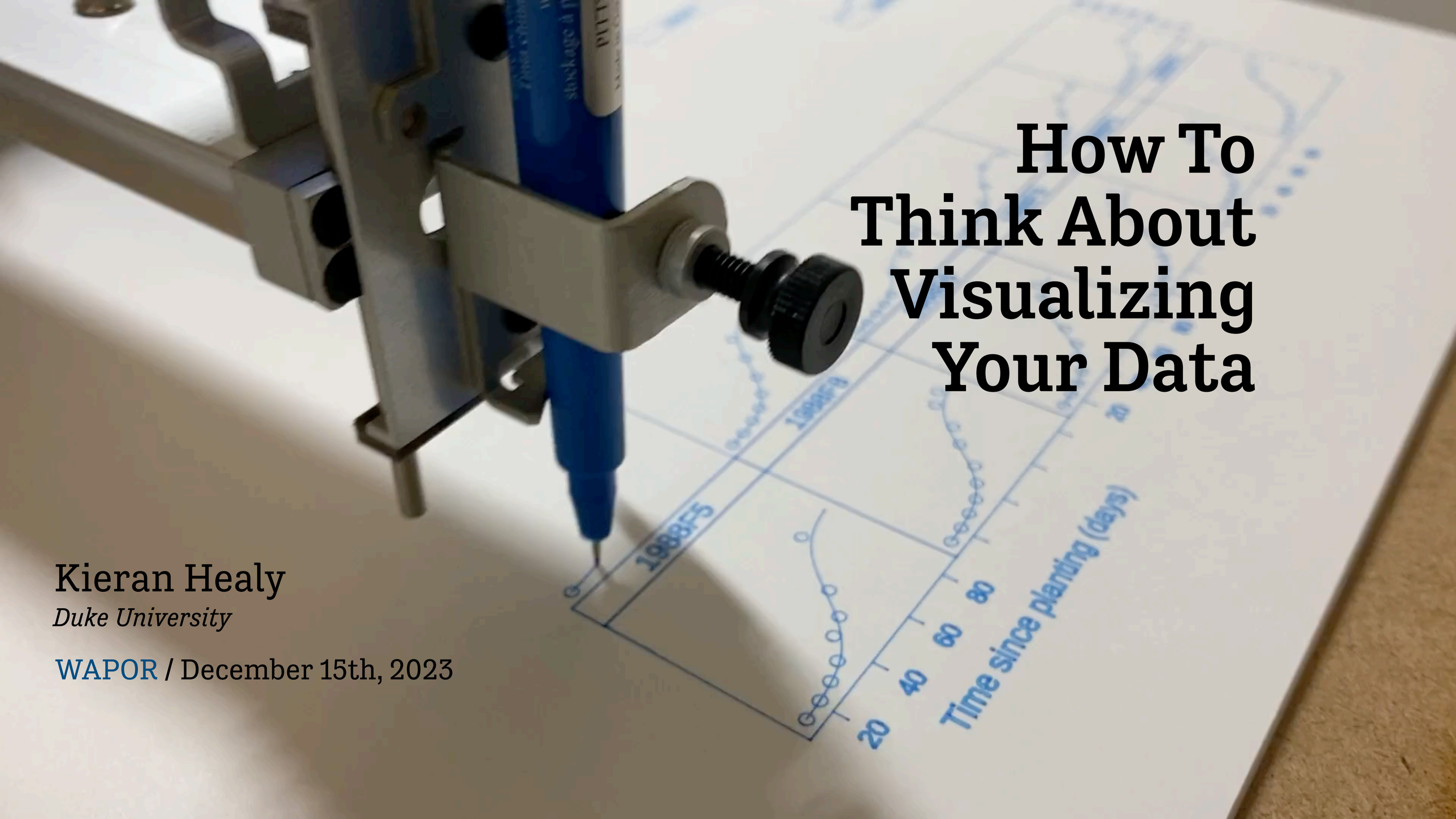


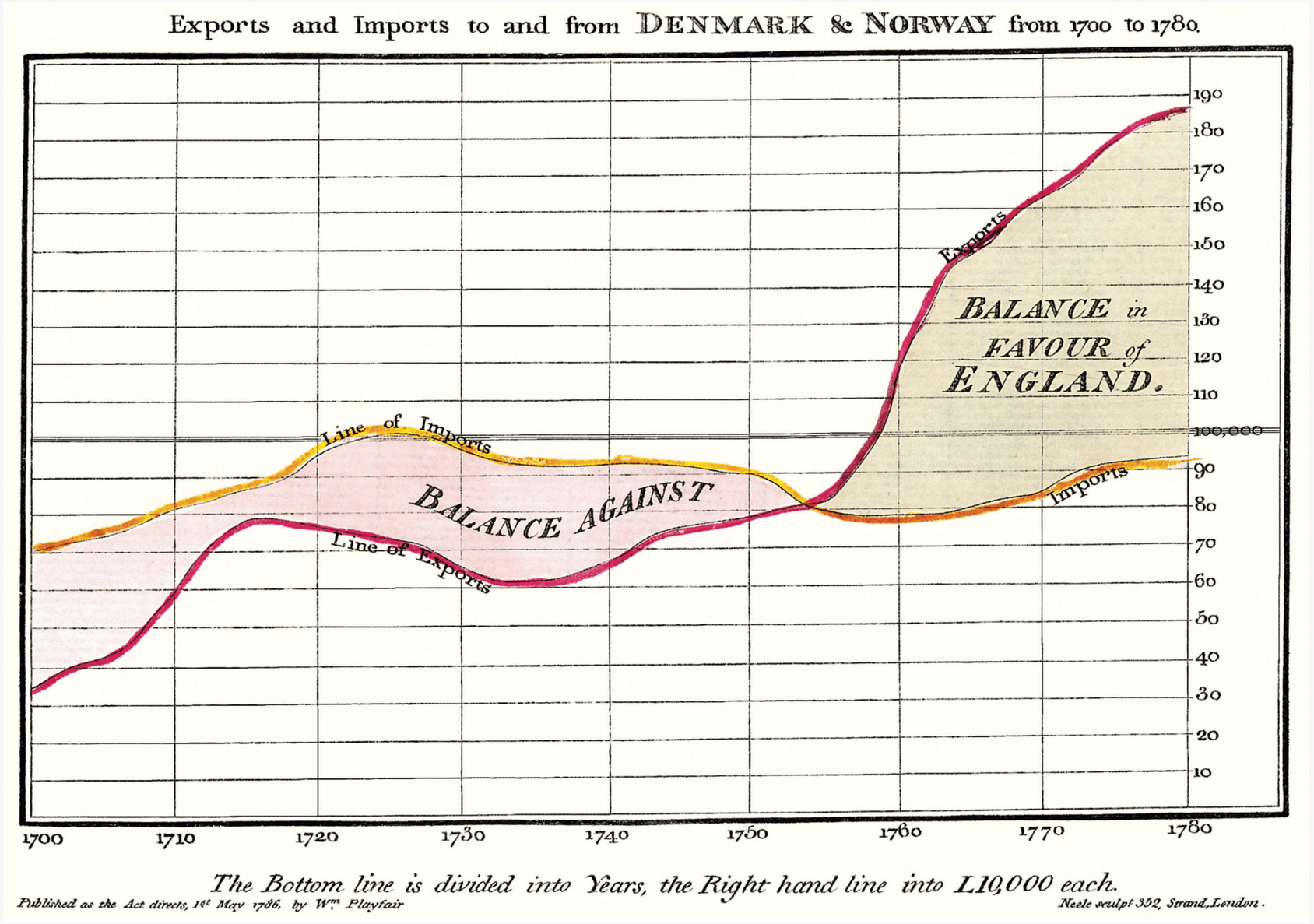
How To Think About Visualizing Your Data

Kieran Healy
Duke University

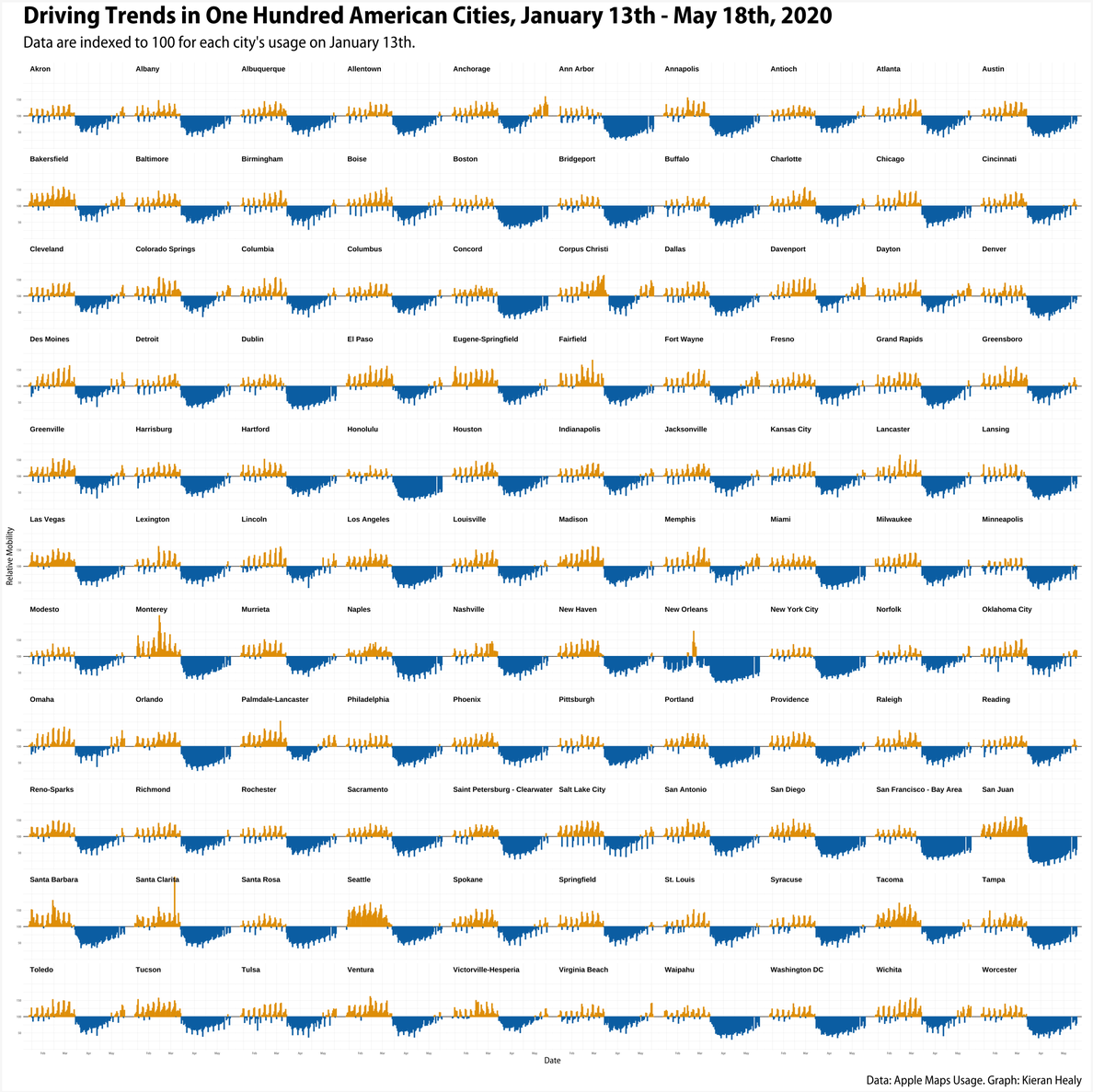
WAPOR / December 15th, 2023

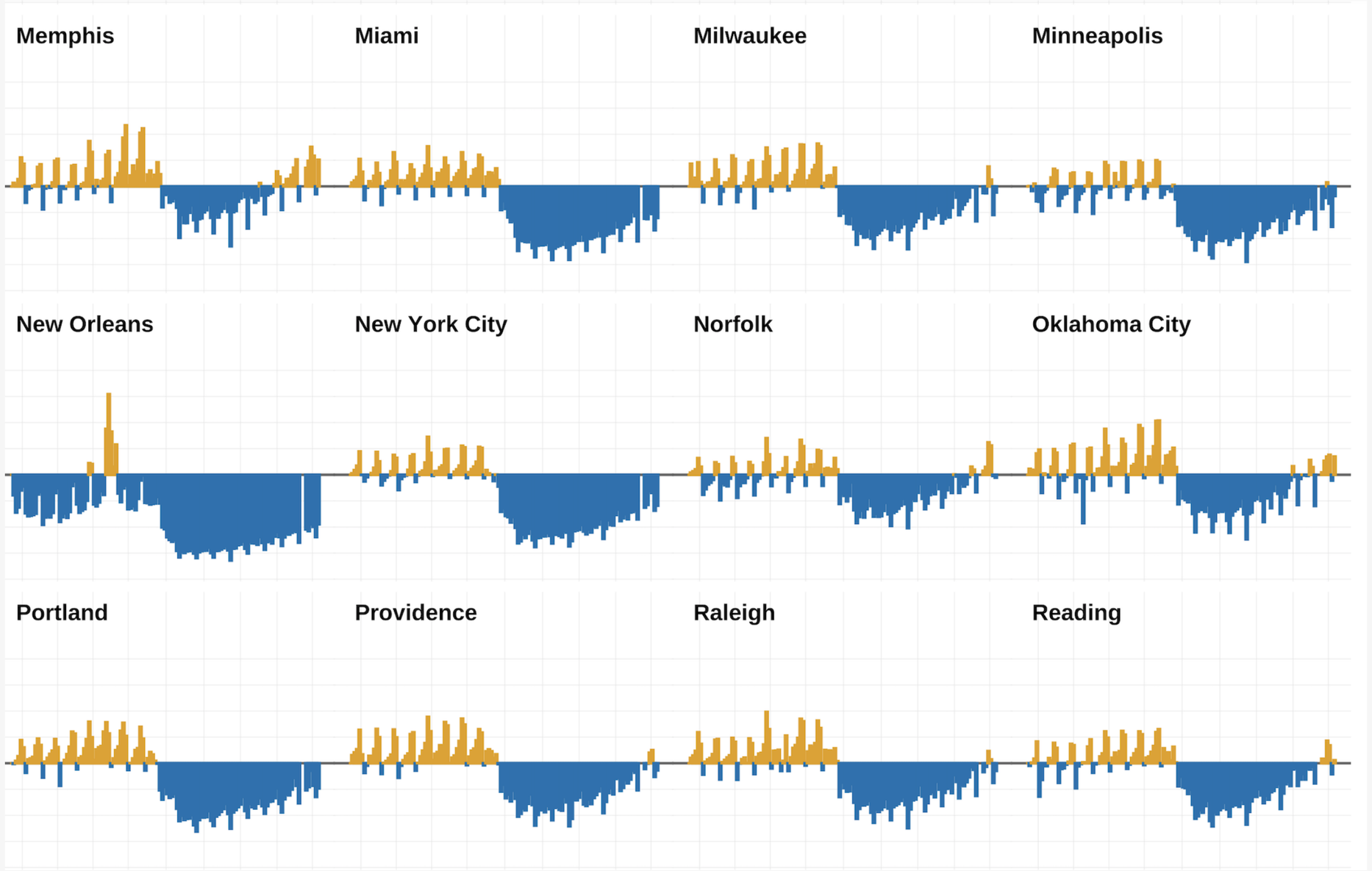
The Problem

YOU SHOULD LOOK AT YOUR DATA



William Playfair


Driving Trends in One Hundred American Cities, January 13th - May 18th, 2020
Data are indexed to 100 for each city's usage on January 13th.
Akron
Albany
Albuquerque
Allentown
Anchorage
Ann Arbor
Annapolis
Antioch
Atlanta
Austin
Bakersfield
Baltimore
Birmingham
Boise
Boston
Bridgeport
Buffalo
Charlotte
Chicago
Cincinnati
Cleveland
Colorado Springs
Columbia
Columbus
Concord
Corpus Christi
Dallas
Davenport
Dayton
Denver
Des Moines
Detroit
Dublin
El Paso
Eugene-Springfield
Fairfield
Fort Wayne
Fresno
Grand Rapids
Greensboro
Greenville
Harrisburg
Hartford
Honolulu
Houston
Indianapolis
Jacksonville
Kansas City
Lancaster
Lansing
Las Vegas
Lexington
Lincoln
Los Angeles
Louisville
Madison
Memphis
Miami
Milwaukee
Minneapolis
Modesto
Monterey
Murrieta
Naples
Nashville
New Haven
New Orleans
New York City
Norfolk
Oklahoma City
Omaha
Orlando
Palmdale-Lancaster
Philadelphia
Phoenix
Pittsburgh
Portland
Providence
Raleigh
Reading
Reno-Sparks
Richmond
Rochester
Sacramento
Saint Petersburg - Clearwater
Salt Lake City
San Antonio
San Diego
San Francisco - Bay Area
San Juan
Santa Barbara
Santa Clarita
Santa Rosa
Seattle
Spokane
Springfield
St. Louis
Syracuse
Tacoma
Tampa
Toledo
Tucson
Tulsa
Ventura
Victorville-Hesperia
Virginia Beach
Waipahu
Washington DC
Wichita
Worcester
Relative Mobility
Feb
Mar
Apr
May
Date
Data: Apple Maps Usage. Graph: Kieran Healy

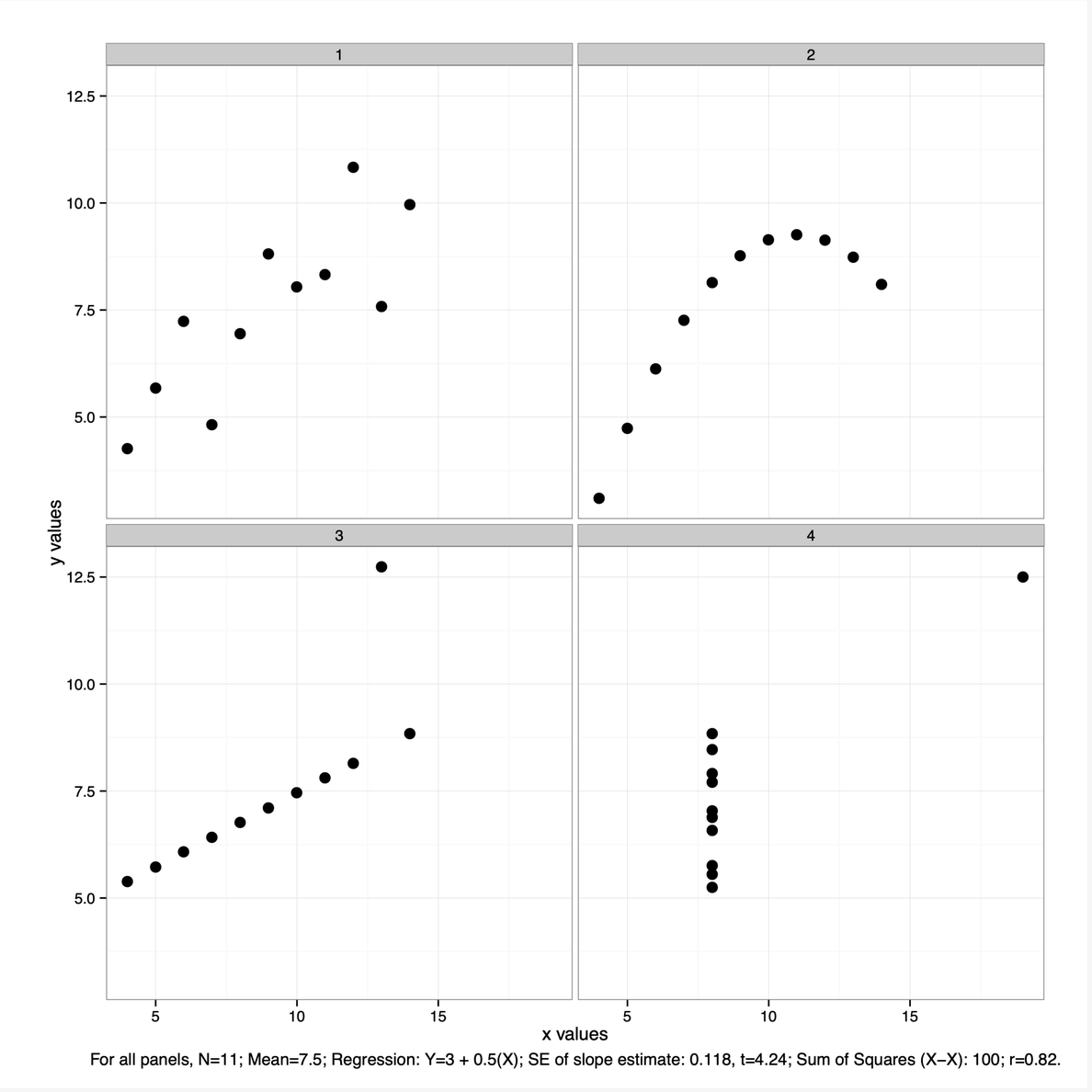

Memphis
Miami
Milwaukee
Minneapolis
New Orleans
New York City
Norfolk
Oklahoma City
Portland
Providence
Raleigh
Reading

Zoomed in

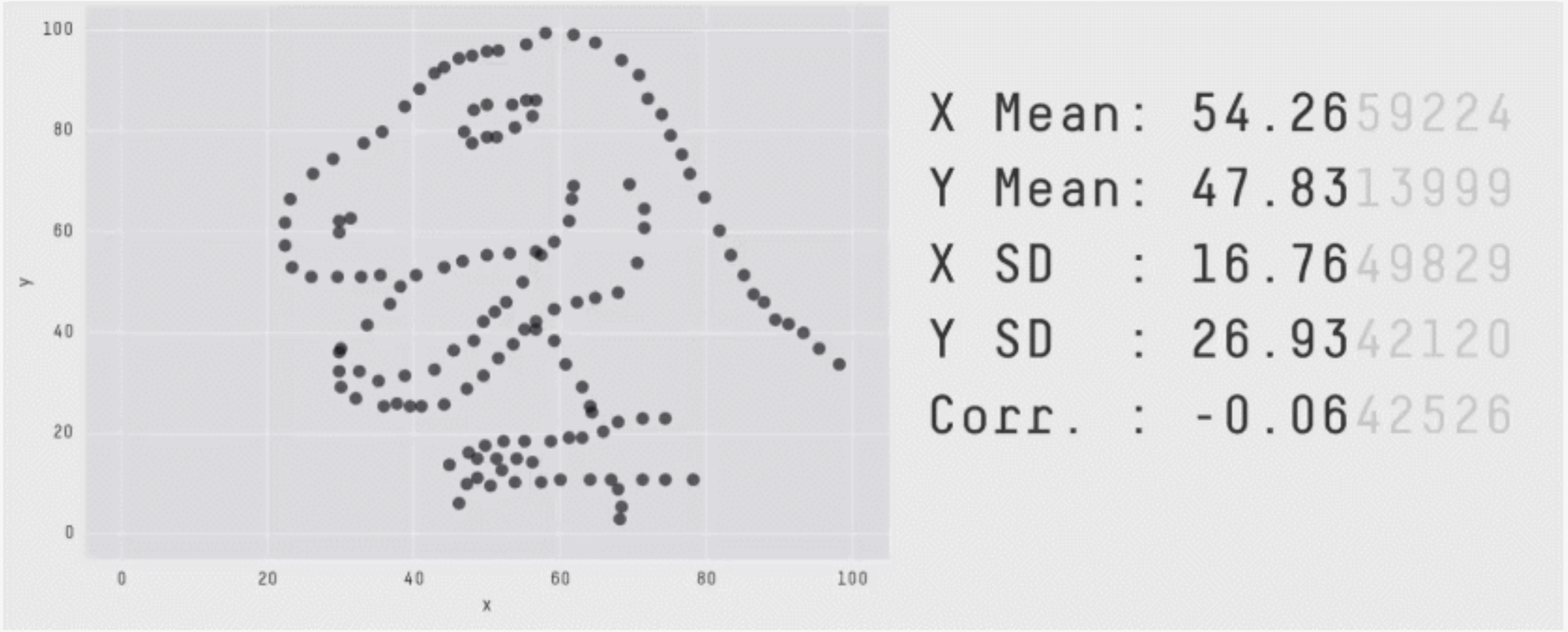
SEEING IS NOT AS SIMPLE AS IT LOOKS

Two circles; no cheating.

Why this stuff matters



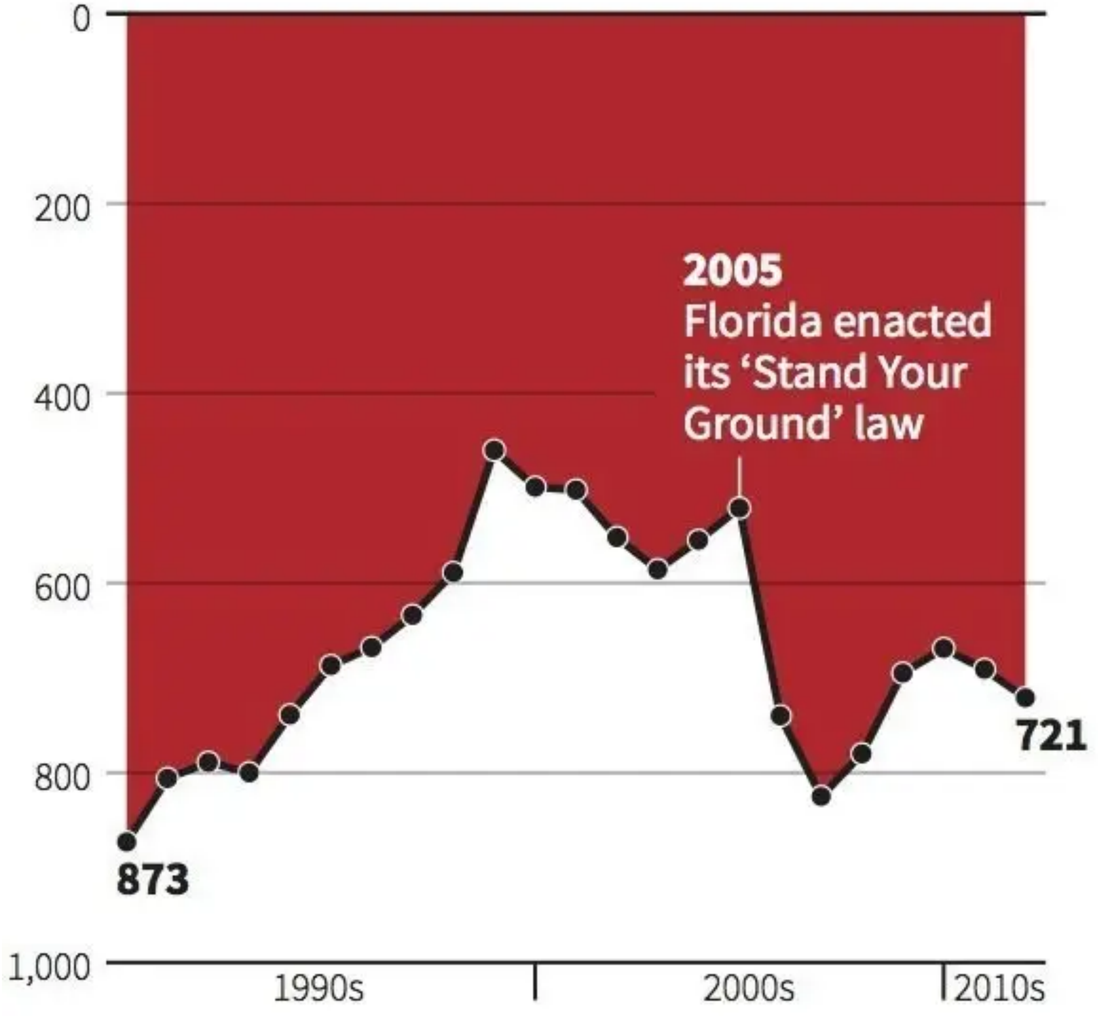
Anscombe’s Quartet



Cairo; Matejka & Fitzmaurice

Gun deaths in Florida

Number of murders committed using firearms

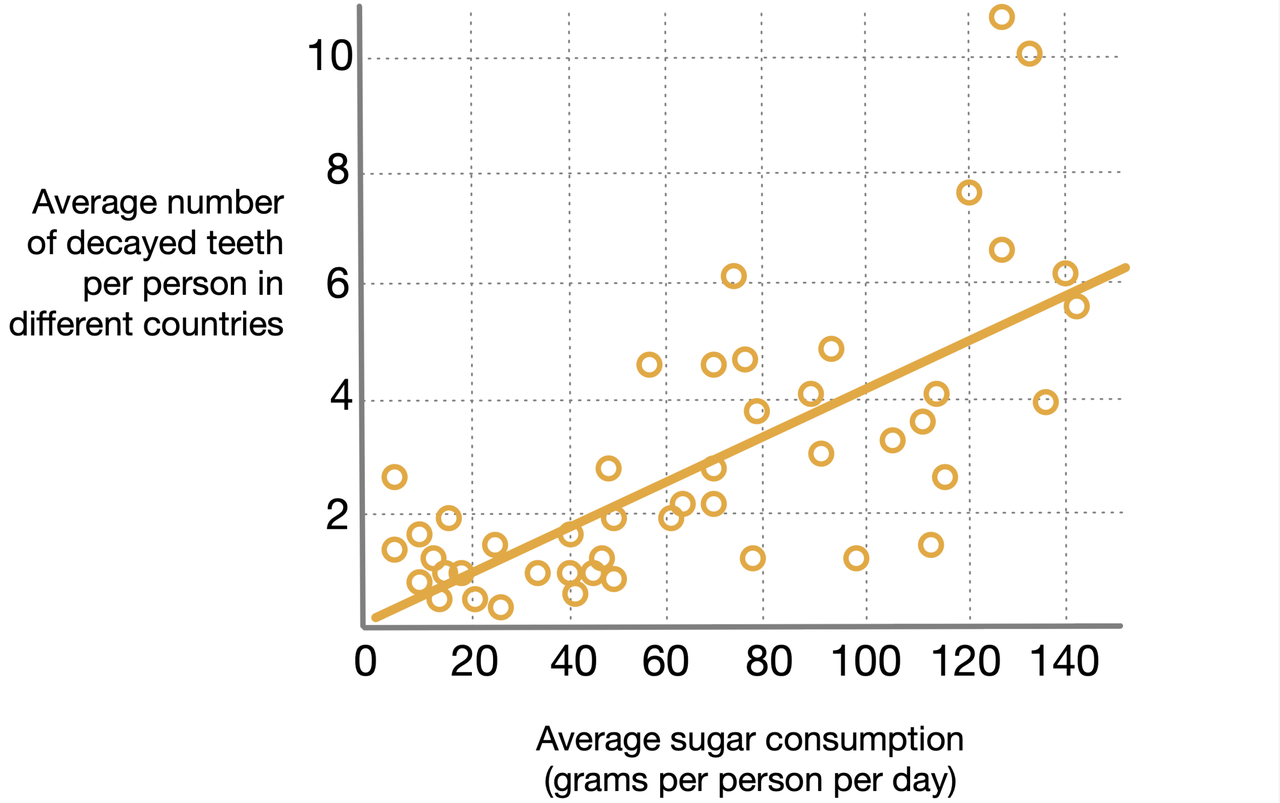


Source: Florida Department of Law Enforcement

C. Chan 16/02/2014

REUTERS

Which of the following statements best describes the data in the graph below?



Pew Research

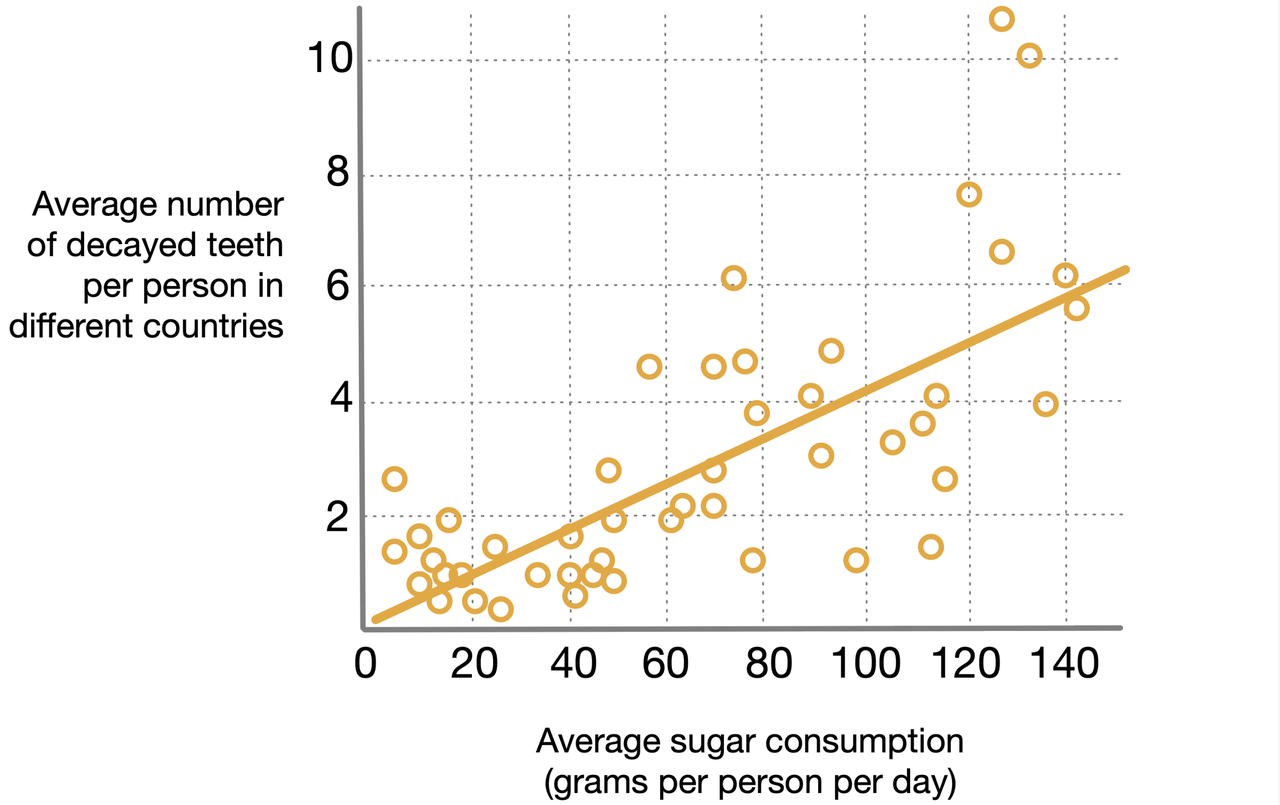
A. In recent years, the rate of cavities has increased in many countries

B. In some countries, people brush their teeth more frequently than in other countries

C. The more sugar people eat, the more likely they are to get cavities

D. In recent years, the consumption of sugar has increased in many countries

Which of the following statements best describes the data in the graph below?



Pew Research

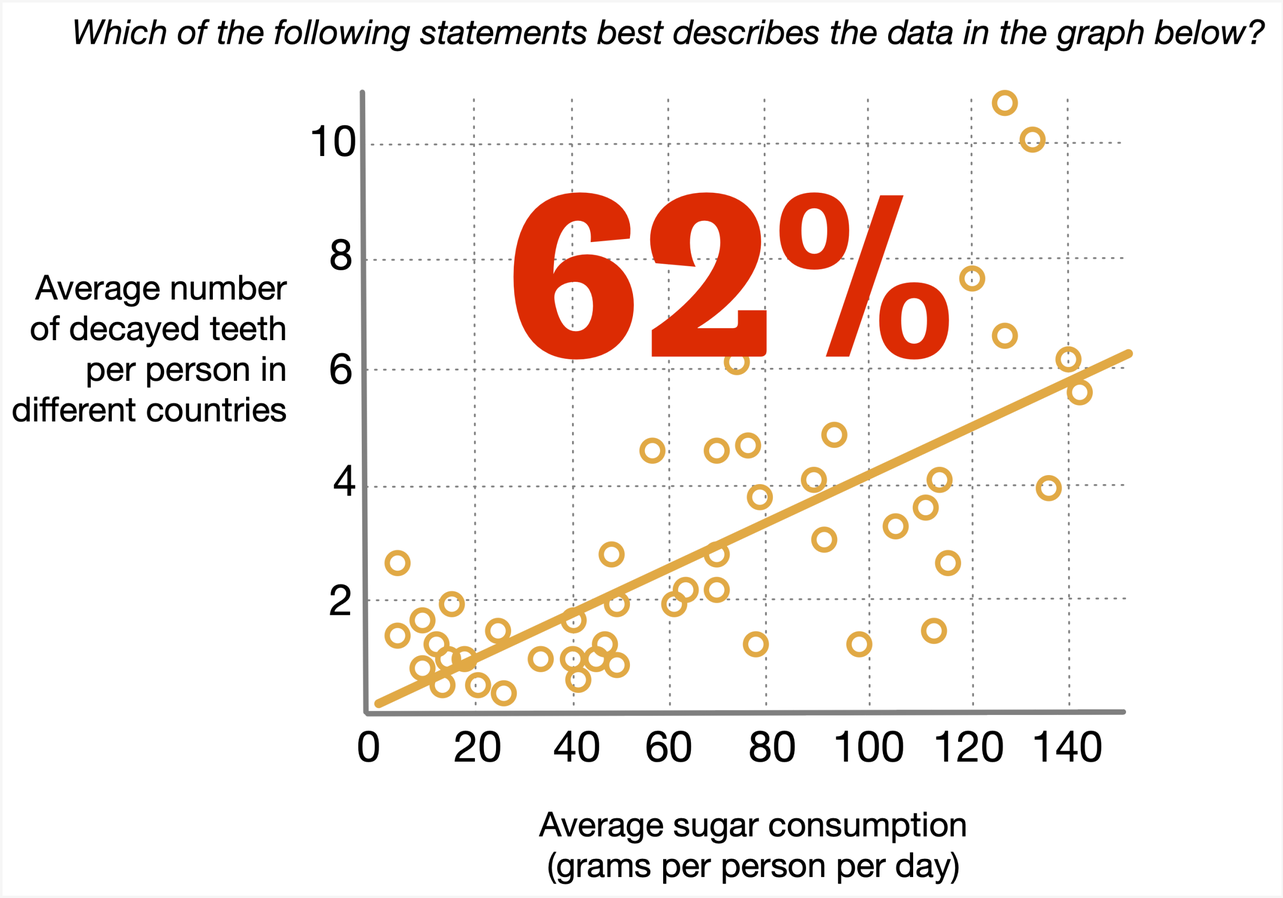
A. In recent years, the rate of cavities has increased in many countries

B. In some countries, people brush their teeth more frequently than in other countries

C. **The more sugar people eat, the more likely they are to get cavities**

D. In recent years, the consumption of sugar has increased in many countries

Which of the following statements best describes the data in the graph below?



Pew Research

A. In recent years, the rate of cavities has increased in many countries

B. In some countries, people brush their teeth more frequently than in other countries

C. **The more sugar people eat, the more likely they are to get cavities**

D. In recent years, the consumption of sugar has increased in many countries

Some charts are better than others

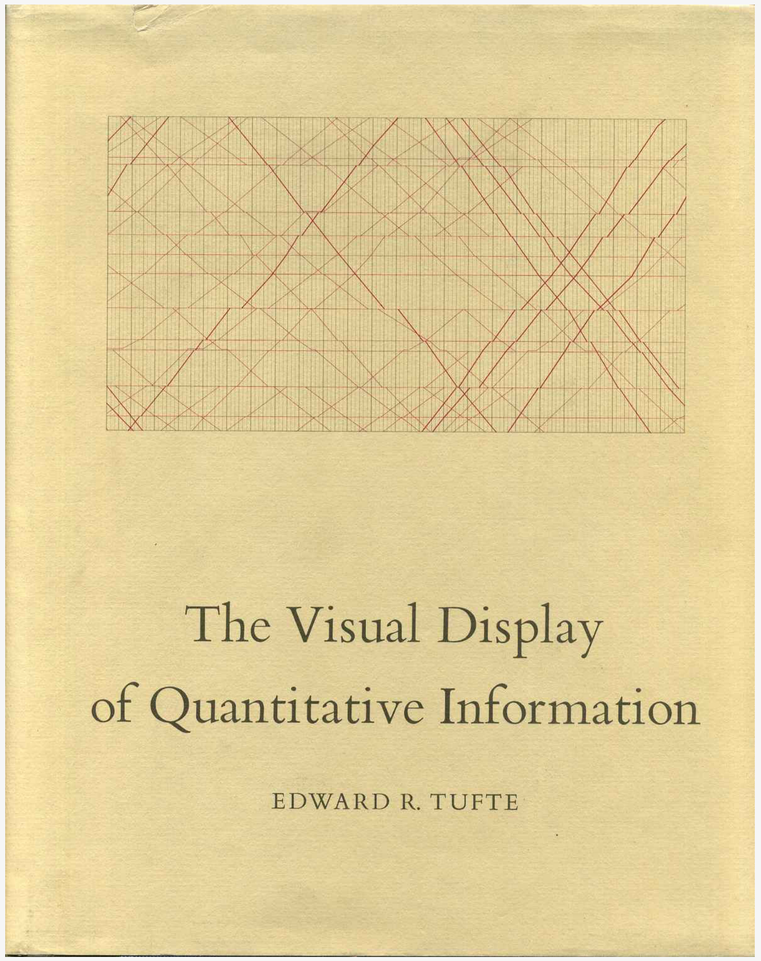
BAD TASTE

BAD DATA

BAD PERCEPTION

1. Bad Taste

Simplify, simplify?

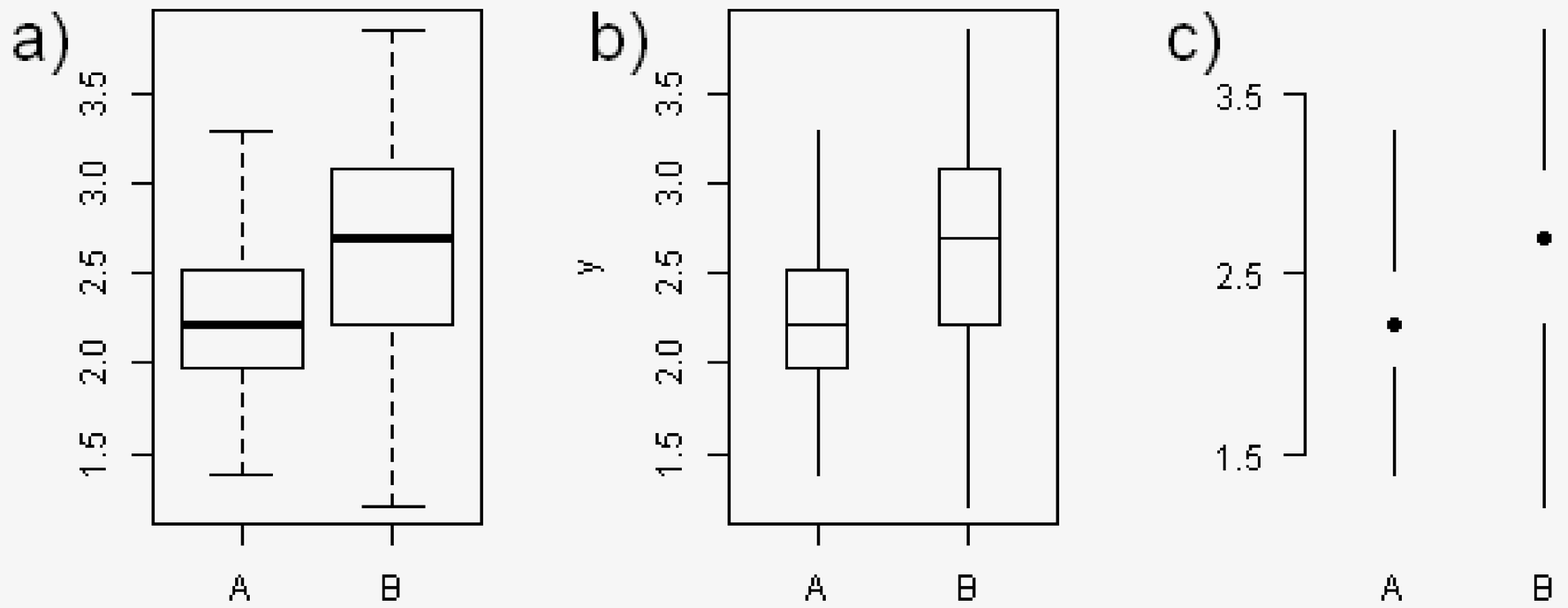


Edward Tufte

“Chart junk”

Edward Tufte

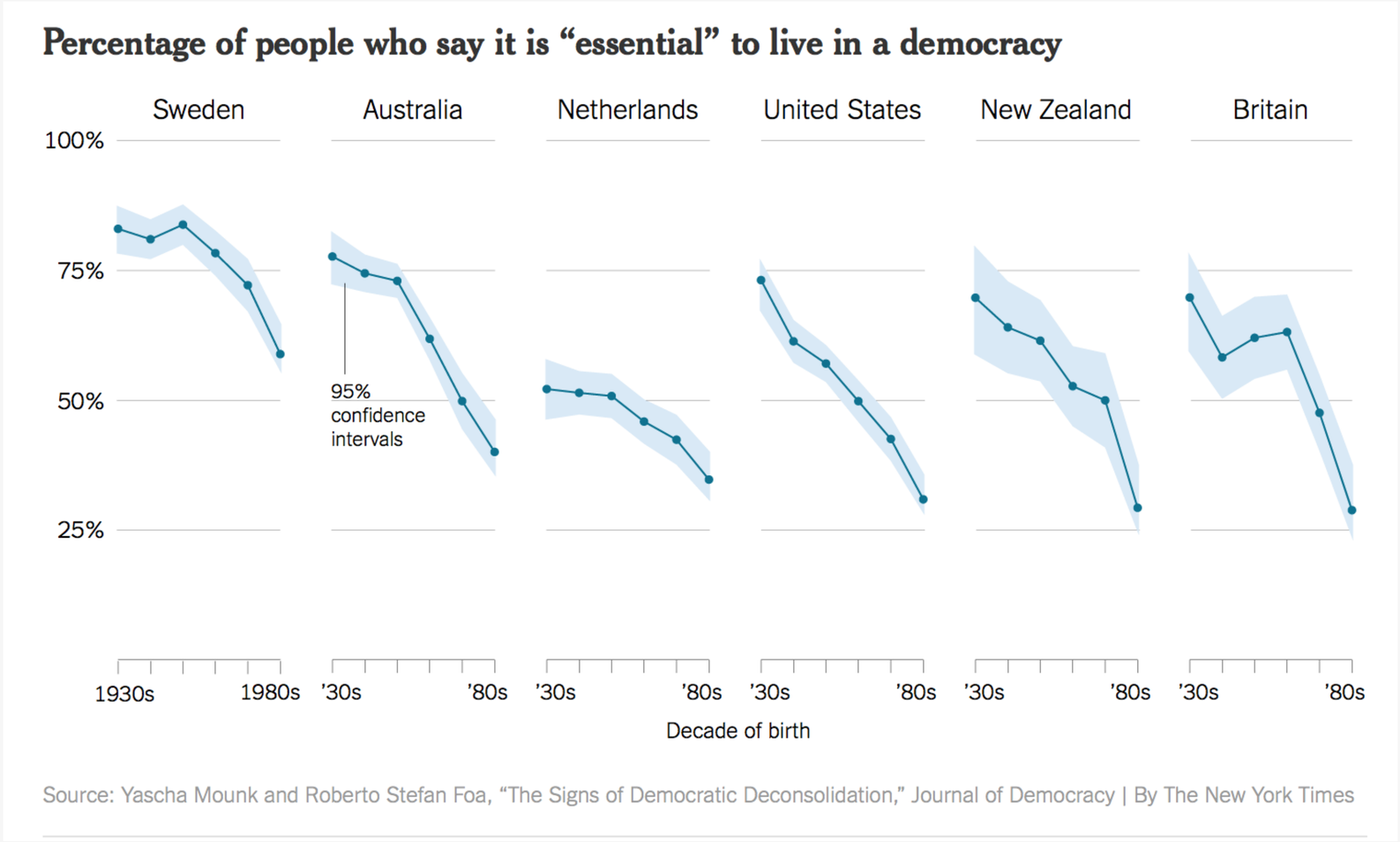
… And its limits



Three kinds of boxplot

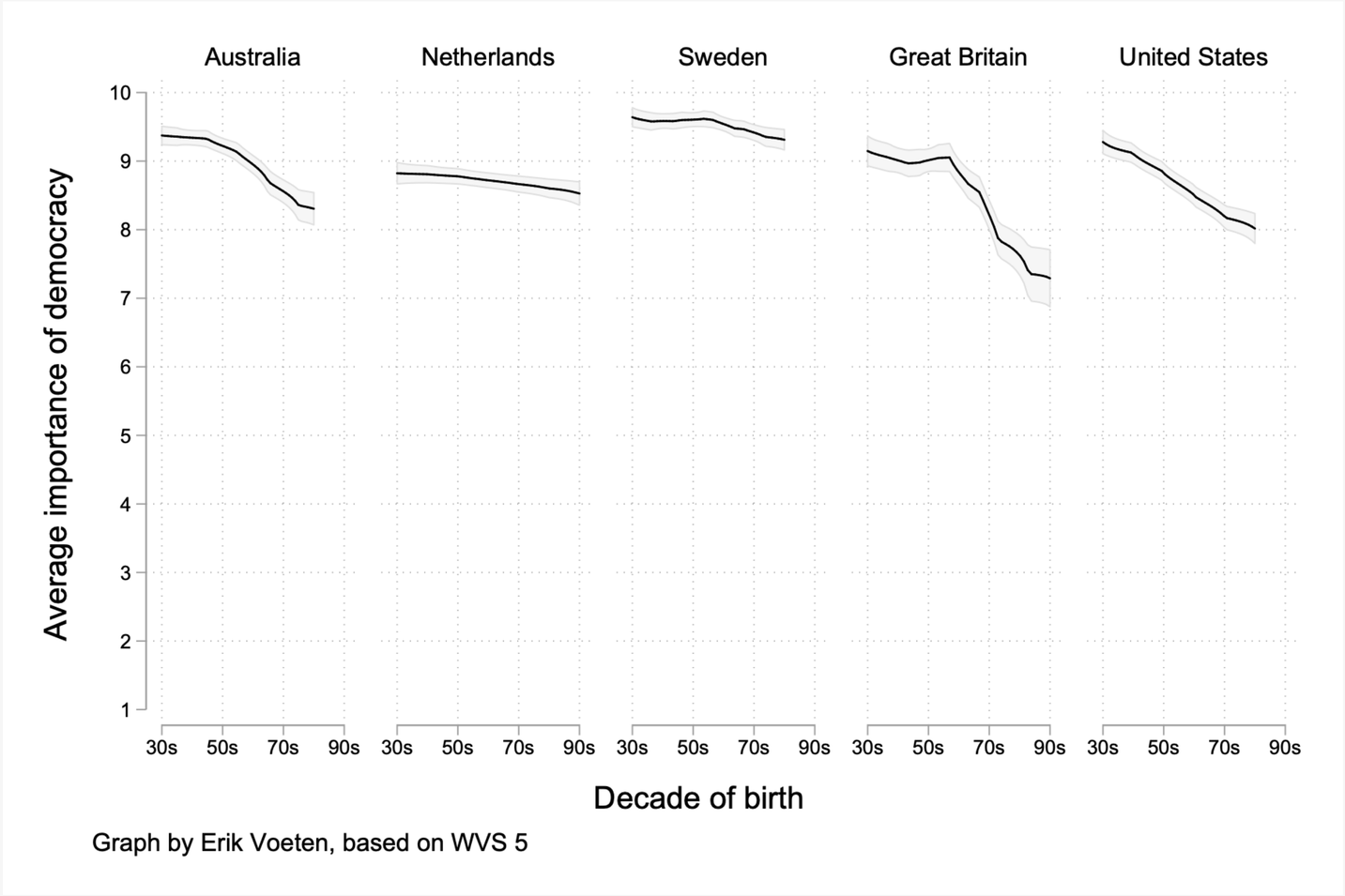
2. Bad Data

Junk-free junk charts



New York Times

Junk-free junk charts



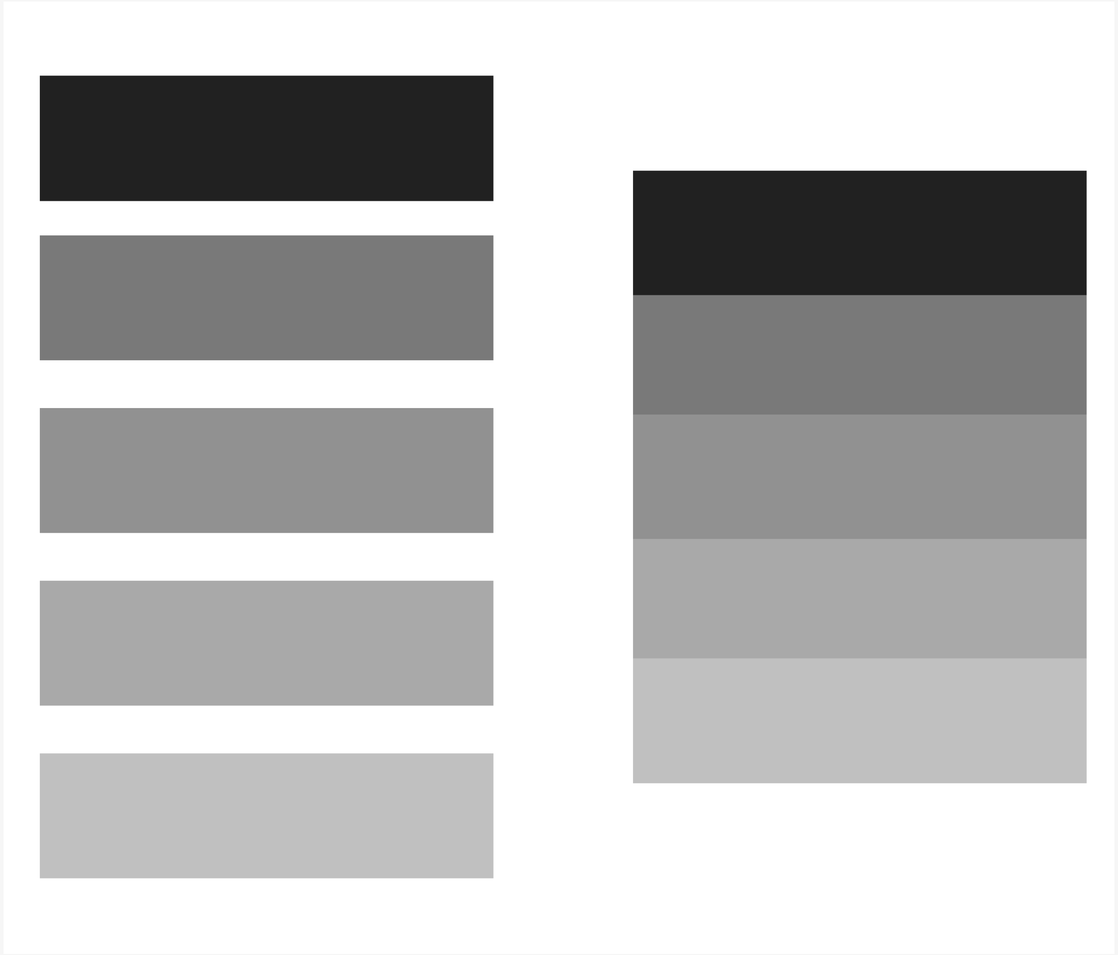
Erik Voeten

3. Bad Perception

Contrasts and Edges

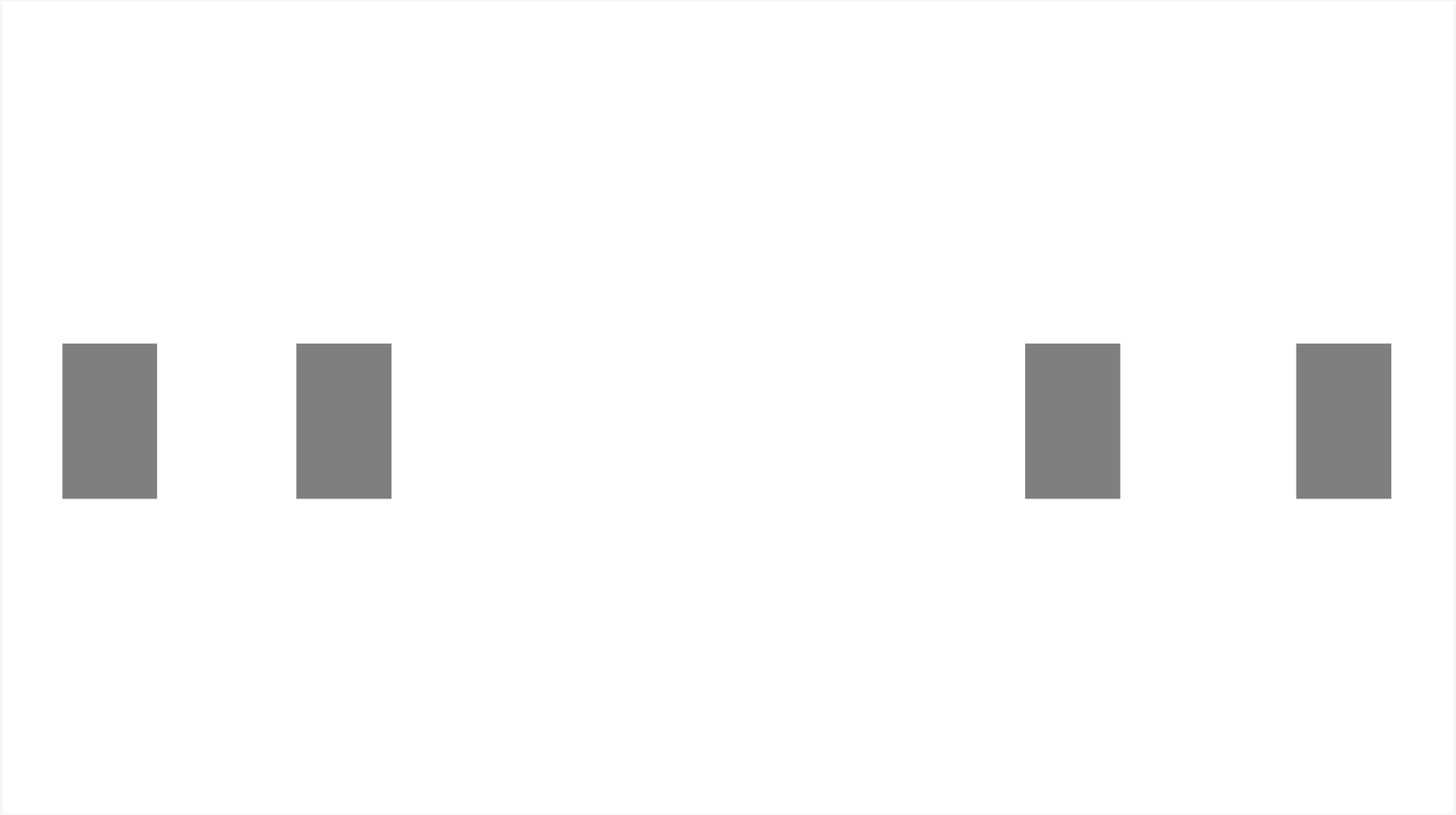
Mach Bands

Contrasts and Edges



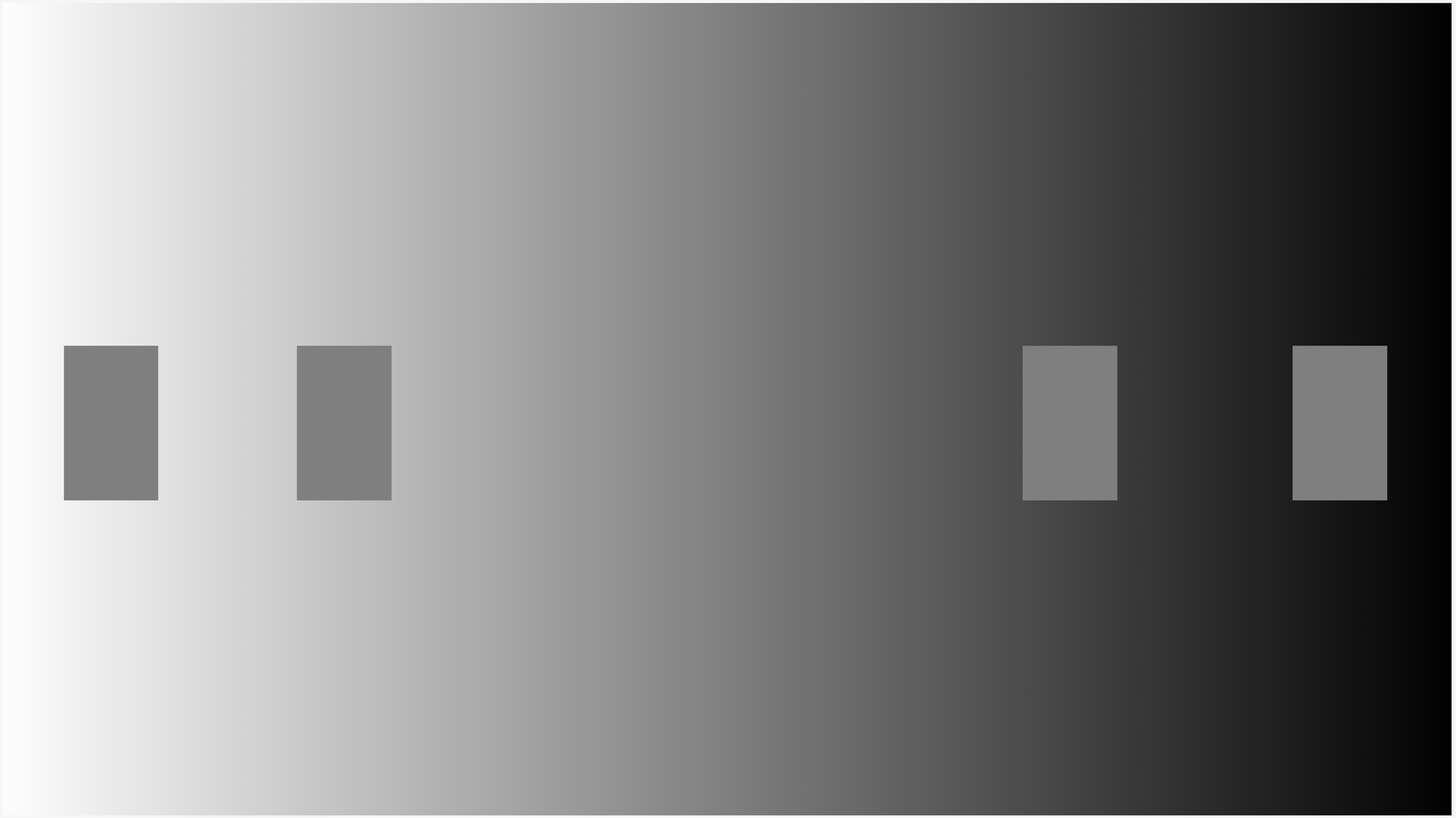
Mach Bands

Contrasts and Edges



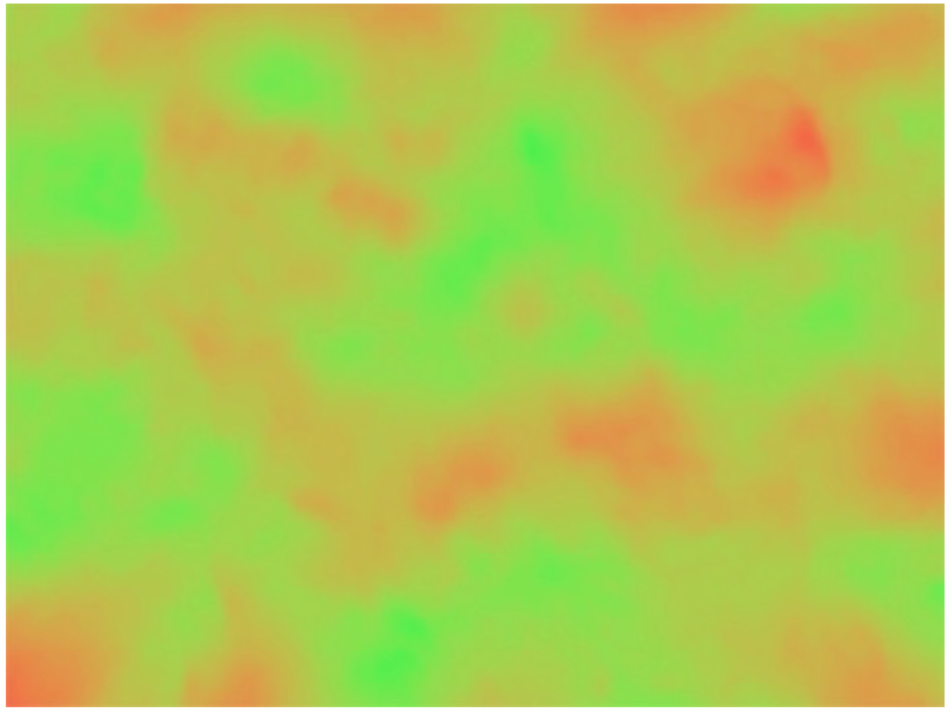
Four gray rectangles

Contrasts and Edges



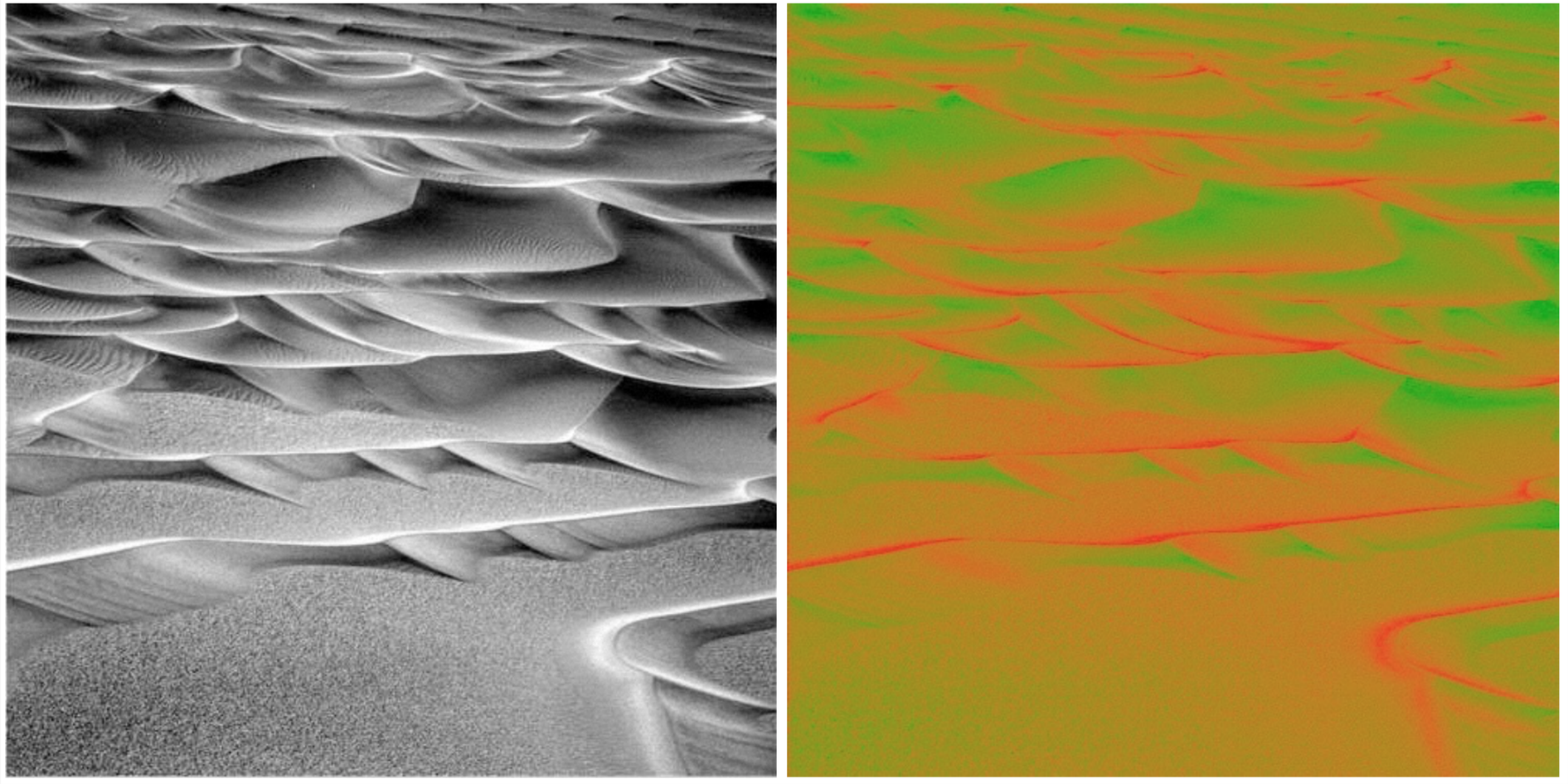
With a gradient background

Contrast and Color



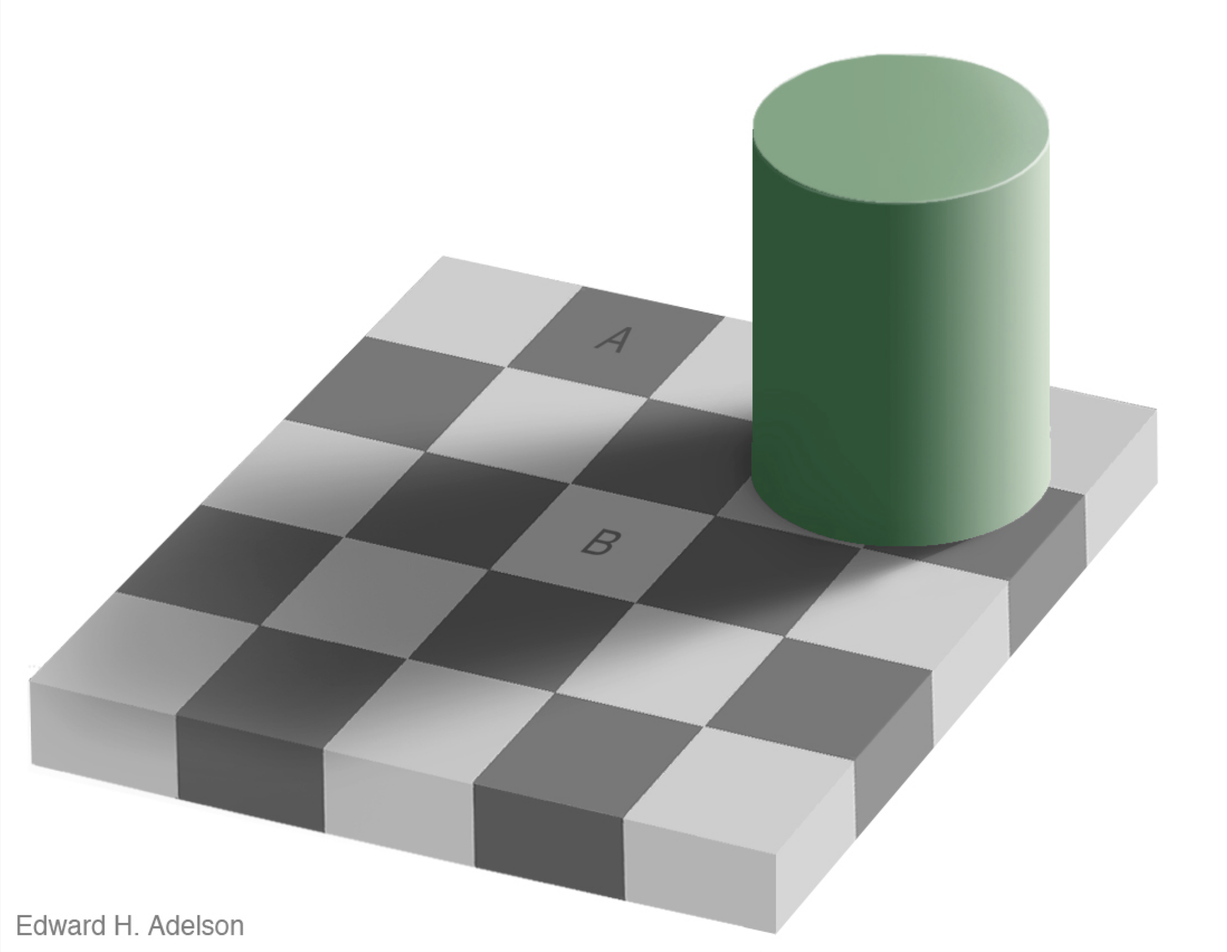
Colin Ware

Contrast and Color



Colin Ware

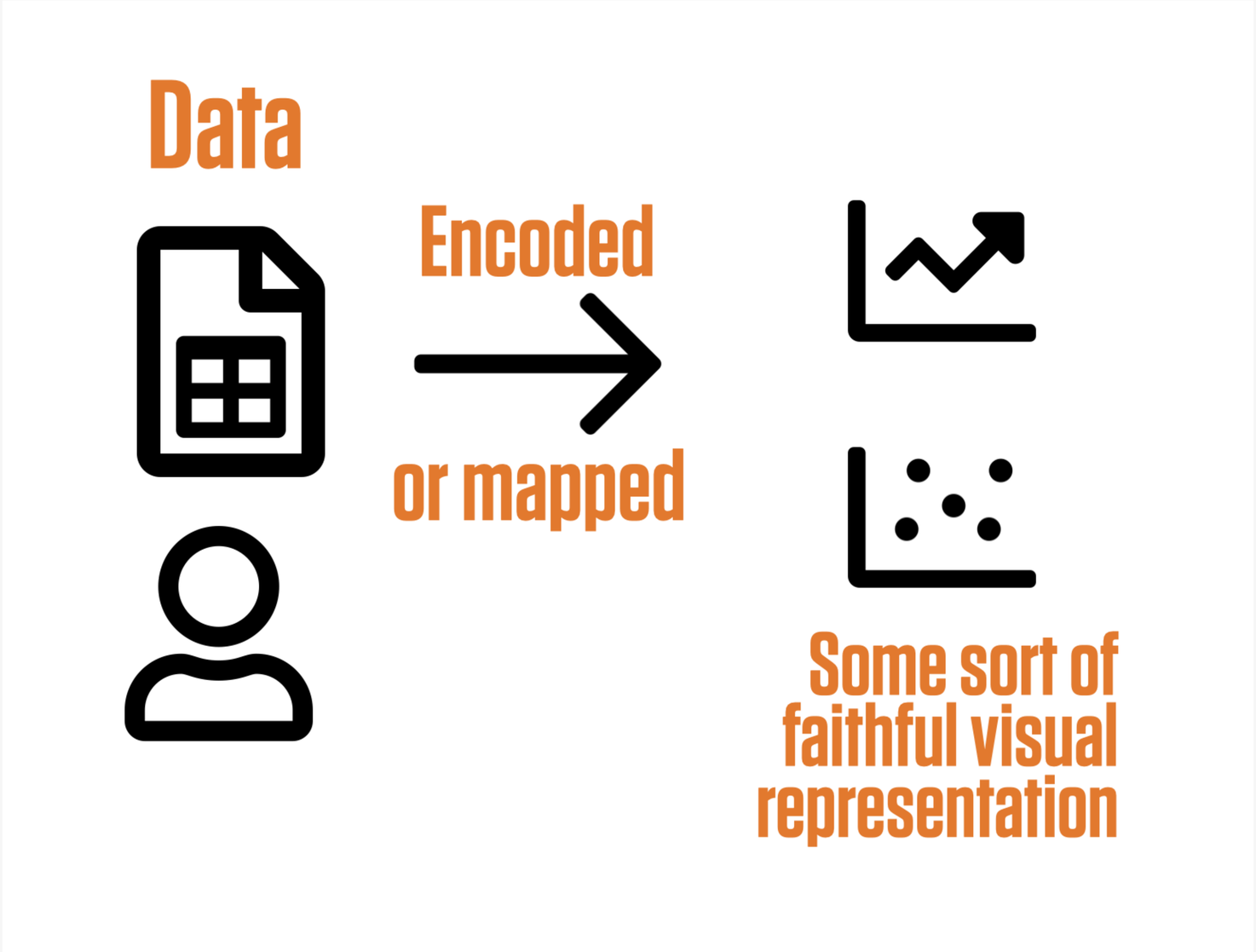
Adelson’s checkerboard



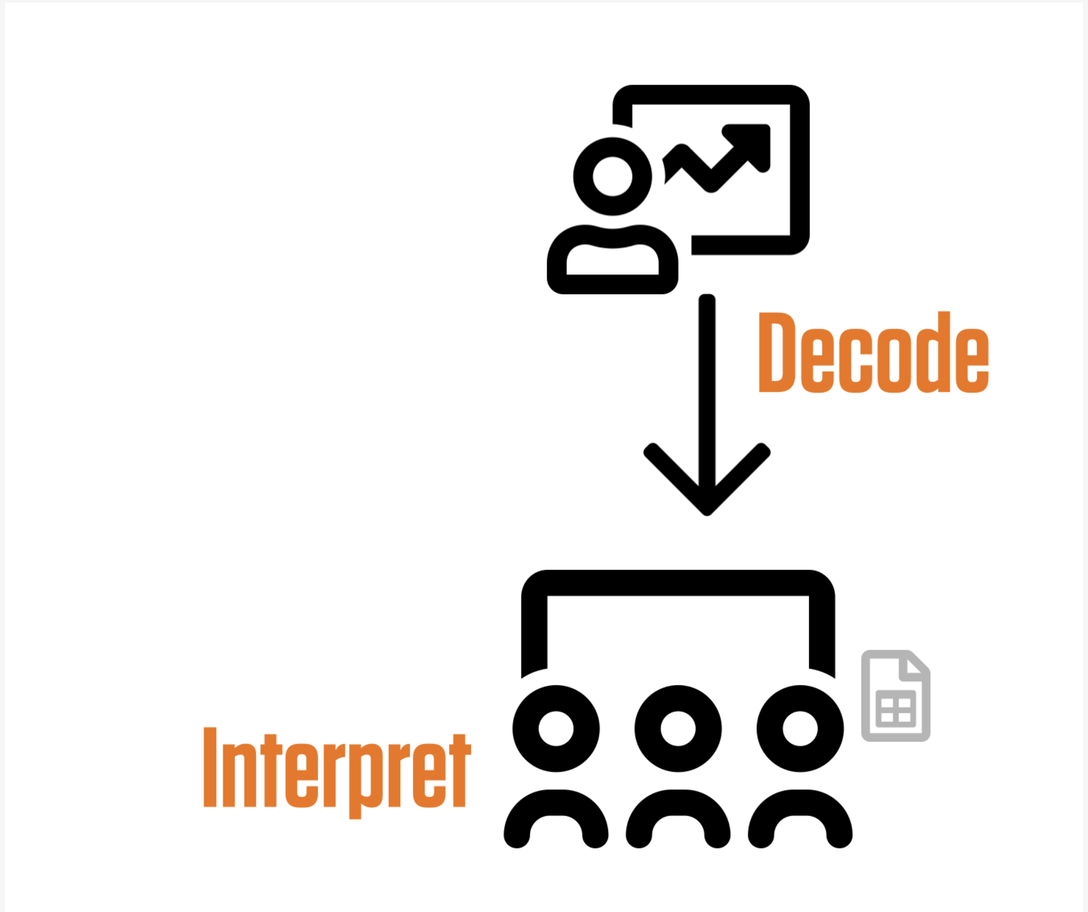
Edward Adelson

Visual Tasks for Decoding Graphs

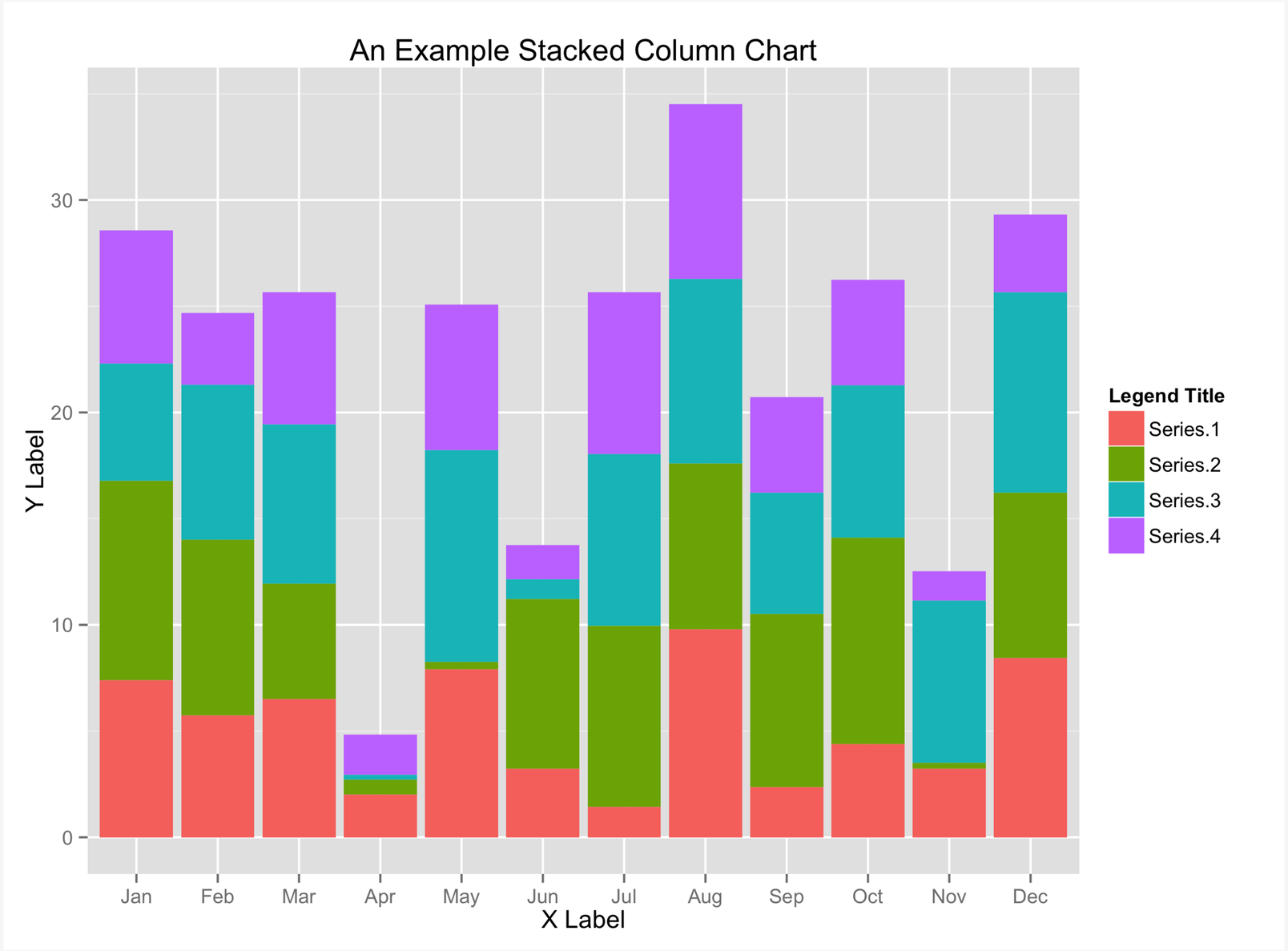
What's a graph, anyway?


Data
Encoded
or mapped
Some sort of faithful visual representation

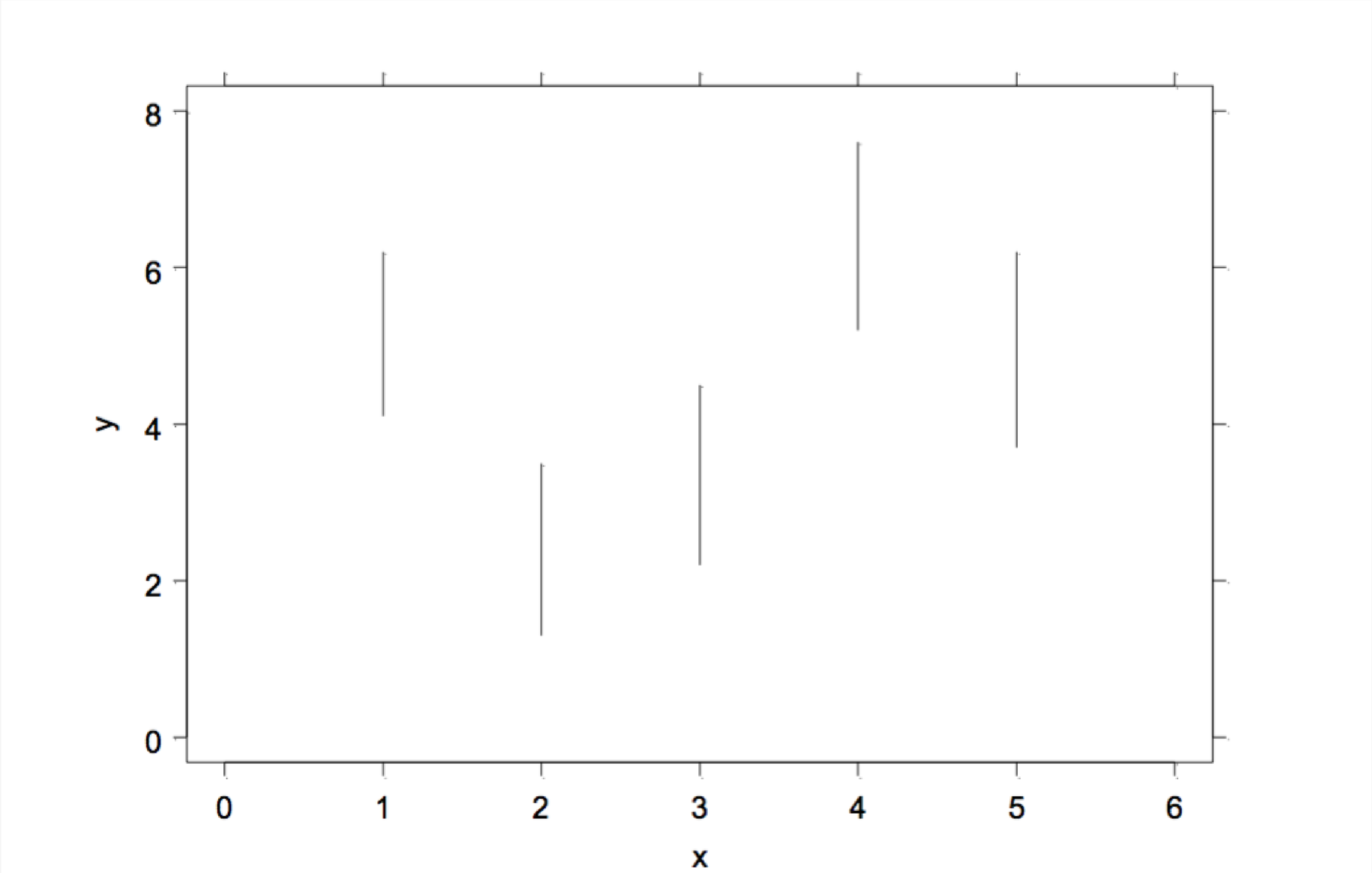
Encode ...



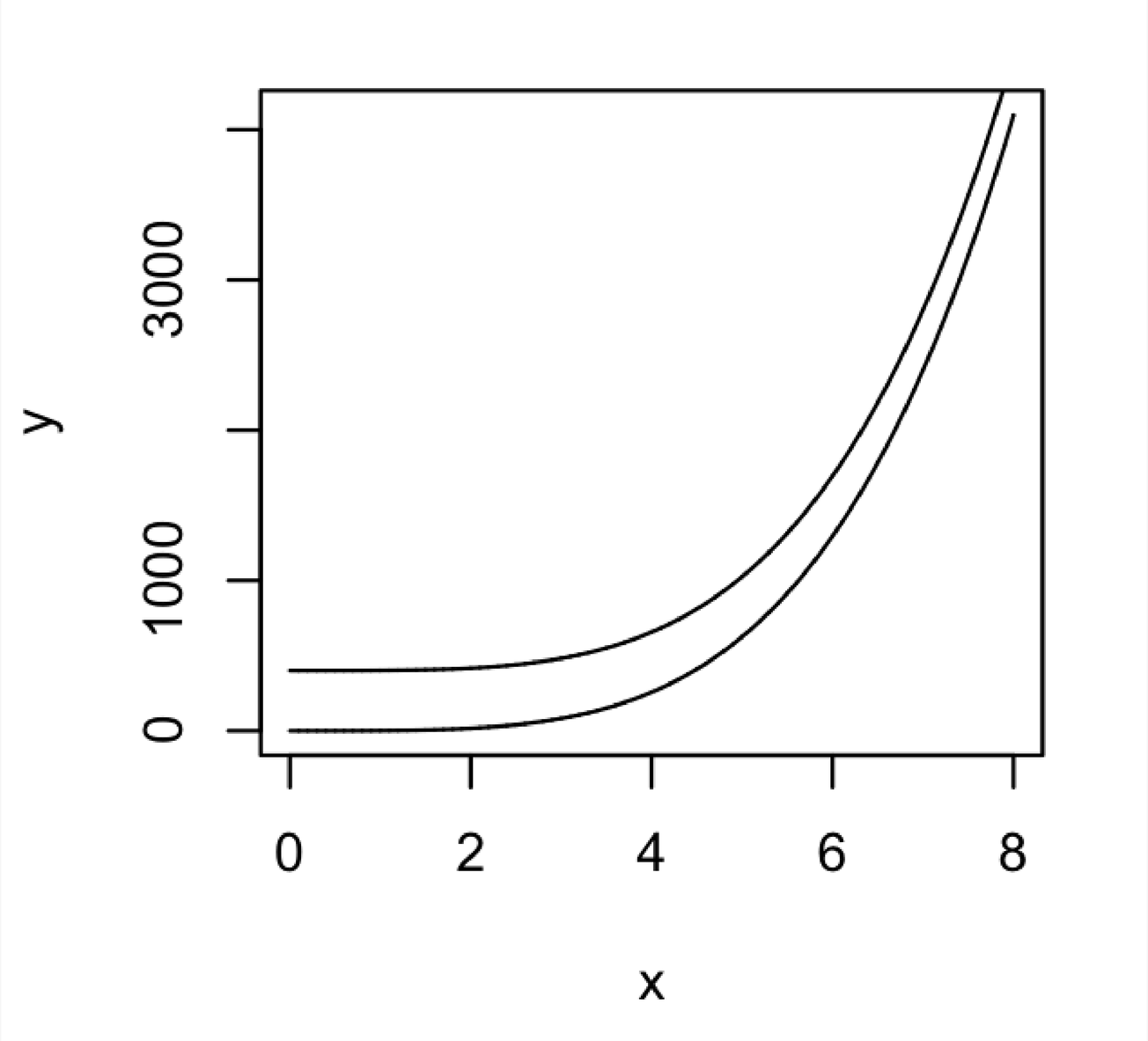
… and decode



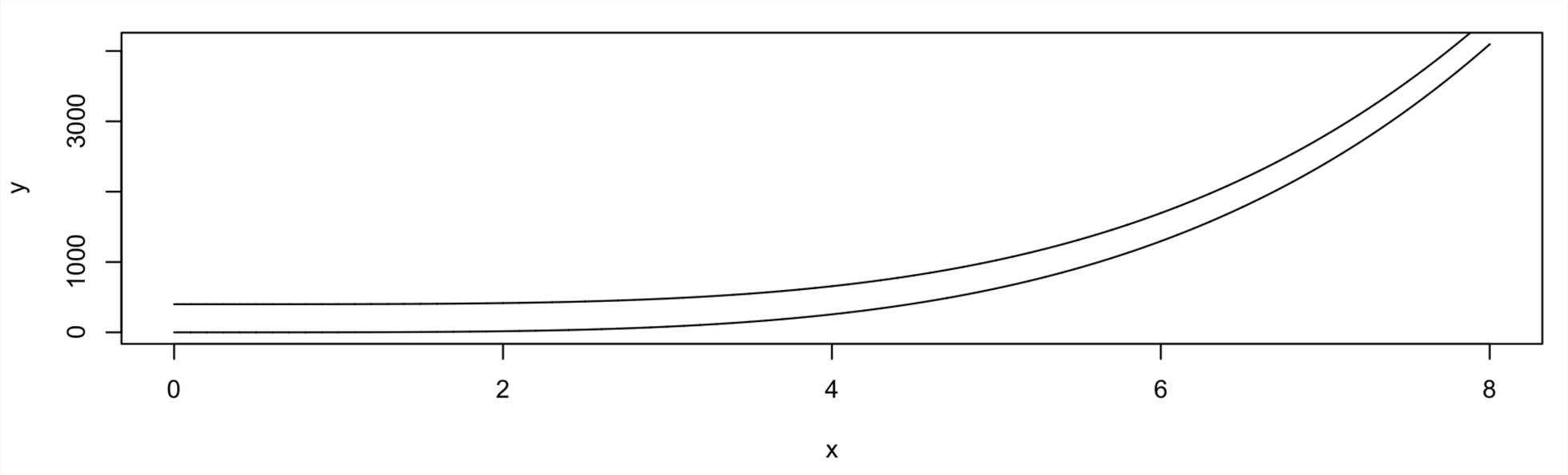
Stacked columns



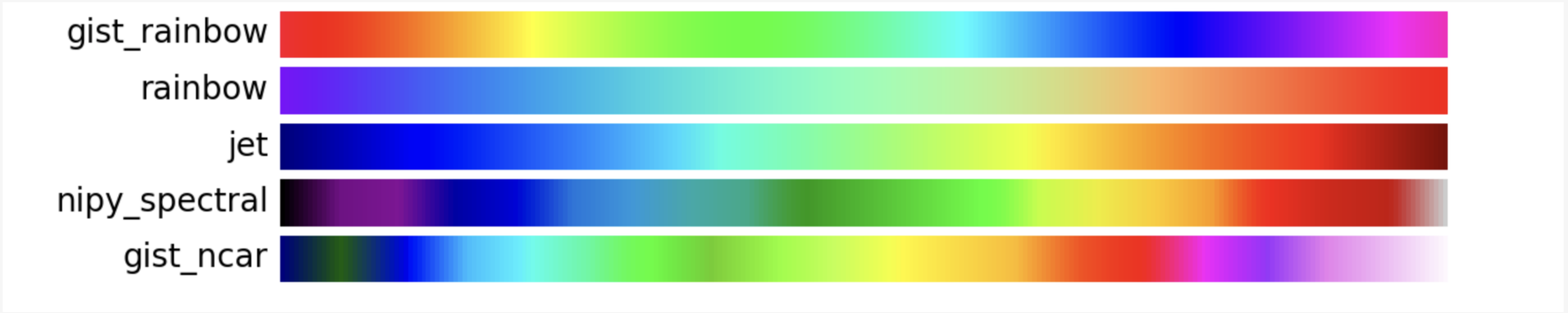
William Cleveland



William Cleveland

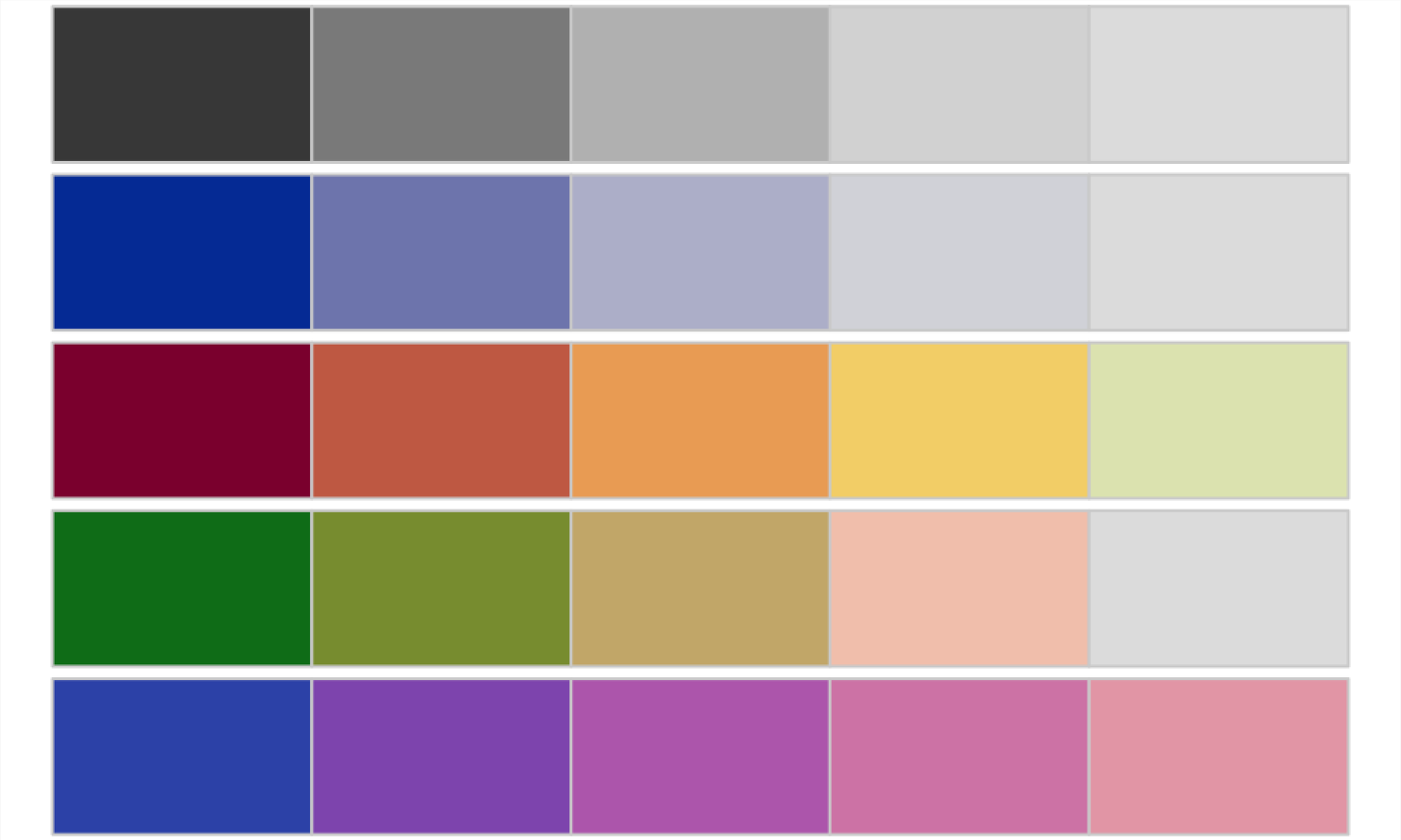


William Cleveland



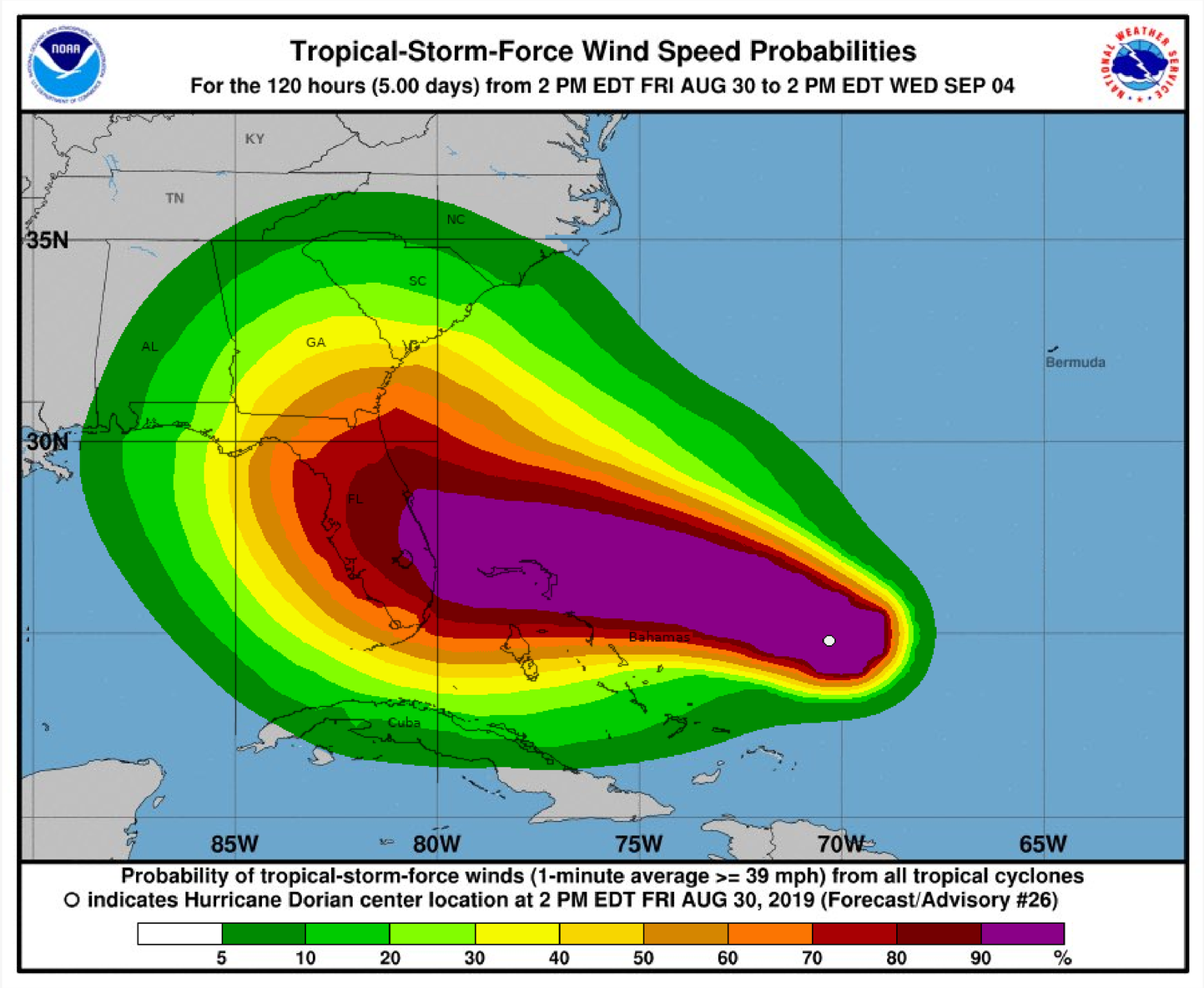
Rainbow gradients are not linear in the luminance channel

Luminance-balanced gradients

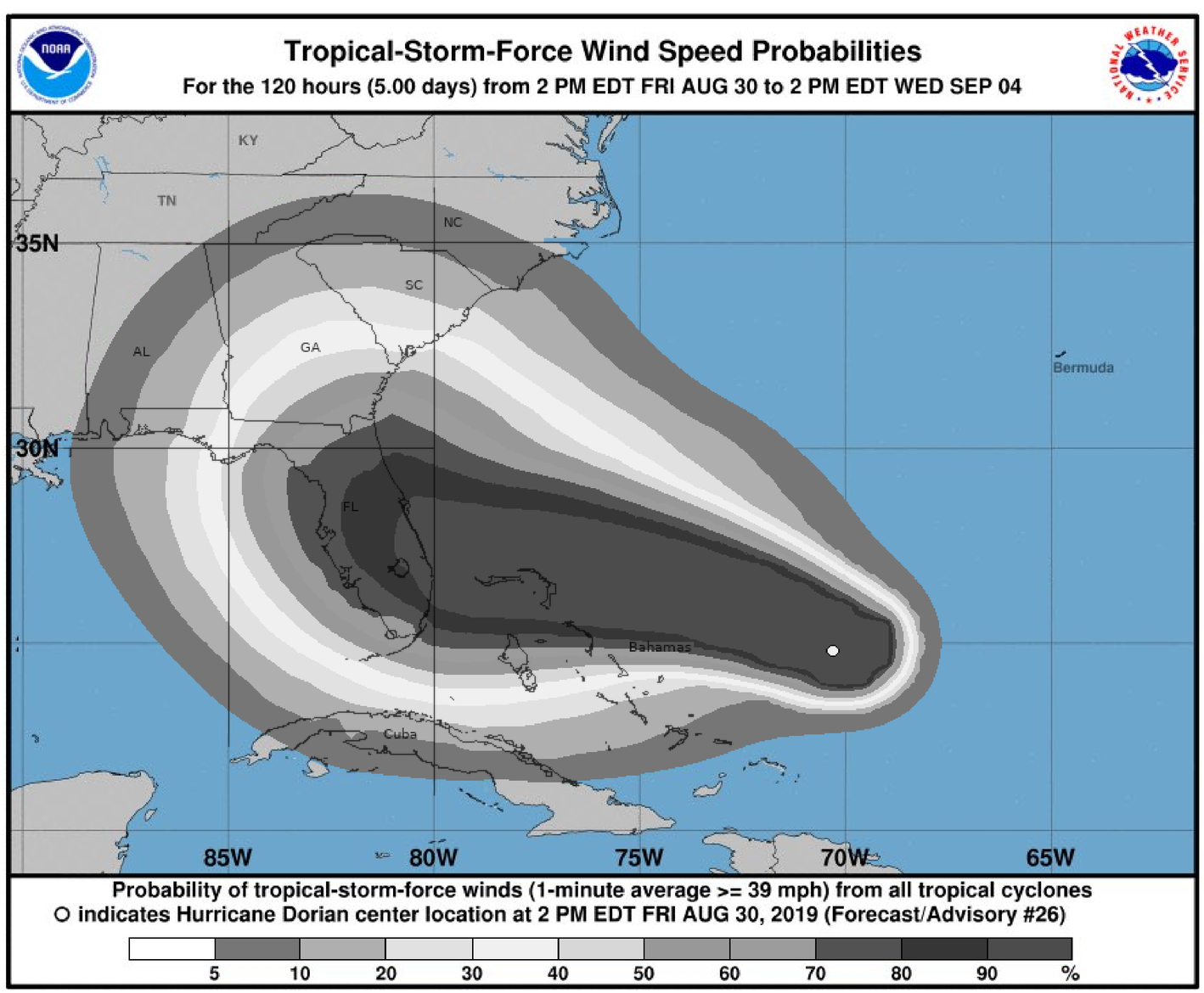


Luminance-balanced sequential palettes

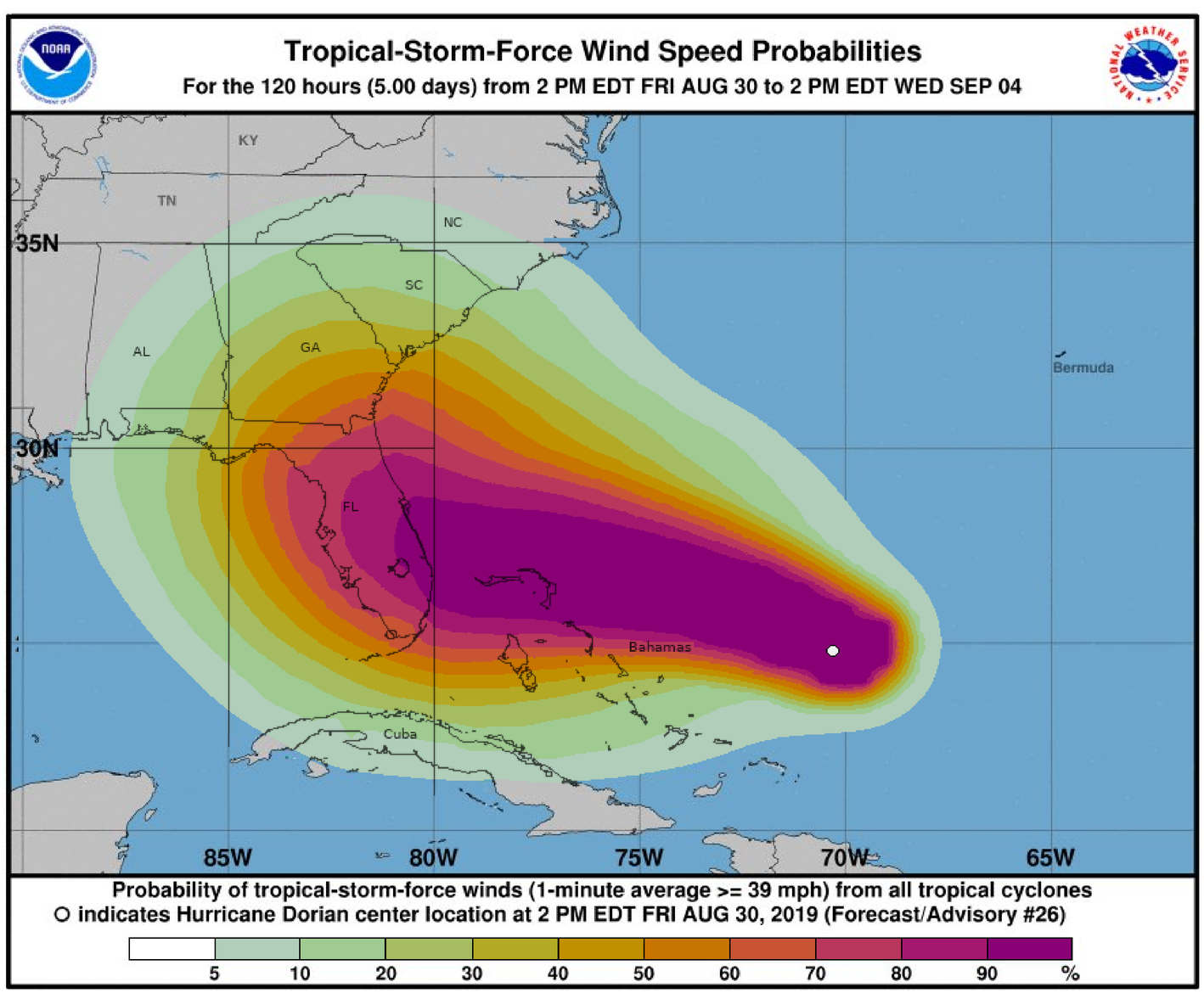
Luminance-balanced diverging and qualitative/unordered palettes



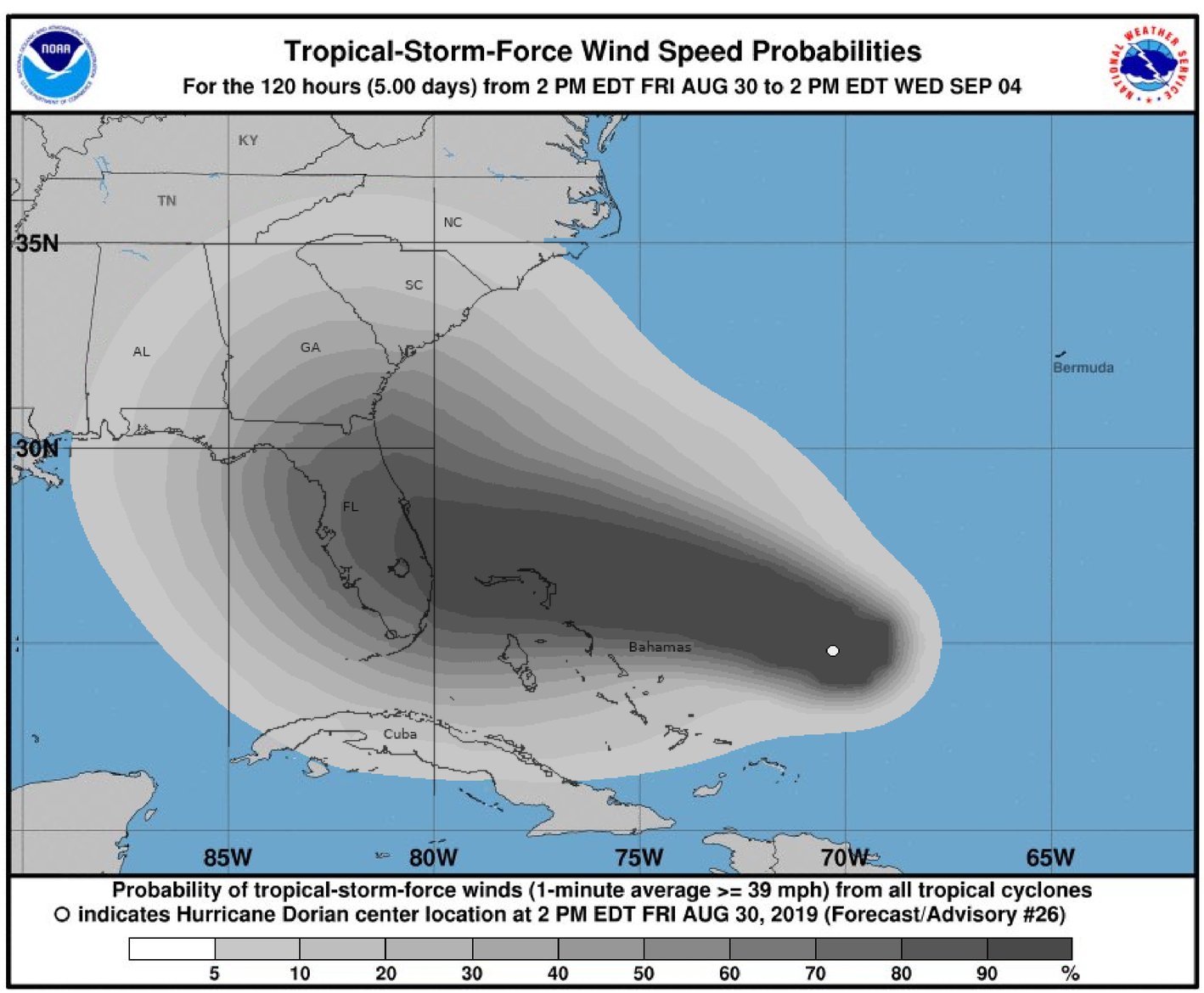
National Weather Service



Achim Zeileis

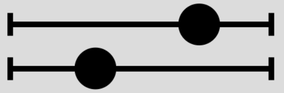
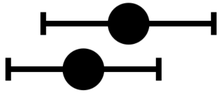
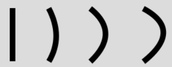


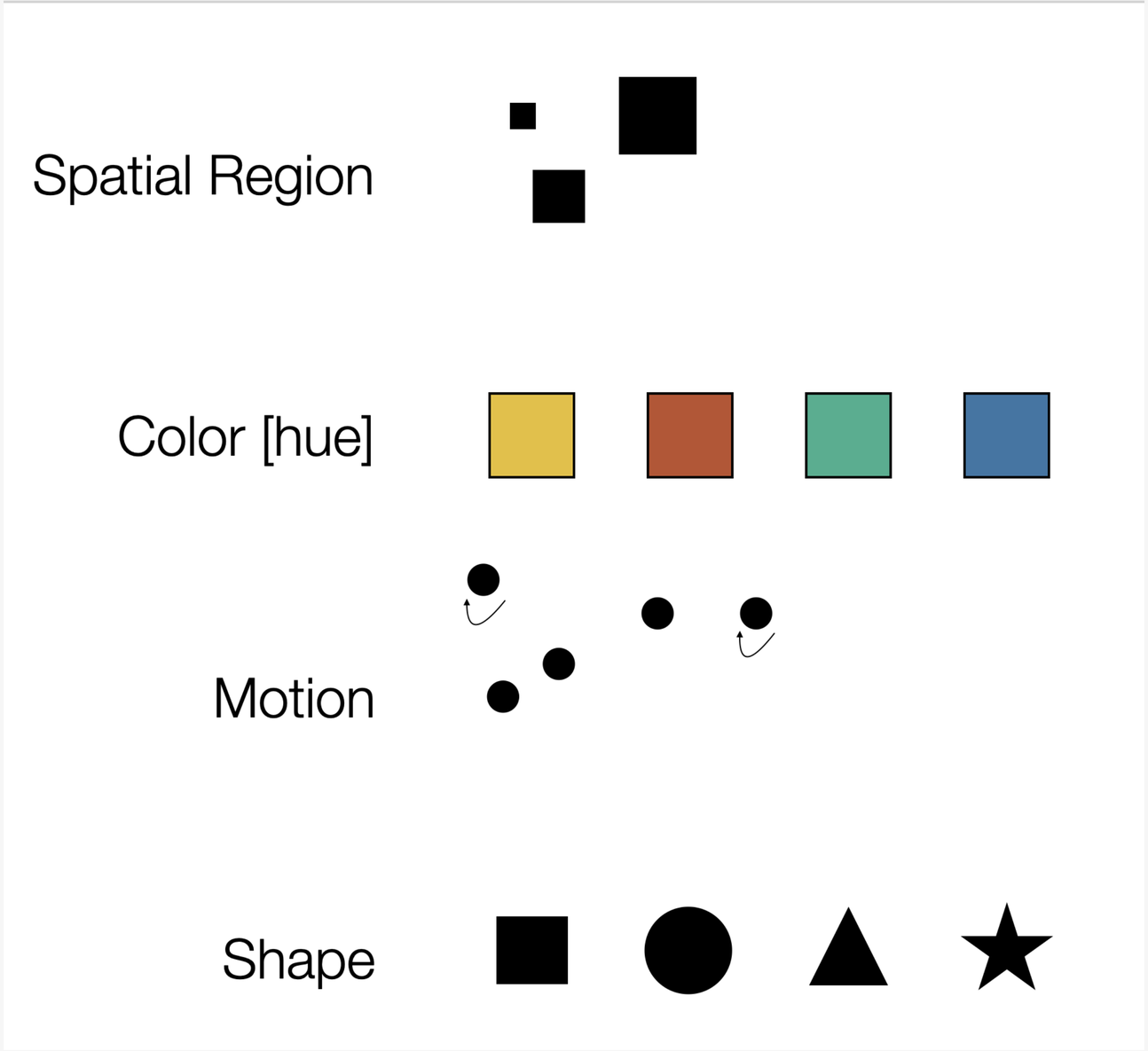
Achim Zeileis



Achim Zeileis

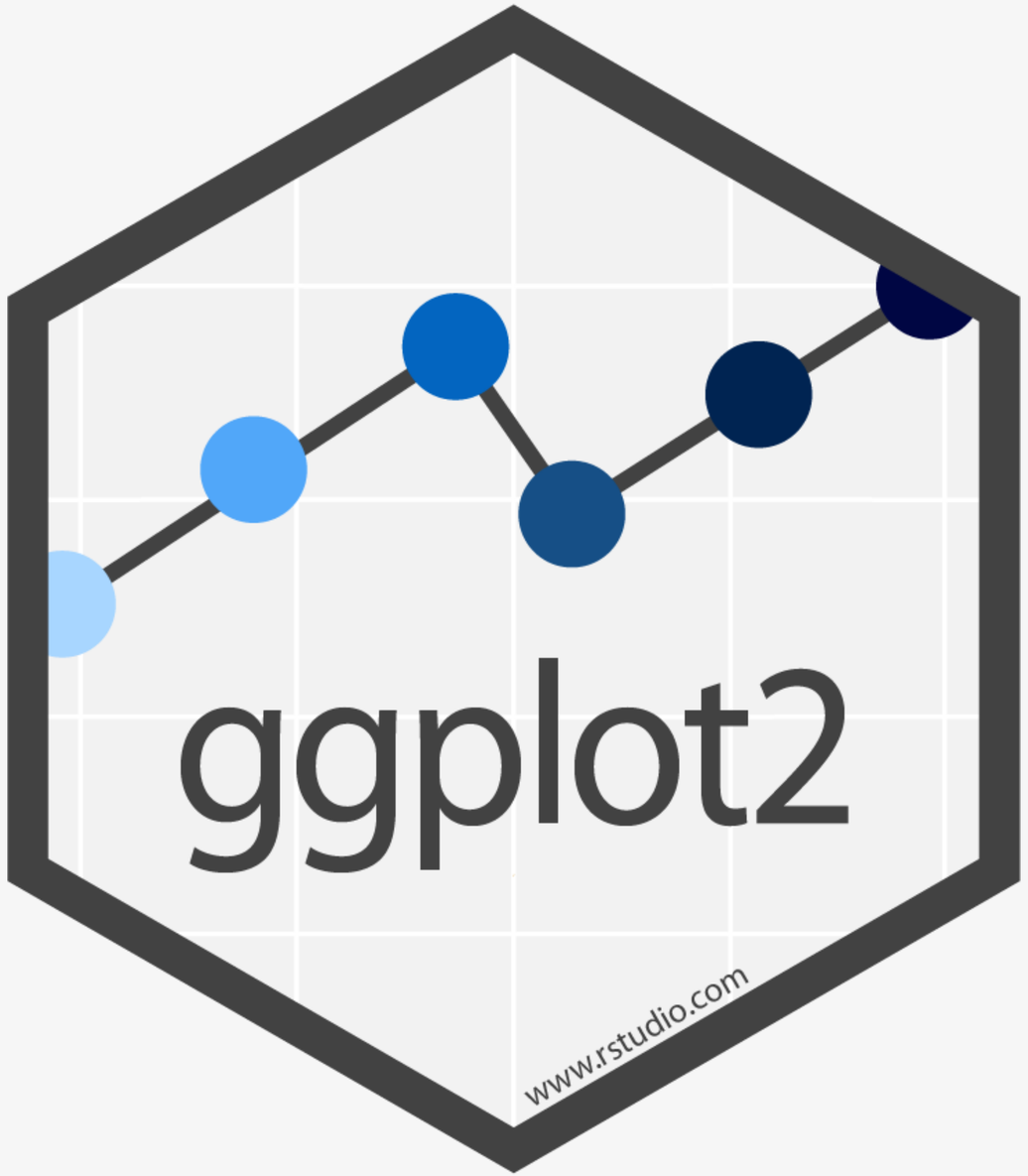
A rough hierarchy of mappings for data

Position on a common scale	
Position on unaligned scale	
Length (1D as size)	
Tilt or Angle	
Area (2D as size)	
Depth (3D as Position)	
Color luminance [brightness]	
Color saturation [intensity]	
Curvature	
Volume (3D as size)	


Spatial Region
Color [hue]
Motion
Shape

Learning How

The R Project for Statistical Computing

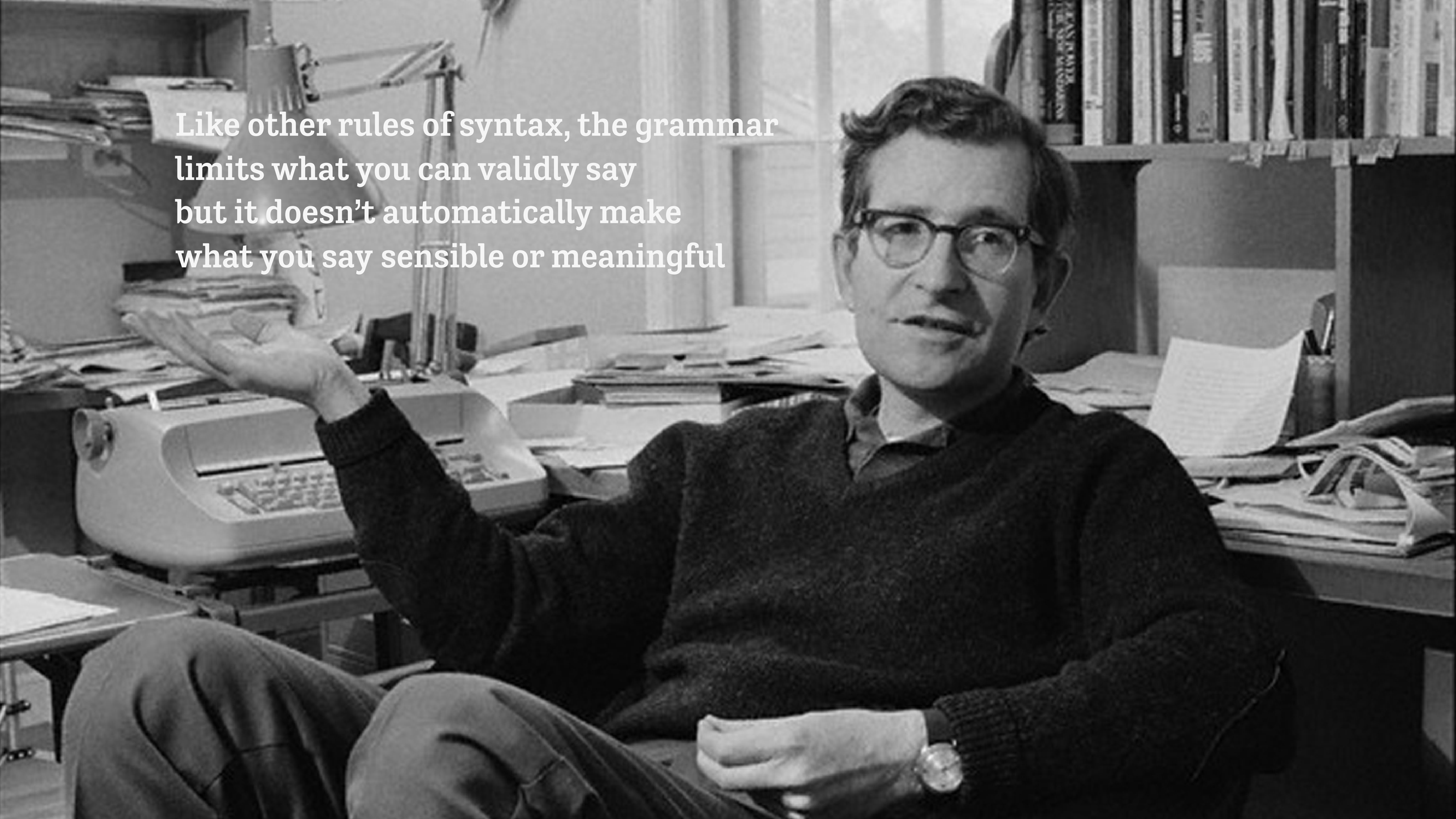


The ggplot2 package

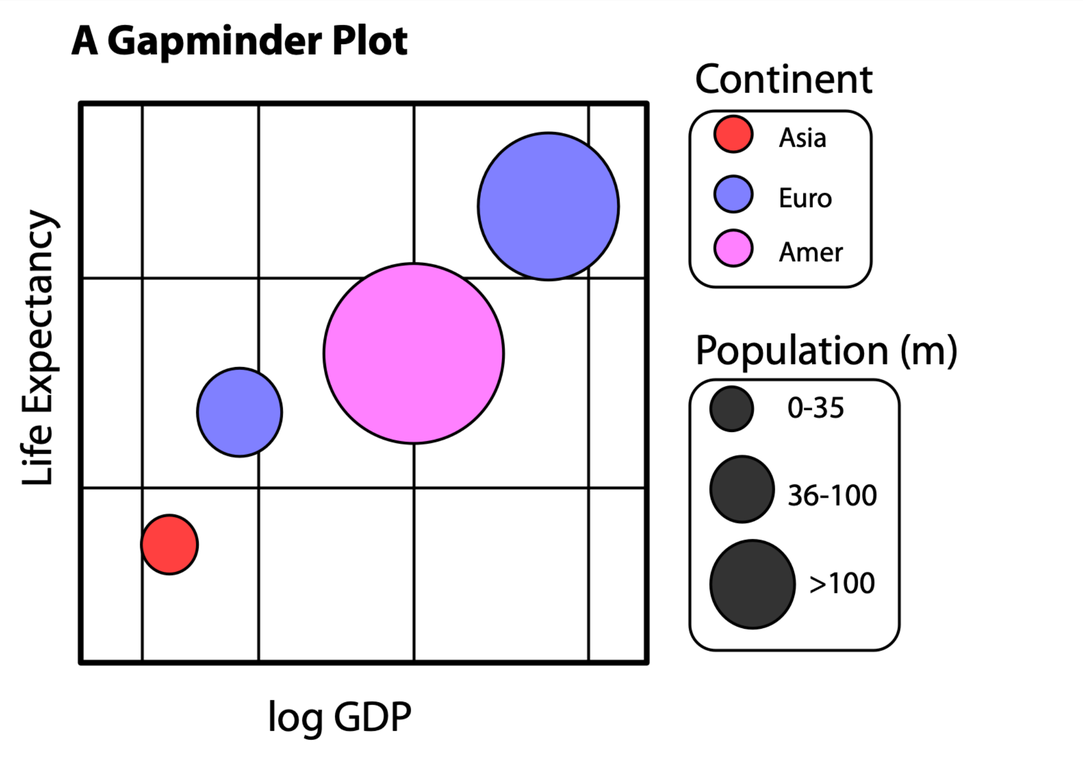
ggplot
implements a
grammar of
graphics

A grammar of graphics

The grammar is a set of rules for how to produce graphics from data, by *mapping* data to or *representing* it by geometric objects (like points and lines) that have aesthetic attributes (like position, color, size, and shape), together with further rules for transforming data if needed, for adjusting scales and their guides, and for projecting results onto some coordinate system.


Like other rules of syntax, the grammar
limits what you can validly say
but it doesn’t automatically make
what you say sensible or meaningful

What we need our code to make



Data **represented** by visual elements;

like *position*, *length*, *color*, and *size*;

Each measured on some **scale**;

Each scale with a labeled **guide**;

With the plot itself also **titled** and labeled.

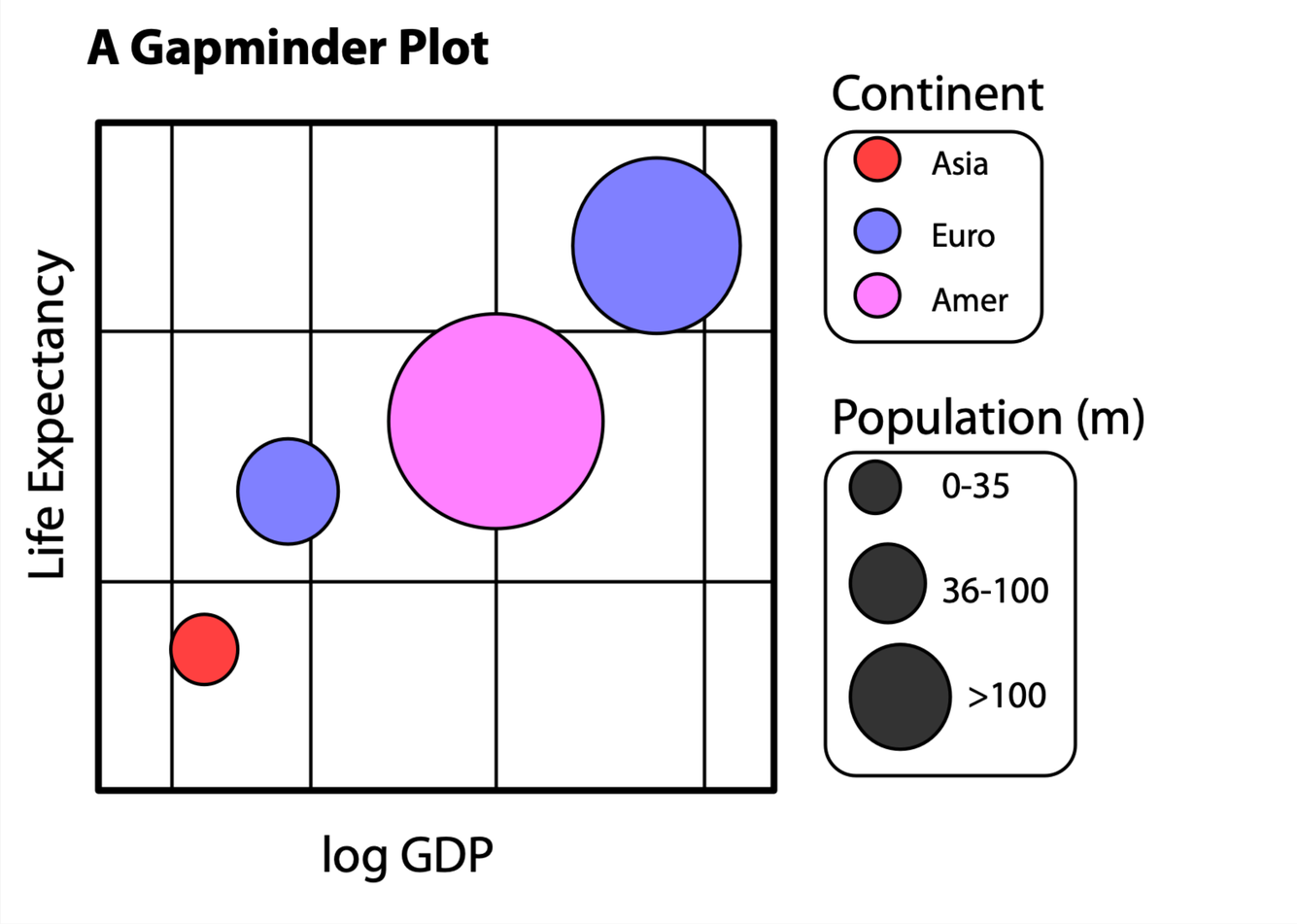
How does ggplot do this?

ggplot's flow of action

gdp	lifexp	pop	continent
340	65	31	Euro
227	51	200	Amer
909	81	80	Euro
126	40	20	Asia

What we start with

ggplot's flow of action



Where we're going

ggplot's flow of action

1. Tidy Data

```
p ← ggplot(data = gapminder, ...
```

gdp	lifexp	pop	continent
340	65	31	Euro
227	51	200	Amer
909	81	80	Euro
126	40	20	Asia

1. Get the data in the right shape. This is usually **long** format.

2. Map the Aesthetics

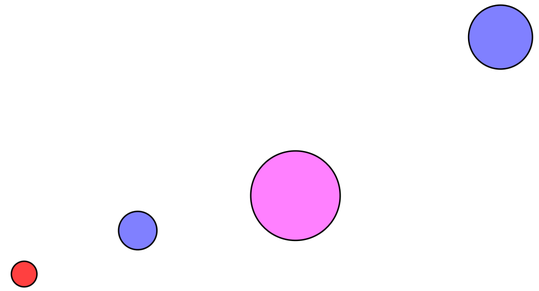
```
p ← ggplot(data = gapminder,
           mapping = aes(x = gdp,
                         y = lifexp,
                         size = pop,
                         color = continent))
```

x-axis will represent	y-axis will represent	size will represent	color will represent
gdp	lifexp	pop	continent
340	65	31	Euro

2. Decide how your variables will be **represented** by things you can see.

3. Pick a Geom

```
p + geom_point()
```



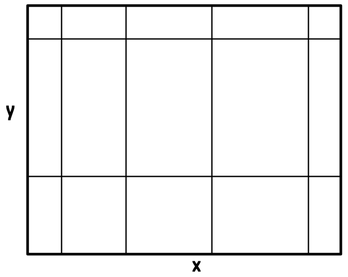
3. Decide **what kind** of plot, or series of plots, you want to draw.

Core steps

ggplot's flow of action

4. Fix Scales and Co-Ordinate System

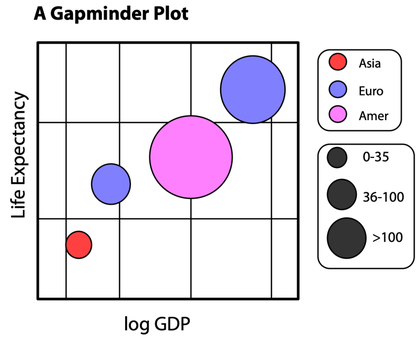
```
p + coord_cartesian() +
    scale_x_log10()
```



4. Adjust **scales and their markings.** Not just x and y but also color, size, etc,

5. Add Labels and Adjust Guides

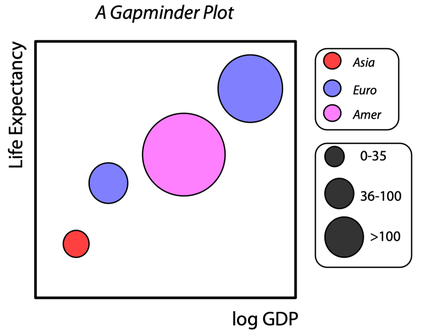
```
p + labs(x = "log GDP",
         y = "Life Expectancy",
         title = "A Gapminder Plot")
```



5. Label your plot and adjust how the guides are displayed.

6. Add or Adjust Themes

```
p + theme_minimal()
```



6. Style or adjust plot elements that are **not directly representing** your data.

Optional steps

You can think of this as a series of transformations, carried out by functions

Example: Gapminder

Start with the data

```
gapminder
```

```
# A tibble: 1,704 × 6
   country     continent  year lifeExp      pop gdpPercap
   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
 1 Afghanistan Asia       1952    28.8  8425333      779.
 2 Afghanistan Asia       1957    30.3  9240934      821.
 3 Afghanistan Asia       1962    32.0 10267083      853.
 4 Afghanistan Asia       1967    34.0 11537966      836.
 5 Afghanistan Asia       1972    36.1 13079460      740.
 6 Afghanistan Asia       1977    38.4 14880372      786.
 7 Afghanistan Asia       1982    39.9 12881816      978.
 8 Afghanistan Asia       1987    40.8 13867957      852.
 9 Afghanistan Asia       1992    41.7 16317921      649.
10 Afghanistan Asia       1997    41.8 22227415      635.
# ℹ 1,694 more rows
```

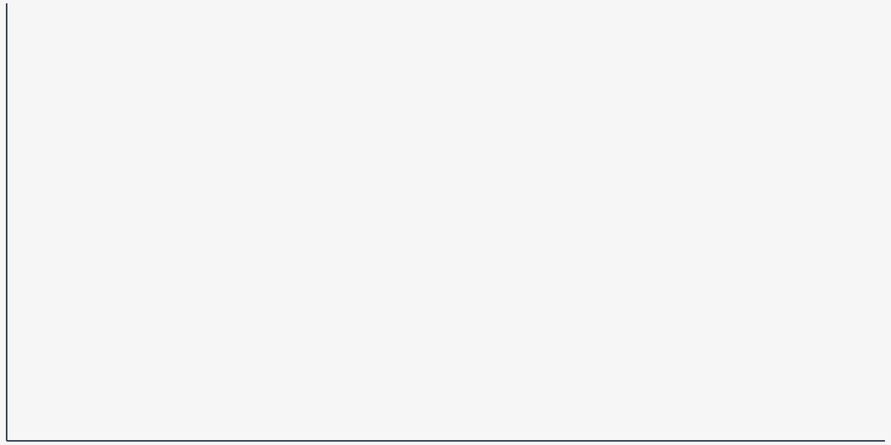
```
dim(gapminder)
```

```
[1] 1704    6
```

Create a plot object

Data is the `gapminder` table.

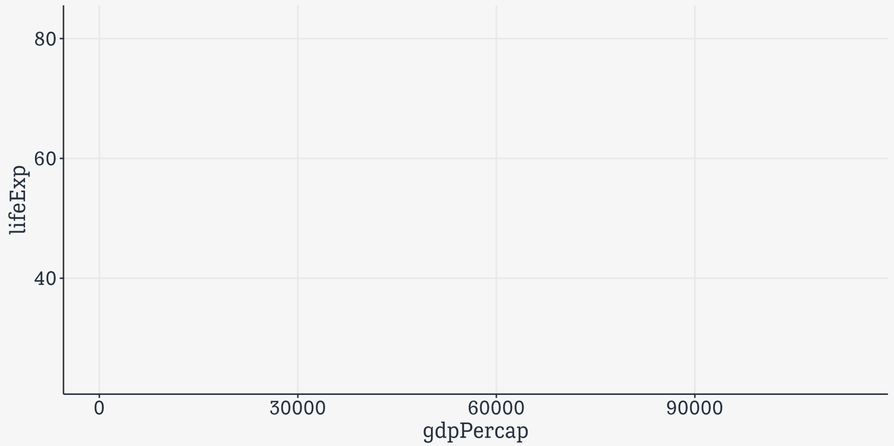
```
gapminder |>
  ggplot()
```



Map variables to aesthetics

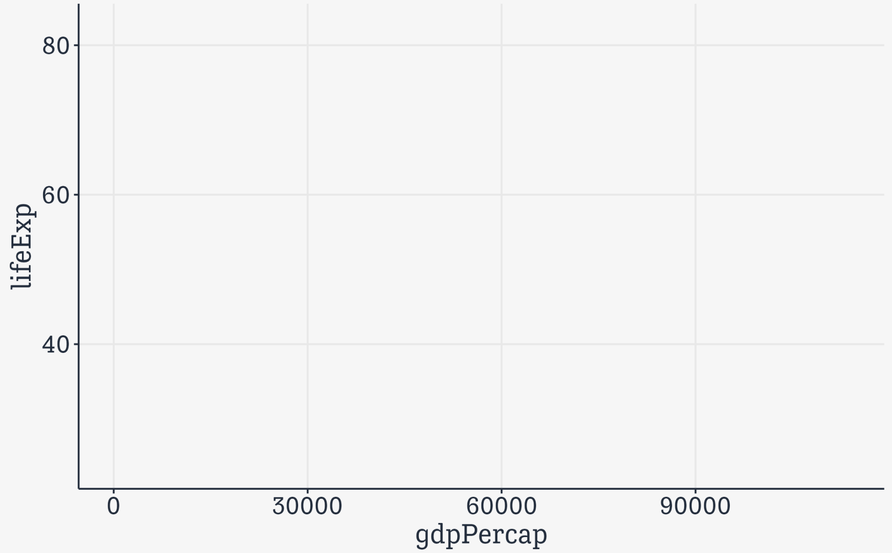
Tell `ggplot` which visual elements represent which columns

```
gapminder |>
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent))
```



What sort of plot?

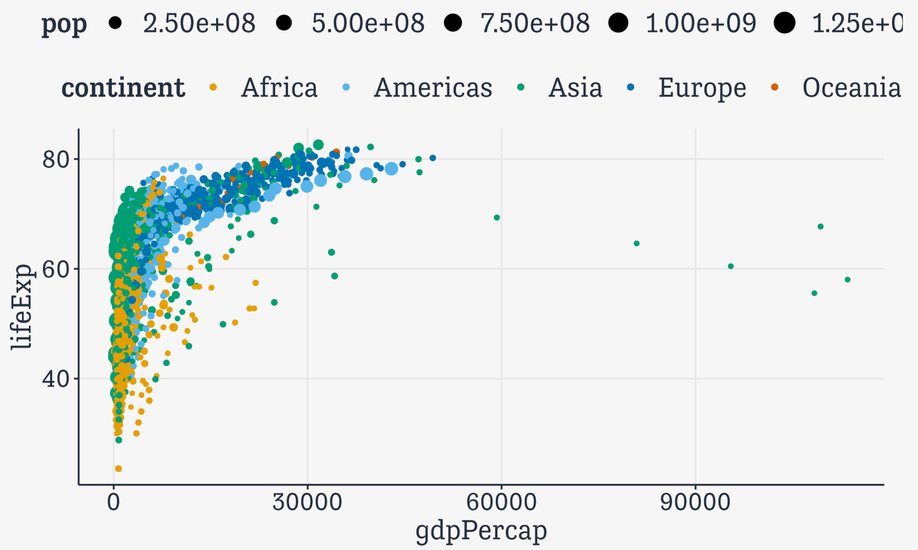
```
gapminder |>
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent))
```



This empty plot has no geoms.

Pick a geom

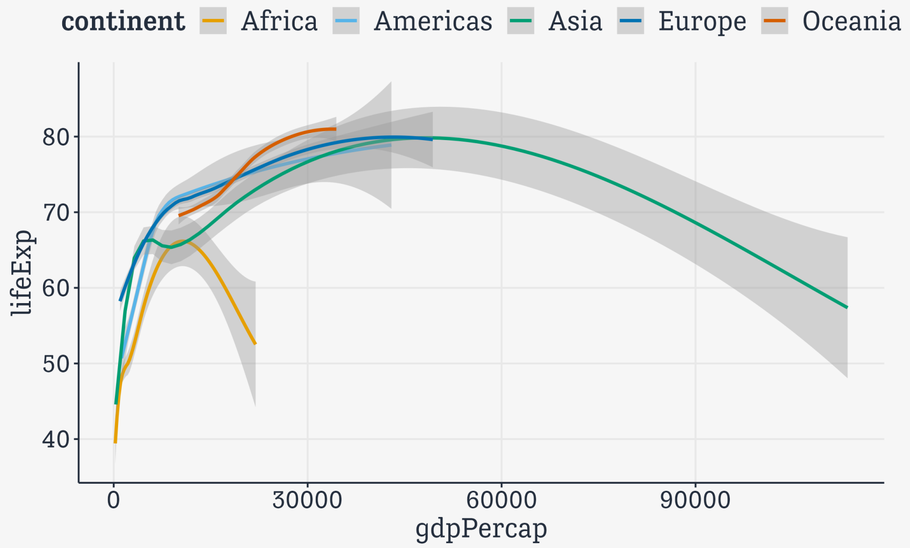
```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point()
```



A scatterplot of Life Expectancy vs GDP

Try a different one

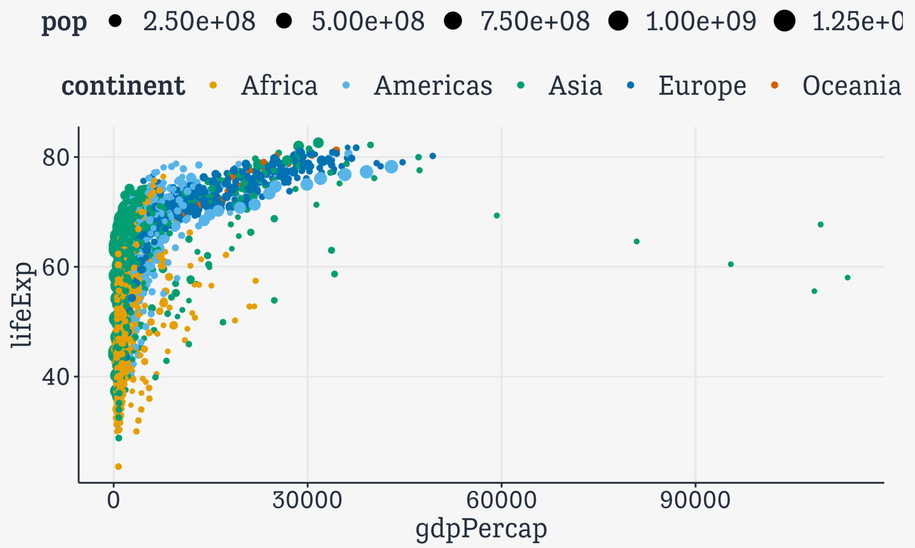
```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_smooth()
```



A smoothed lineplot of Life Expectancy vs GDP

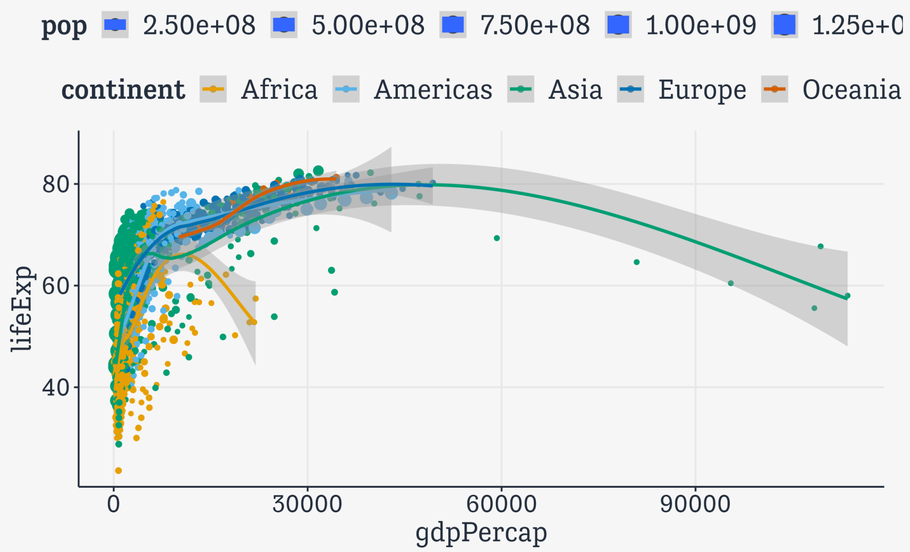
Build your plots layer by layer

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point()
```



This process is additive

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point() +
  geom_smooth()
```

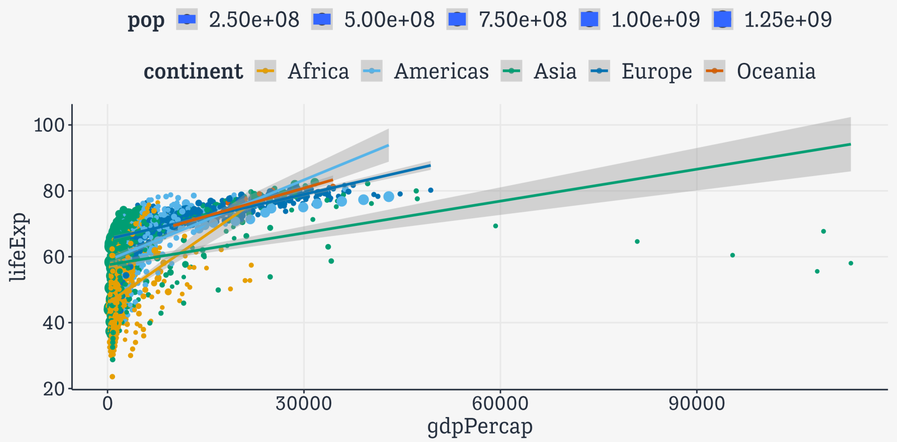


Point and smoother together

Every geom is a function

Functions take arguments

```
gapminder |>
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point() +
  geom_smooth(method = "lm")
```



An ill-advised linear fit

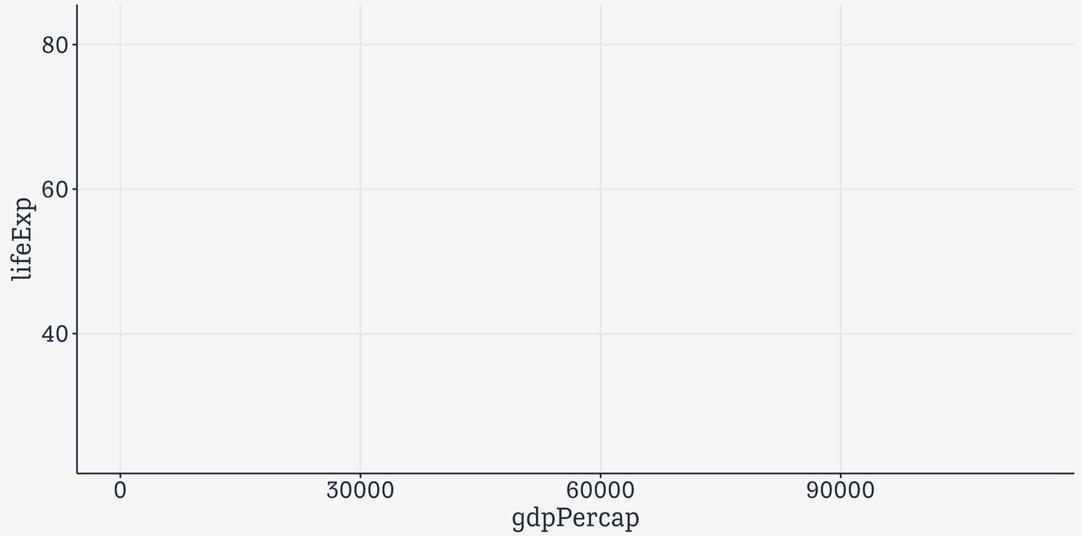
Keep Layering

```
gapminder
```

```
# A tibble: 1,704 × 6
   country     continent  year lifeExp      pop gdpPercap
   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
 1 Afghanistan Asia       1952    28.8  8425333      779.
 2 Afghanistan Asia       1957    30.3  9240934      821.
 3 Afghanistan Asia       1962    32.0 10267083      853.
 4 Afghanistan Asia       1967    34.0 11537966      836.
 5 Afghanistan Asia       1972    36.1 13079460      740.
 6 Afghanistan Asia       1977    38.4 14880372      786.
 7 Afghanistan Asia       1982    39.9 12881816      978.
 8 Afghanistan Asia       1987    40.8 13867957      852.
 9 Afghanistan Asia       1992    41.7 16317921      649.
10 Afghanistan Asia       1997    41.8 22227415      635.
# ℹ 1,694 more rows
```

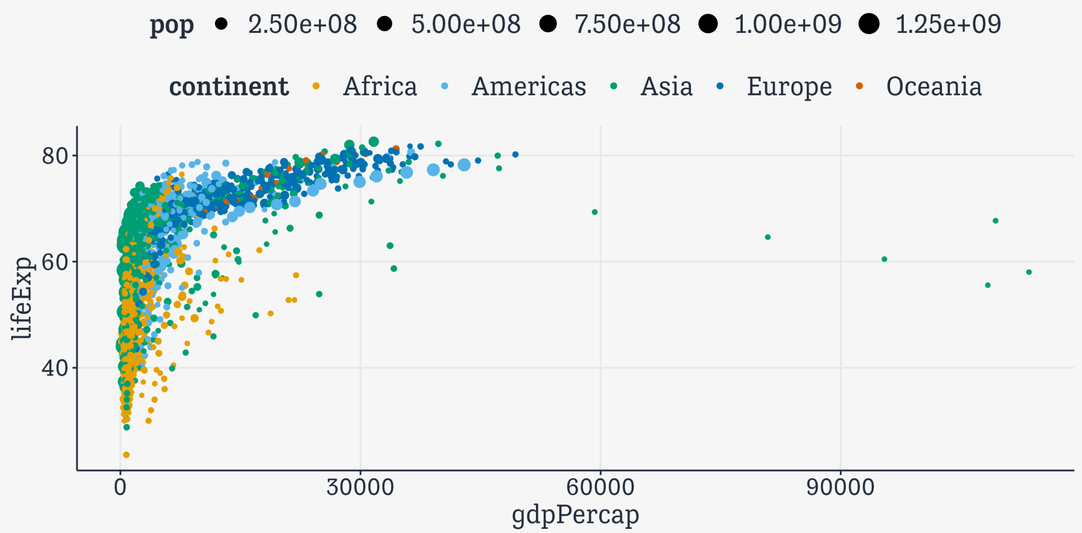
Keep Layering

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent))
```



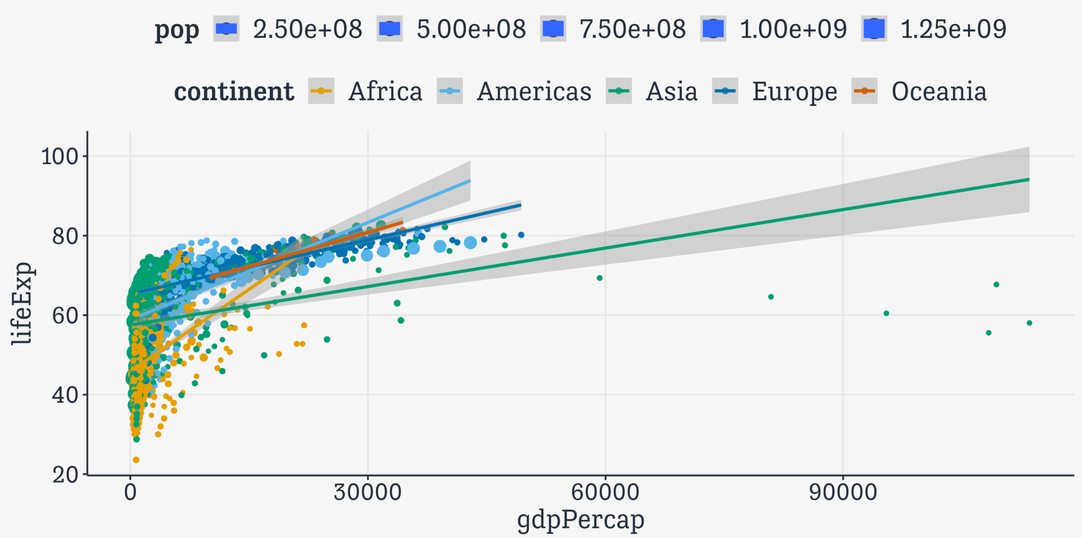
Keep Layering

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point()
```



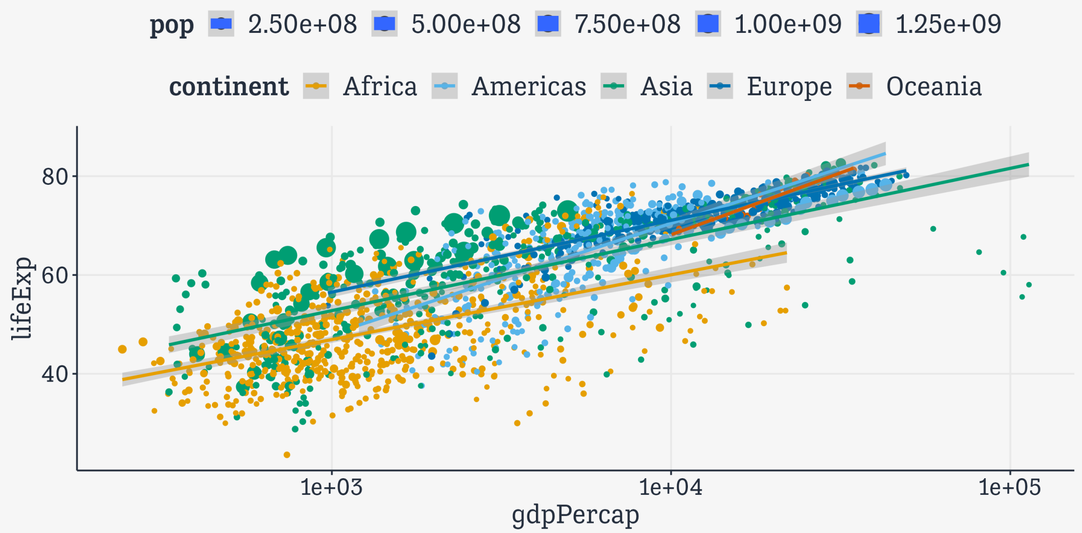
Keep Layering

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point() +
  geom_smooth(method = "lm")
```



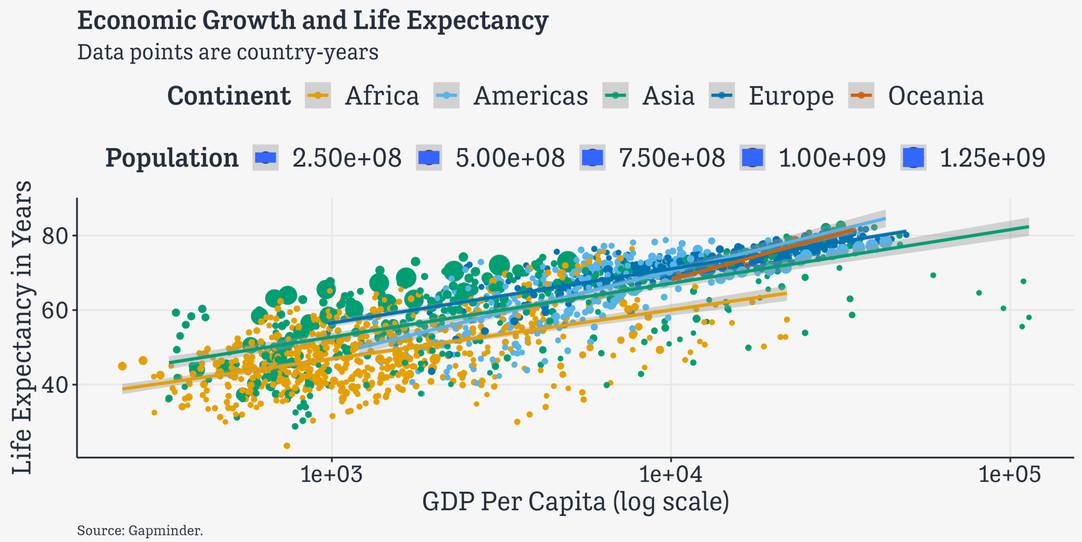
Keep Layering

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_x_log10()
```



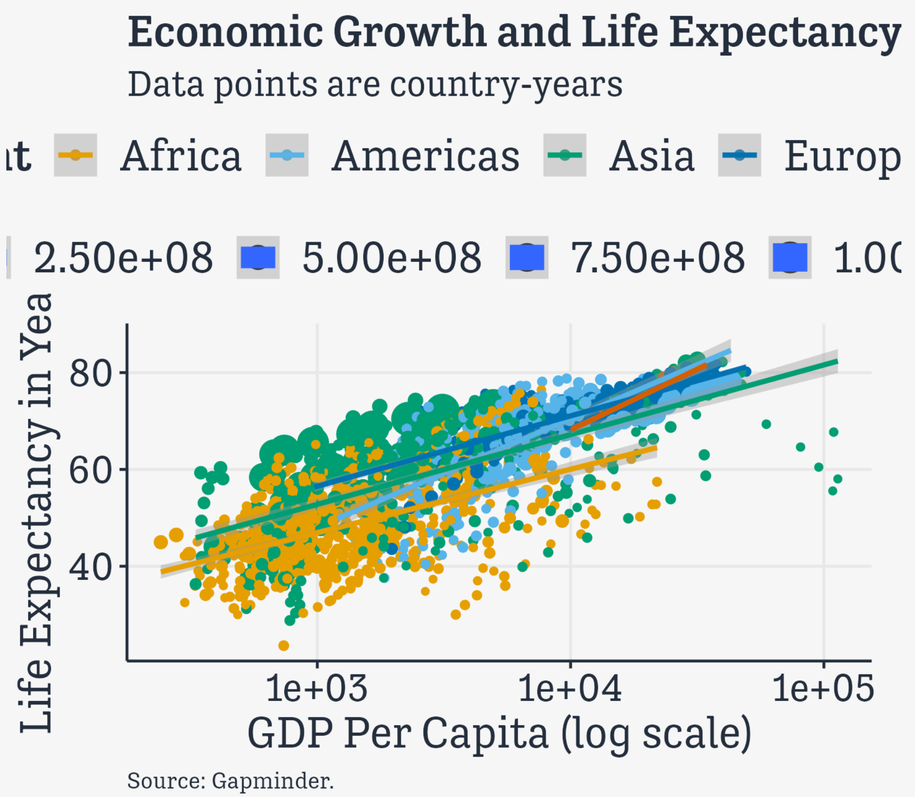
Keep Layering

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_x_log10() +
  labs(x = "GDP Per Capita (log scale)",
       y = "Life Expectancy in Years",
       size = "Population",
       color = "Continent",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```



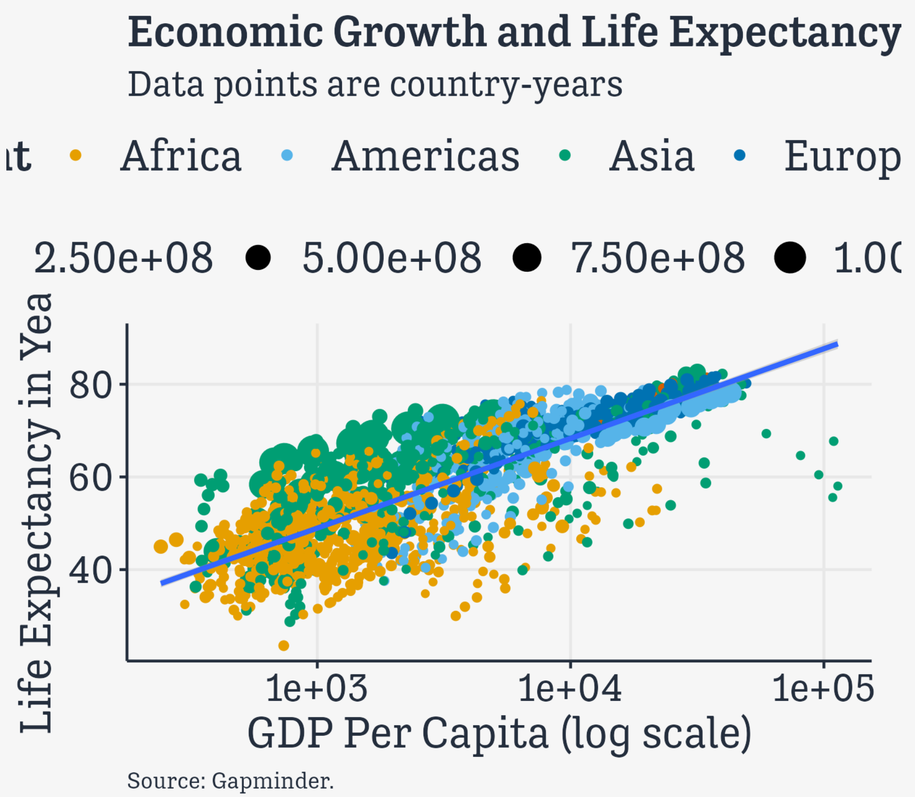
Nearly there

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp,
                       size = pop,
                       color = continent)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_x_log10() +
  labs(x = "GDP Per Capita (log scale)",
       y = "Life Expectancy in Years",
       size = "Population",
       color = "Continent",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```



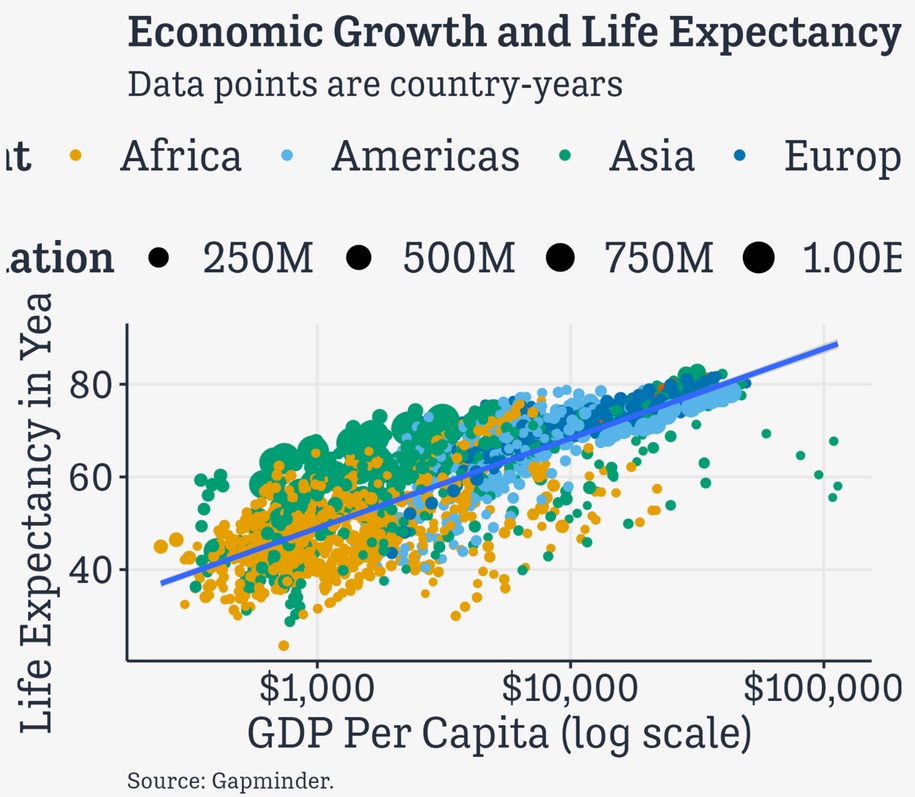
Simplify to one line

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp)) +
  geom_point(mapping =
               aes(size = pop,
                   color = continent)) +
  geom_smooth(method = "lm") +
  scale_x_log10() +
  labs(x = "GDP Per Capita (log scale)",
       y = "Life Expectancy in Years",
       size = "Population",
       color = "Continent",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```



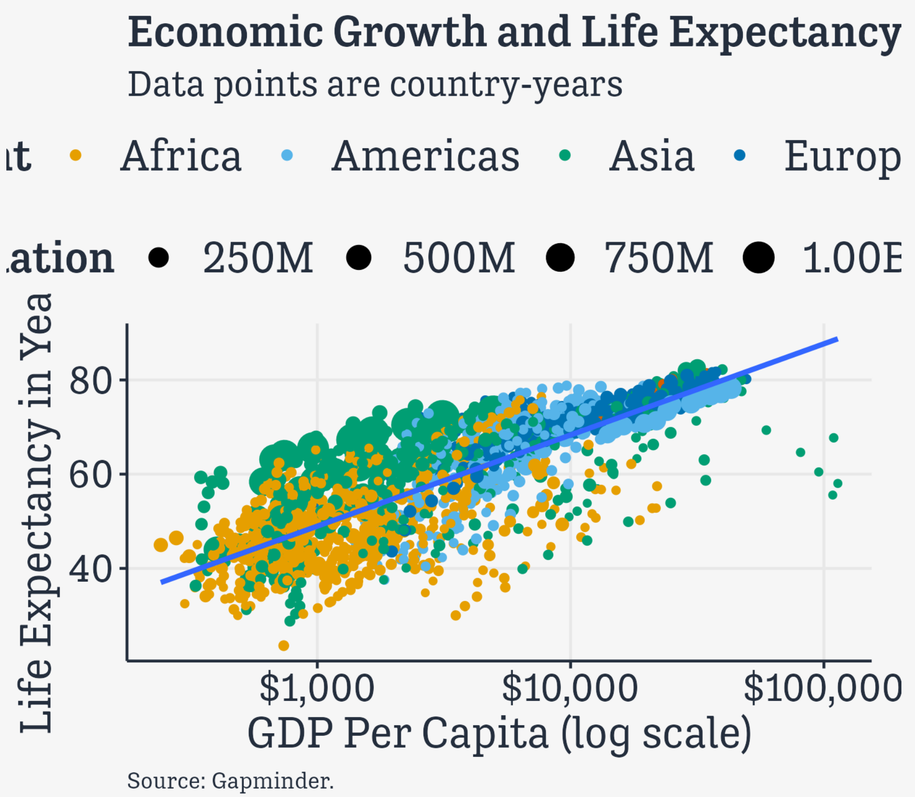
Fix the guide labels

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp)) +
  geom_point(mapping =
               aes(size = pop,
                   color = continent)) +
  geom_smooth(method = "lm") +
  scale_x_log10(labels = label_dollar()) +  #<<
  scale_size(labels =
               label_number(scale_cut = cut_short_scale())) +
  labs(x = "GDP Per Capita (log scale)",
       y = "Life Expectancy in Years",
       size = "Population",
       color = "Continent",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```



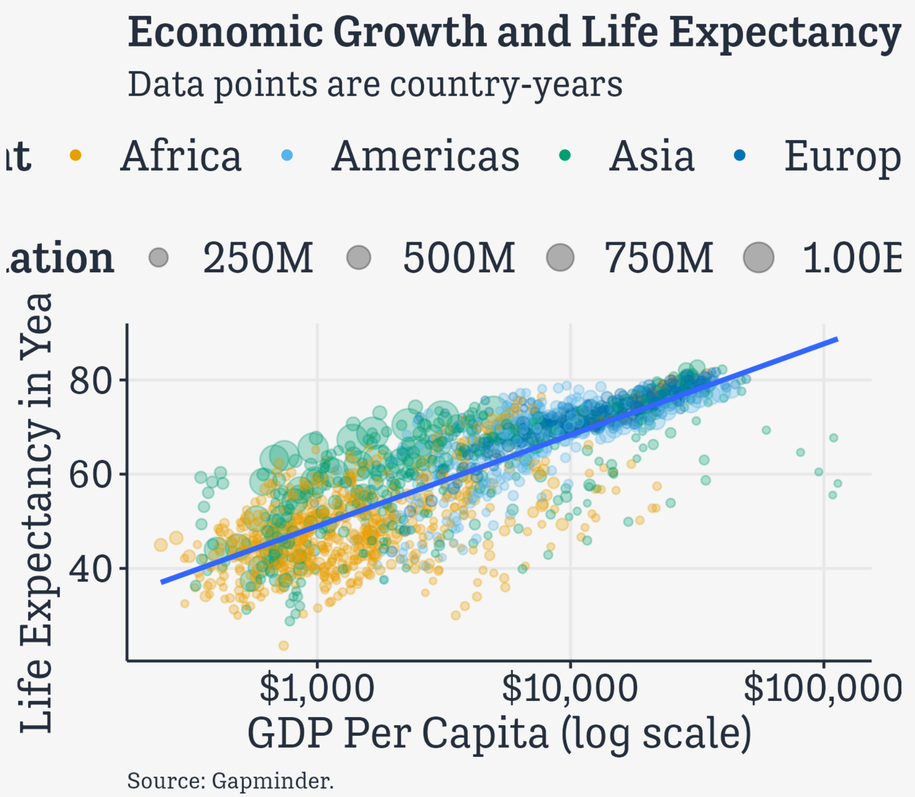
Remove the lm SE band

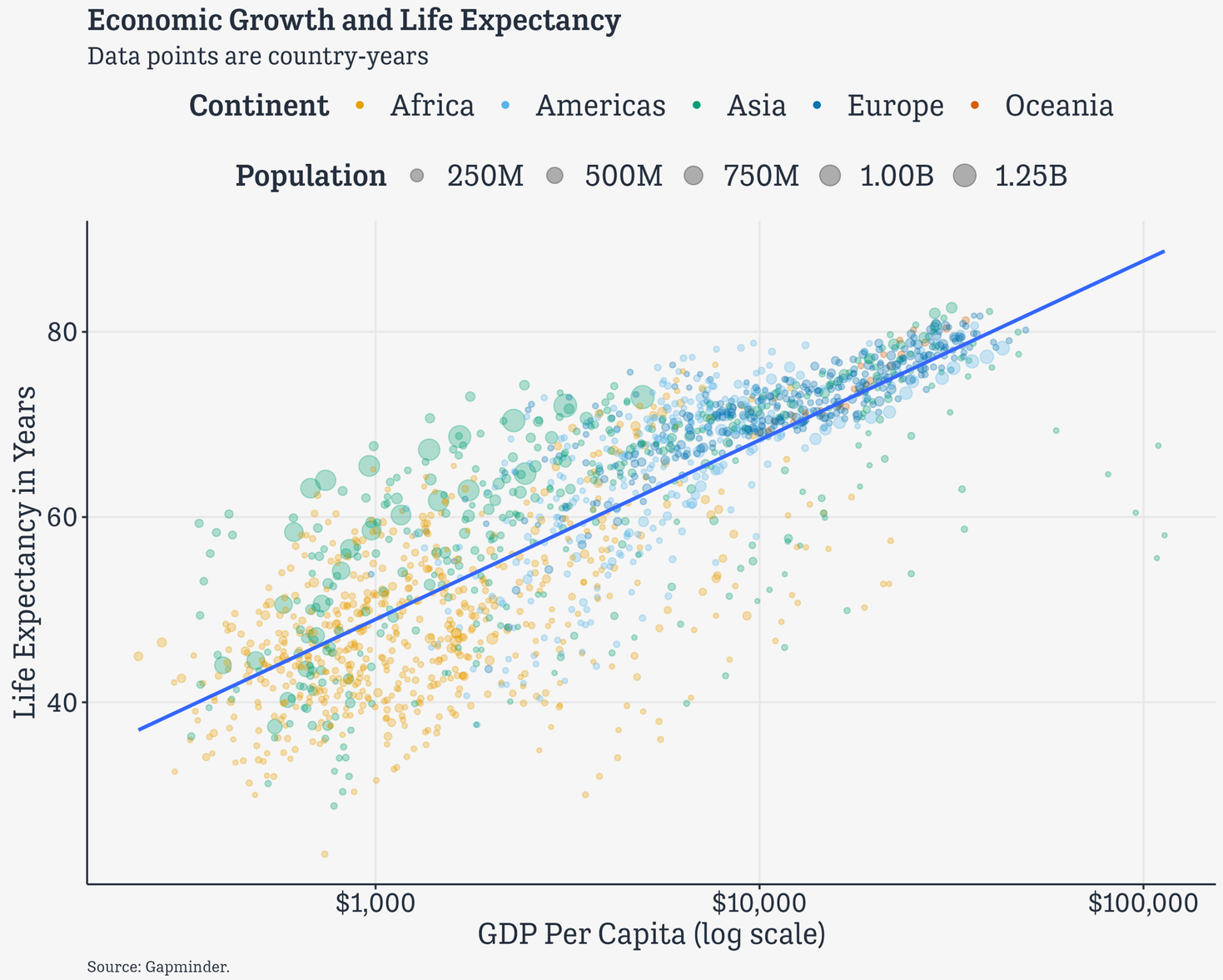
```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp)) +
  geom_point(mapping =
               aes(size = pop,
                   color = continent)) +
  geom_smooth(method = "lm", se = FALSE) +
  scale_x_log10(labels = label_dollar()) +  #<<
  scale_size(labels =
               label_number(scale_cut = cut_short_scale())) +
  labs(x = "GDP Per Capita (log scale)",
       y = "Life Expectancy in Years",
       size = "Population",
       color = "Continent",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```



Use the alpha channel

```
gapminder ▷
  ggplot(mapping = aes(x = gdpPercap,
                       y = lifeExp)) +
  geom_point(mapping =
               aes(size = pop,
                   color = continent),
             alpha = 0.3) +
  geom_smooth(method = "lm", se = FALSE) +
  scale_x_log10(labels = label_dollar()) +
  scale_size(labels =
               label_number(scale_cut = cut_short_scale())) +
  guides(color = guide_legend(override.aes = list(alpha = 1))) +
  labs(x = "GDP Per Capita (log scale)",
       y = "Life Expectancy in Years",
       size = "Population",
       color = "Continent",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```





Completed Plot

Example: Faceting

Facets are a powerful tool

```
gapminder
```

```
# A tibble: 1,704 × 6
   country     continent  year lifeExp      pop gdpPercap
   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
 1 Afghanistan Asia       1952    28.8  8425333      779.
 2 Afghanistan Asia       1957    30.3  9240934      821.
 3 Afghanistan Asia       1962    32.0 10267083      853.
 4 Afghanistan Asia       1967    34.0 11537966      836.
 5 Afghanistan Asia       1972    36.1 13079460      740.
 6 Afghanistan Asia       1977    38.4 14880372      786.
 7 Afghanistan Asia       1982    39.9 12881816      978.
 8 Afghanistan Asia       1987    40.8 13867957      852.
 9 Afghanistan Asia       1992    41.7 16317921      649.
10 Afghanistan Asia       1997    41.8 22227415      635.
# ℹ 1,694 more rows
```

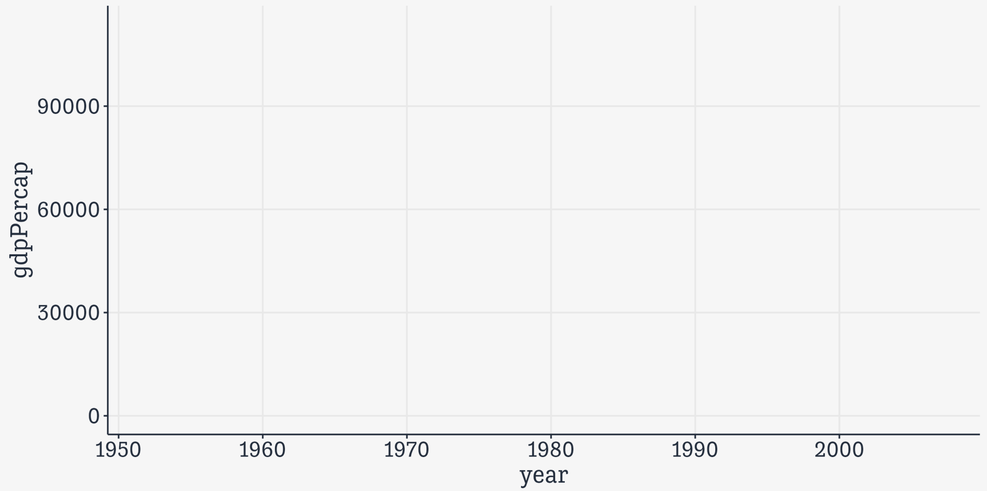
Facets are a powerful tool

```
gapminder ▷
  filter(continent ≠ "Oceania")
```

```
# A tibble: 1,680 × 6
   country     continent  year lifeExp      pop gdpPercap
   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
 1 Afghanistan Asia       1952    28.8  8425333      779.
 2 Afghanistan Asia       1957    30.3  9240934      821.
 3 Afghanistan Asia       1962    32.0 10267083      853.
 4 Afghanistan Asia       1967    34.0 11537966      836.
 5 Afghanistan Asia       1972    36.1 13079460      740.
 6 Afghanistan Asia       1977    38.4 14880372      786.
 7 Afghanistan Asia       1982    39.9 12881816      978.
 8 Afghanistan Asia       1987    40.8 13867957      852.
 9 Afghanistan Asia       1992    41.7 16317921      649.
10 Afghanistan Asia       1997    41.8 22227415      635.
# ℹ 1,670 more rows
```

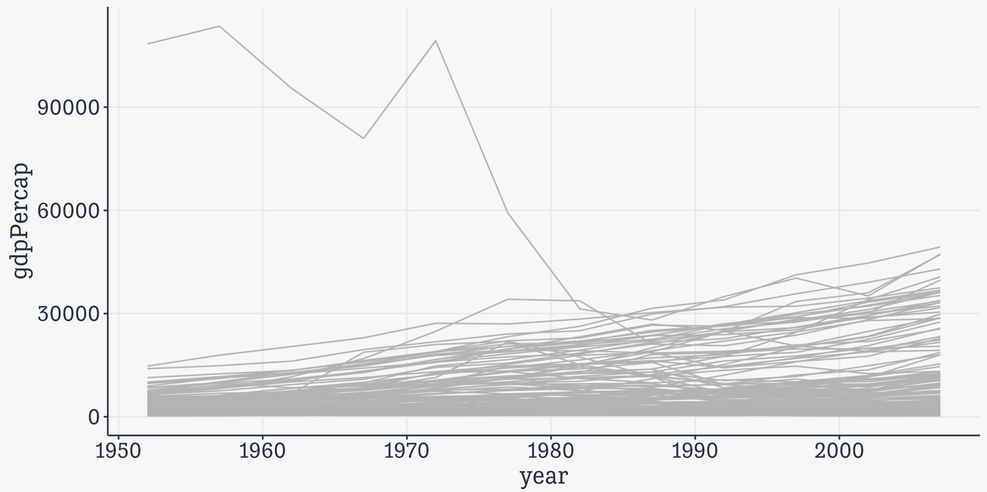
Facets are a powerful tool

```
gapminder ▷
  filter(continent ≠ "Oceania") ▷
  ggplot(mapping = aes(x = year,
                       y = gdpPercap))
```



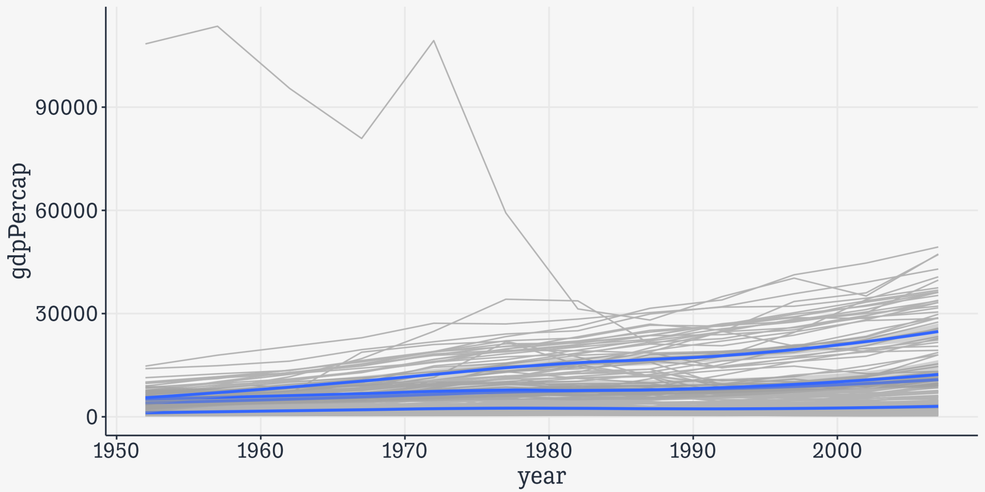
Facets are a powerful tool

```
gapminder ▷
  filter(continent ≠ "Oceania") ▷
  ggplot(mapping = aes(x = year,
                       y = gdpPercap)) +
  geom_line(mapping = aes(group = country),
            color = "gray70")
```



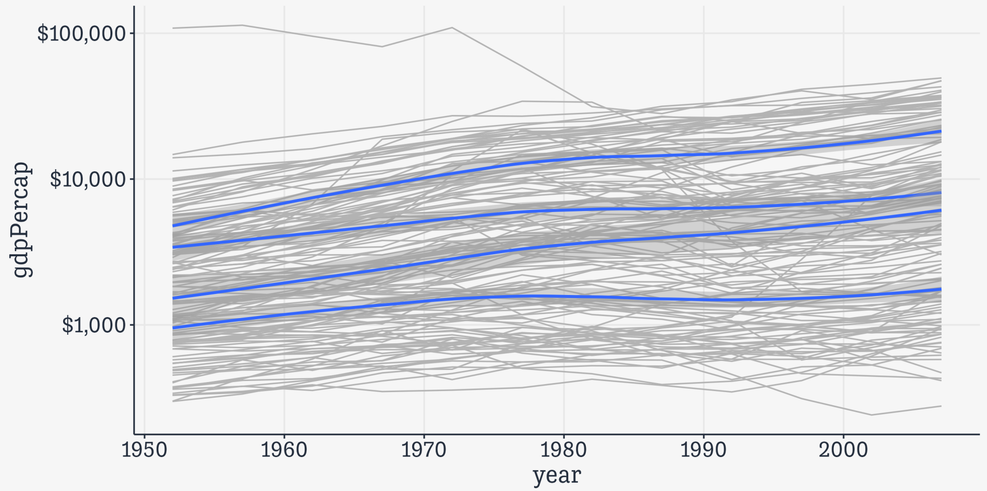
Facets are a powerful tool

```
gapminder ▷
  filter(continent ≠ "Oceania") ▷
  ggplot(mapping = aes(x = year,
                       y = gdpPercap)) +
  geom_line(mapping = aes(group = country),
            color = "gray70") +
  geom_smooth(mapping = aes(group = continent),
              method = "loess")
```



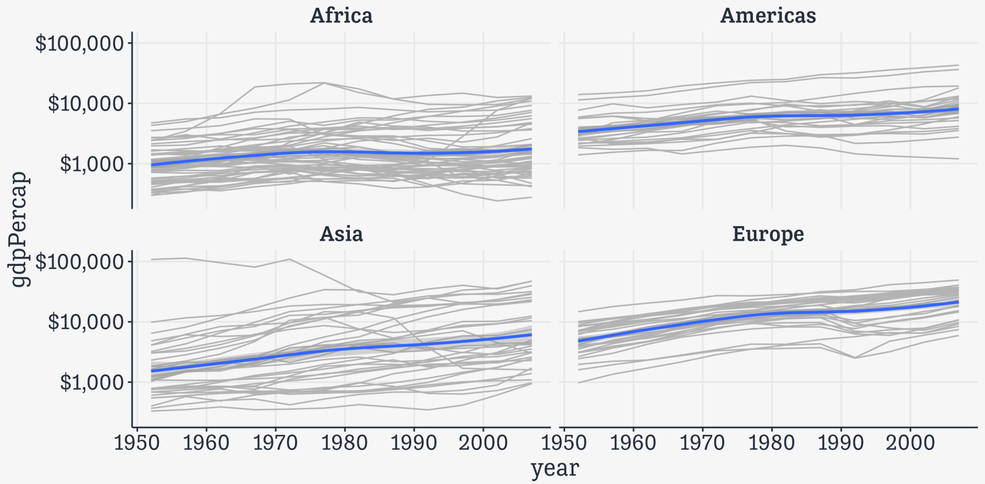
Facets are a powerful tool

```
gapminder ▷
  filter(continent ≠ "Oceania") ▷
  ggplot(mapping = aes(x = year,
                       y = gdpPercap)) +
  geom_line(mapping = aes(group = country),
            color = "gray70") +
  geom_smooth(mapping = aes(group = continent),
              method = "loess") +
  scale_y_log10(labels = label_dollar())
```



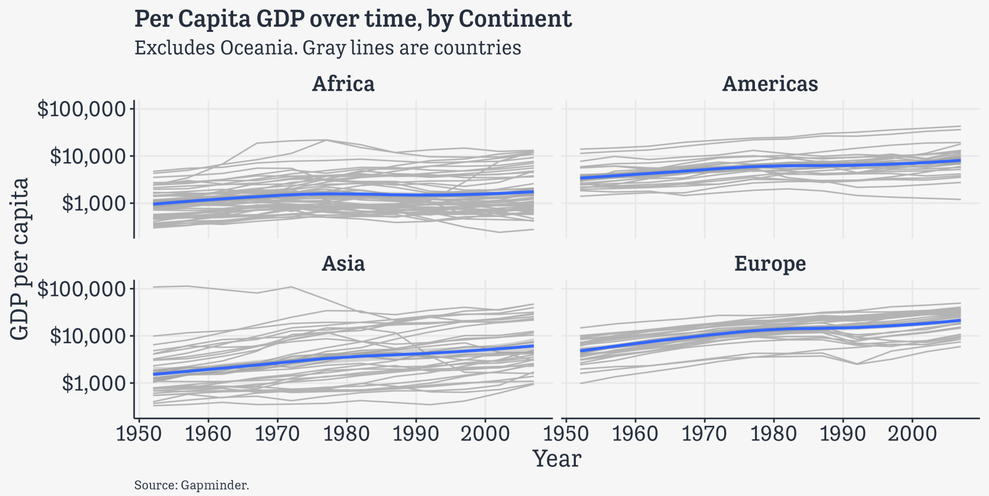
Facets are a powerful tool

```
gapminder |>
  filter(continent != "Oceania") |>
  ggplot(mapping = aes(x = year,
                       y = gdpPercap)) +
  geom_line(mapping = aes(group = country),
            color = "gray70") +
  geom_smooth(mapping = aes(group = continent),
              method = "loess") +
  scale_y_log10(labels = label_dollar()) +
  facet_wrap(~ continent)
```



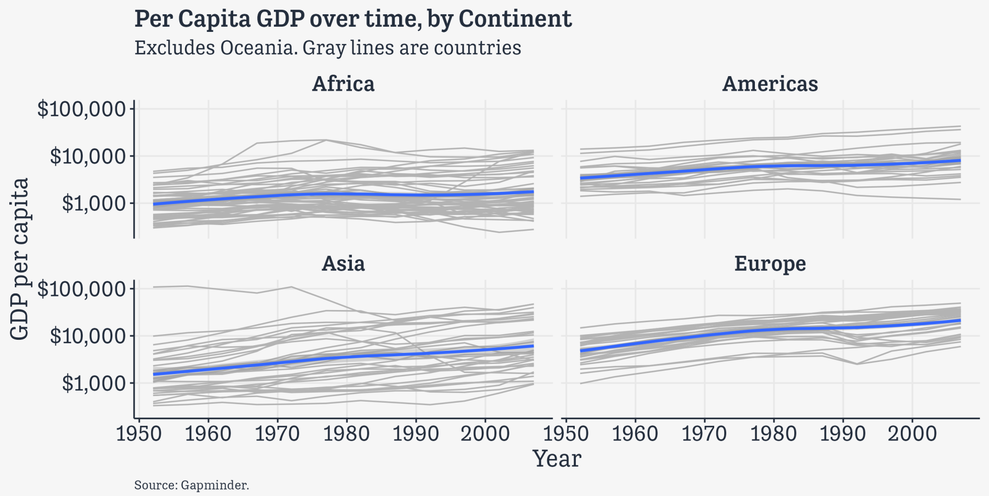
Facets are a powerful tool

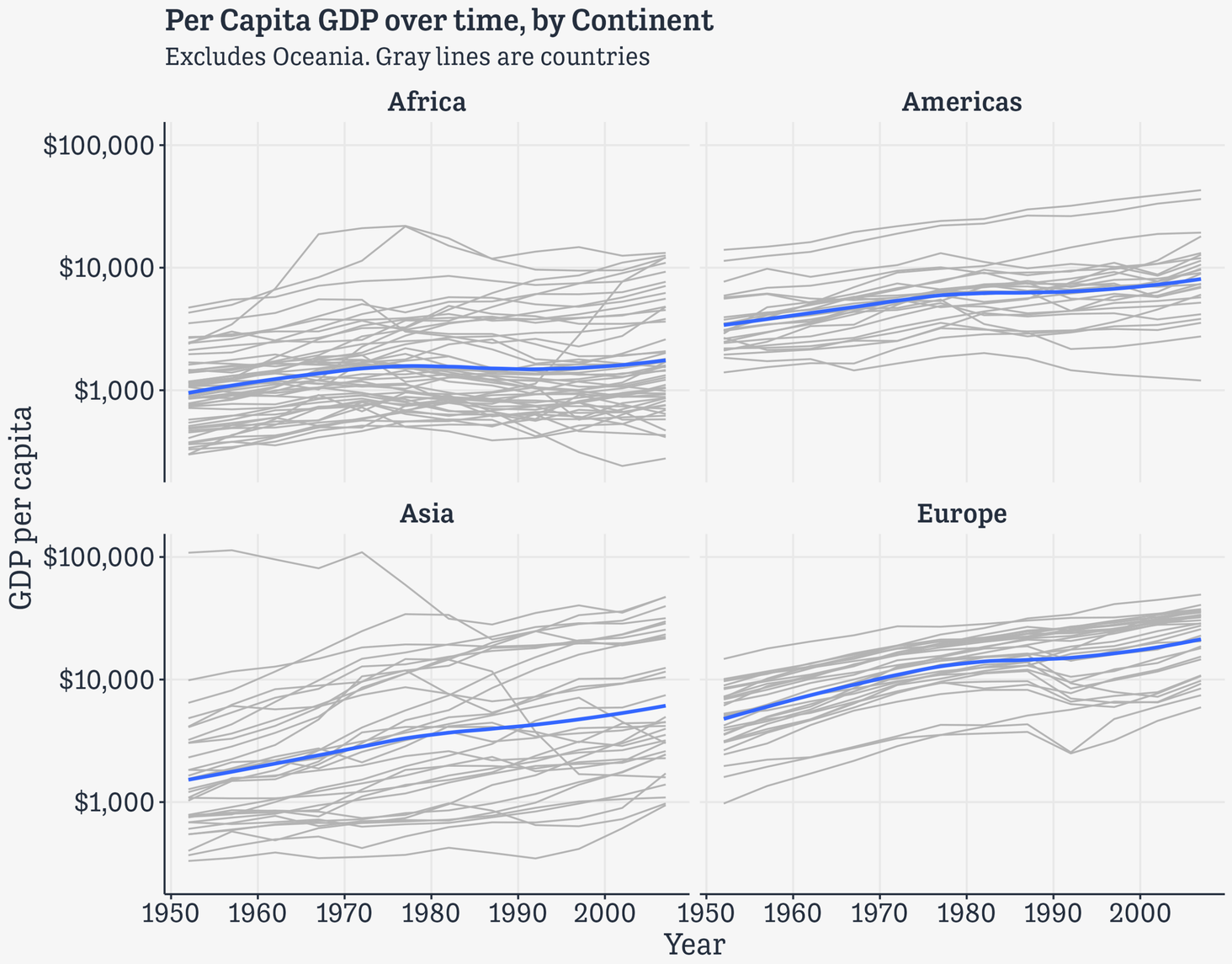
```
gapminder ▷
  filter(continent ≠ "Oceania") ▷
  ggplot(mapping = aes(x = year,
                       y = gdpPercap)) +
  geom_line(mapping = aes(group = country),
            color = "gray70") +
  geom_smooth(mapping = aes(group = continent),
              method = "loess") +
  scale_y_log10(labels = label_dollar()) +
  facet_wrap(~ continent) +
  labs(x = "Year",
         y = "GDP per capita",
         title = "Per Capita GDP over time, by Continent",
         subtitle = "Excludes Oceania. Gray lines are countr
         caption = "Source: Gapminder.")
```



Facets are a powerful tool

```
gapminder ▷
  filter(continent ≠ "Oceania") ▷
  ggplot(mapping = aes(x = year,
                       y = gdpPercap)) +
  geom_line(mapping = aes(group = country),
            color = "gray70") +
  geom_smooth(mapping = aes(group = continent),
              method = "loess") +
  scale_y_log10(labels = label_dollar()) +
  facet_wrap(~ continent) +
  labs(x = "Year",
         y = "GDP per capita",
         title = "Per Capita GDP over time, by Continent",
         subtitle = "Excludes Oceania. Gray lines are countr
         caption = "Source: Gapminder.")
```





Completed Plot

Facets are often better than Guides

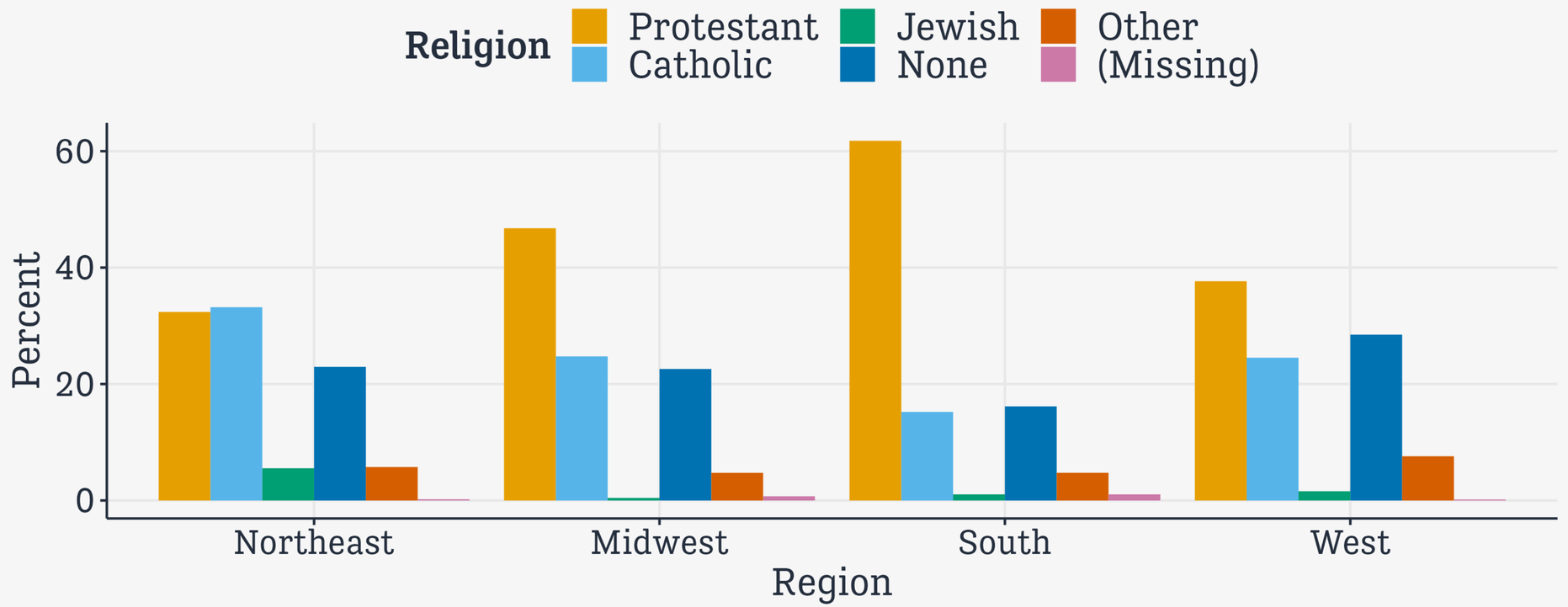
Consider a basic crosstab

```
rel_by_region ▷ select(-n) ▷
  pivot_wider(names_from = religion, values_from = pct) ▷
  janitor::adorn_totals(where = "col")
```

```
bigregion Protestant Catholic Jewish None Other (Missing) Total
Northeast         32       33   5.53   23   5.7      0.20   100
  Midwest         47       25   0.43   23   4.7      0.72   100
    South         62       15   1.05   16   4.8      1.05   100
     West         38       25   1.58   28   7.6      0.16   100
```

We might write ...

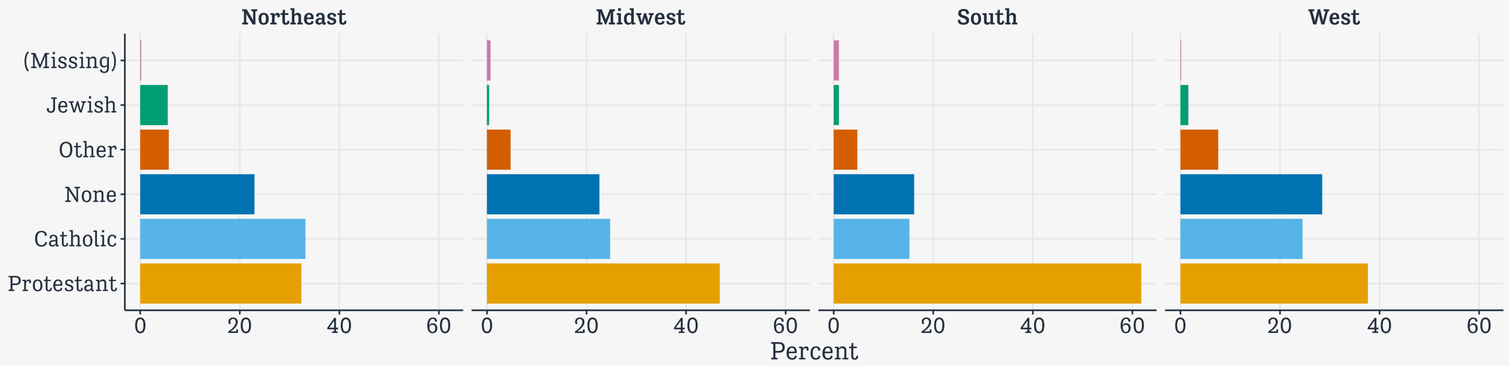
```
rel_by_region ▷
  ggplot(mapping = aes(x = bigregion, y = pct, fill = religion)) +
  geom_col(position = "dodge") + labs(x = "Region", y = "Percent", fill = "Religion")
```



Is this an effective graph? Not really!

Try faceting instead

```
rel_by_region ▷
  ggplot(mapping = aes(x = pct, y = reorder(religion, -pct), fill = religion)) +
  geom_col() + guides(fill = "none") +
  facet_wrap(~ bigregion, nrow = 1) + labs(x = "Percent", y = NULL)
```



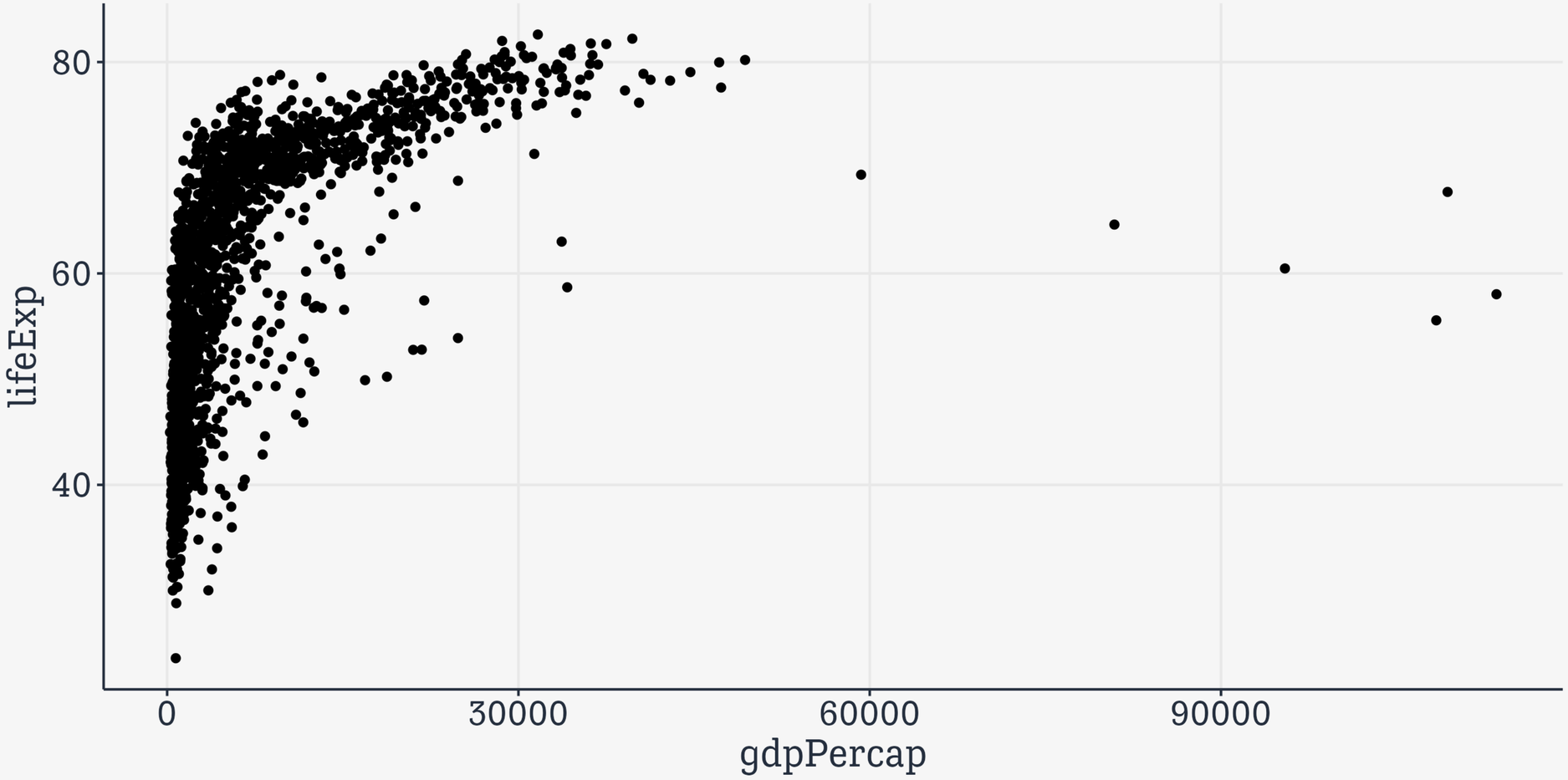
Facets reduce the number of guides and legends the viewer needs to consult. Notice how this graph no longer requires the bars to be in color.

Reorder your categories by the value of what they summarize. This is true for all kinds of summary graphs involving categorical variables, not just facets.

Try putting the categories on the y-axis. This is a very useful trick. It makes graphs compact and table-like, and avoids x-axis labels being in the wrong orientation, or you having to figure out how to put them at an angle.

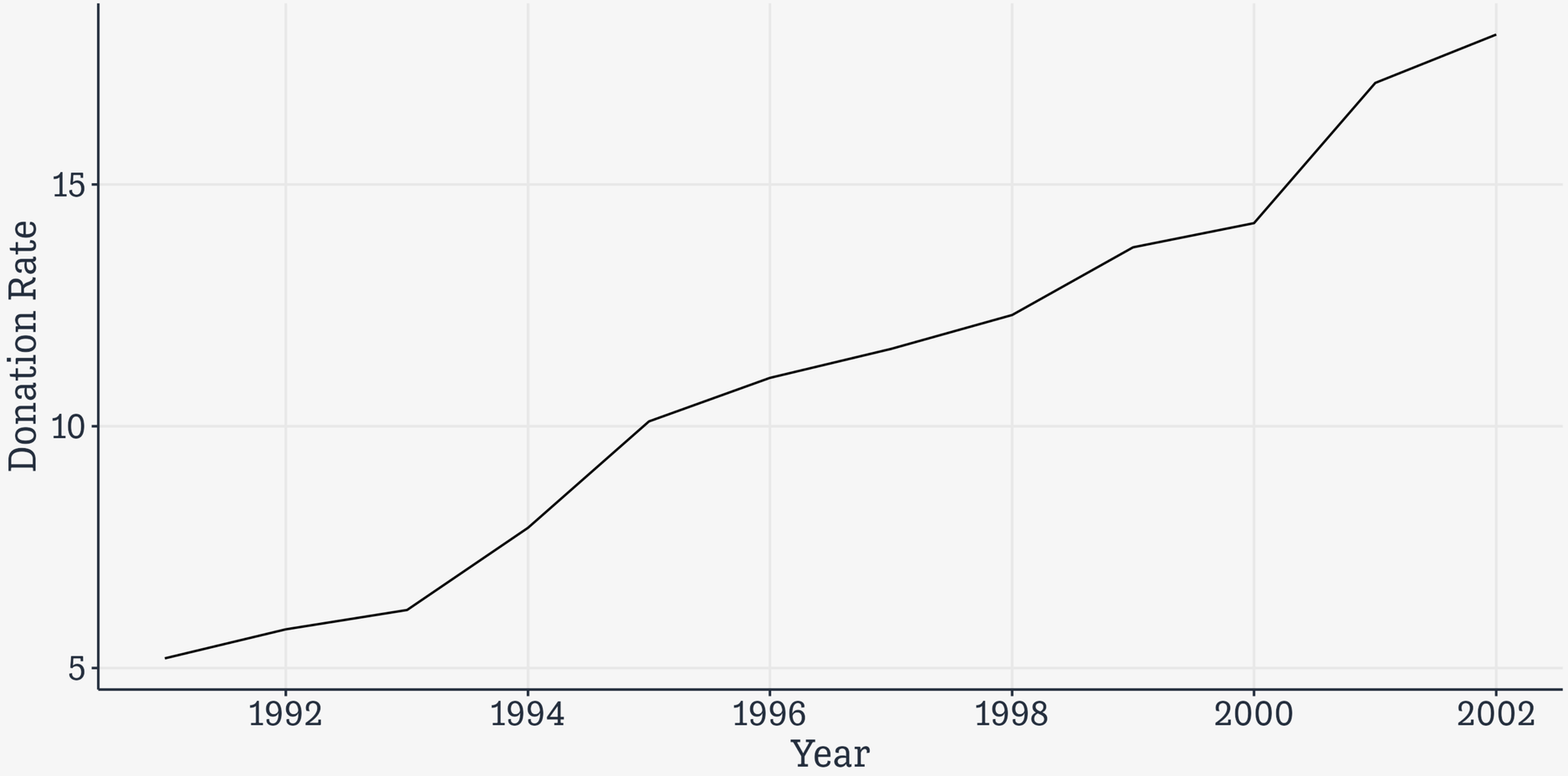
Graphing in practice

Workhorses



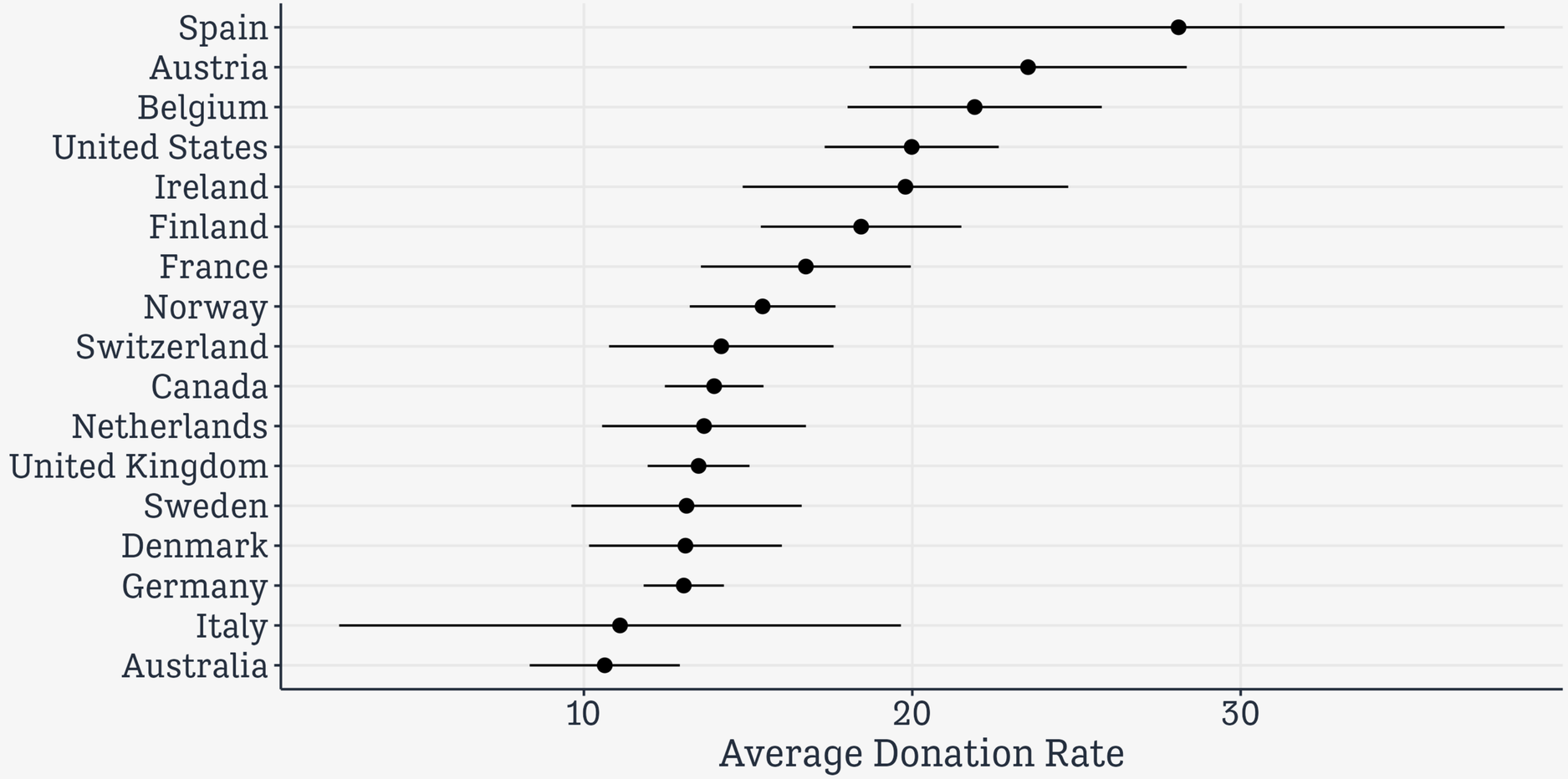
A scatterplot

Workhorses



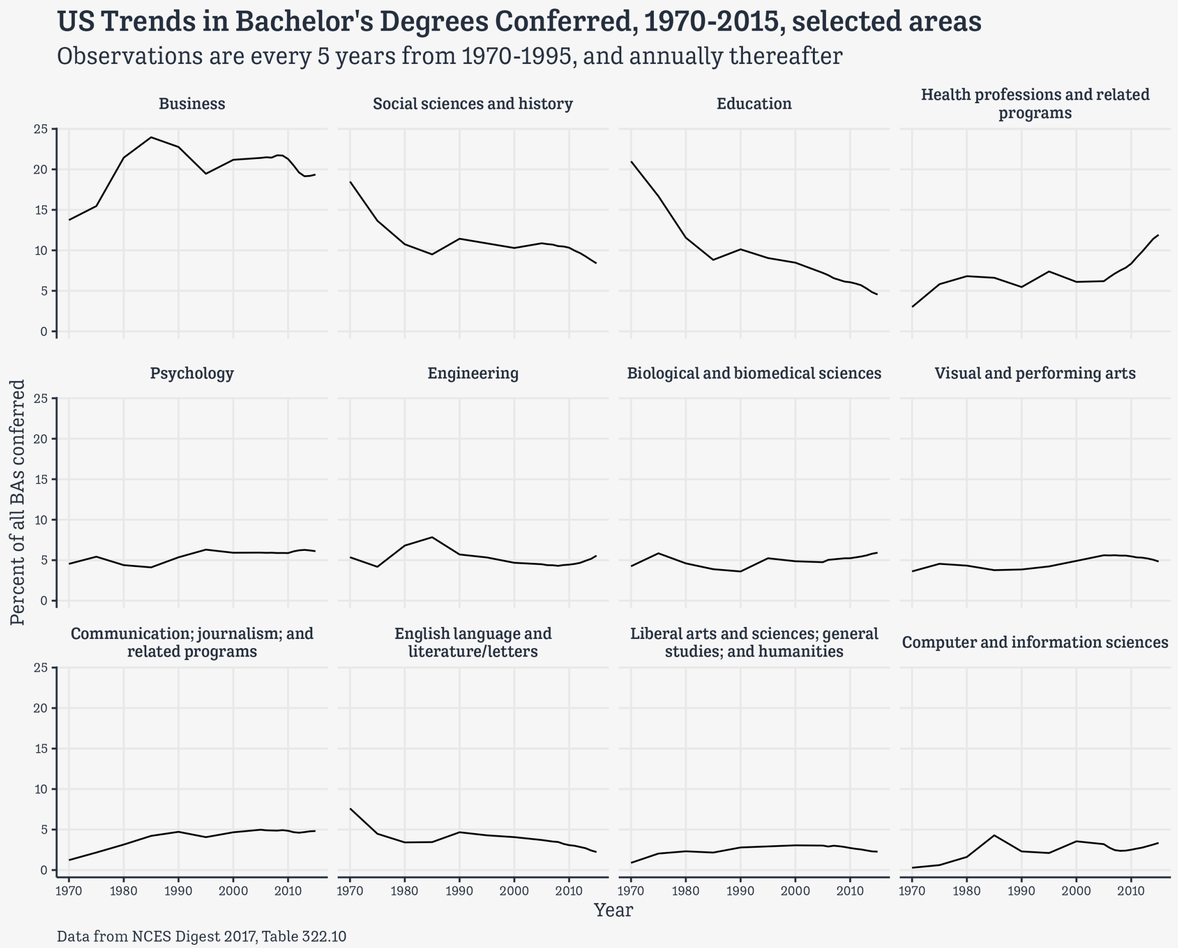
Trendlines

Workhorses



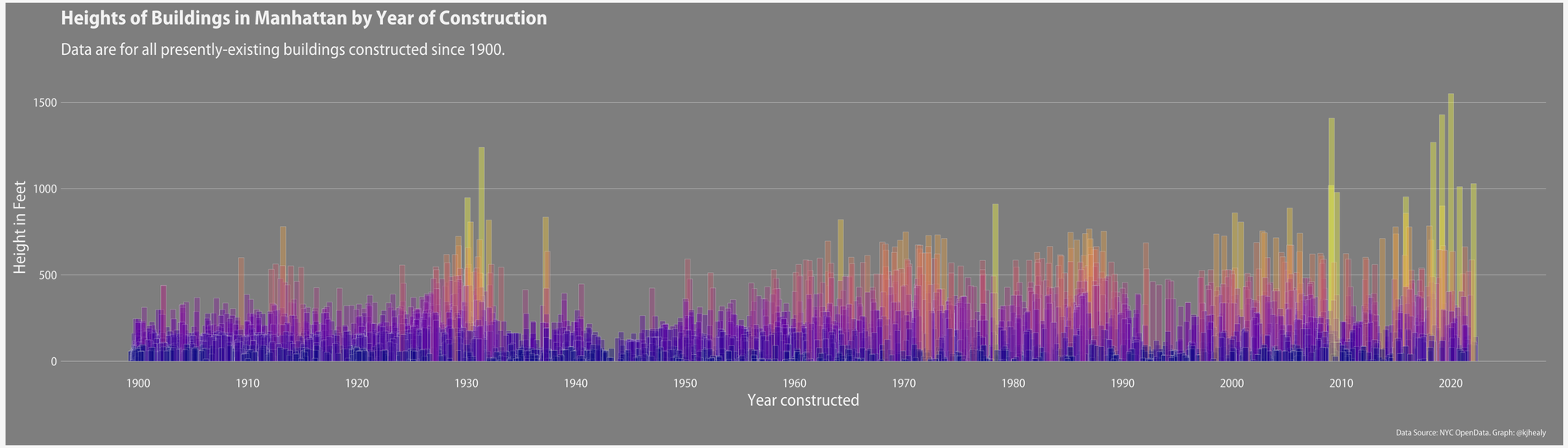
Point-and-range across categories

Workhorses



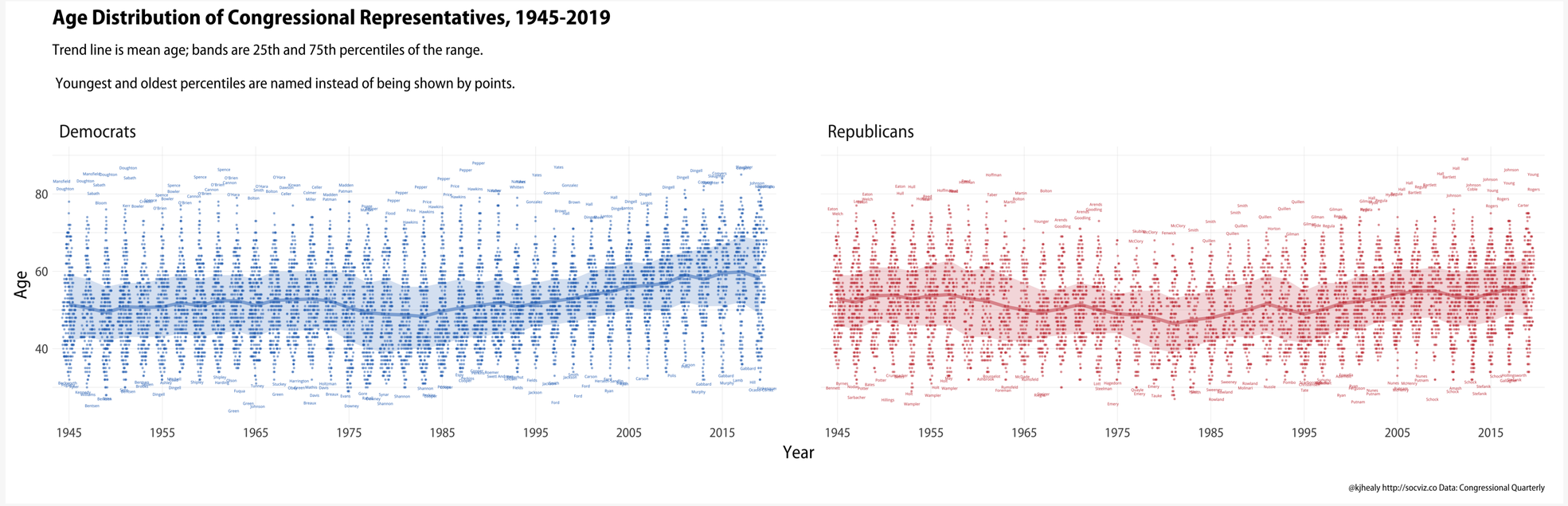
Faceting

Show Ponies



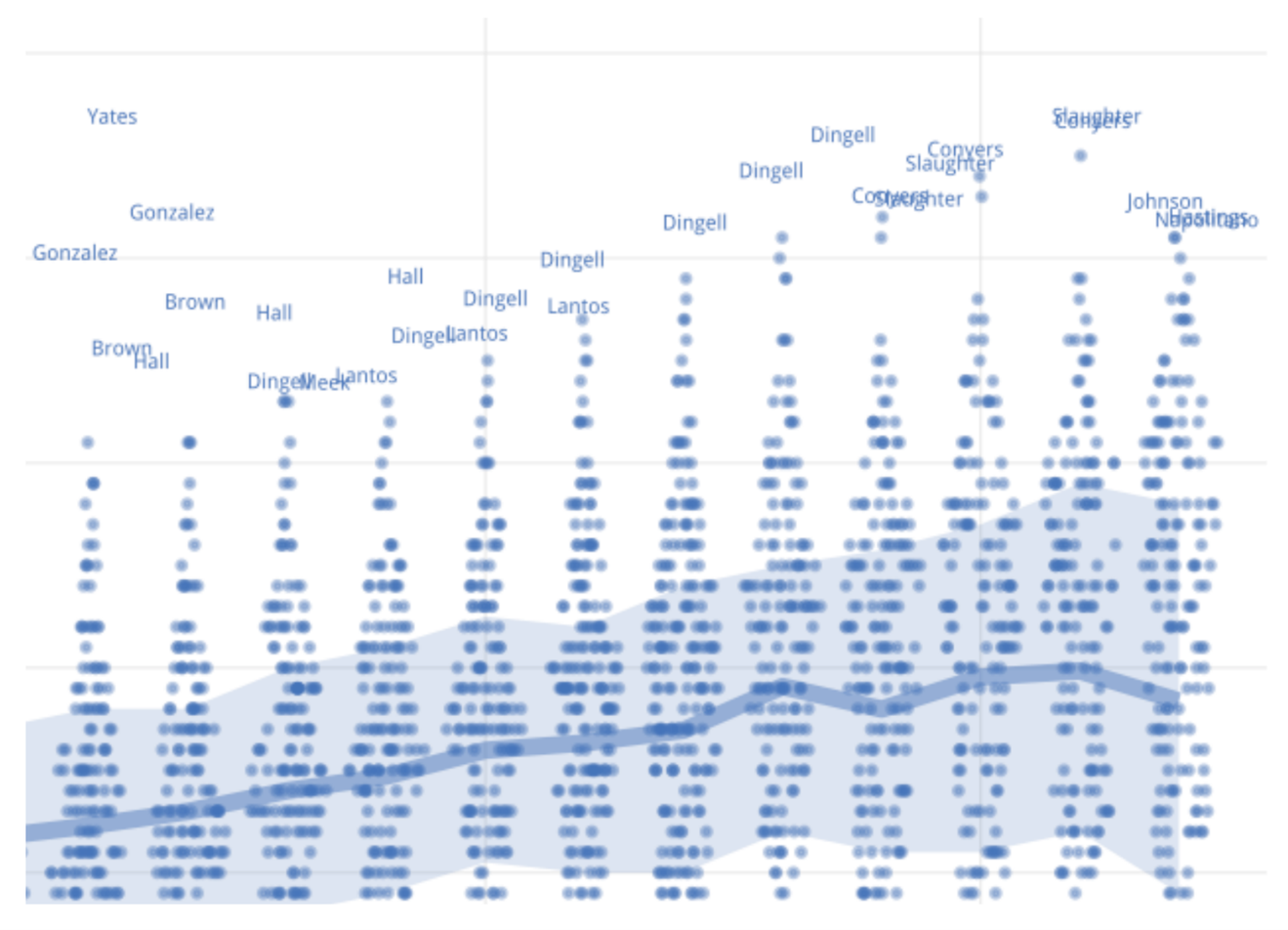
Manhattan Building Heights by Year of Construction

Show Ponies



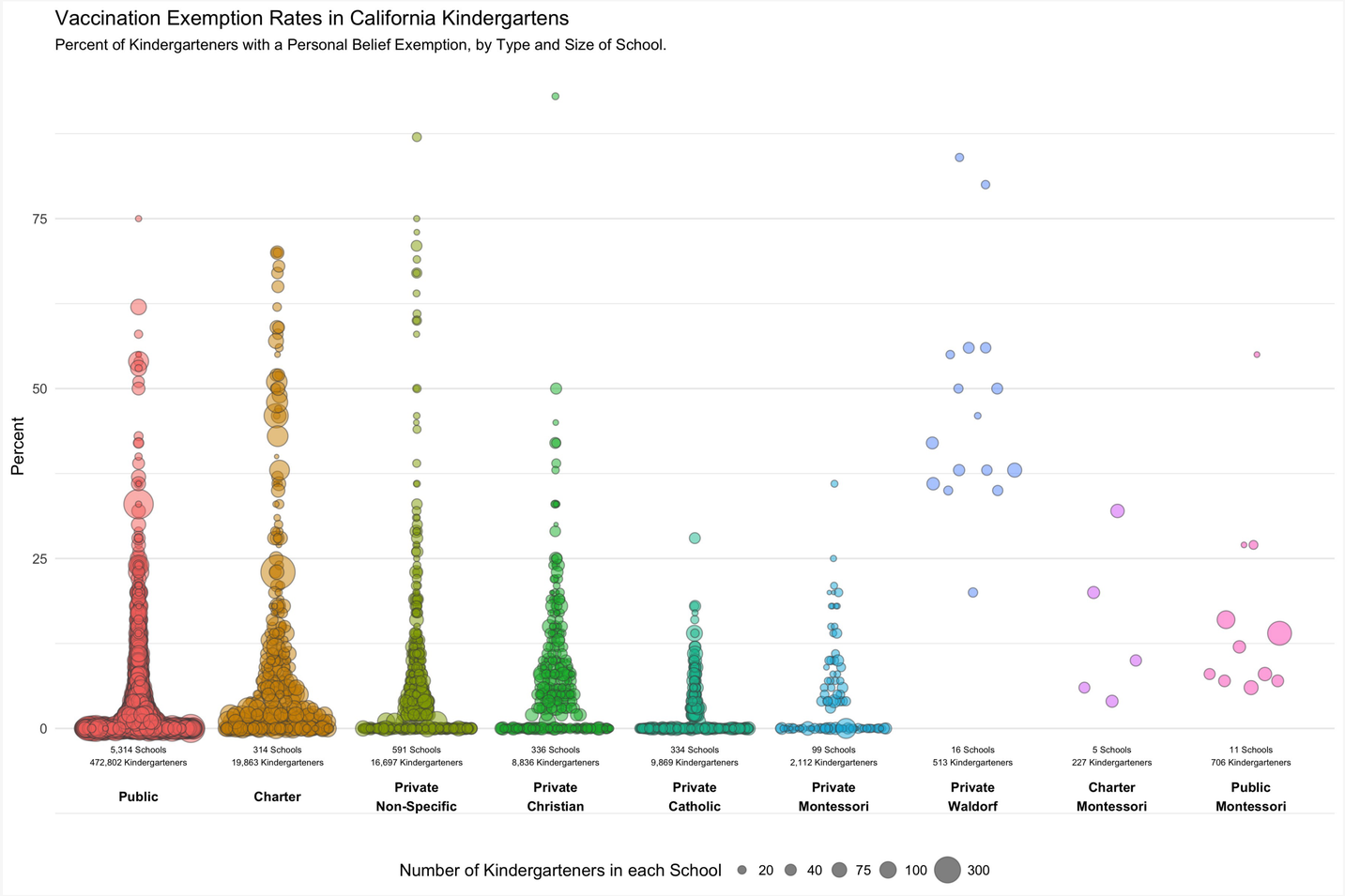
Congressional comparison

Show Ponies



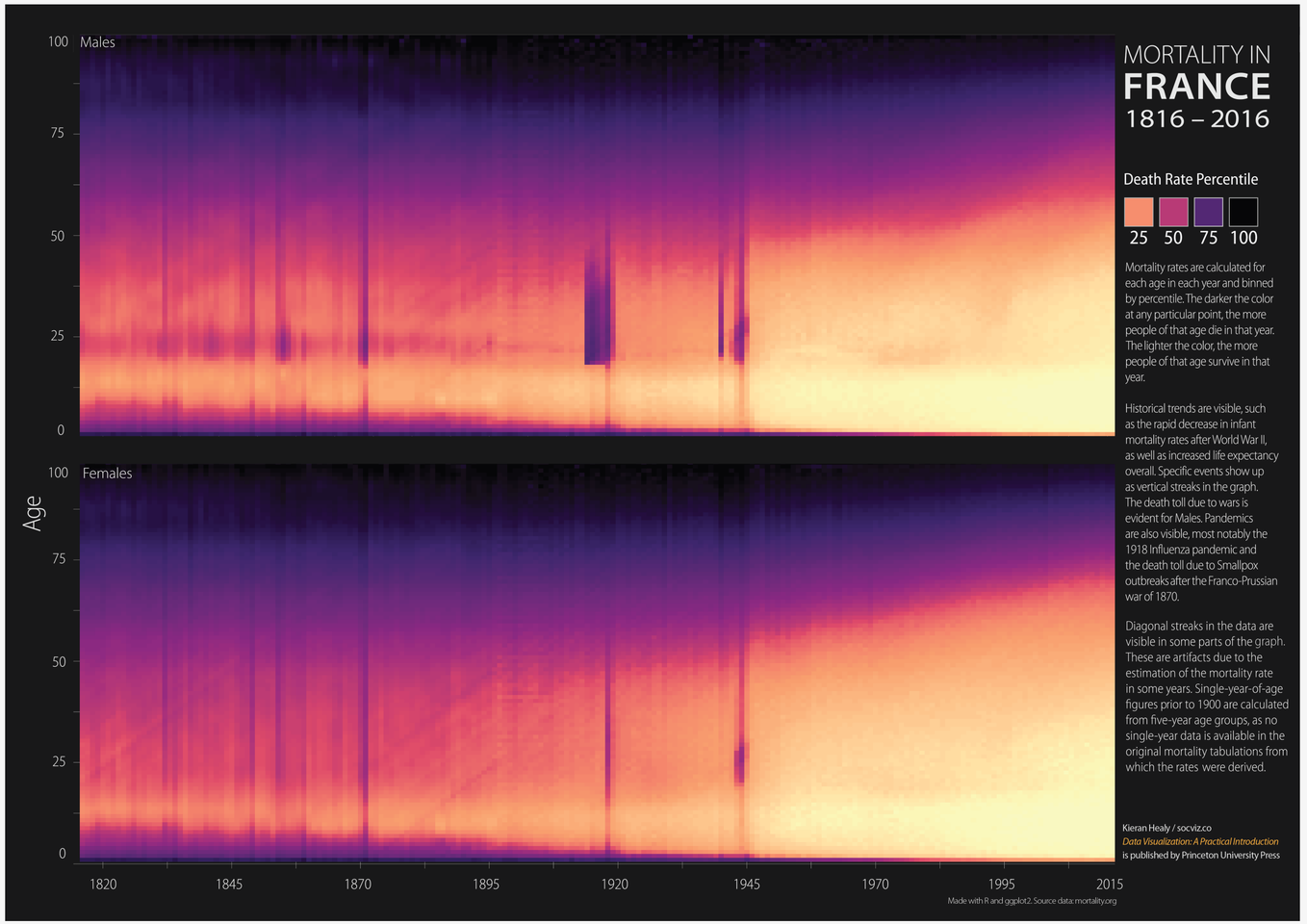
Several plots at once

Show Ponies



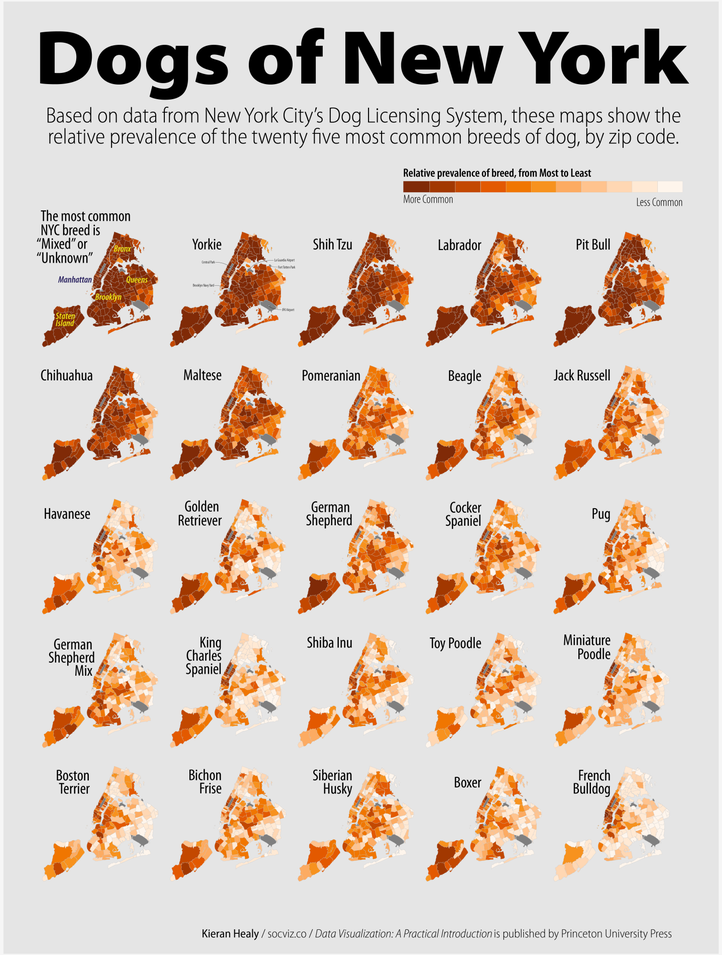
Beeswarm plot

Show Ponies



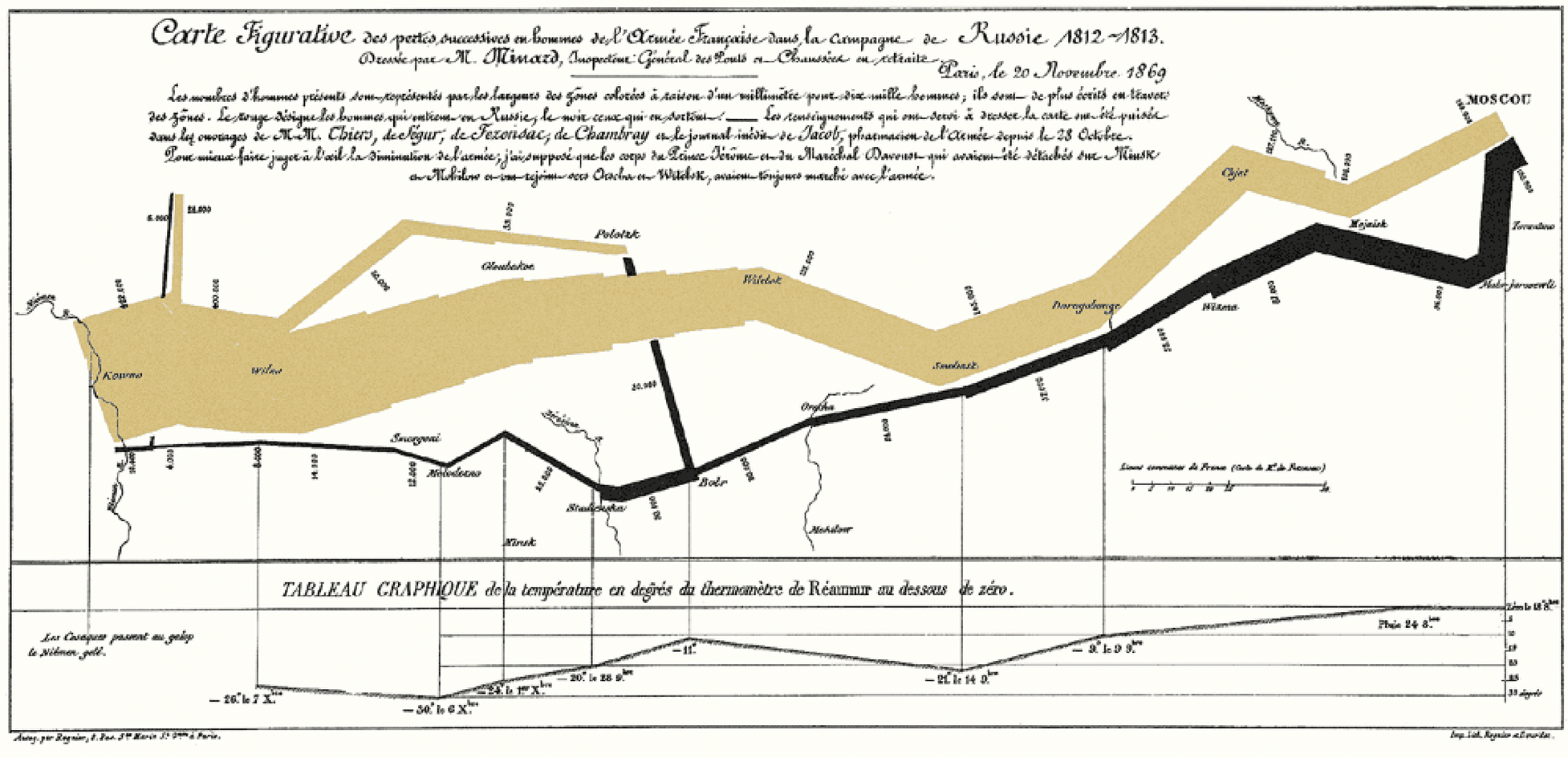
Lexis Surface

Show Ponies



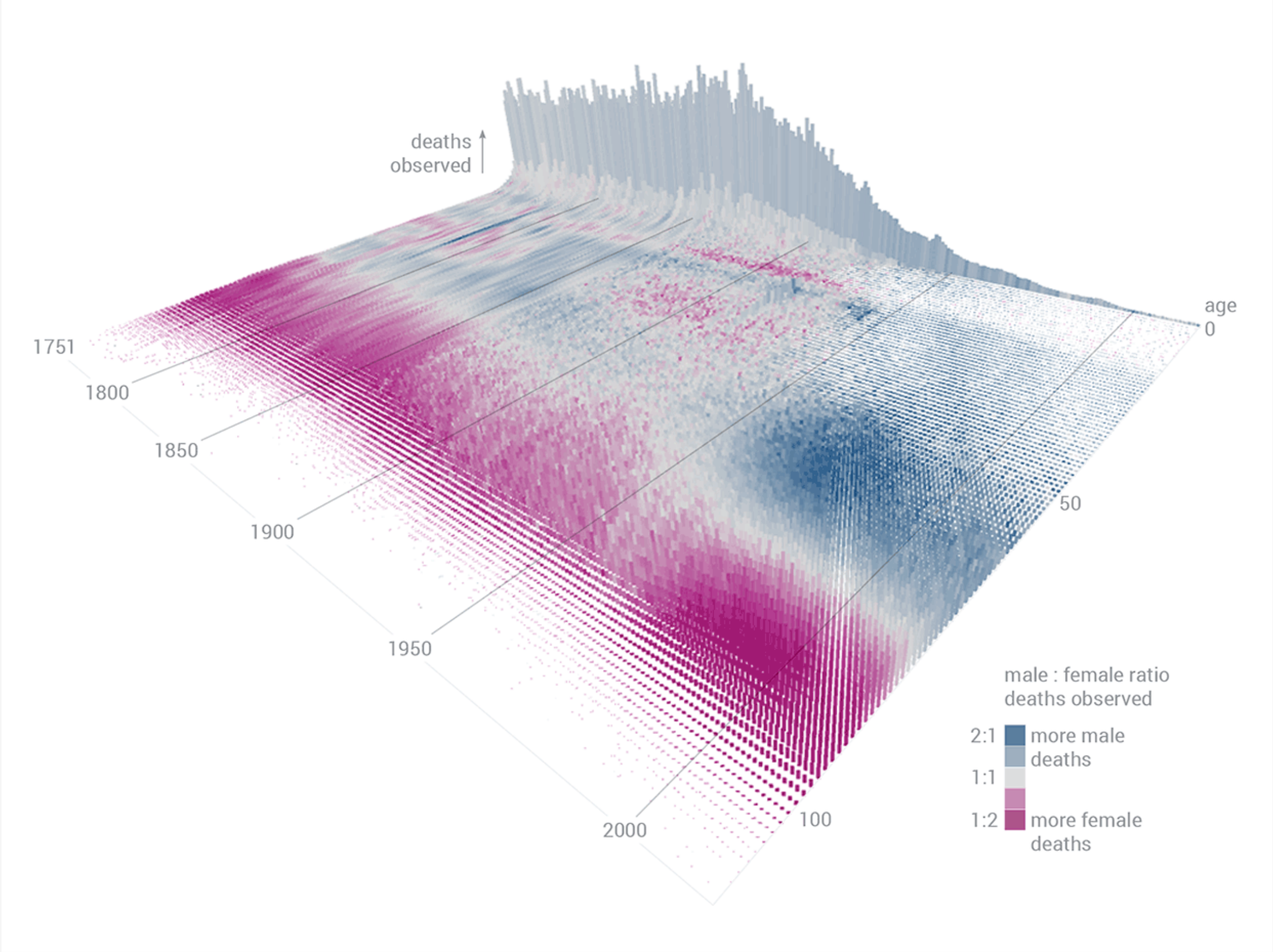
Faceted maps

Unicorns …



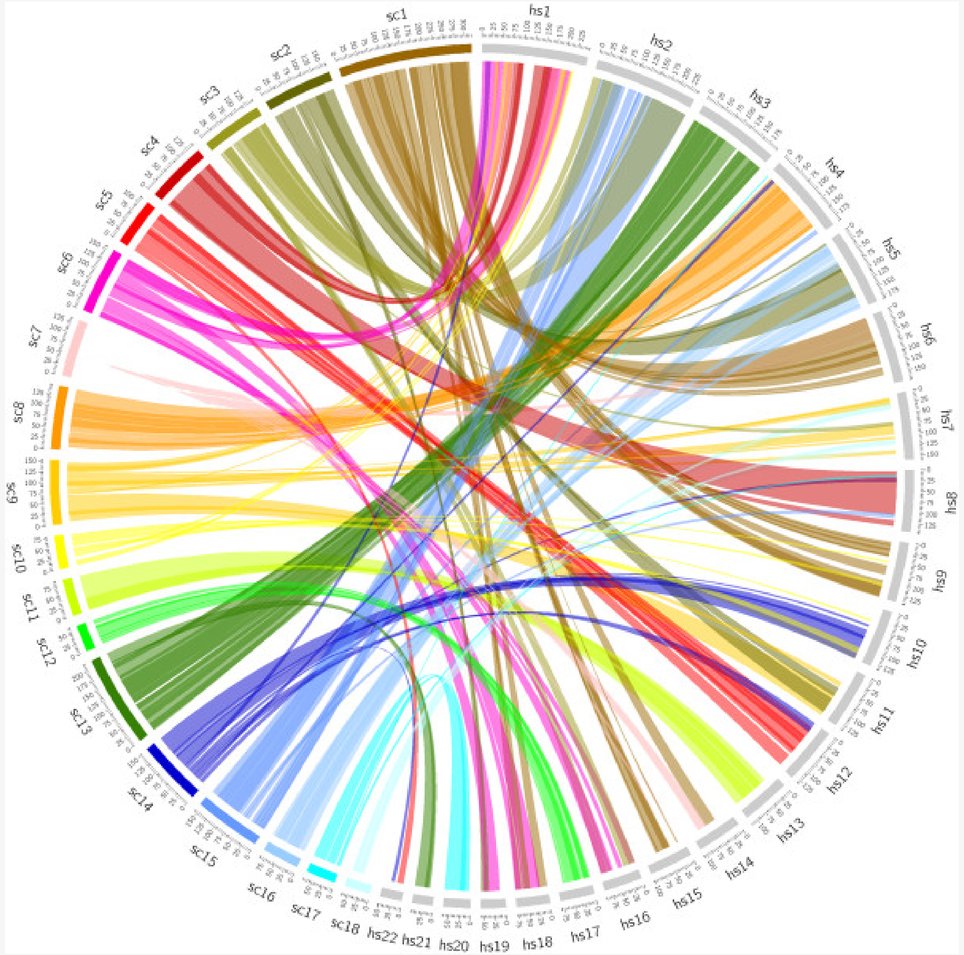
The inevitable Minard

Unicorns ...



Swedish mortality

… or monsters



Network chords

Summary

Your toolkit can’t make you honest, or correct

But you should still look at (and show) your data

Consider who your audience is, including when it's yourself

Present findings in substantive terms

Show degrees of confidence or uncertainty

Show the data when you can

But remember, these points apply just as well to presenting data in *any* format: tables, models, text, whatever.

Graphs are not special in this respect!

Learn More

https://socviz.co

Full draft of *Data Visualization: A Practical Introduction* (Princeton 2019)

https://kieranhealy.org

Examples, R data packages (including for the GSS), code