

# Comparisons

## Session 8

PMAP 8551/4551: Data Visualization with R  
Andrew Young School of Policy Studies  
Spring 2026

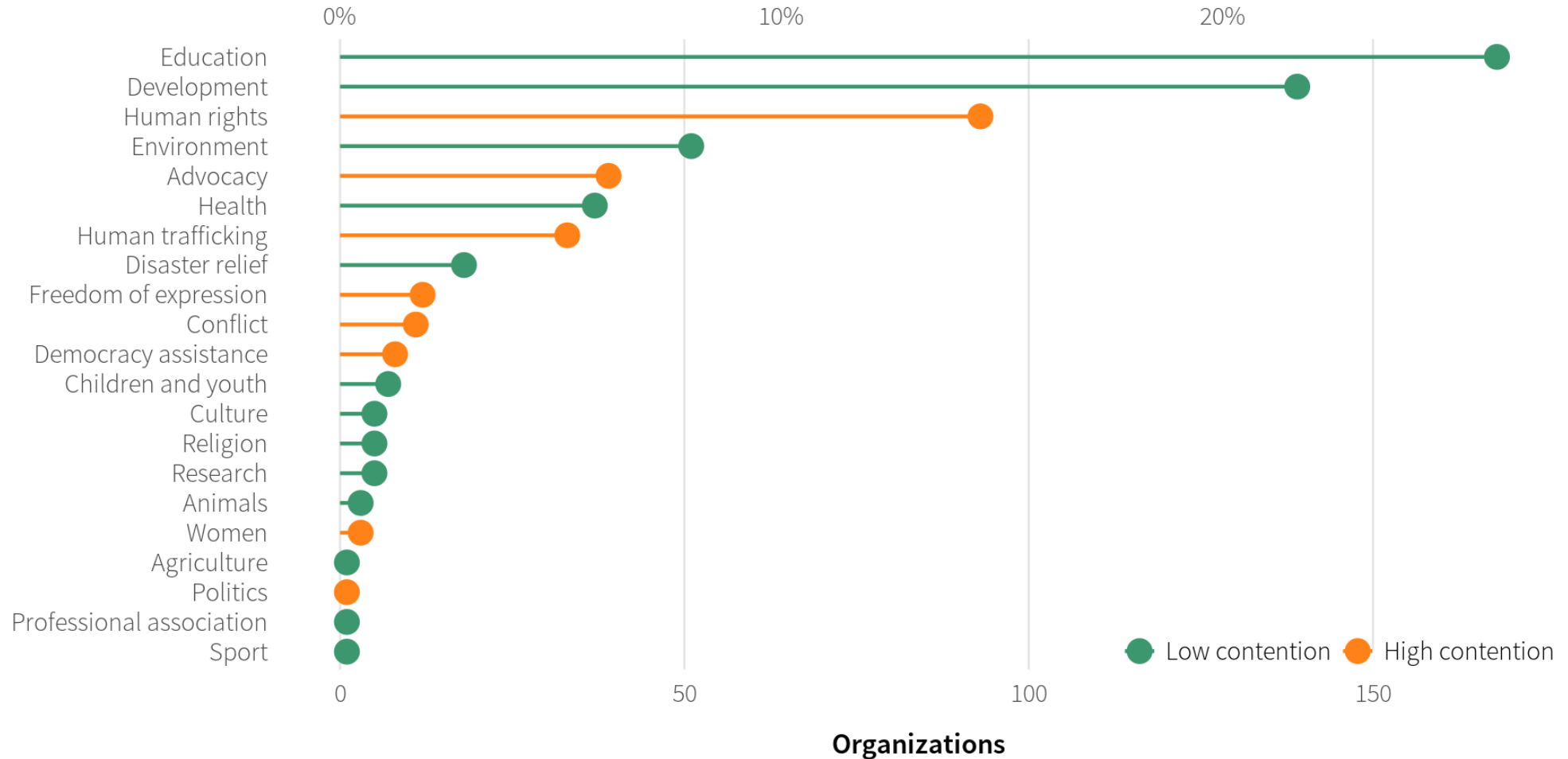
# Plan for today

**Visualizing comparisons**

**Reproducible examples**

# Visualizing comparisons

# Lollipops and bars





# Small multiples

## How Trump compares with past presidents

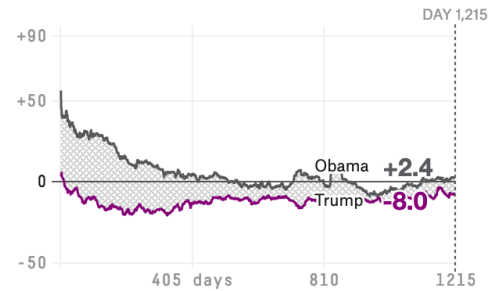
○ Approval rating ○ Disapproval rating ● Net approval

1,215 days

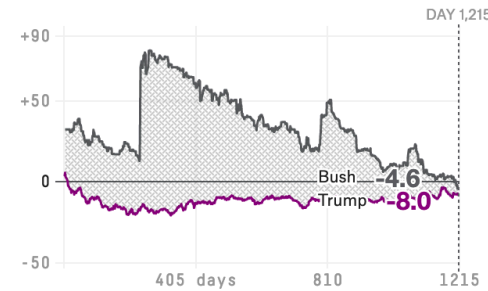
4 years

8 years

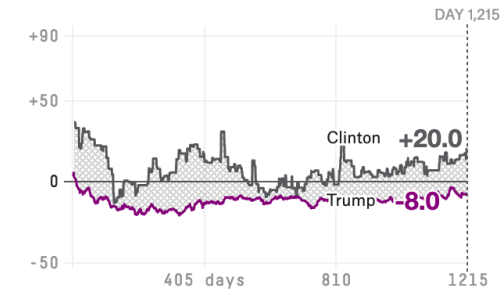
Barack Obama 2009-17



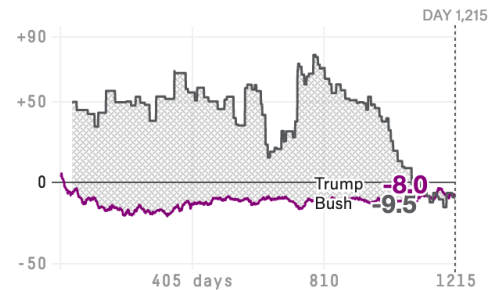
George W. Bush 2001-09



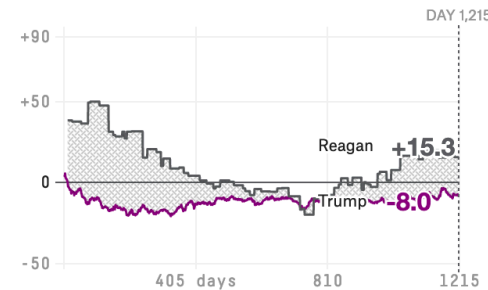
Bill Clinton 1993-2001



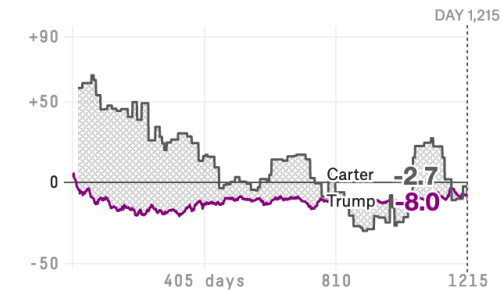
George H.W. Bush 1989-93



Ronald Reagan 1981-89

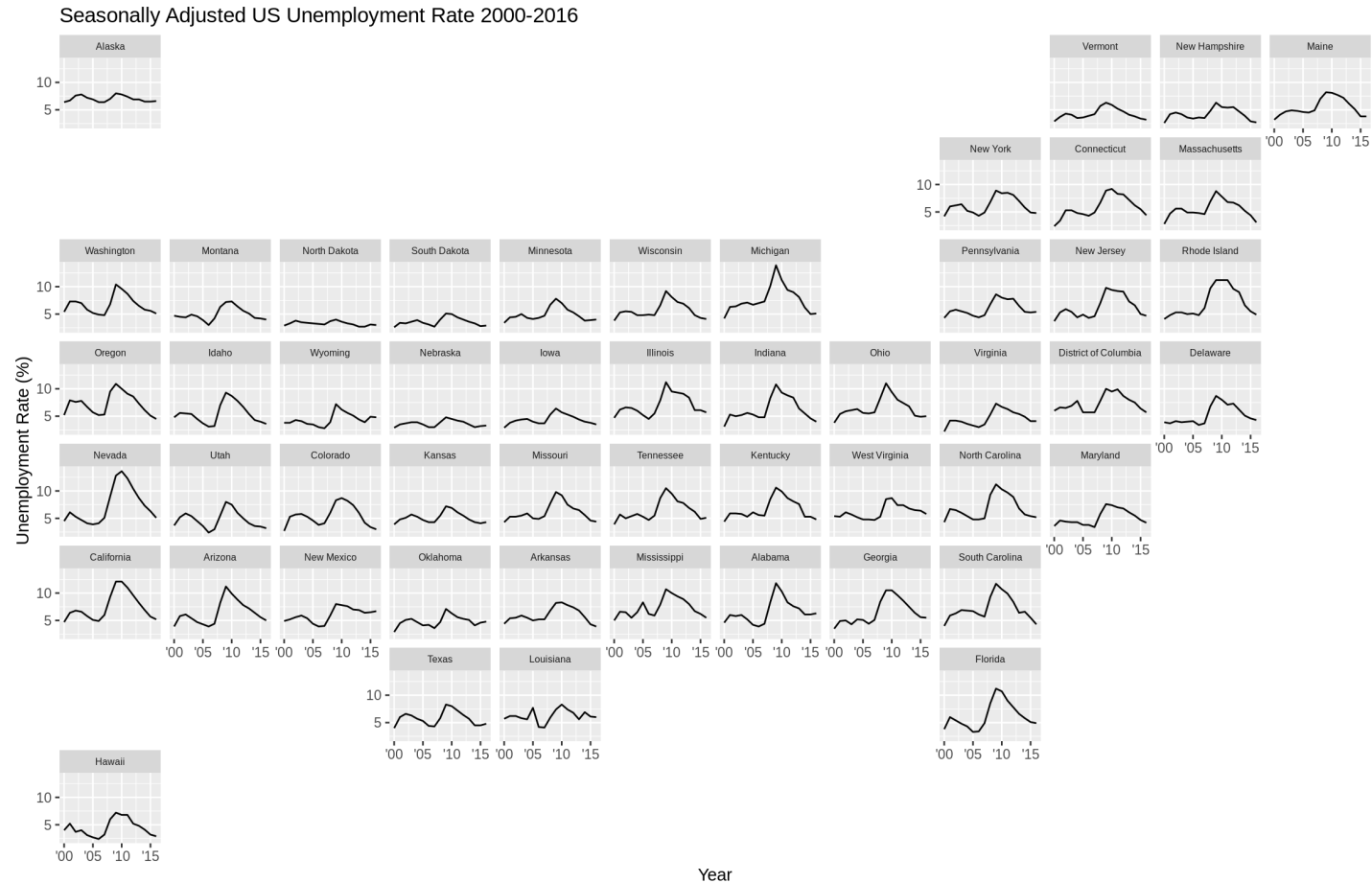


Jimmy Carter 1977-81



FiveThirtyEight, Trump approval ratings


# Small multiples with larger shapes



`facet_geo()` in the **geofacet** package

# Sparklines

Mauricio Pochettino has lead Spurs on their best run **8TH**  **2ND** in 24 years of the Premier League

Alibaba stock is at 5 yr high **93.89**  **152.11** as of July 2017

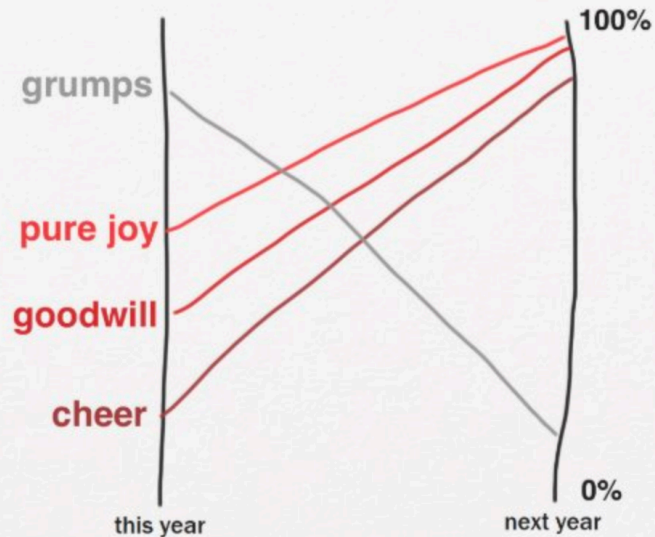
The FTSE100 Brexit bounce **5562**  **7501** continues one year on from the vote last summer





# Slopegraphs

**May your New Year see  
a significant increase in  
all the good stuff.**



# Slopegraphs

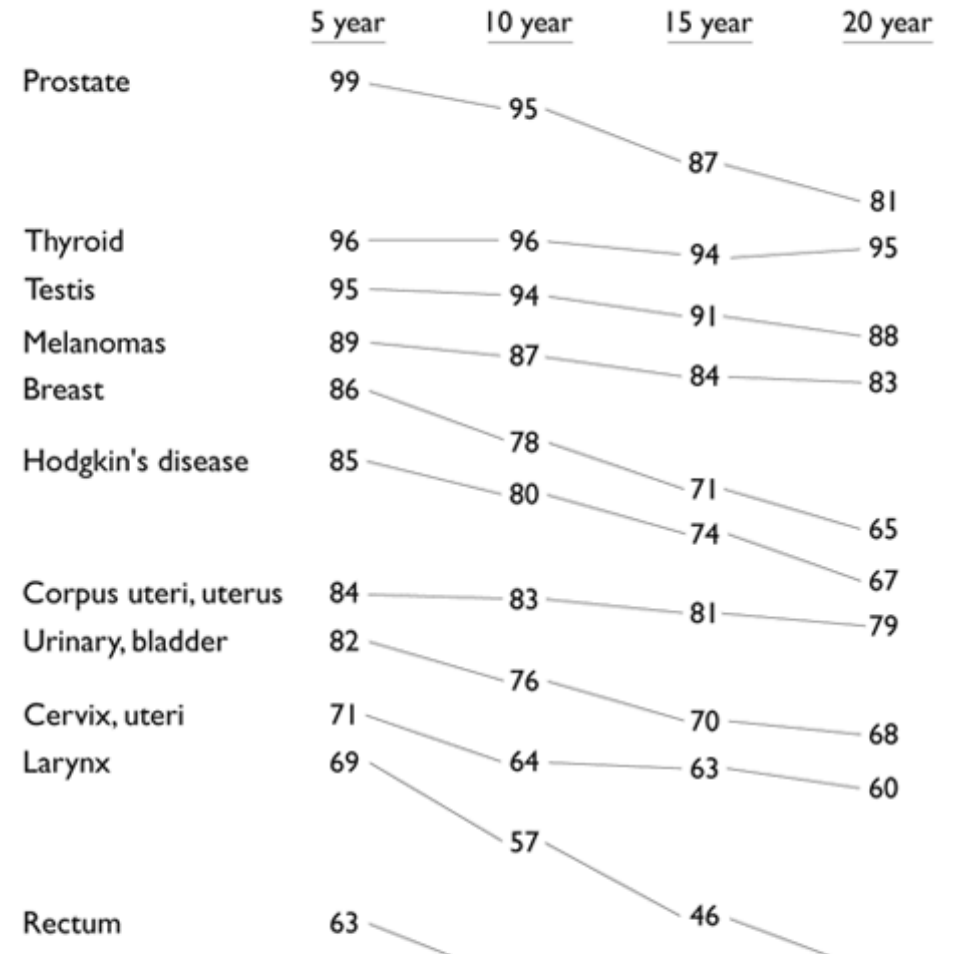


# Slopegraphs

Estimates of relative survival rates, by cancer site

	% survival rates and their standard errors							
	5 year		10 year		15 year		20 year	
Prostate	98.8	0.4	95.2	0.9	87.1	1.7	81.1	3.0
Thyroid	96.0	0.8	95.8	1.2	94.0	1.6	95.4	2.1
Testis	94.7	1.1	94.0	1.3	91.1	1.8	88.2	2.3
Melanomas	89.0	0.8	86.7	1.1	83.5	1.5	82.8	1.9
Breast	86.4	0.4	78.3	0.6	71.3	0.7	65.0	1.0
Hodgkin's disease	85.1	1.7	79.8	2.0	73.8	2.4	67.1	2.8
Corpus uteri, uterus	84.3	1.0	83.2	1.3	80.8	1.7	79.2	2.0
Urinary, bladder	82.1	1.0	76.2	1.4	70.3	1.9	67.9	2.4
Cervix, uteri	70.5	1.6	64.1	1.8	62.8	2.1	60.0	2.4
Larynx	68.8	2.1	56.7	2.5	45.8	2.8	37.8	3.1
Rectum	62.6	1.2	55.2	1.4	51.8	1.8	49.2	2.3
Kidney, renal pelvis	61.8	1.3	54.4	1.6	49.8	2.0	47.3	2.6
Colon	61.7	0.8	55.4	1.0	53.9	1.2	52.3	1.6
Non-Hodgkin's	57.8	1.0	46.3	1.2	38.3	1.4	34.3	1.7
Oral cavity, pharynx	56.7	1.3	44.2	1.4	37.5	1.6	33.0	1.8
Ovary	55.0	1.3	49.3	1.6	49.9	1.9	49.6	2.4
Leukemia	42.5	1.2	32.4	1.3	29.7	1.5	26.2	1.7
Brain, nervous system	32.0	1.4	29.2	1.5	27.6	1.6	26.1	1.9

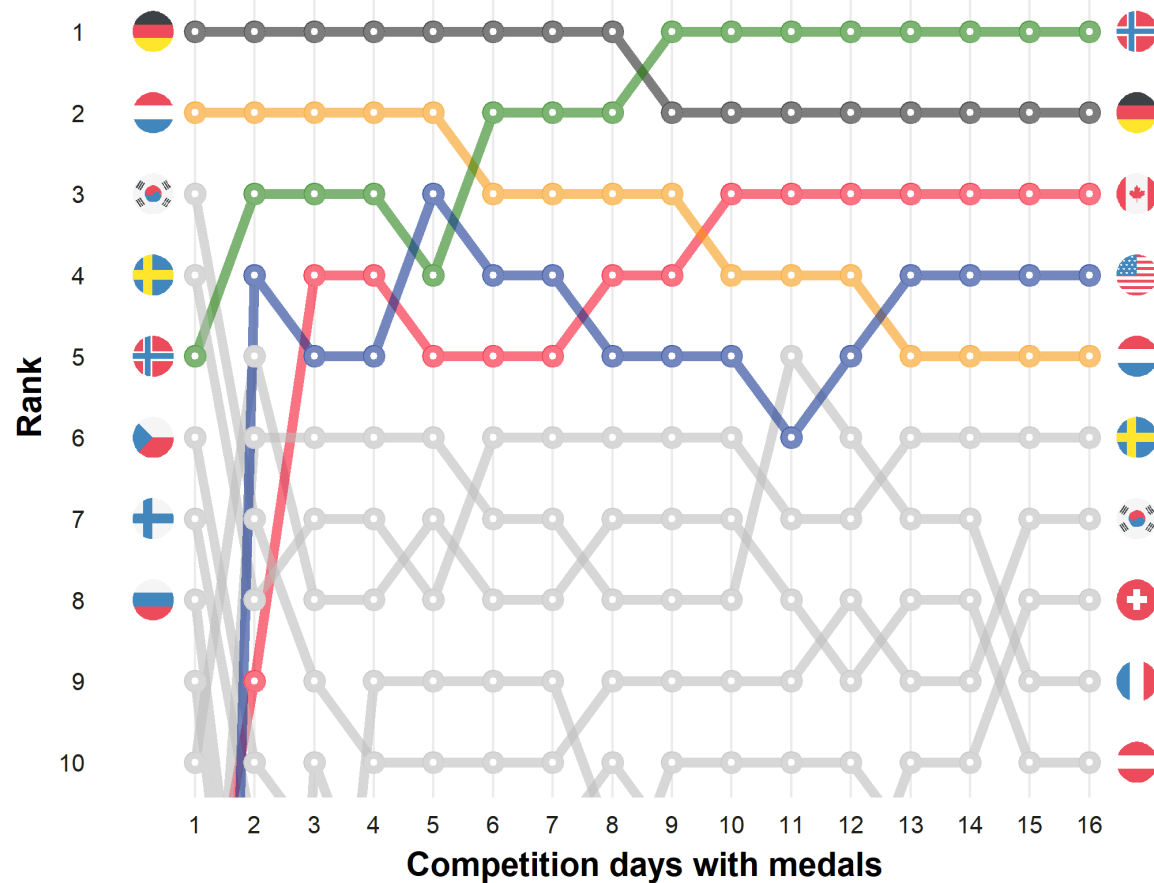
Estimates of % survival rates



# Bump charts

## PyeongChang 2018 Olympic Winter Games

Countries ranked by overall medals after each competition day



# Reproducible examples

# This is 100% normal!



**Brandon Rohrer**

@\_brohrer\_



I was just asked whether I ever have to look things up when I code. I want to go on record saying that, aside from canned white board coding examples, I can't write two lines of code without referring to Google or stack overflow. I would be lost without them.

3:57 PM · Oct 8, 2018 · [Twitter for iPhone](#)

**298** Retweets   **1.5K** Likes



# Broken cake



**Help! My cake broke!**

**vs.**

**Help! I followed these  
6 steps and my cake broke!**

**Same principle  
applies to code**

# Reprexes

## Reproducible examples

**Something anyone can run on their computer  
to reproduce the problem you're facing**



# Debugging and replexes

Simplify your code down to something very basic

Add additional things until stuff breaks

Use a subset of your data or invent fake data

Restart your session and see if it runs in a new session

Ask the internet for help using your toy example

**75% of the time you'll find what's wrong as you make the replex!**

# Making datasets with `tribble()`

`.pull-right[`

```
my_data <- tribble(  
  ~animal, ~number,  
  "cat", 5,  
  "dog", 4,  
  "bear", 7,  
  "bison", 1  
)
```

`my_data`

```
## [38;5;250m3 [39m bear      7  
## [38;5;250m4 [39m bison     1
```

```
## [38;5;246m# A tibble: 4 × 2 [39m  
##   animal number  
##   [3m [38;5;246m<chr> [39m [23m  
## [38;5;250m1 [39m cat           5  
## [38;5;250m2 [39m dog           4
```

# Example reprex

```
my_data <- tribble(  
  ~animal, ~number,  
  "cat", 5,  
  "dog", 4,  
  "bear", 7,  
  "bison", 1  
)
```

*# This plot has a fill legend, but I want to remove it because it's redundant*

*# What's the best way to get rid of the fill?*

```
ggplot(fake_data, aes(x = animal, y = number, fill = animal)) +  
  geom_col()
```

*# I add something here, but what?*