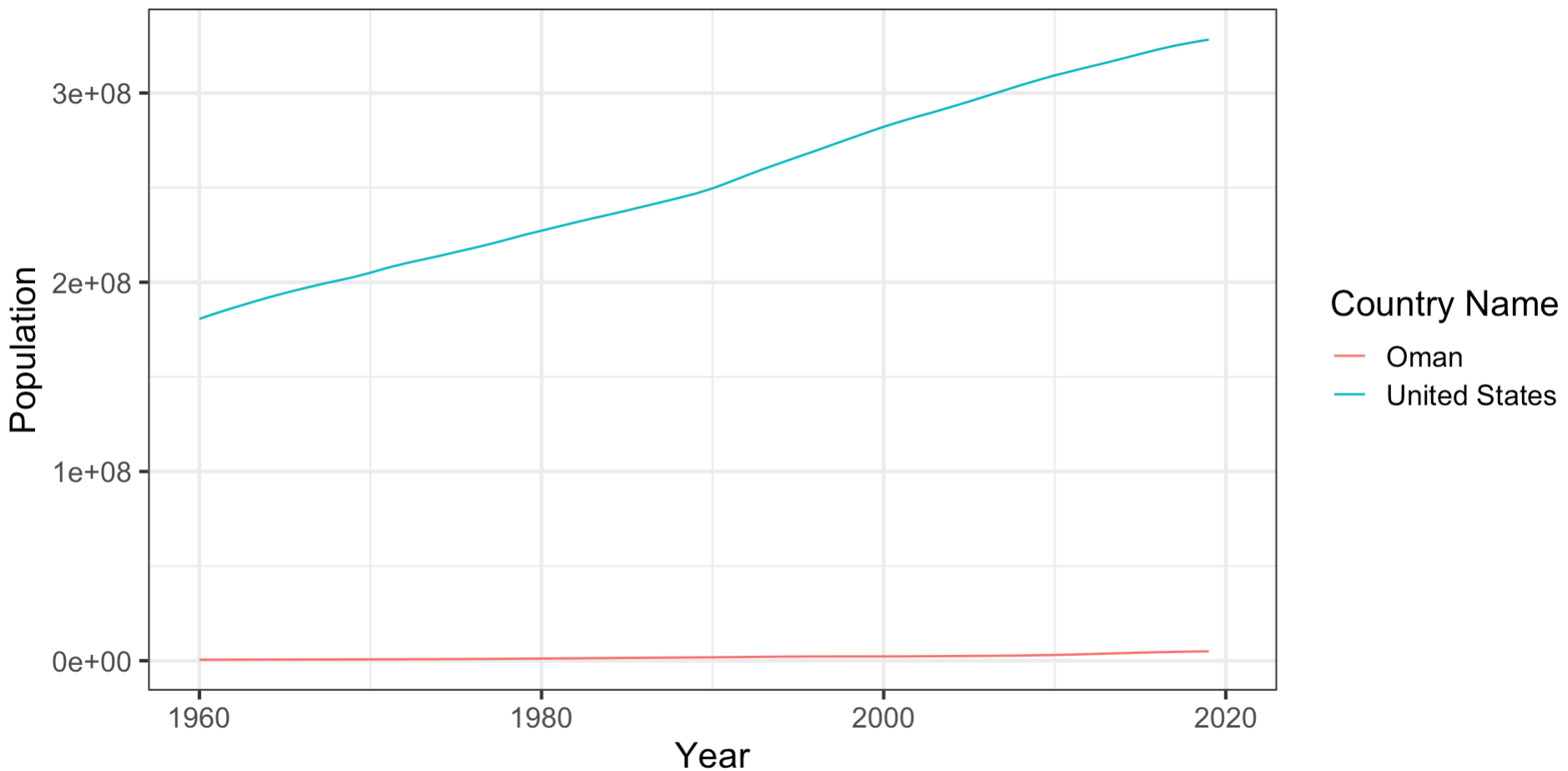
- Same plot? Different plot?
- Absolute vs relative?
- `patchwork` and `facet_wrap` as tools

Absolute vs relative levels

Absolute levels:

```
pop |>
  filter(`Country Name` %in% c("Oman", "United States")) |>
  ggplot() + geom_line(aes(x=Year, y=Population, col=`Country Name`)) +
  theme_bw(base_size=16)
```

Absolute vs relative levels

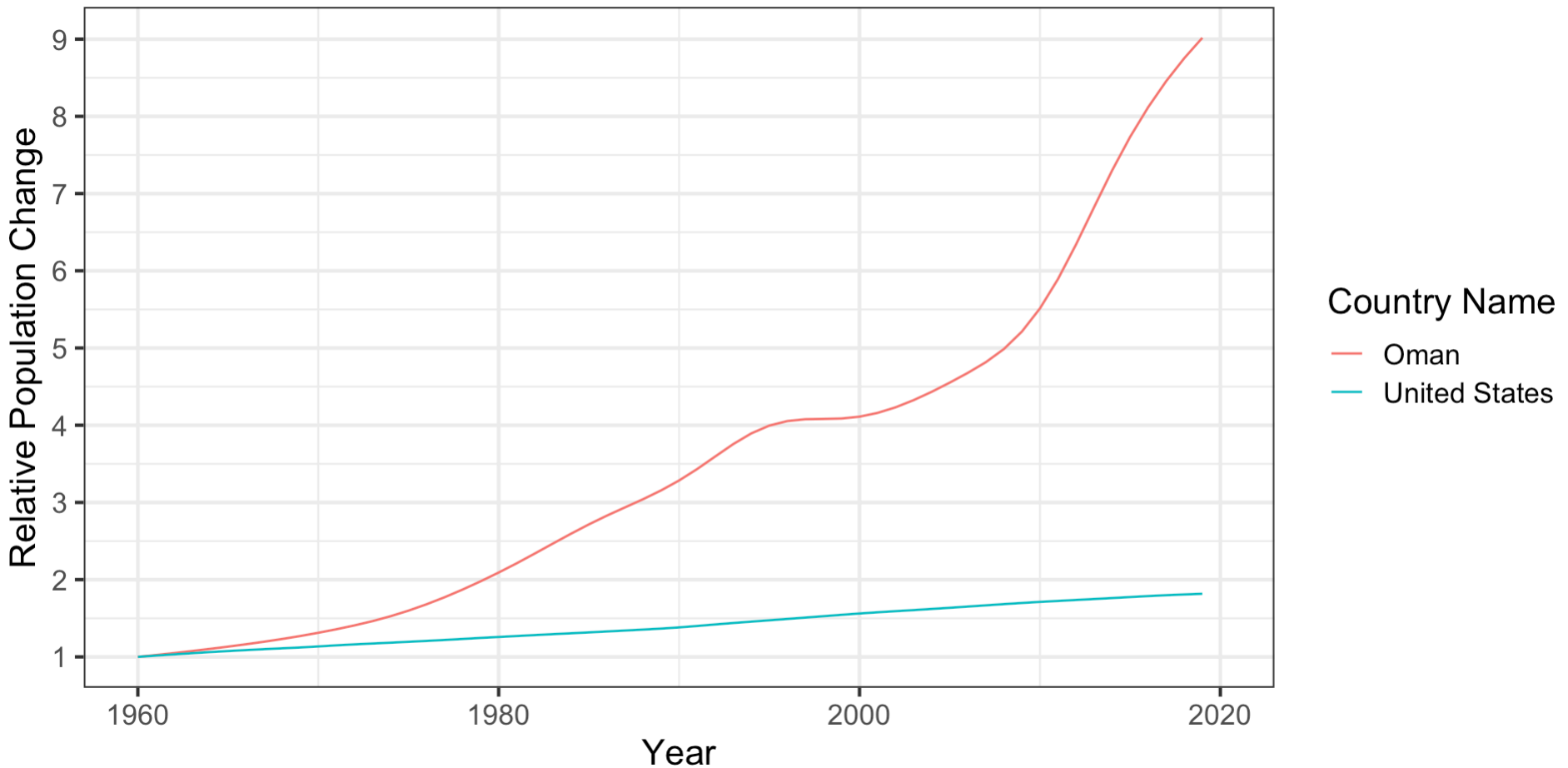


Absolute vs relative levels

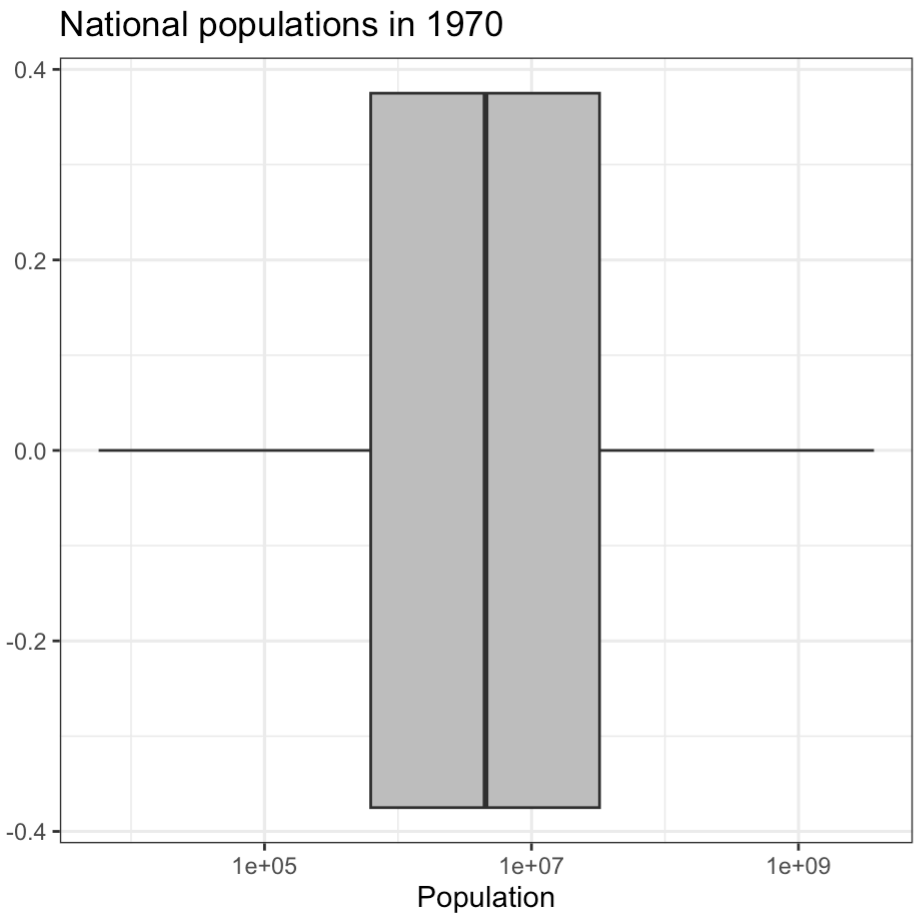
Relative levels:

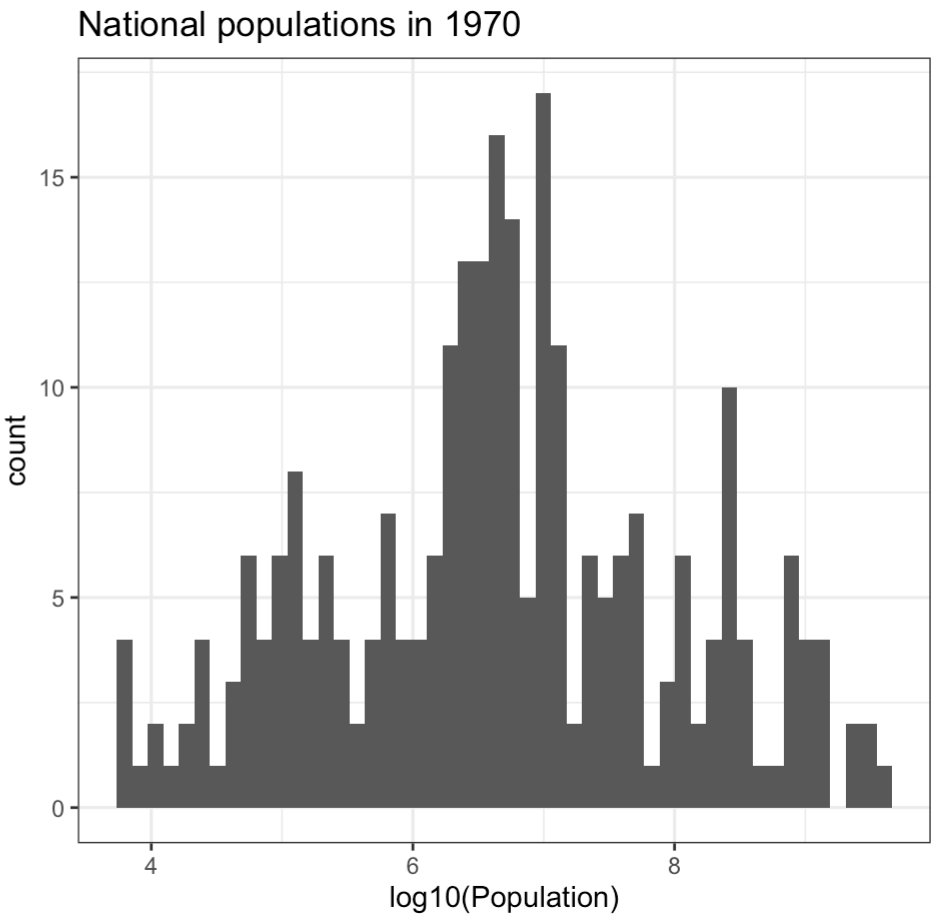
```
pop |>
  group_by(`Country Name`) |>
  mutate(`Relative Population Change` = Population / first(Population[Year=
  filter(`Country Name` %in% c("Oman", "United States")) |>
  ggplot() + geom_line(aes(x=Year, y=`Relative Population Change`, col=`Cou
  theme_bw(base_size=16) +
  scale_y_continuous(breaks=1:10)
```

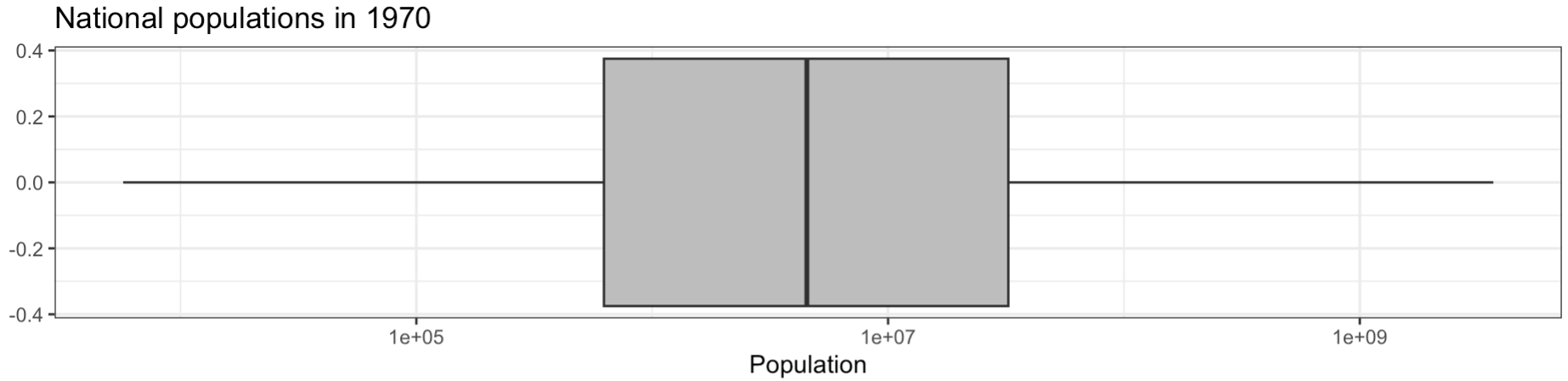
Absolute vs relative levels

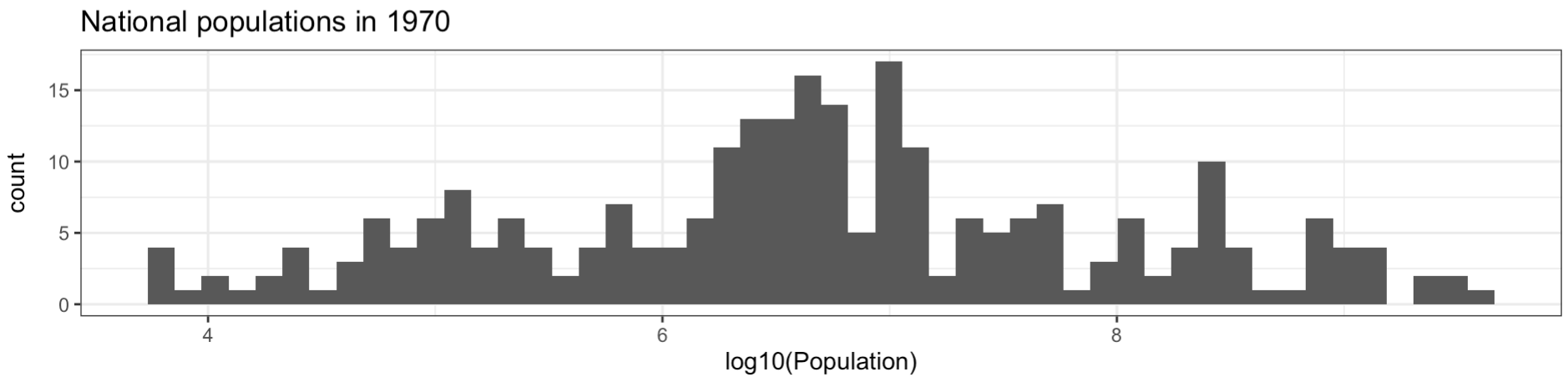


Patchwork


National populations in 1970
0.4
0.2
0.0
-0.2
-0.4
1e+05
1e+07
1e+09
Population


National populations in 1970
15
10
5
0
count
4
6
8
log10(Population)

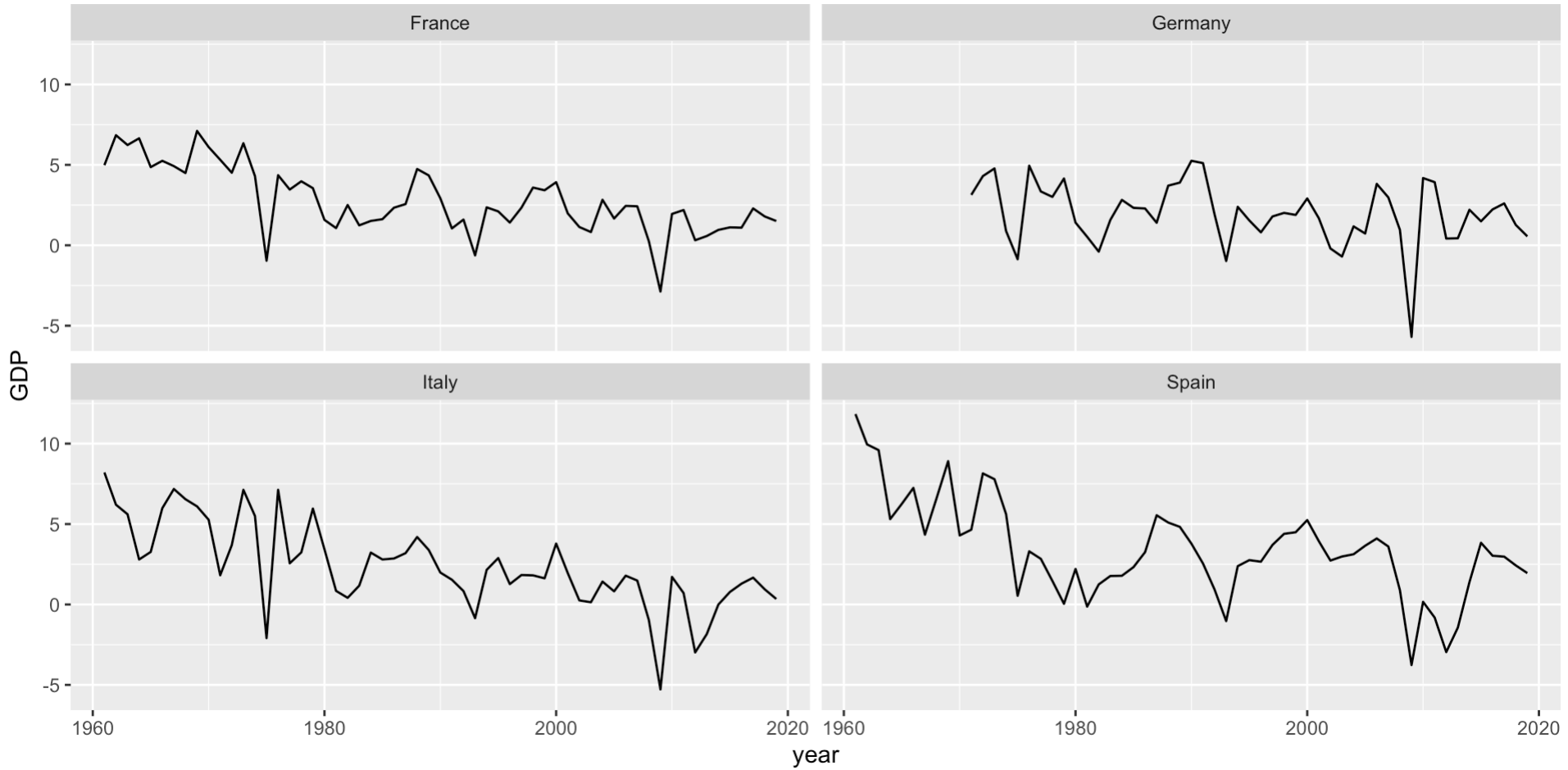

National populations in 1970
0.4
0.2
0.0
-0.2
-0.4
1e+05
1e+07
1e+09
Population


National populations in 1970
count
15
10
5
0
4
6
8
log10(Population)

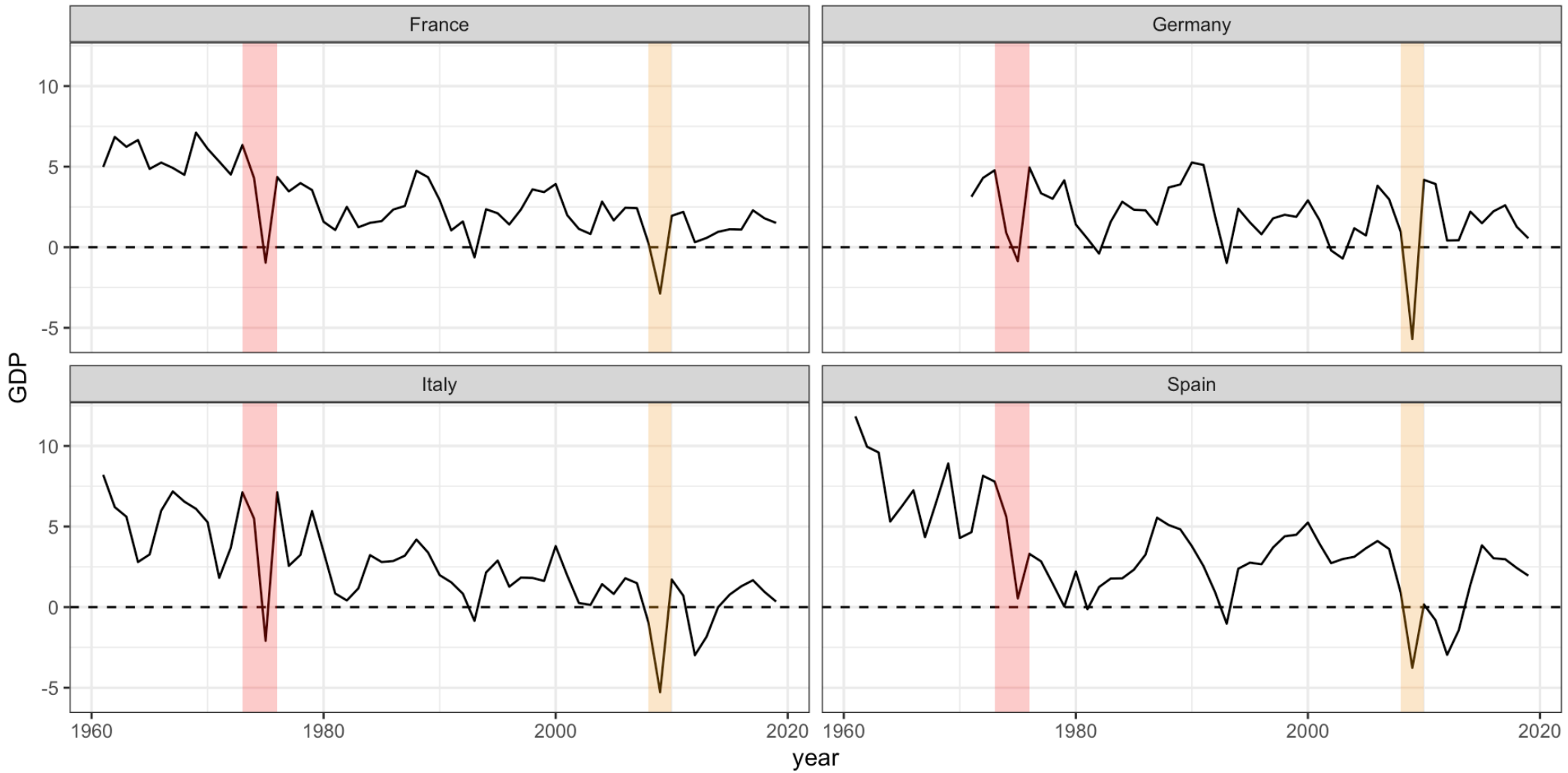
Faceting

```
gdp2 |>
mutate(year = ymd(paste0(year, "-01-01"))) |>
filter(`Country Name` %in% c("Italy", "France", "Germany", "Spain")) |>
ggplot() + geom_line(aes(x=year, y=GDP)) +
facet_wrap(~`Country Name`)
```

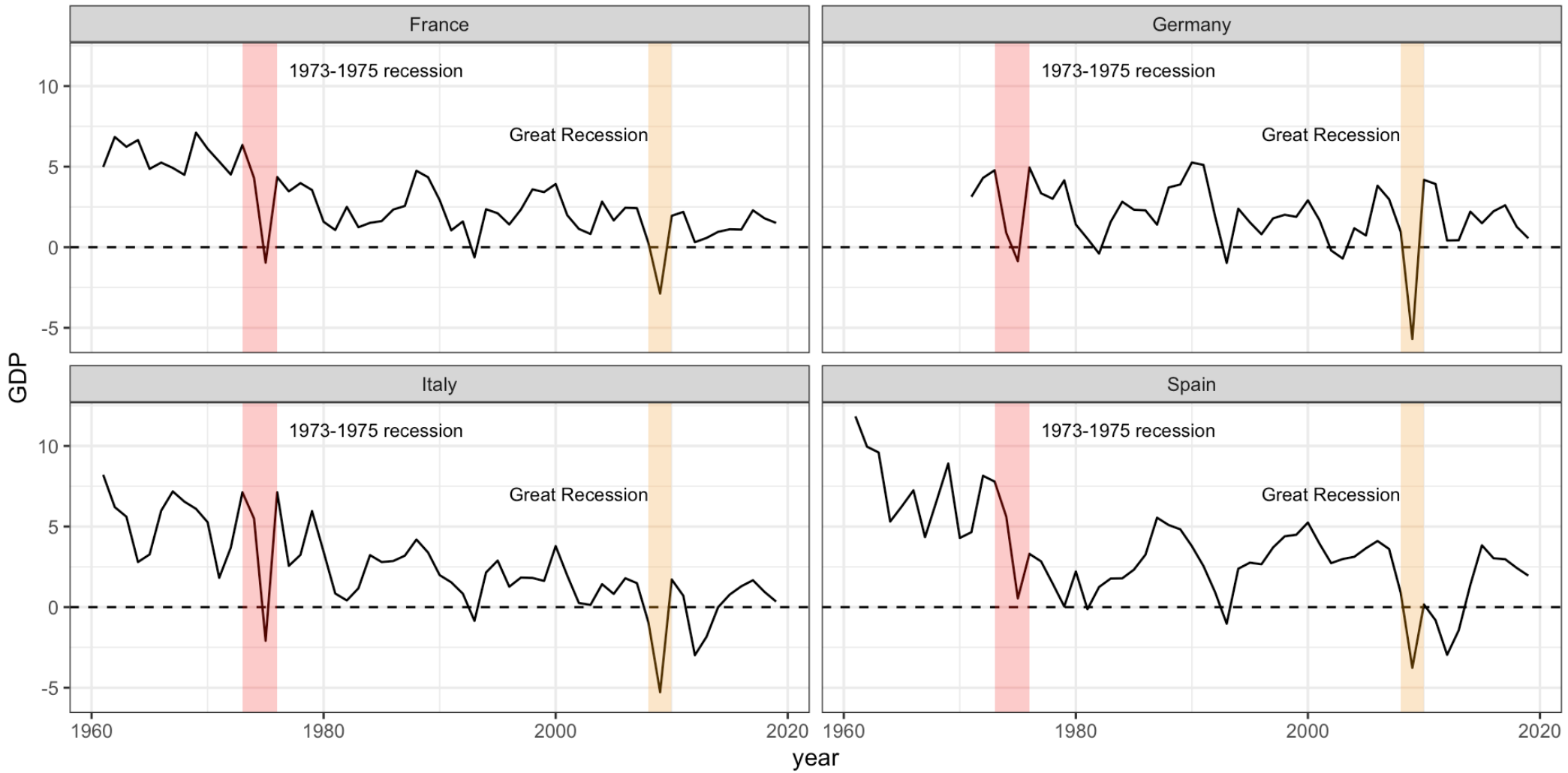
Faceting


France
Germany
Italy
Spain
GDP
10
5
0
-5
1960
1980
2000
2020
year

GDP


France
Germany
Italy
Spain
GDP
10
5
0
-5
1960
1980
2000
2020
year

GDP


France
Germany
Italy
Spain
1973-1975 recession
Great Recession
GDP
year
10
5
0
-5
1960
1980
2000
2020

Principle of Proportional Ink

> **The principle of proportional ink:** The sizes of shaded areas in a visualization need to be proportional to the data values they represent.

Redundant Coding

Summary

This has been a primer on visualization basics. We touched on the following points:

- The elements of statistical graphics are axes, geometric objects, aesthetics, and text
- Statistical graphics are constructed by mapping dataframe columns to geometric objects and aesthetics
  - Allows for display of complex information
  - Specified in Altair as marks and encodings on a chart
- The basic statistical graphics are histograms, boxplots, scatterplots, line plots, and barplots.
  - Think of these as building blocks you can use to develop more sophisticated visualizations
- Effective data visualizations are novel, informative, efficient/economical, and pleasing.

Summary

- Make sure text and numbers are large enough!
- Avoid putting too much in a single plot
- Label axes and include units of measurements
- Scale axes appropriately
- Add a legend or caption to the plot explaining symbols or colors, etc
- Each plot should be self-explanatory without having to refer to the text
- Use ggplot2 + companion libraries