Relationships

Session 7

PMAP 8921: Data Visualization with R
Andrew Young School of Policy Studies
Summer 2025

Plan for today

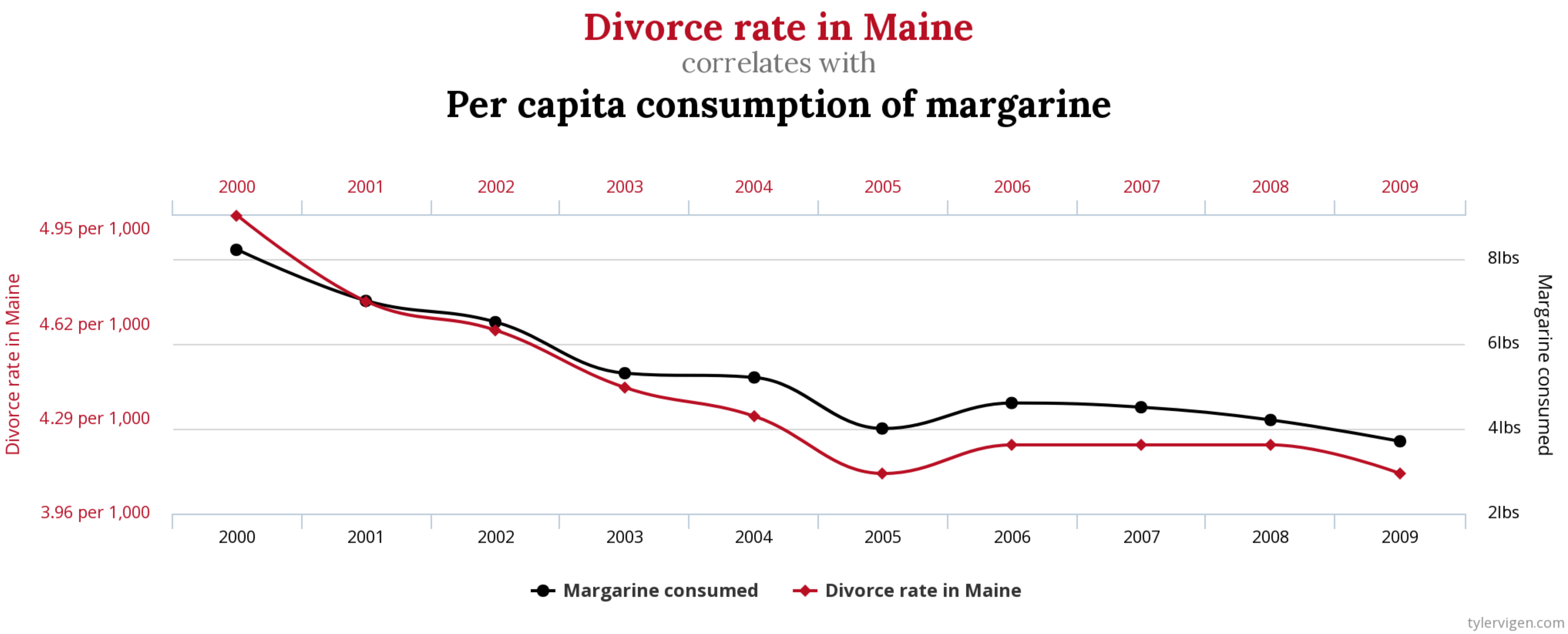
The dangers of dual y-axes

Visualizing correlations

Visualizing regressions

The dangers of dual y-axes

Stop eating margarine!



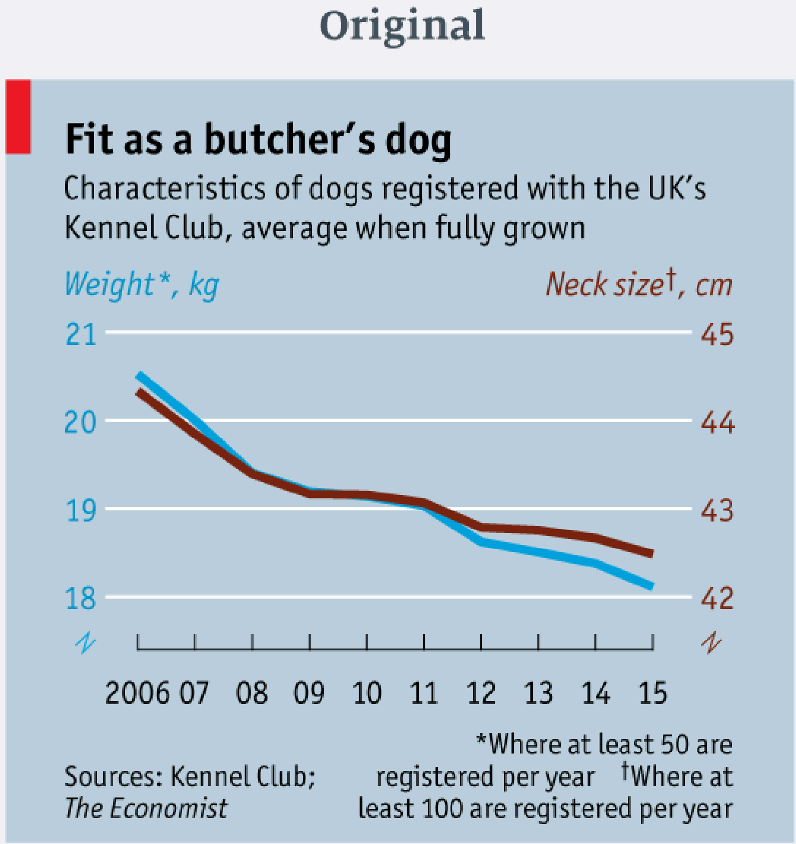
Source: Tyler Vigen's spurious correlations

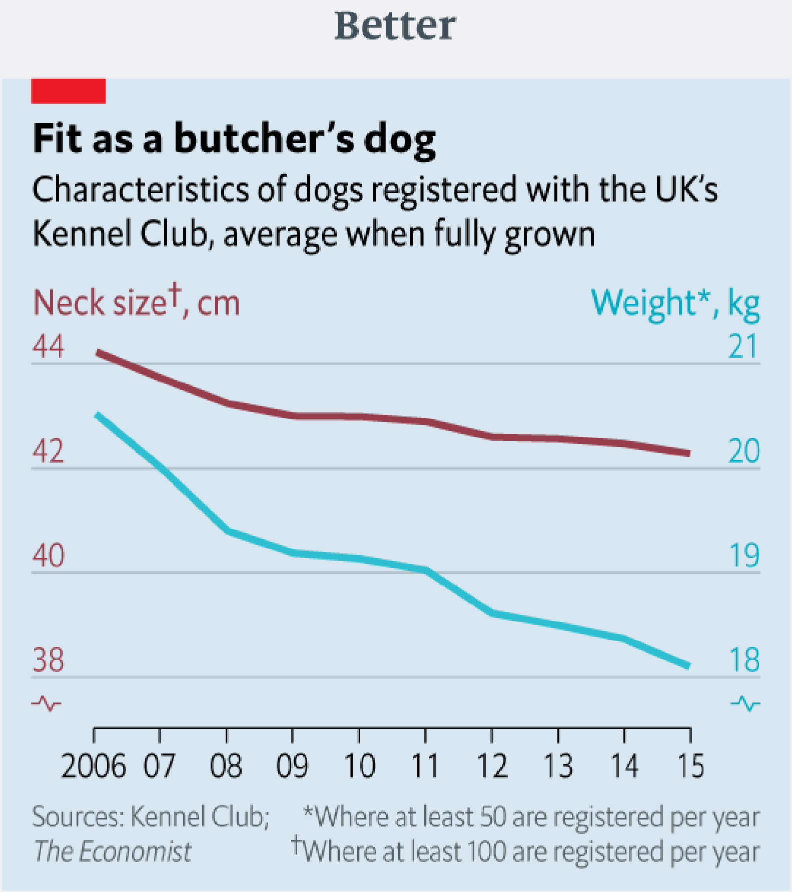
Why not use double y-axes?

You have to choose where the y-axes start and stop, which means…

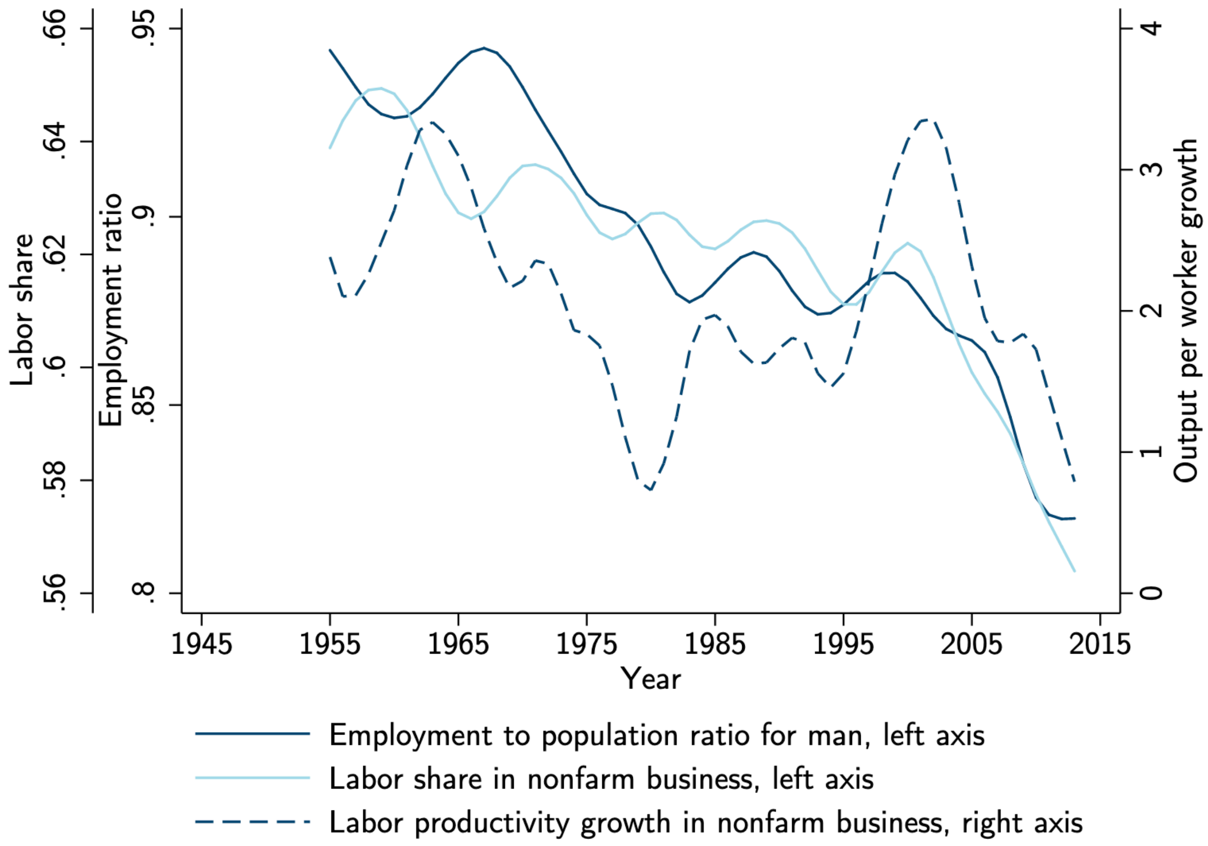
…you can force the two trends to line up however you want!

It even happens in *The Economist!*





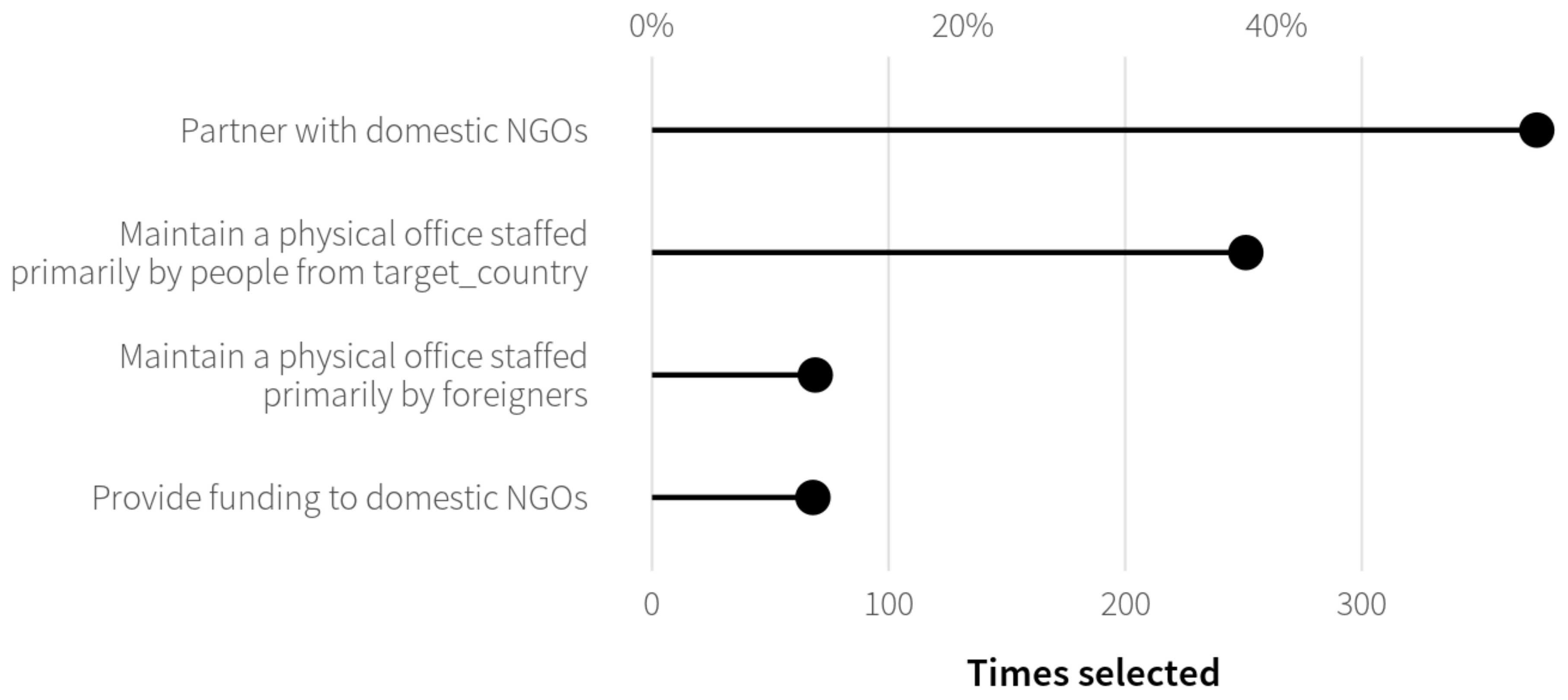
The rare triple y-axis!



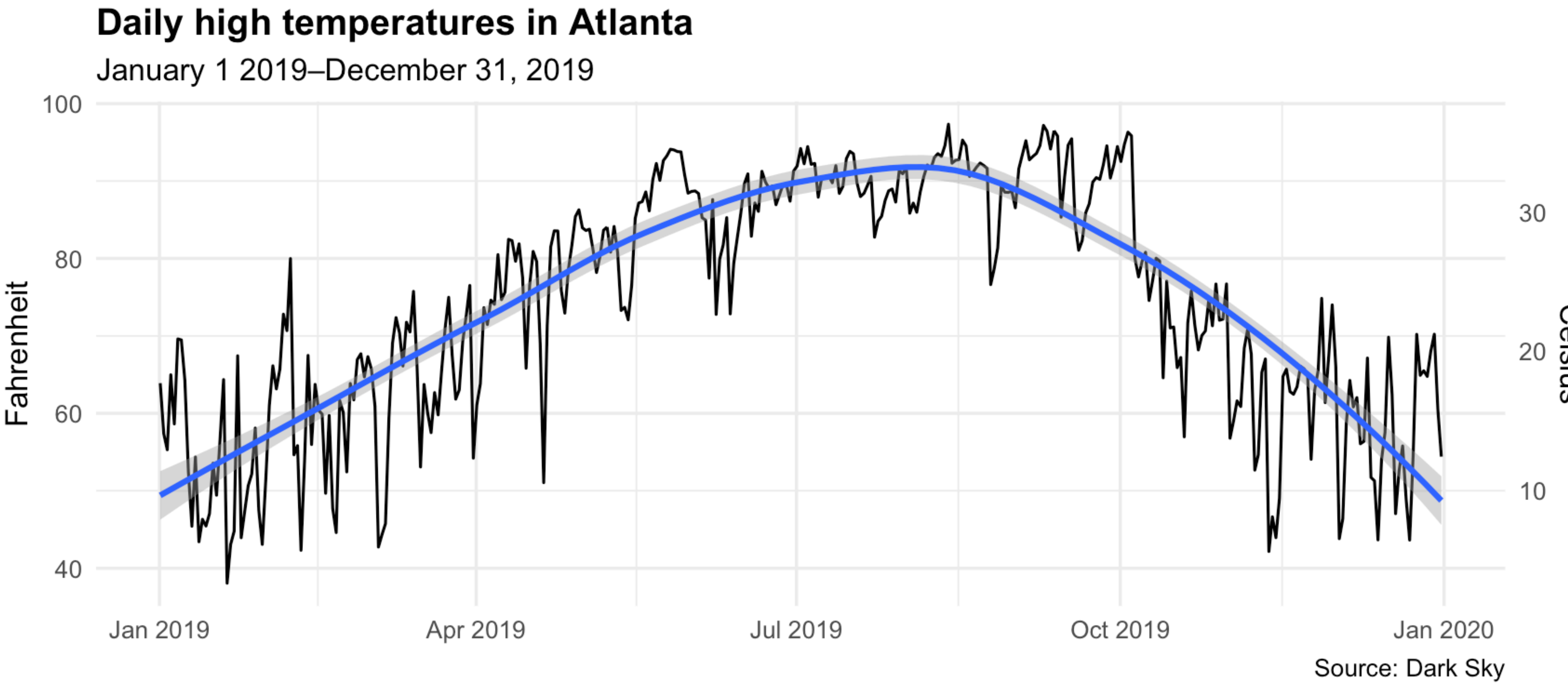
Source: Daron Acemoglu and Pascual Restrepo, "The Race Between Man and Machine: Implications of Technology for Growth, Factor Shares and Employment"

When is it legal?

When the two axes measure the same thing

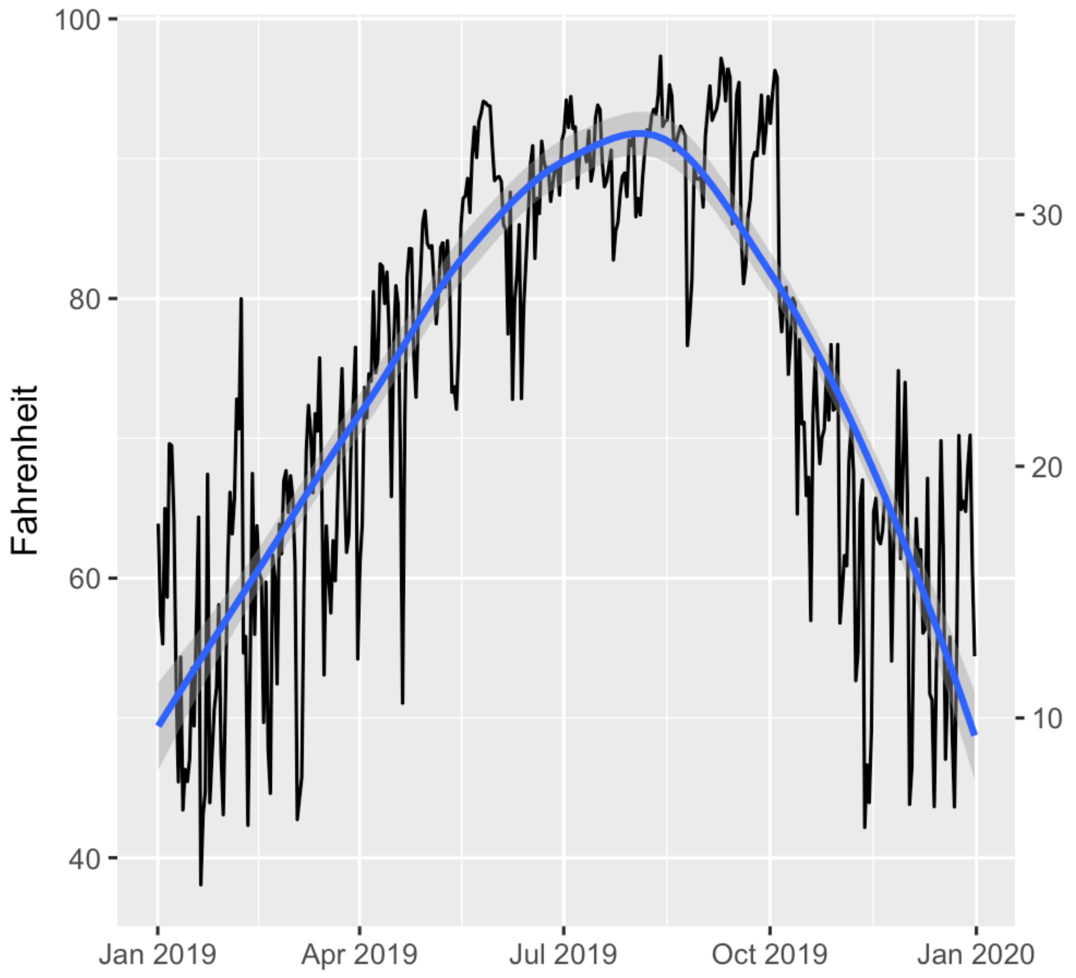


When is it legal?



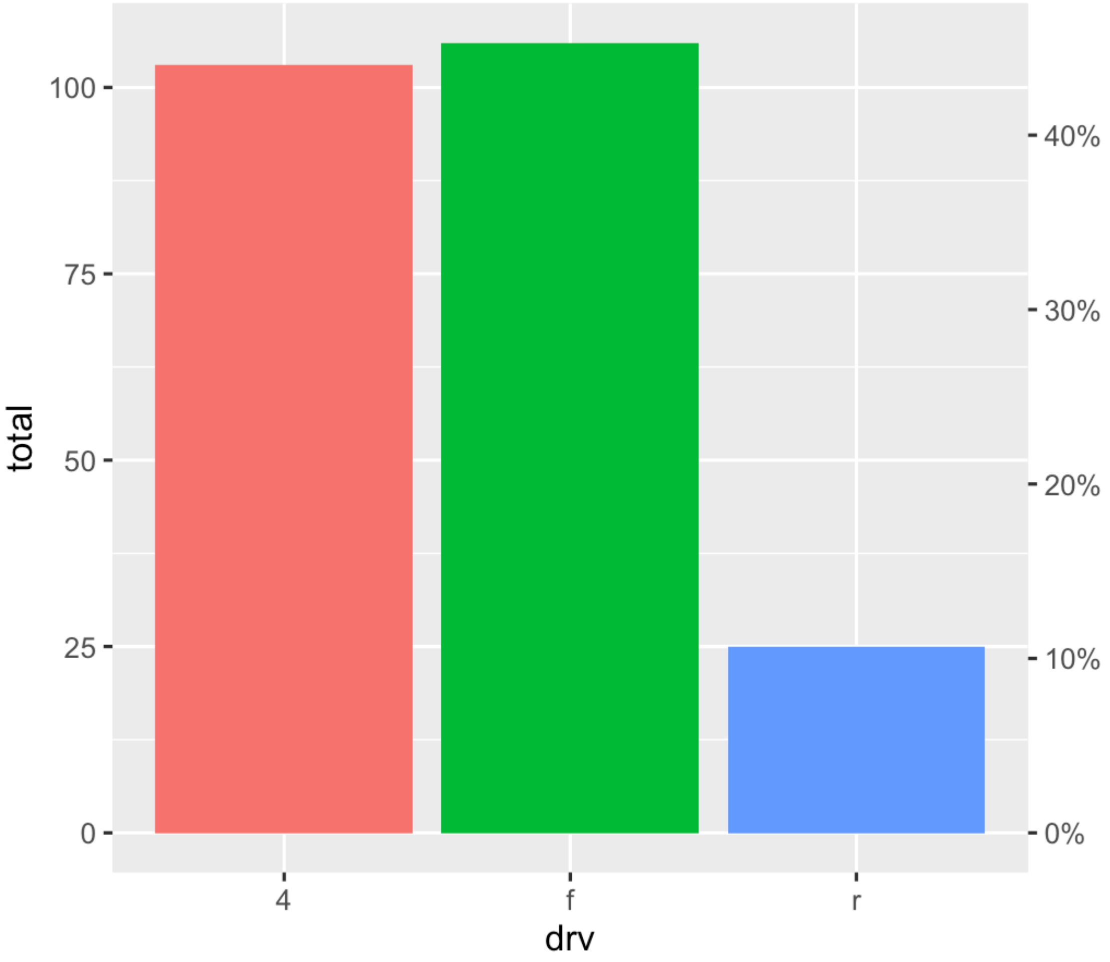
Adding a second scale in R

```r
# From the uncertainty example
weather_atl <-
  read_csv("data/atl-weather-2019.csv")

ggplot(weather_atl,
       aes(x = time, y = temperatureHigh)) +
  geom_line() +
  geom_smooth() +
  scale_y_continuous(
    sec.axis =
      sec_axis(trans = ~ (32 - .) * -5/9,
               name = "Celsius")
  ) +
  labs(x = NULL, y = "Fahrenheit")
```

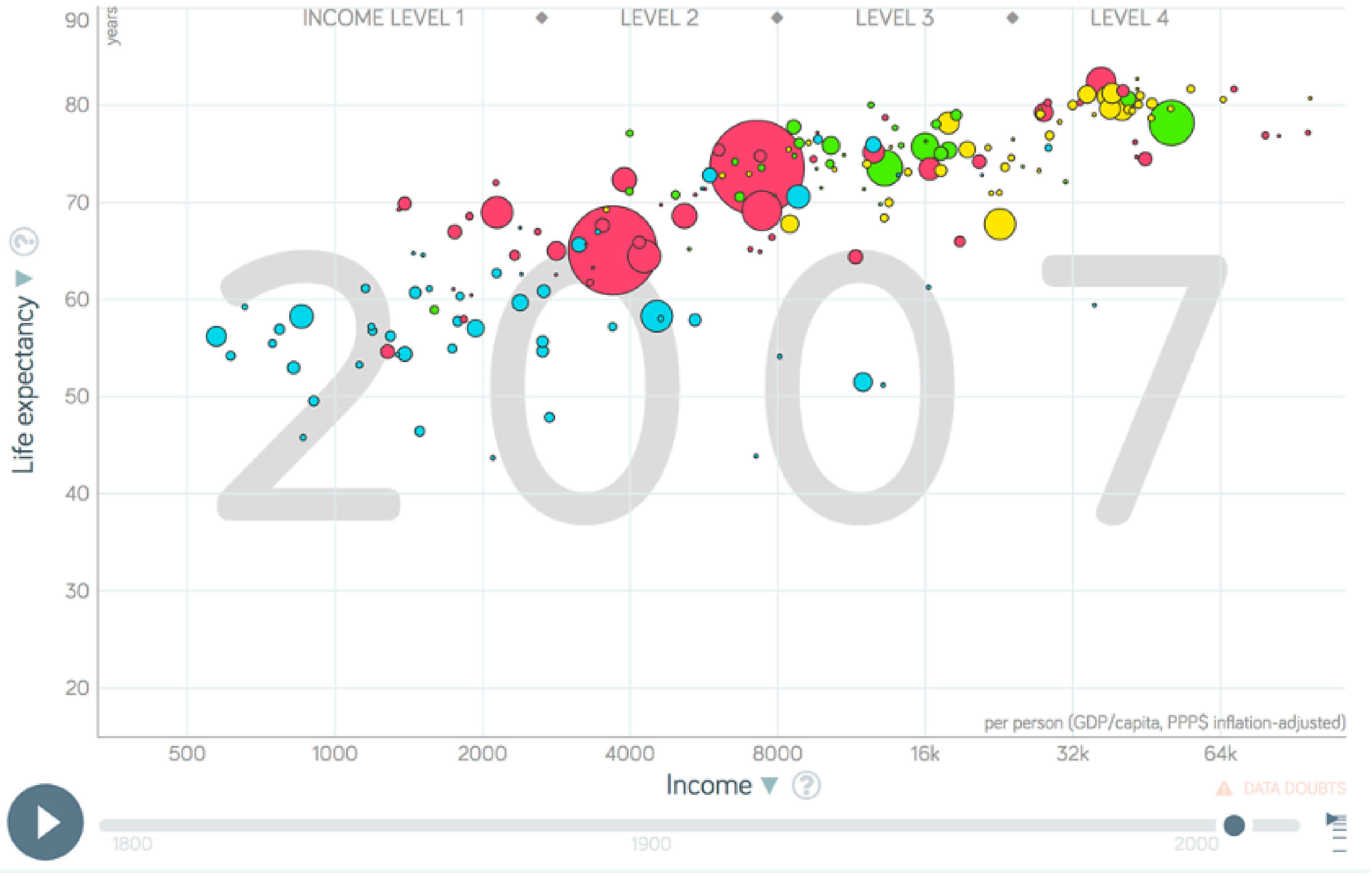


Adding a second scale in R

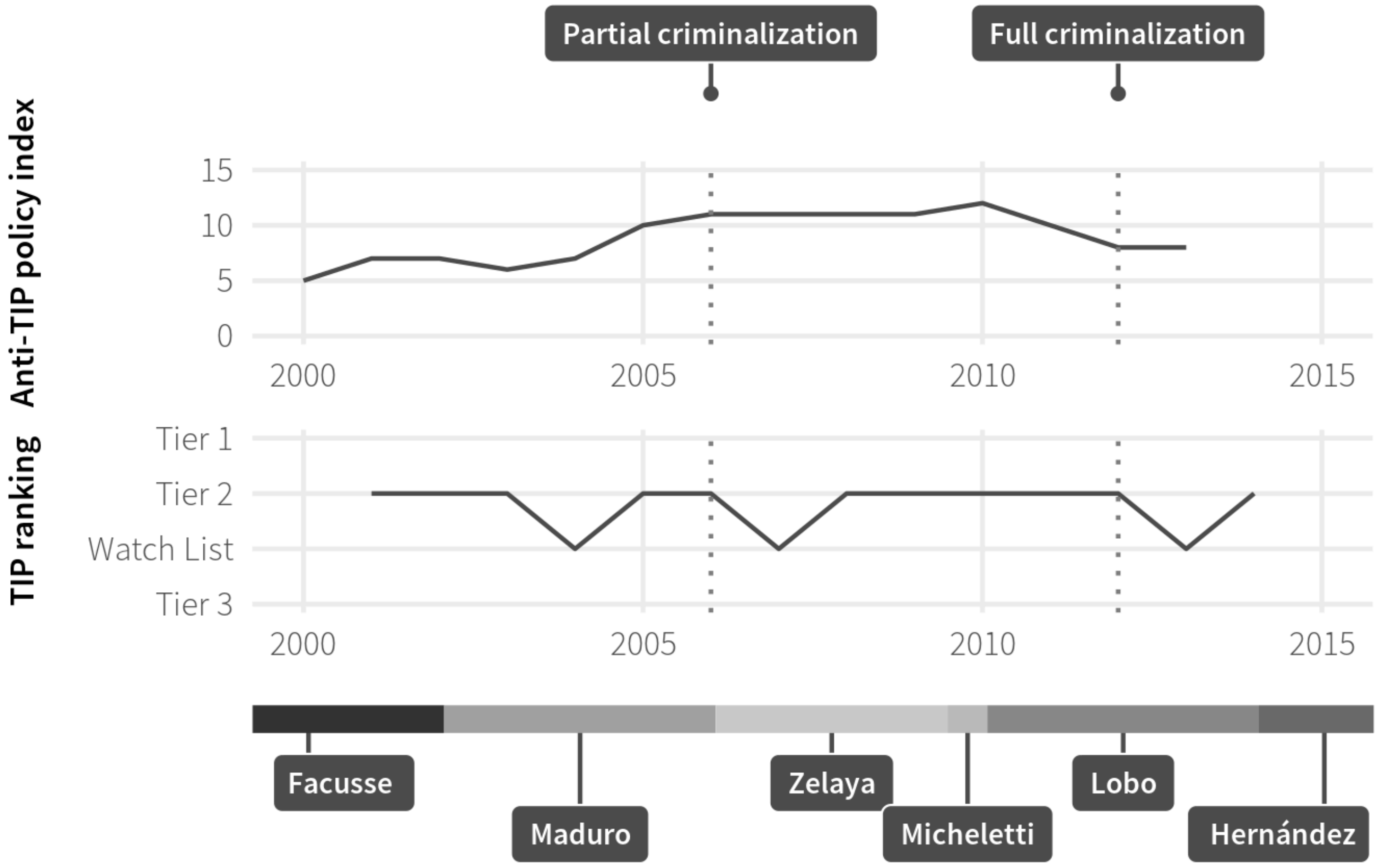
```r
car_counts <- mpg |>
  group_by(drv) |>
  summarize(total = n())

total_cars <- sum(car_counts$total)

ggplot(car_counts,
       aes(x = drv, y = total,
           fill = drv)) +
  geom_col() +
  scale_y_continuous(
    sec.axis = sec_axis(
      trans = ~ . / total_cars,
      labels = scales::label_percent())
  ) +
  guides(fill = "none")
```



Alternative 1: Use another aesthetic



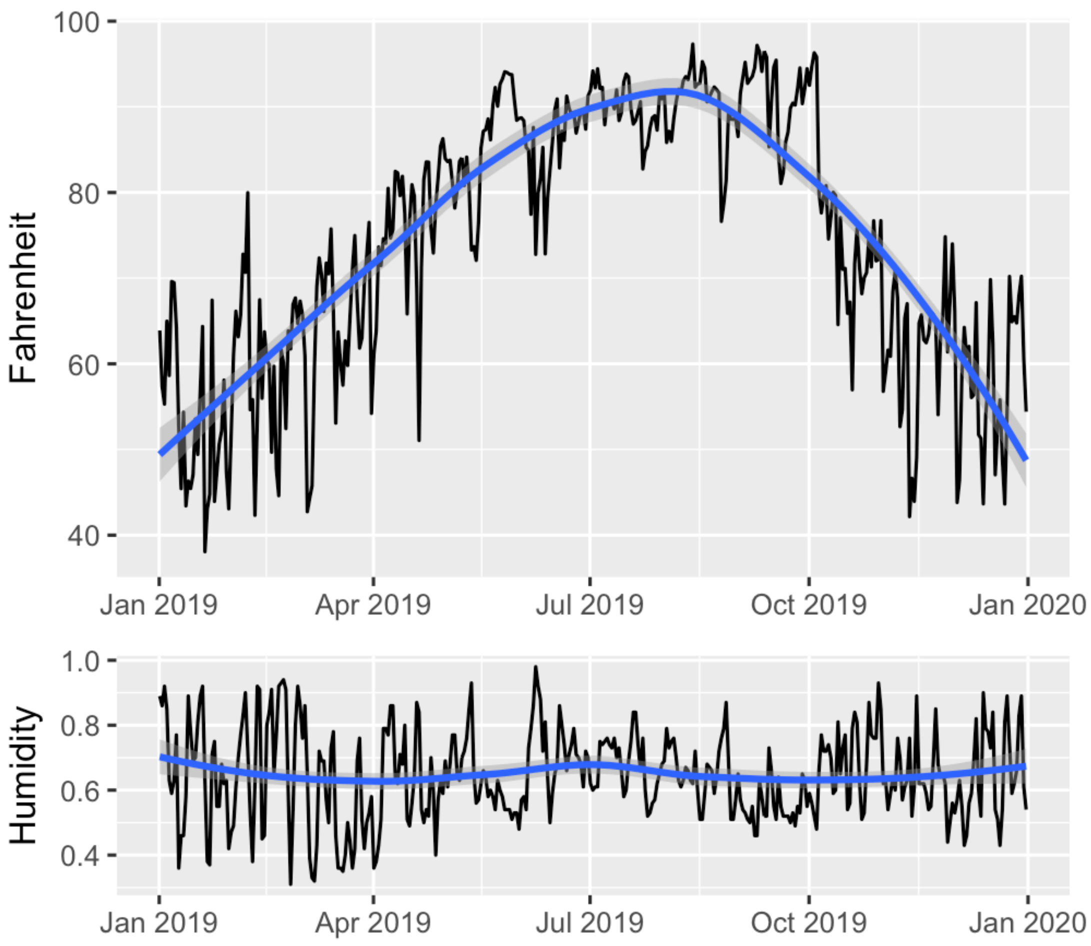
Alternative 2: Use multiple plots



Anti-trafficking policy timeline in Honduras

Alternative 2: Use multiple plots

```r
library(patchwork)

temp_plot <- ggplot(weather_atl,
                    aes(x = time,
                        y = temperatureHigh)) +
  geom_line() + geom_smooth() +
  labs(x = NULL, y = "Fahrenheit")

humid_plot <- ggplot(weather_atl,
                     aes(x = time,
                         y = humidity)) +
  geom_line() + geom_smooth() +
  labs(x = NULL, y = "Humidity")

temp_plot + humid_plot +
  plot_layout(ncol = 1,
              heights = c(0.7, 0.3))
```



Visualizing correlations

What is correlation?

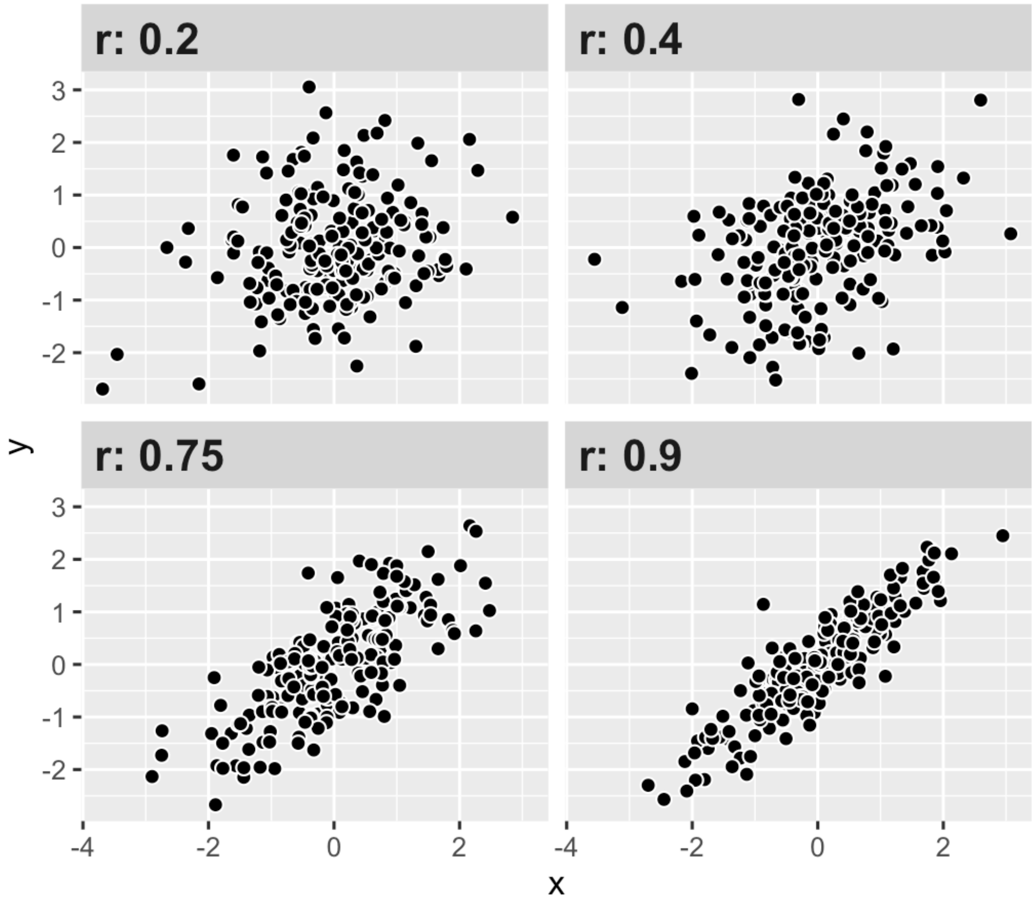
$$r_{x,y} = \frac{\text{cov}(x, y)}{\sigma_x \sigma_y}$$

As the value of X goes up, Y tends to go up (or down) a lot/a little/not at all

Says nothing about *how much* Y changes when X changes

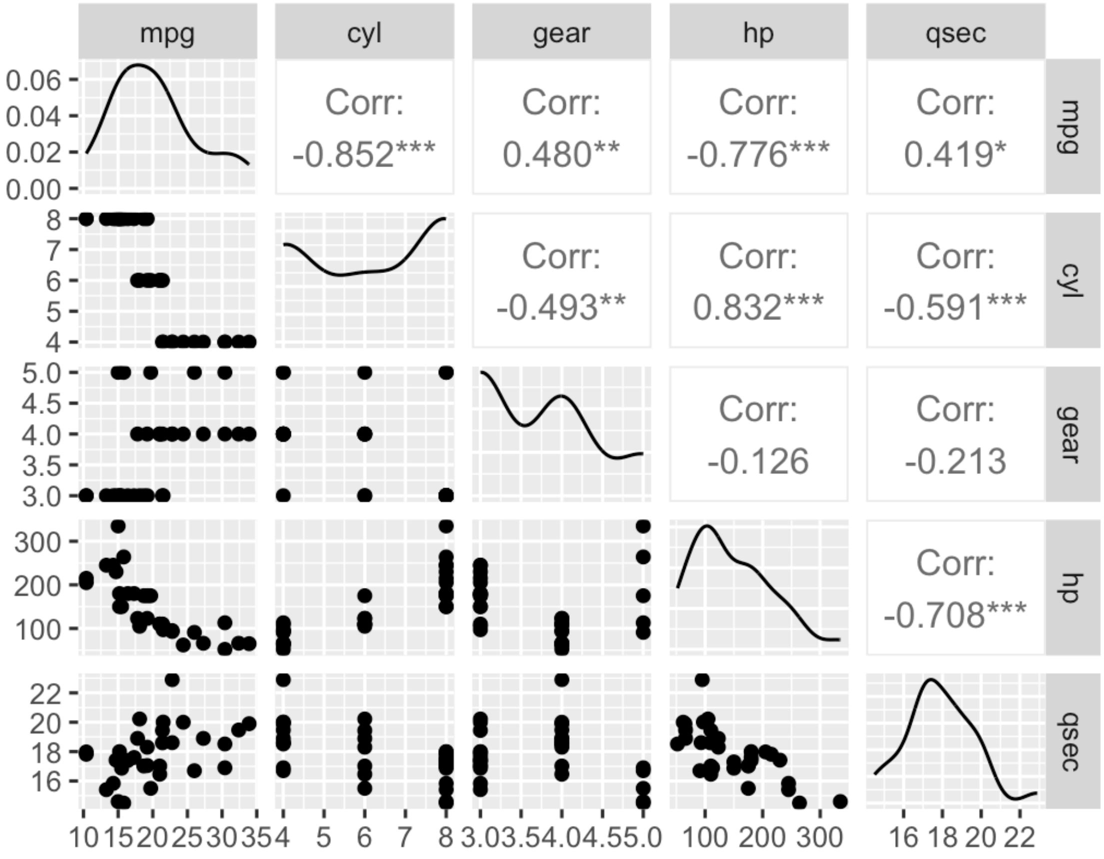
Correlation values

r	Rough meaning
±0.1–0.3	Modest
±0.3–0.5	Moderate
±0.5–0.8	Strong
±0.8–0.9	Very strong

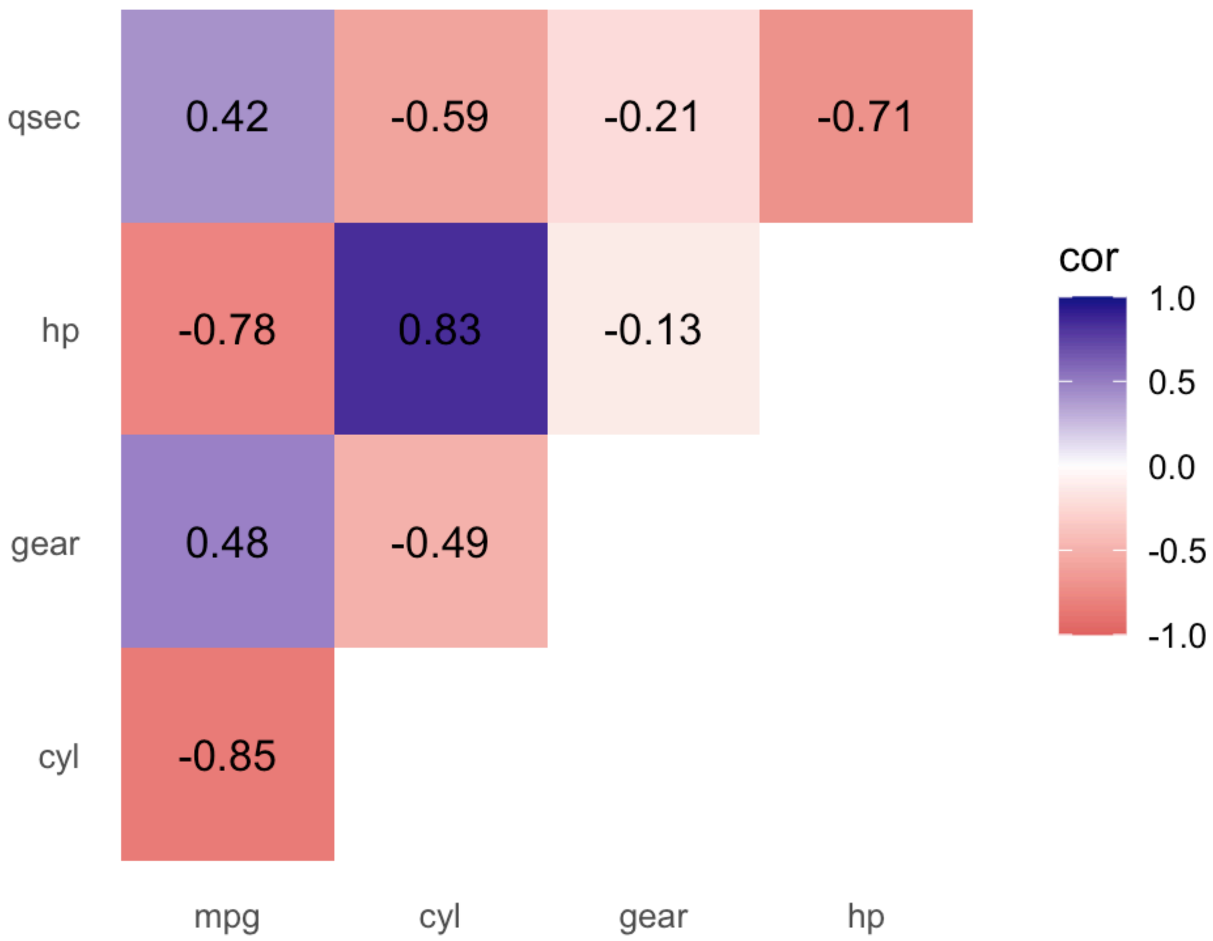


Scatterplot matrices

```r
library(GGally)

cars_smaller <- mtcars |>
  select(mpg, cyl, gear, hp, qsec)

ggpairs(cars_smaller)
```



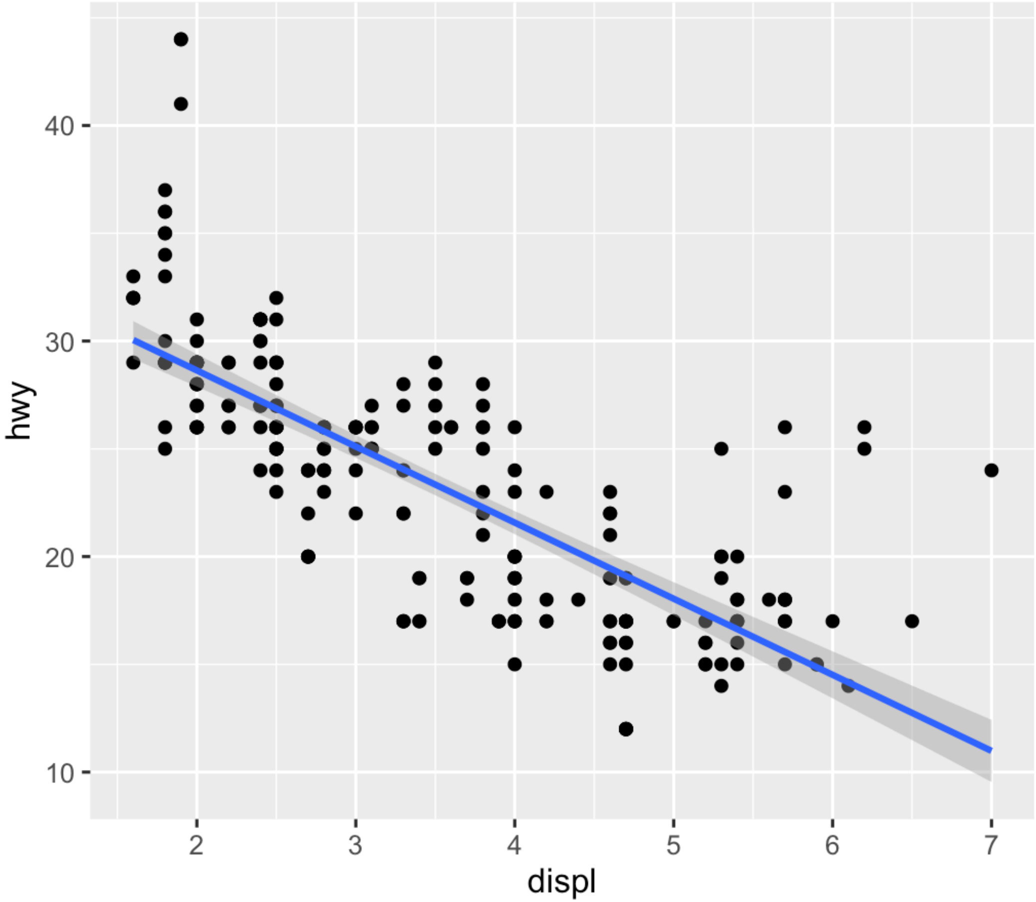
Correlograms: Heatmaps

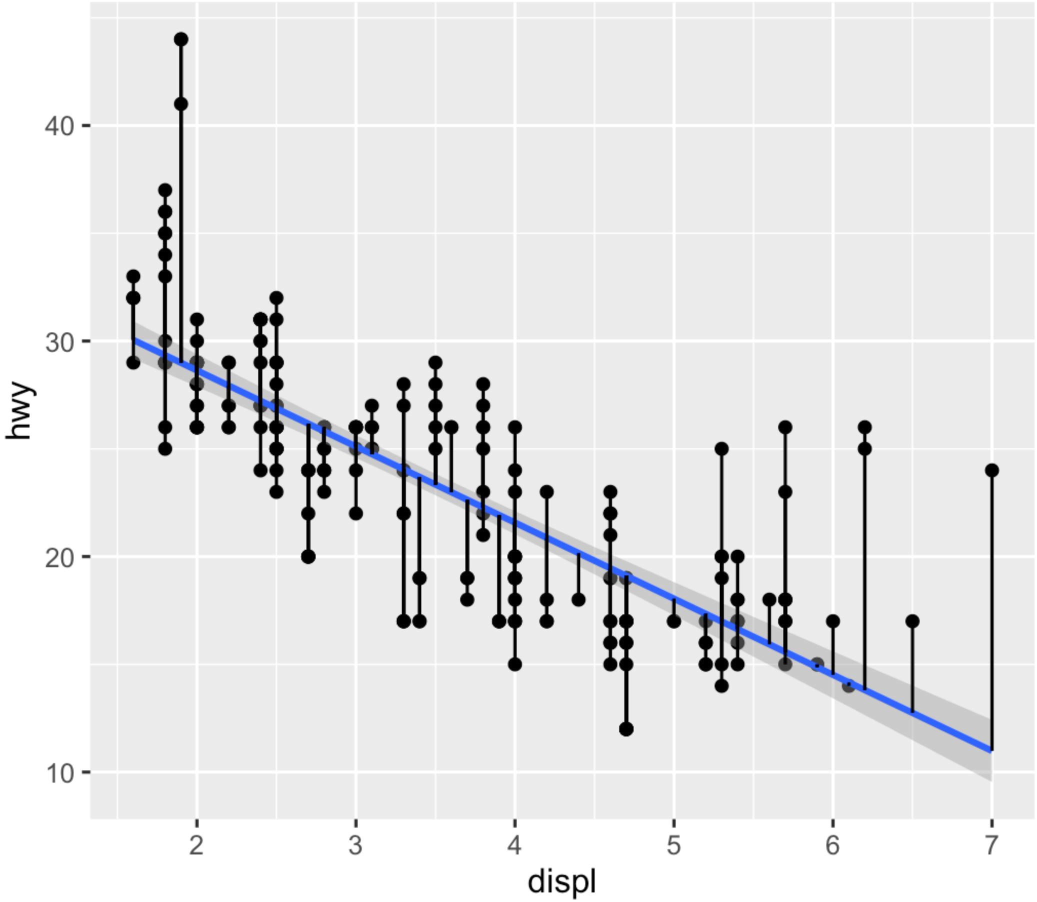


Correlograms: Points

Visualizing regressions

Drawing lines





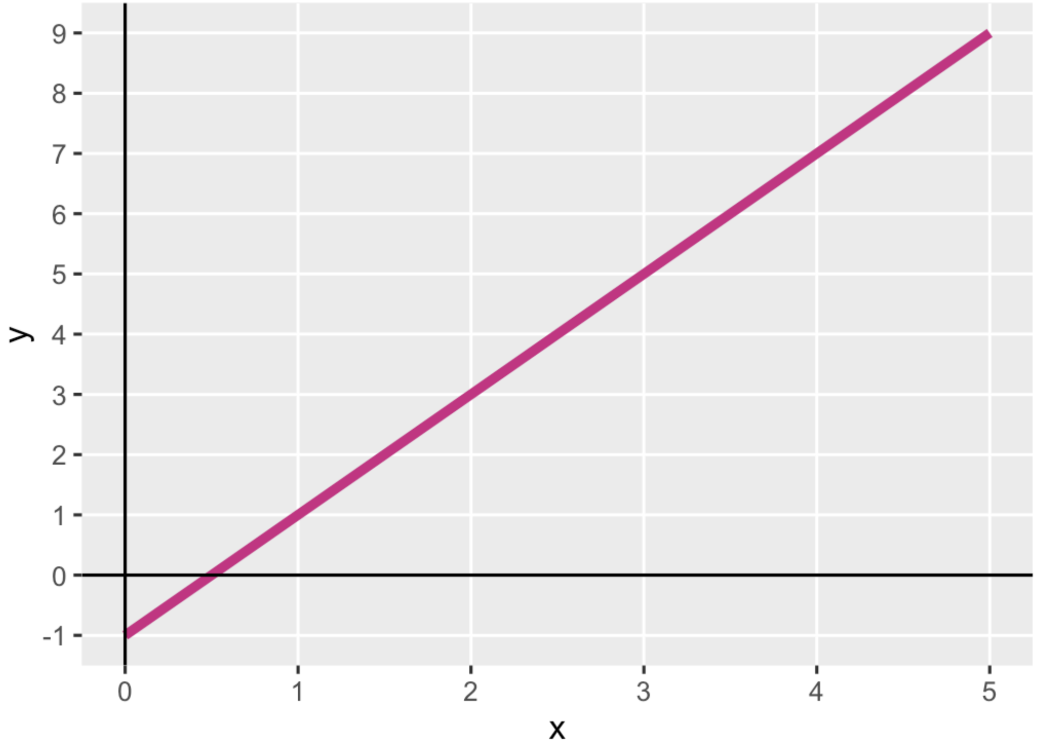
Drawing lines with math

$$y = mx + b$$

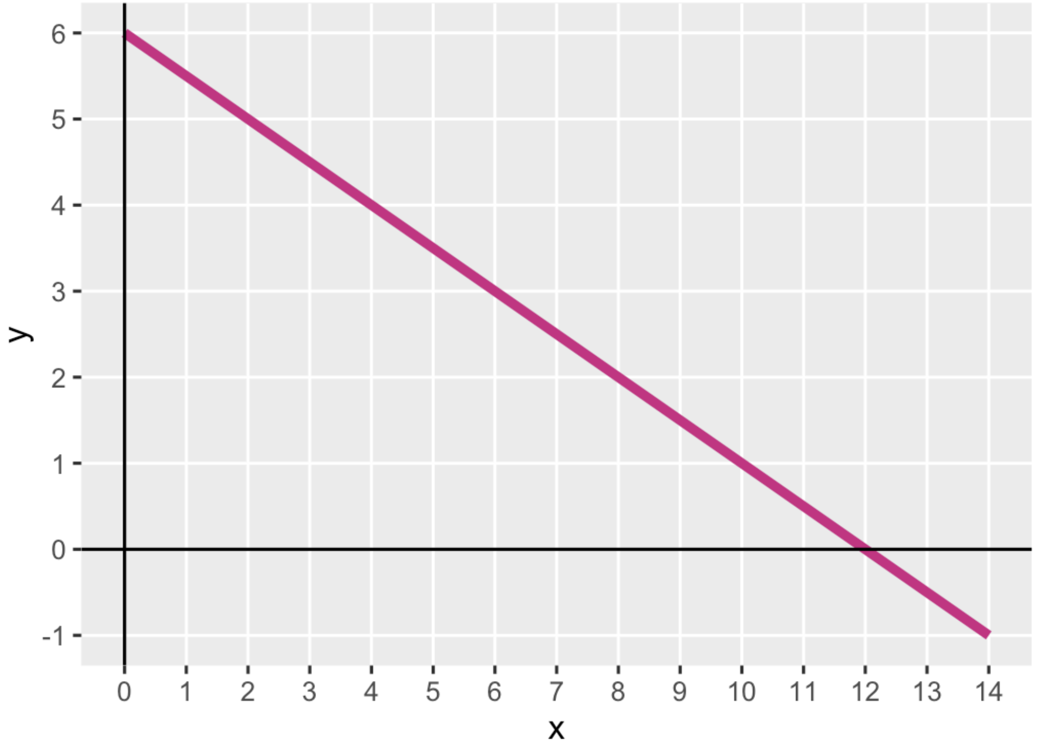
y	A number
x	A number
m	Slope ($\frac{\text{rise}}{\text{run}}$)
b	y-intercept

Slopes and intercepts

$$y = 2x - 1$$



$$y = -0.5x + 6$$



Drawing lines with stats

$$\hat{y} = \beta_0 + \beta_1 x_1 + \varepsilon$$

y	$\hat{y}$	Outcome variable (DV)
x	x_1	Explanatory variable (IV)
m	β_1	Slope
b	β_0	y-intercept
	ε	Error (residuals)

Building models in R

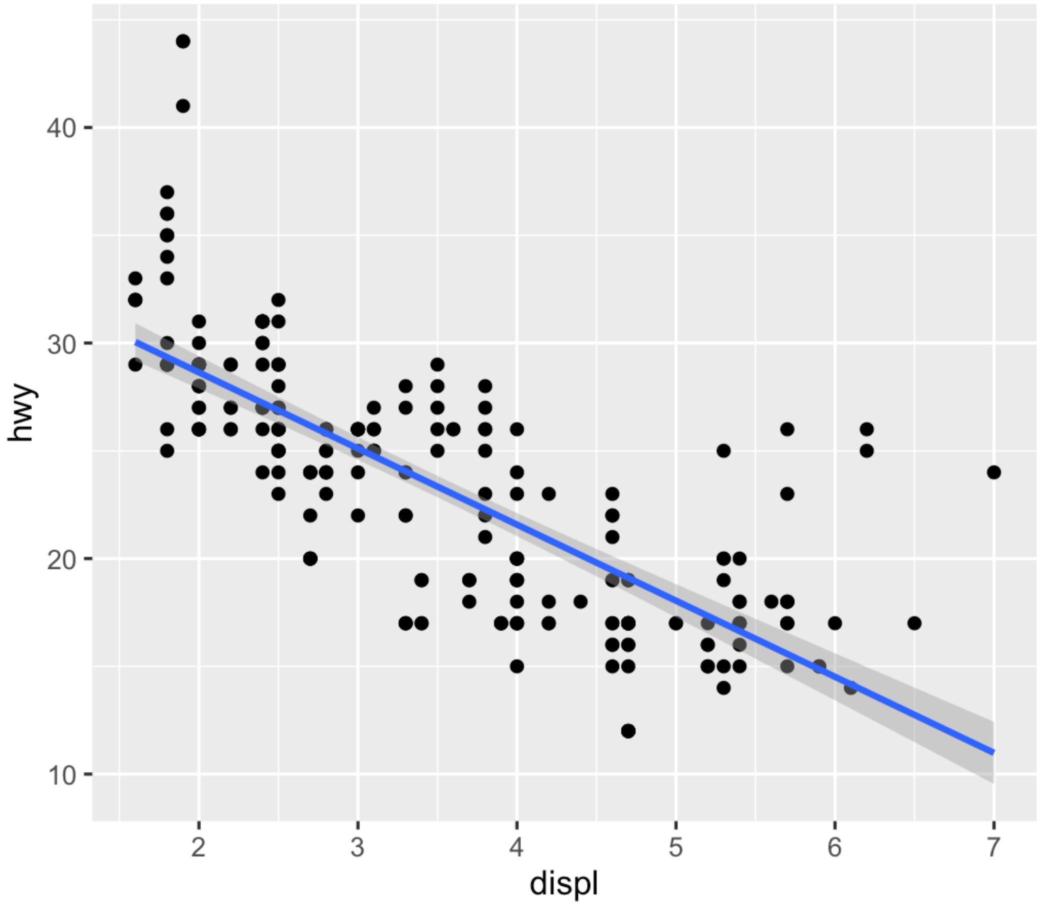
```r
name_of_model <- lm(<Y> ~ <X>, data = <DATA>)

summary(name_of_model)  # See model details
```

```r
library(broom)

# Convert model results to a data frame for plotting
tidy(name_of_model)

# Convert model diagnostics to a data frame
glance(name_of_model)
```

Modeling displacement and MPG

$$\hat{\text{hwy}} = \beta_0 + \beta_1 \text{displ} + \varepsilon$$

```
car_model <- lm(hwy ~ displ,
                data = mpg)
```



Modeling displacement and MPG

.small-code[

```
tidy(car_model, conf.int = TRUE)
```

```
##  [38;5;246m# A tibble: 2 × 7 [39m
##   term          estimate std.error statistic   p.value conf.low
##    [3m [38;5;246m<chr> [39m [23m              [3m [38;5;246m<dbl> [3
## [38;5;250m1 [39m (Intercept)    35.7      0.720      49.6 2.12
## [38;5;250m2 [39m displ         - [31m3 [39m [31m. [39m [31m53 [3
```

]

.small-code[

Translating results to math

.pull-left[

.small-code[

```
##  [38;5;246m# A tibble: 2 × 2 [39m
##   term           estimate
##    [3m [38;5;246m<chr> [39m [23m           [3m [38;5;246m<dbl> [3
##  [38;5;250m1 [39m (Intercept)     35.7
##  [38;5;250m2 [39m displ           - [31m3 [39m [31m. [39m [31m53 [3
```

]

$$\hat{\text{hwy}} = 35.7 + (-3.53) \times \text{displ} + \varepsilon$$

Template for single variables

A one unit increase in X is *associated* with a β_1 increase (or decrease) in Y, on average

$$\hat{\text{hwy}} = \beta_0 + \beta_1 \text{displ} + \varepsilon$$

$$\hat{\text{hwy}} = 35.7 + (-3.53) \times \text{displ} + \varepsilon$$

This is easy to visualize! It's a line!

Multiple regression

We're not limited to just one explanatory variable!

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon$$

```
car_model_big <- lm(hwy ~ displ + cyl + drv,
                    data = mpg)
```

$$\hat{\text{hwy}} = \beta_0 + \beta_1 \text{displ} + \beta_2 \text{cyl} + \beta_3 \text{drv:f} + \beta_4 \text{drv:r} + \varepsilon$$

Modeling lots of things and MPG

.small-code[

```
tidy(car_model_big, conf.int = TRUE)
```

```
##  [38;5;246m# A tibble: 5 × 7 [39m
##   term          estimate std.error statistic  p.value conf.low c
##    [3m [38;5;246m<chr> [39m [23m                  [3m [38;5;246m<dbl> [3
##  [38;5;250m1 [39m (Intercept)    33.1      1.03      32.1  9.49
##  [38;5;250m2 [39m displ          - [31m1 [39m [31m. [39m [31m12 [3
##  [38;5;250m3 [39m cyl            - [31m1 [39m [31m. [39m [31m45 [3
##  [38;5;250m4 [39m drvf            5.04     0.513      9.83 3.07
##  [38;5;250m5 [39m drvr            4.89     0.712      6.86 6.20
```

Sliders and switches

Sliders and switches

Categorical variables

Continuous variables

Template for continuous variables

Holding everything else constant, **a one unit increase in X is *associated* with a β_n increase (or decrease) in Y, on average**

$$
\hat{\text{hwy}} = 33.1 + (-1.12) \times \text{displ} + (-1.45) \times \text{cyl} + (5.04) \times \text{drv:f} + (4.89) \times \text{drv:r} + \varepsilon
$$

On average, a one unit increase in cylinders is associated with 1.45 lower highway MPG, holding everything else constant

Template for categorical variables

Holding everything else constant, Y is β_n units larger (or smaller) in X_n, compared to X_{omitted}, on average

$$\hat{\text{hwy}} = 33.1 + (-1.12) \times \text{displ} + (-1.45) \times \text{cyl} + (5.04) \times \text{drv:f} + (4.89) \times \text{drv:r} + \varepsilon$$

On average, front-wheel drive cars have 5.04 higher highway MPG than 4-wheel-drive cars, holding everything else constant

Good luck visualizng all this!

You can't just draw a line!
There are too many moving parts!

Main problems

Each coefficient has its own estimate and standard errors

Solution: Plot the coefficients and their errors with a *coefficient plot*

The results change as you move each slider up and down and flip each switch on and off

Solution: Plot the *marginal effects* for the coefficients you're interested in

Coefficient plots

Convert the model results to a data frame with `tidy()`

.small-code[

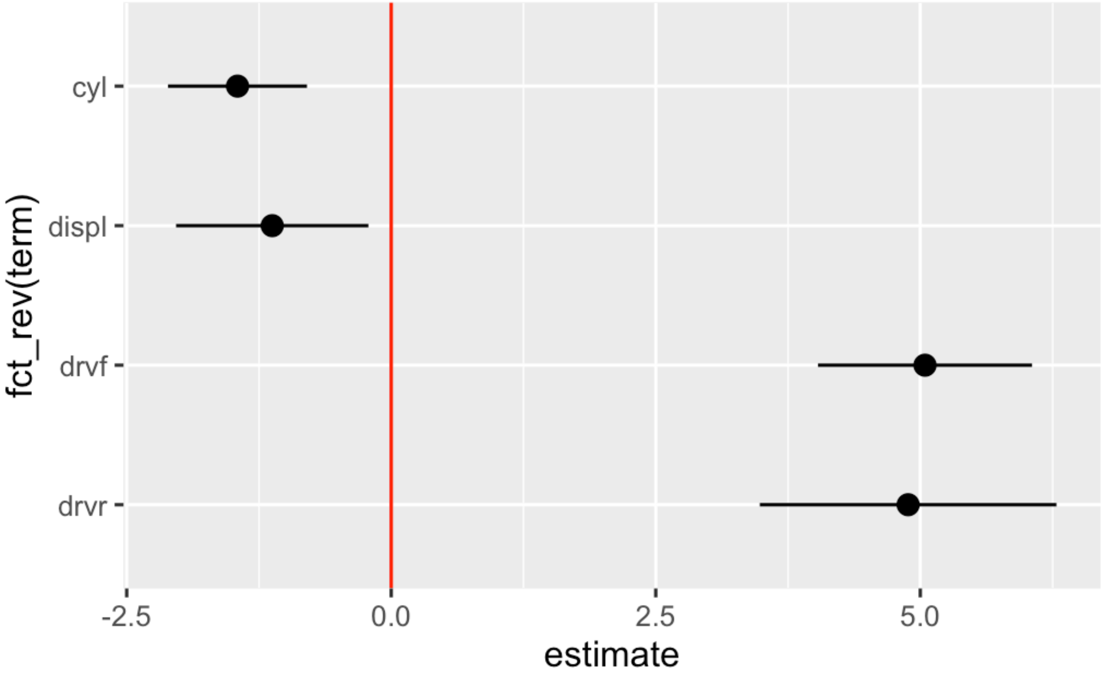
```
car_model_big <- lm(hwy ~ displ + cyl + drv, data = mpg)

car_coefs <- tidy(car_model_big, conf.int = TRUE) |>
  filter(term != "(Intercept)")  # We can typically skip plott
car_coefs
```

```
##  [38;5;246m# A tibble: 4 × 7 [39m
##   term  estimate std.error statistic  p.value conf.low conf.hi
##    [3m [38;5;246m<chr> [39m [23m     [3m [38;5;246m<dbl> [39m [23m
```

Coefficient plots

Plot the estimate and confidence intervals with `geom_pointrange()`

```
ggplot(car_coefs,
       aes(x = estimate,
           y = fct_rev(term))) +
  geom_pointrange(aes(xmin = conf.low,
                      xmax = conf.high)) +
  geom_vline(xintercept = 0, color = "red")
```



Marginal effects plots

Remember that we interpret individual coefficients while holding the others constant

We move one slider while leaving all the other sliders and switches alone

Same principle applies to visualizing the effect

Plug a bunch of values into the model and find the predicted outcome

Plot the values and predicted outcome

Marginal effects plots

Create a data frame of values you want to manipulate and values you want to hold constant

Must include all the explanatory variables in the model

Marginal effects plots

.small-code[

```
cars_new_data <- tibble(displ = seq(2, 7, by = 0.1),
                        cyl = mean(mpg$cyl),
                        drv = "f")

head(cars_new_data)
```

```
##  [38;5;246m# A tibble: 6 × 3 [39m
##   displ    cyl drv
##    [3m [38;5;246m<dbl> [39m [23m  [3m [38;5;246m<dbl> [39m [23m  [
##  [38;5;250m1 [39m   2     5.89 f
##  [38;5;250m2 [39m   2.1   5.89 f
```

Marginal effects plots

Plug each of those rows of data into the model with `augment()`

.small-code[

```
predicted_mpg <- augment(car_model_big, newdata = cars_new_dat
                         se_fit = TRUE)

head(predicted_mpg)
```

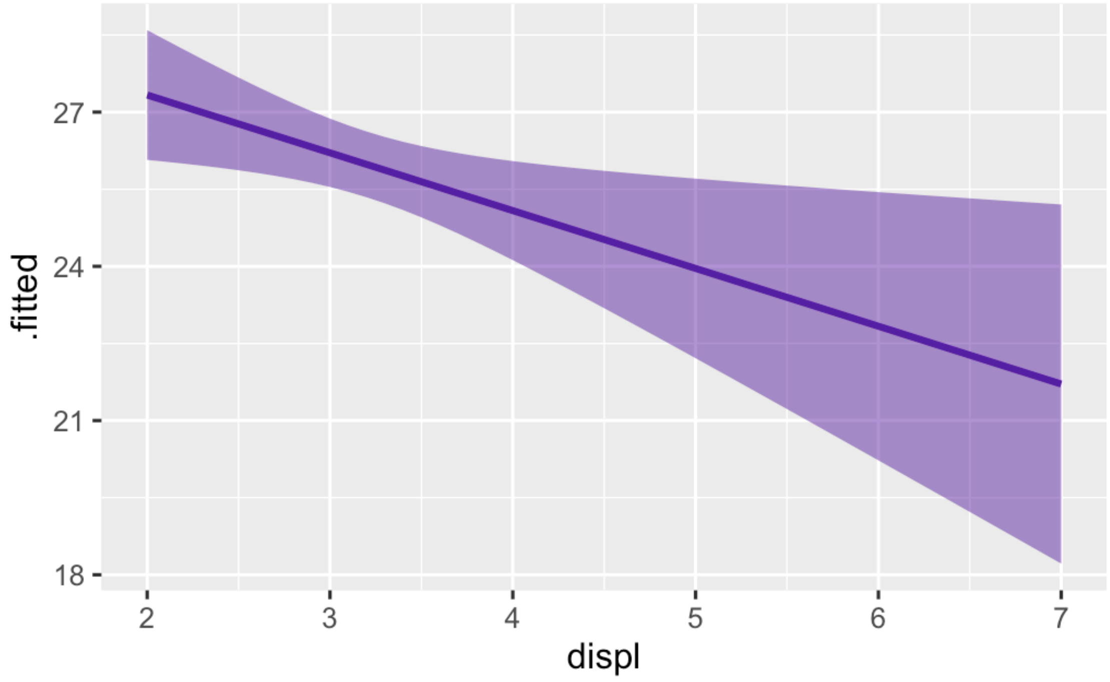
```
##  [38;5;246m# A tibble: 6 × 5 [39m
##   displ   cyl drv   .fitted .se.fit
##    [3m [38;5;246m<dbl> [39m [23m  [3m [38;5;246m<dbl> [39m [23m  [
## [38;5;250m1 [39m    2     5.89 f         27.3    0.644
```

Marginal effects plots

Plot the fitted values for each row

Cylinders held at their mean; assumes front-wheel drive

```
ggplot(predicted_mpg,
       aes(x = displ, y = .fitted)) +
  geom_ribbon(aes(ymin = .fitted +
                    (-1.96 * .se.fit),
                  ymax = .fitted +
                    (1.96 * .se.fit)),
              fill = "#5601A4",
              alpha = 0.5) +
  geom_line(size = 1, color = "#5601A4")
```



Multiple effects at once

We can also move multiple
sliders and switches at the same time!

What's the marginal effect of increasing displacement across the front-, rear-, and four-wheel drive cars?

Multiple effects at once

Create a new dataset with varying displacement *and* varying drive, holding cylinders at its mean

The `expand_grid()` function does this

Multiple effects at once

.small-code[

```
cars_new_data_fancy <- expand_grid(displ = seq(2, 7, by = 0.1),
                                   cyl = mean(mpg$cyl),
                                   drv = c("f", "r", "4"))

head(cars_new_data_fancy)
```

```
##  [38;5;246m# A tibble: 6 × 3 [39m
##    displ    cyl drv
##     [3m [38;5;246m<dbl> [39m [23m  [3m [38;5;246m<dbl> [39m [23m  [
##  [38;5;250m1 [39m   2     5.89 f
##  [38;5;250m2 [39m   2     5.89 r
```

Multiple effects at once

Plug each of those rows of data into the model with `augment()`

.small-code[

```
predicted_mpg_fancy <- augment(car_model_big, newdata = cars_n
                               se_fit = TRUE)

head(predicted_mpg_fancy)
```

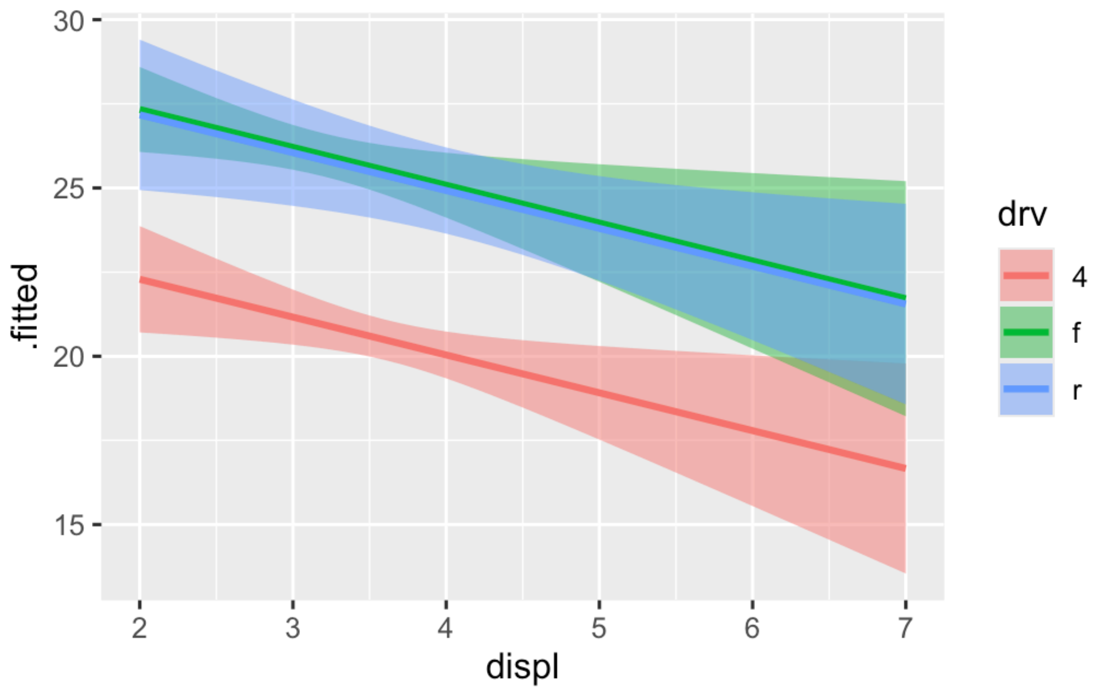
```
##  [38;5;246m# A tibble: 6 × 5 [39m
##   displ   cyl drv   .fitted .se.fit
##    [3m [38;5;246m<dbl> [39m [23m  [3m [38;5;246m<dbl> [39m [23m  [
## [38;5;250m1 [39m    2     5.89 f           27.3    0.644
```

Multiple effects at once

Plot the fitted values for each row

Cylinders held at their mean; colored/filled by drive

```
ggplot(predicted_mpg_fancy,
       aes(x = displ, y = .fitted)) +
  geom_ribbon(aes(ymin = .fitted +
                    (-1.96 * .se.fit),
                  ymax = .fitted +
                    (1.96 * .se.fit),
                  fill = drv),
              alpha = 0.5) +
  geom_line(aes(color = drv), linewidth = 1)
```

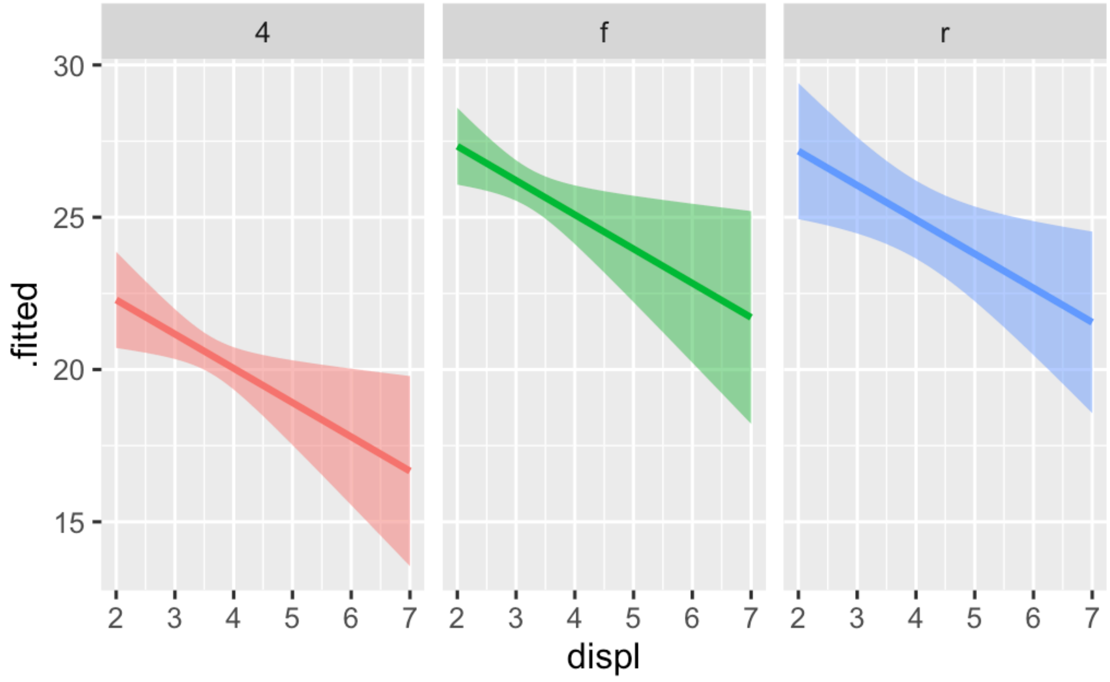


Multiple effects at once

Plot the fitted values for each row

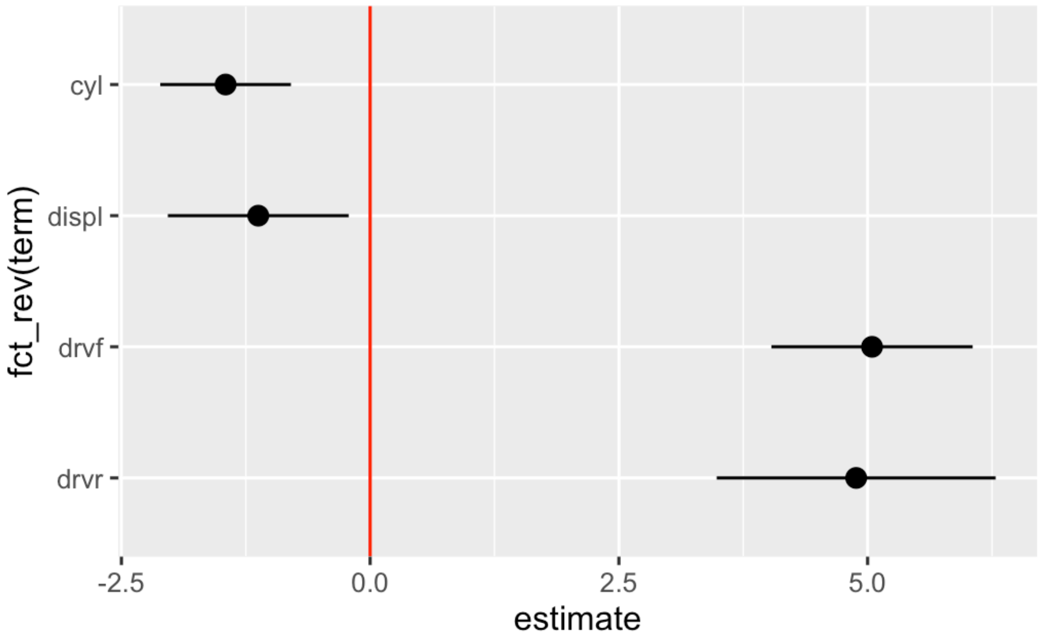
Cylinders held at their mean; colored/filled/facetted by drive

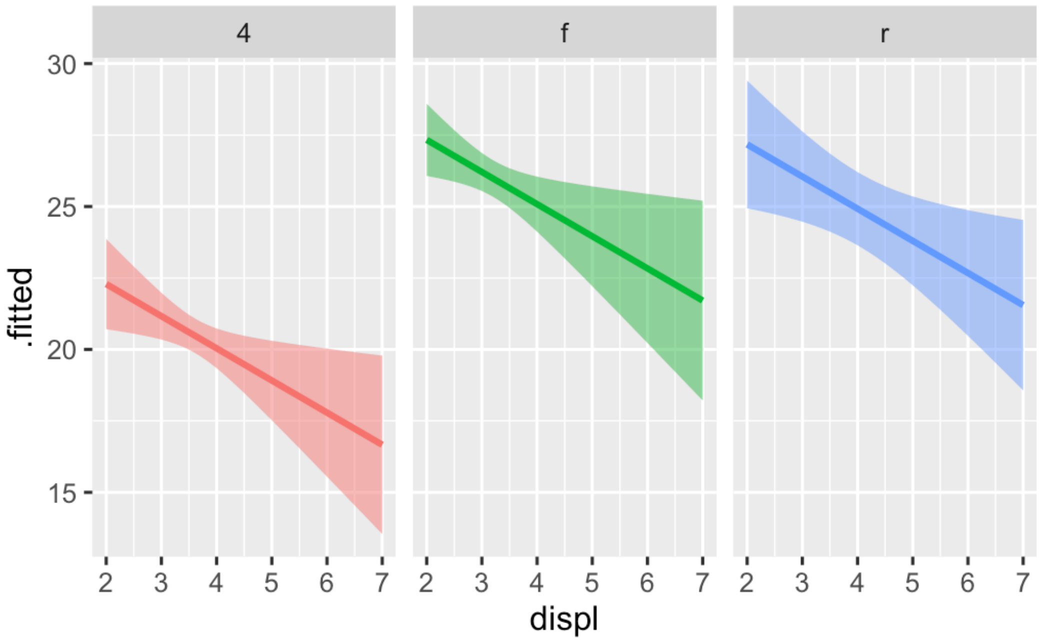
```
ggplot(predicted_mpg_fancy,
       aes(x = displ, y = .fitted)) +
  geom_ribbon(aes(ymin = .fitted +
                    (-1.96 * .se.fit),
                  ymax = .fitted +
                    (1.96 * .se.fit),
                  fill = drv),
              alpha = 0.5) +
  geom_line(aes(color = drv), linewidth = 1)
  guides(fill = "none", color = "none") +
  facet_wrap(vars(drv))
```



Not just OLS!

These plots are for an OLS model built with `lm()`





Any type of statistical model

The same techniques work for pretty much any model R can run

Logistic, probit, and multinomial regression (ordered and unordered)

Multilevel (i.e. mixed and random effects) regression

Bayesian models
(These are extra pretty with the {tidybayes} package)

Machine learning models

If it has coefficients and/or if it makes predictions, you can (and should) visualize it!