

# Relationships

## **Session 7**

PMAP 8921: Data Visualization with R  
Andrew Young School of Policy Studies  
May 2020

# Plan for today

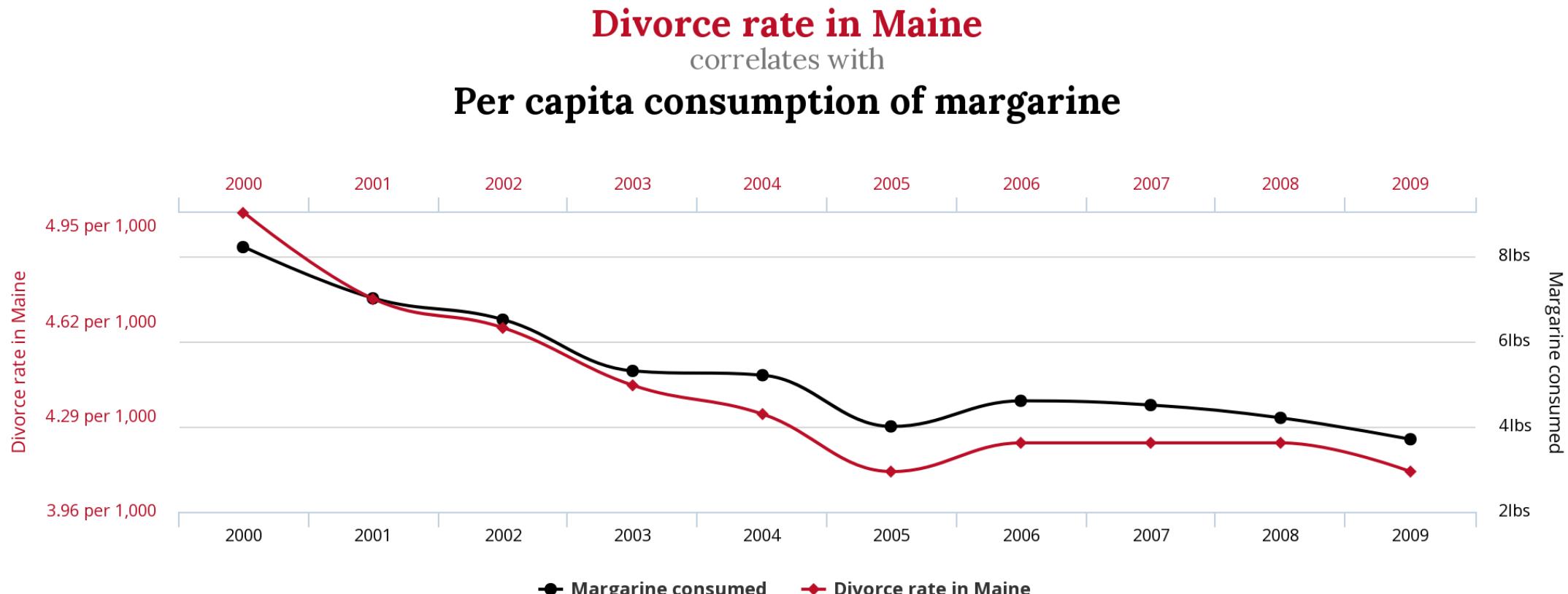
The dangers of dual y-axes

Visualizing correlations

Visualizing regressions

# The dangers of dual y-axes

# Stop eating margarine!



tylervigen.com

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...

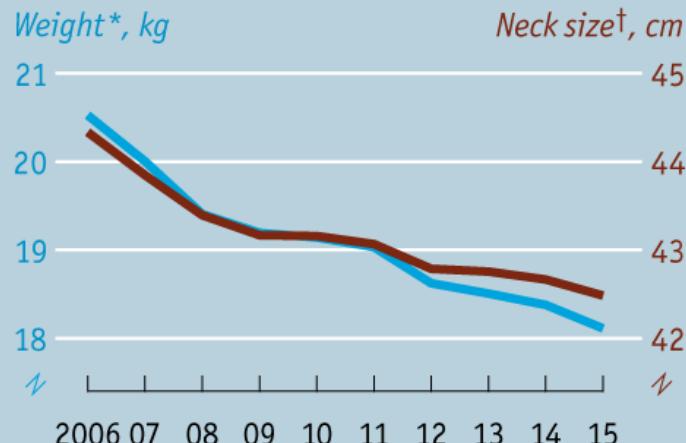
...you can force the two trends to line up however you want!

# It even happens in *The Economist*!

Original

## Fit as a butcher's dog

Characteristics of dogs registered with the UK's Kennel Club, average when fully grown



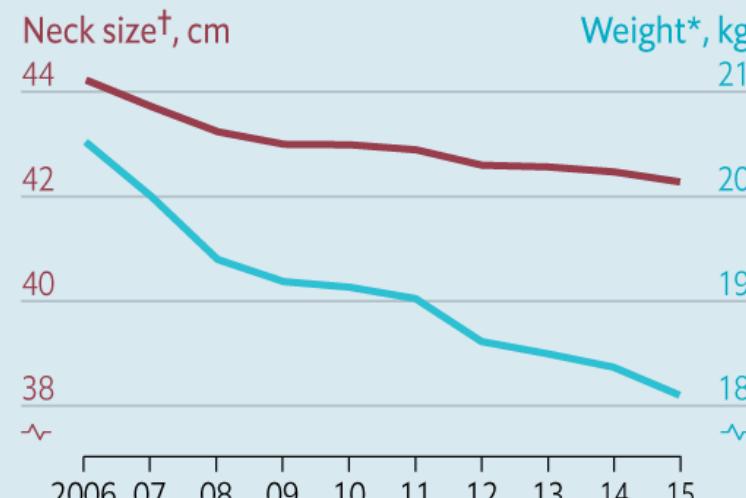
Sources: Kennel Club;  
*The Economist*

\*Where at least 50 are  
registered per year    †Where at  
least 100 are registered per year

Better

## Fit as a butcher's dog

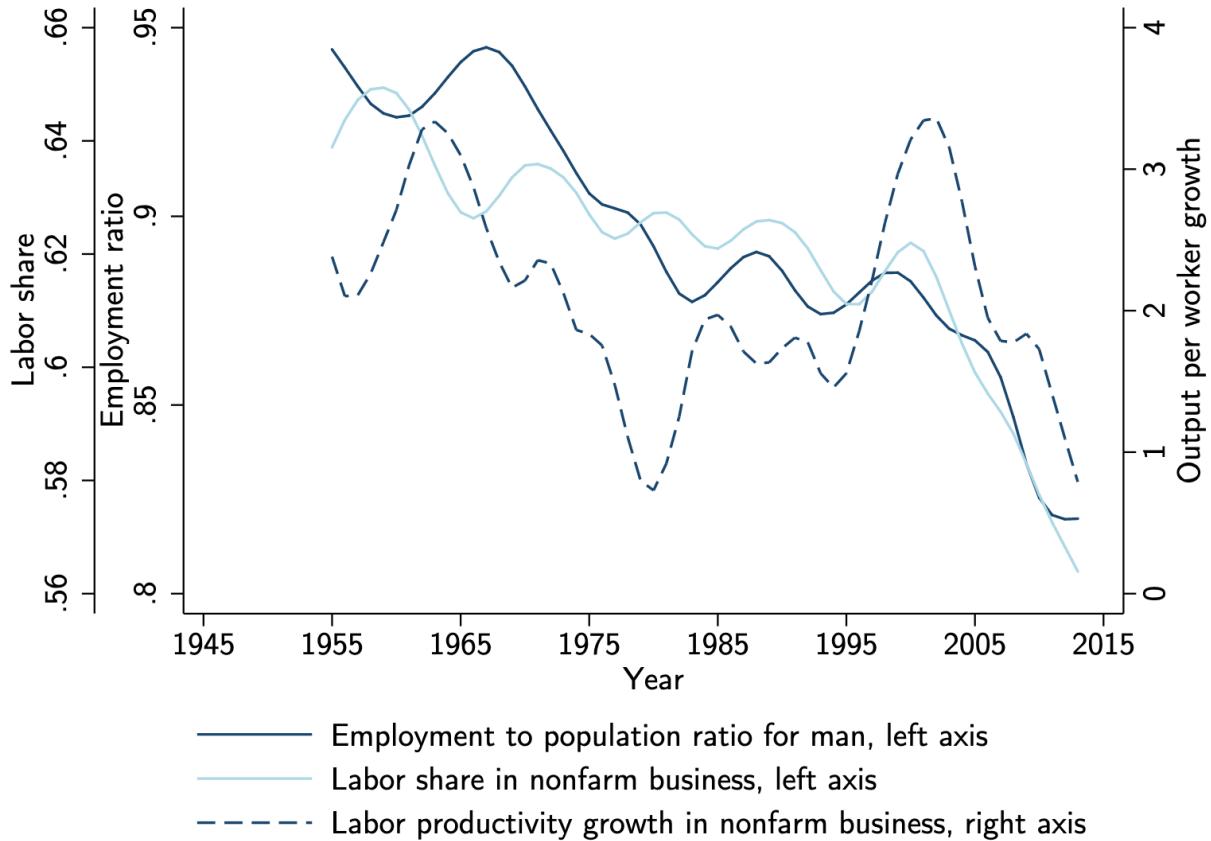
Characteristics of dogs registered with the UK's Kennel Club, average when fully grown



Sources: Kennel Club;  
*The Economist*

\*Where at least 50 are registered per year  
†Where at least 100 are registered per year

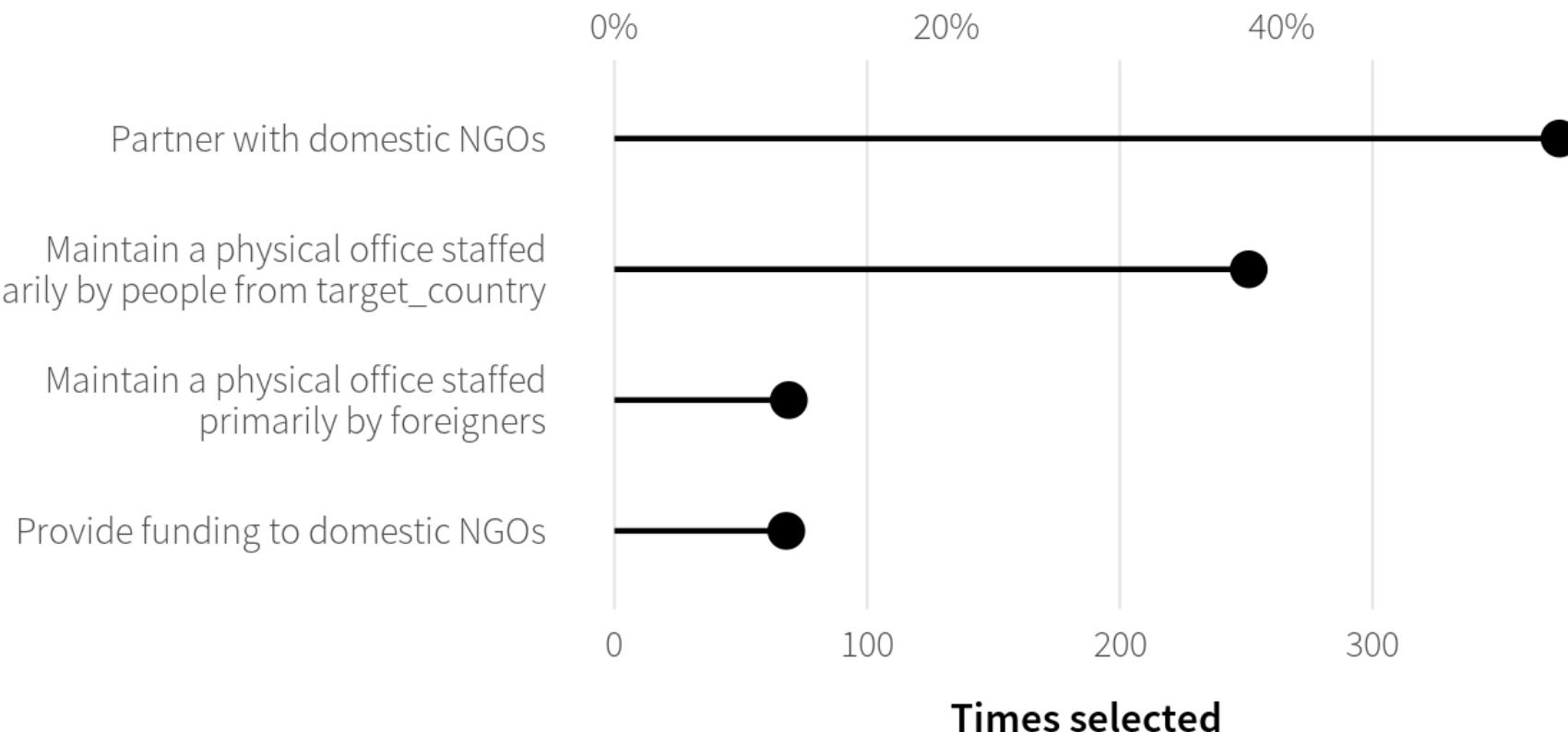
# The rare triple y-axis!



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?

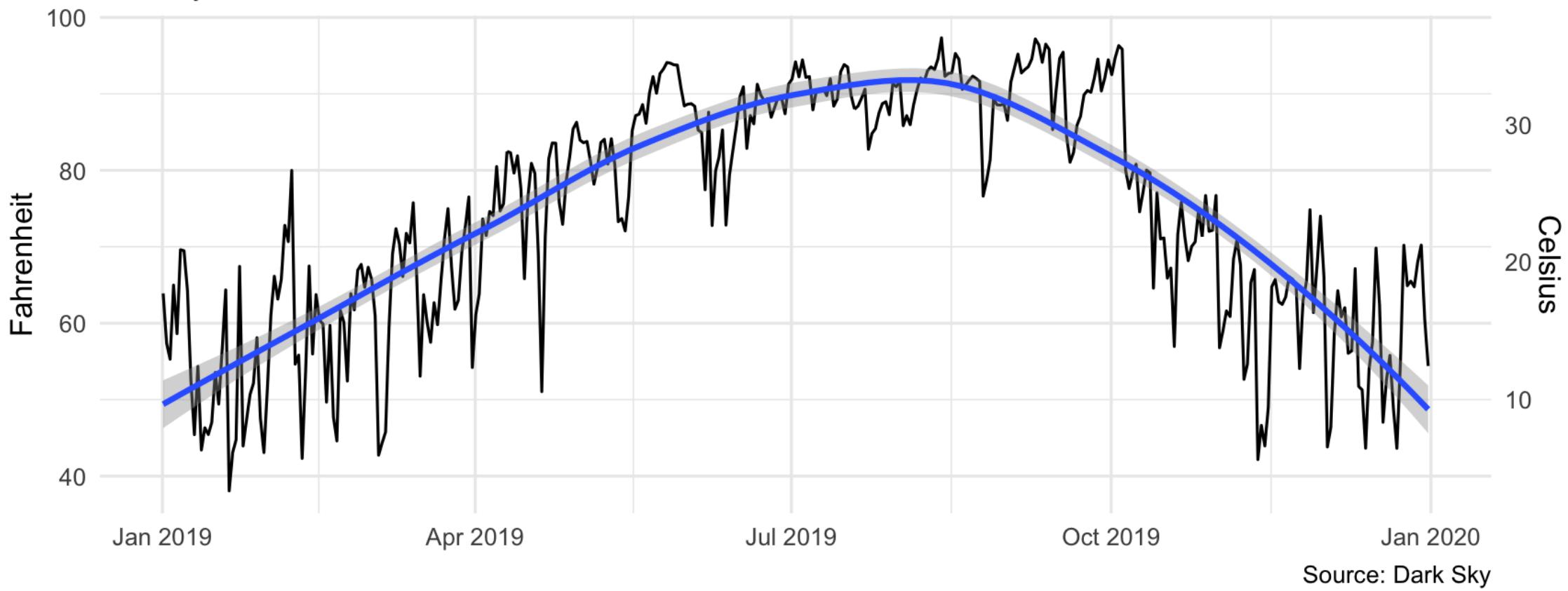
## When the two axes measure the same thing



# When is it legal?

## Daily high temperatures in Atlanta

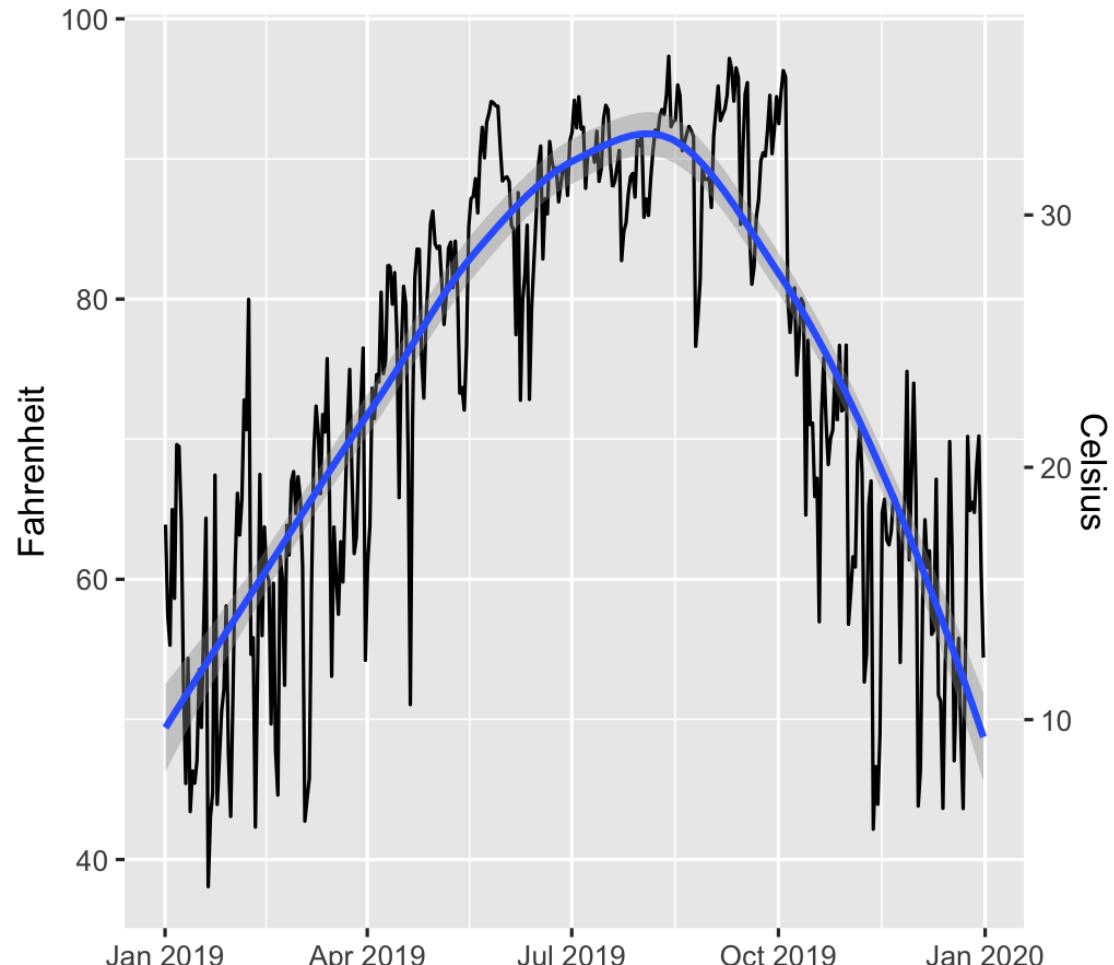
January 1 2019–December 31, 2019



# Adding a second scale in 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")
```

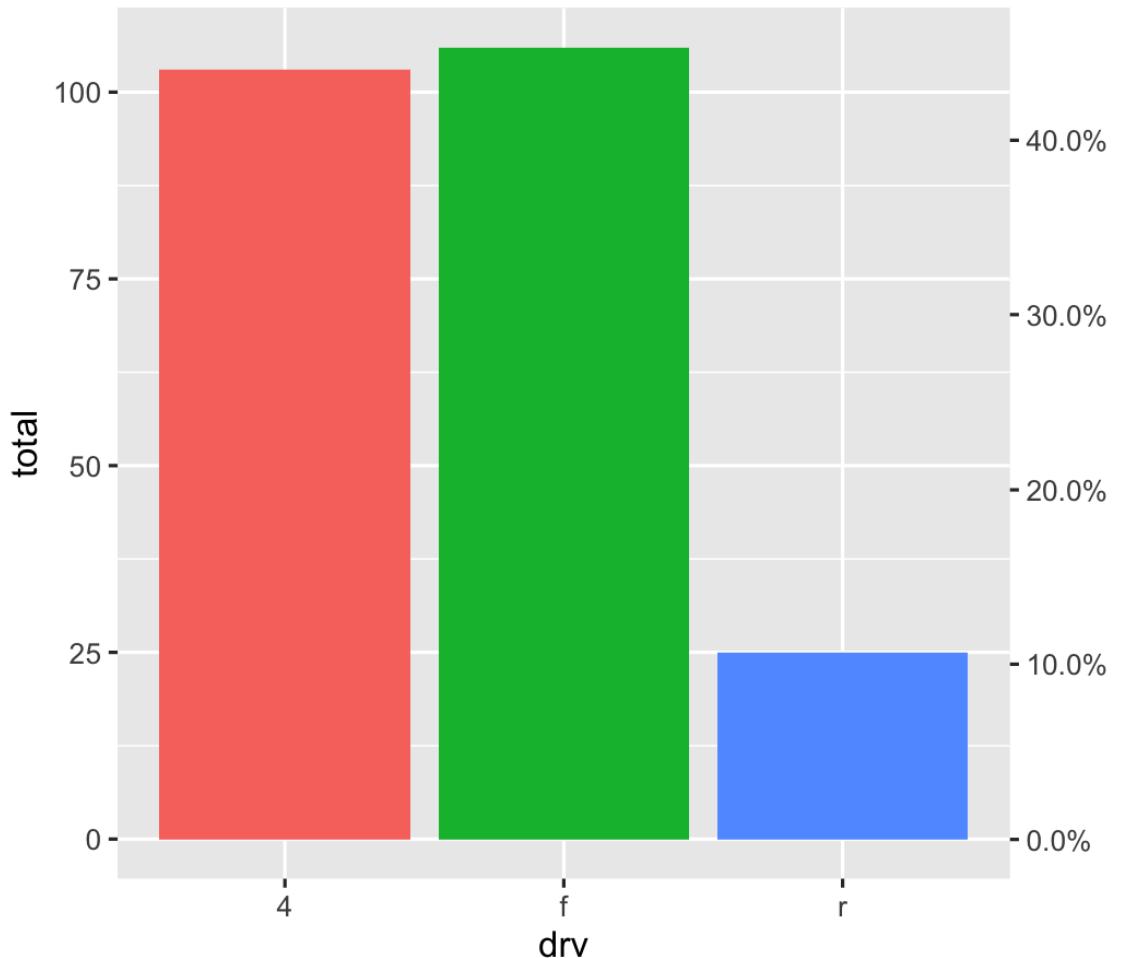


# Adding a second scale in R

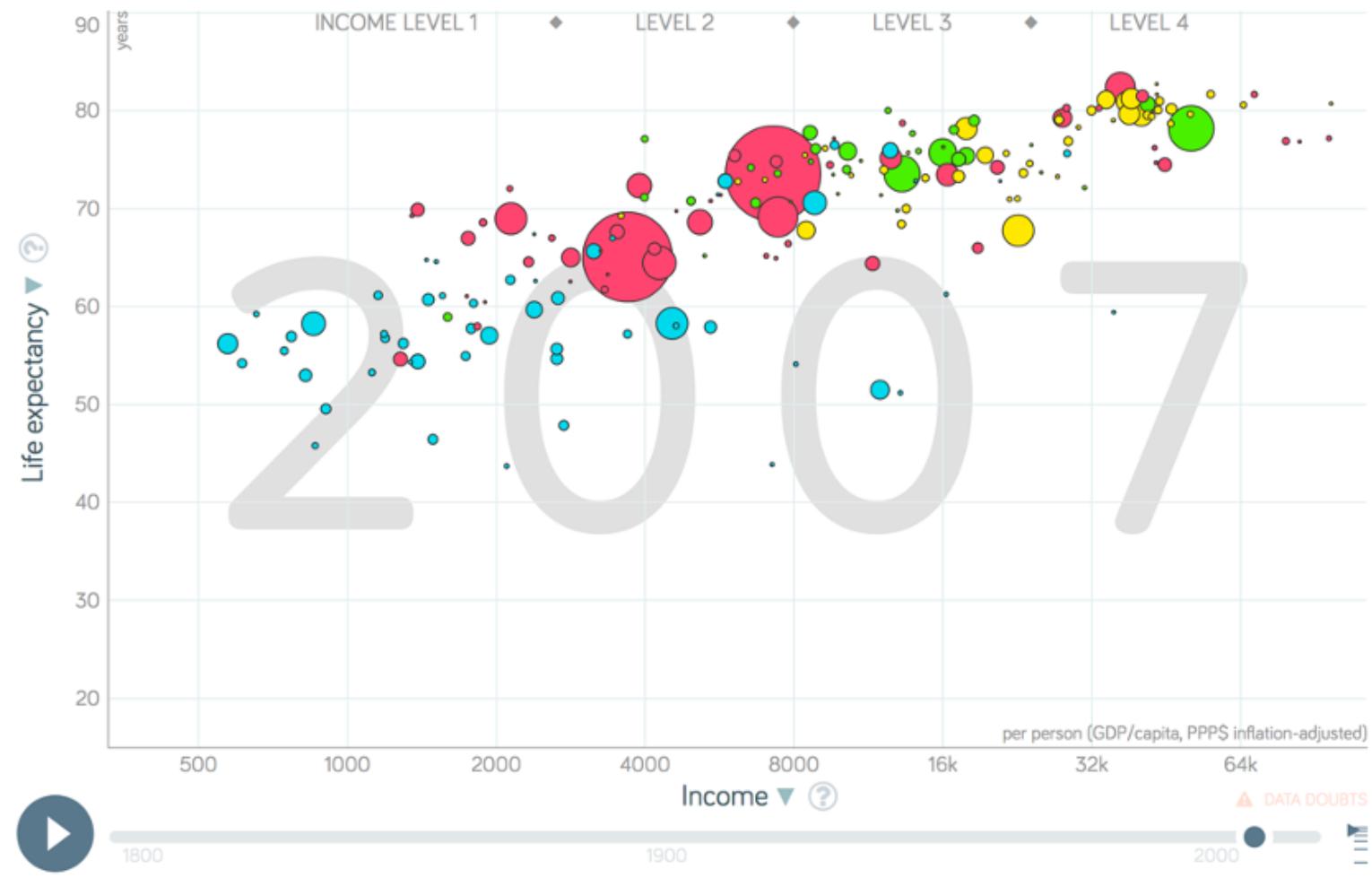
```
car_counts <- mpg %>%
  group_by(drv) %>%
  summarize(total = n())

total_cars <- sum(car_counts$total)

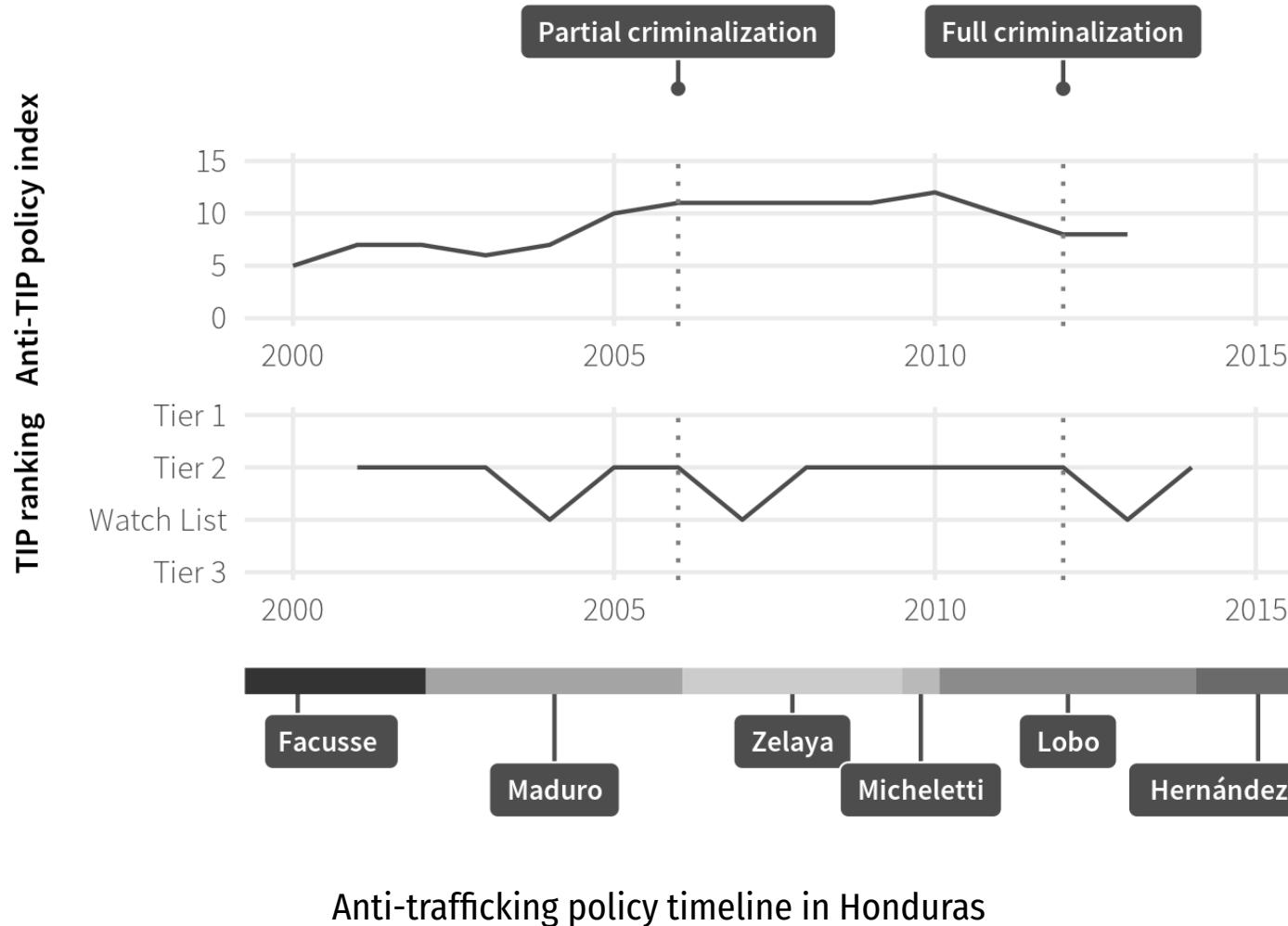
ggplot(car_counts,
       aes(x = drv, y = total,
            fill = drv)) +
  geom_col() +
  scale_y_continuous(
    sec.axis = sec_axis(
      trans = ~ . / total_cars,
      labels = scales::percent)
  ) +
  guides(fill = FALSE)
```



# Alternative 1: Use another aesthetic



# Alternative 2: Use multiple plots



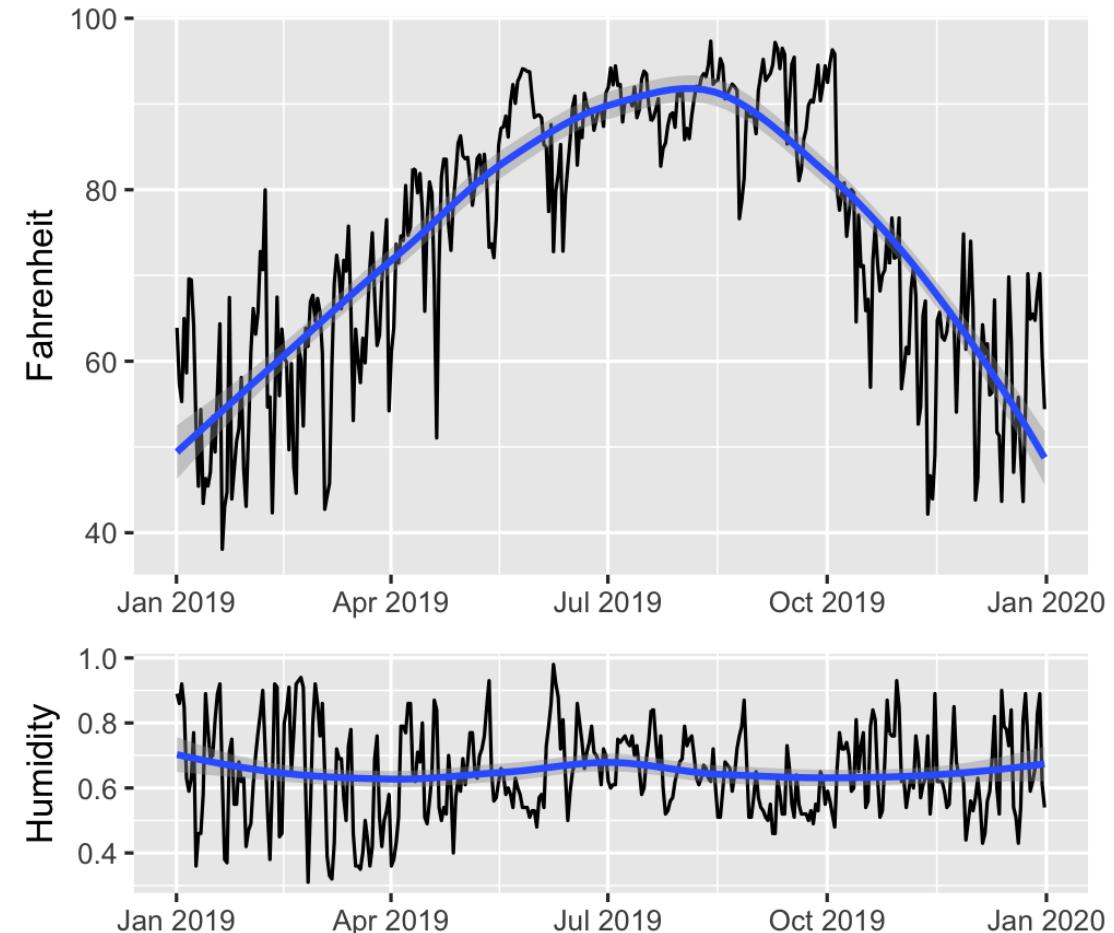
# Alternative 2: Use multiple plots

```
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?

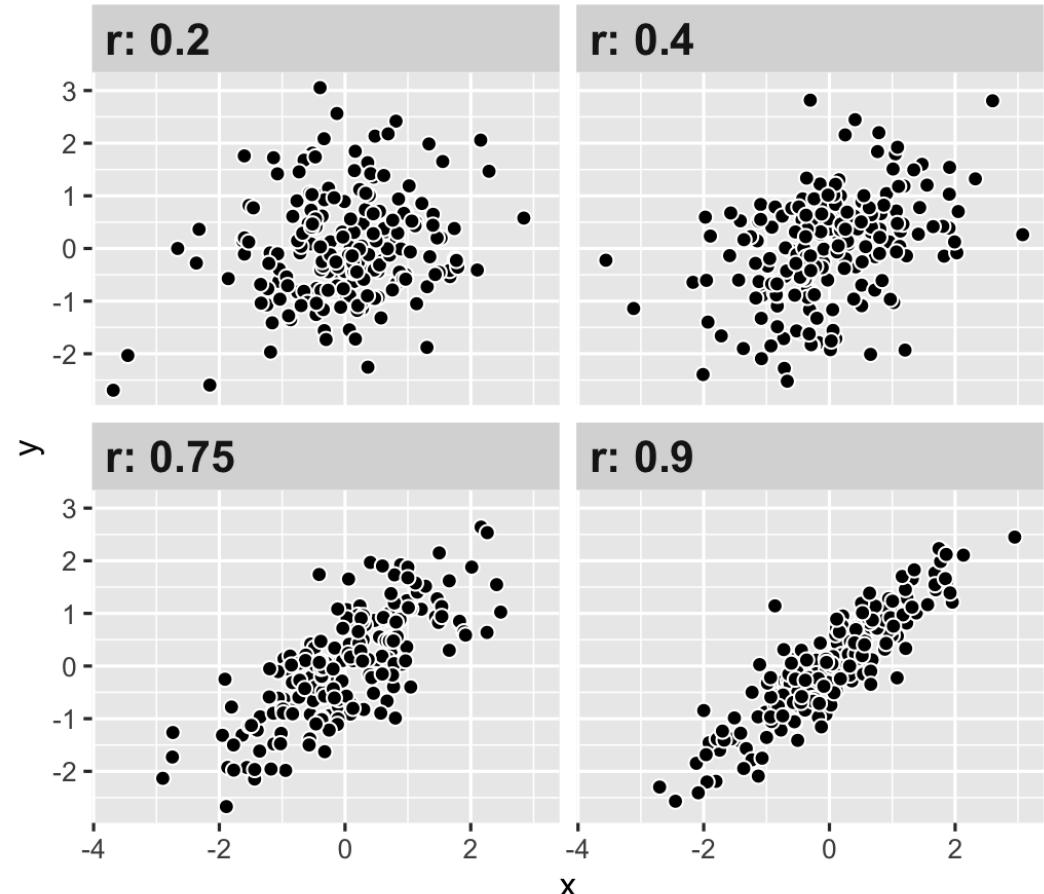
$$r_{x,y} = \frac{\text{cov}(x, y)}{\sigma_x \sigma_y}$$

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

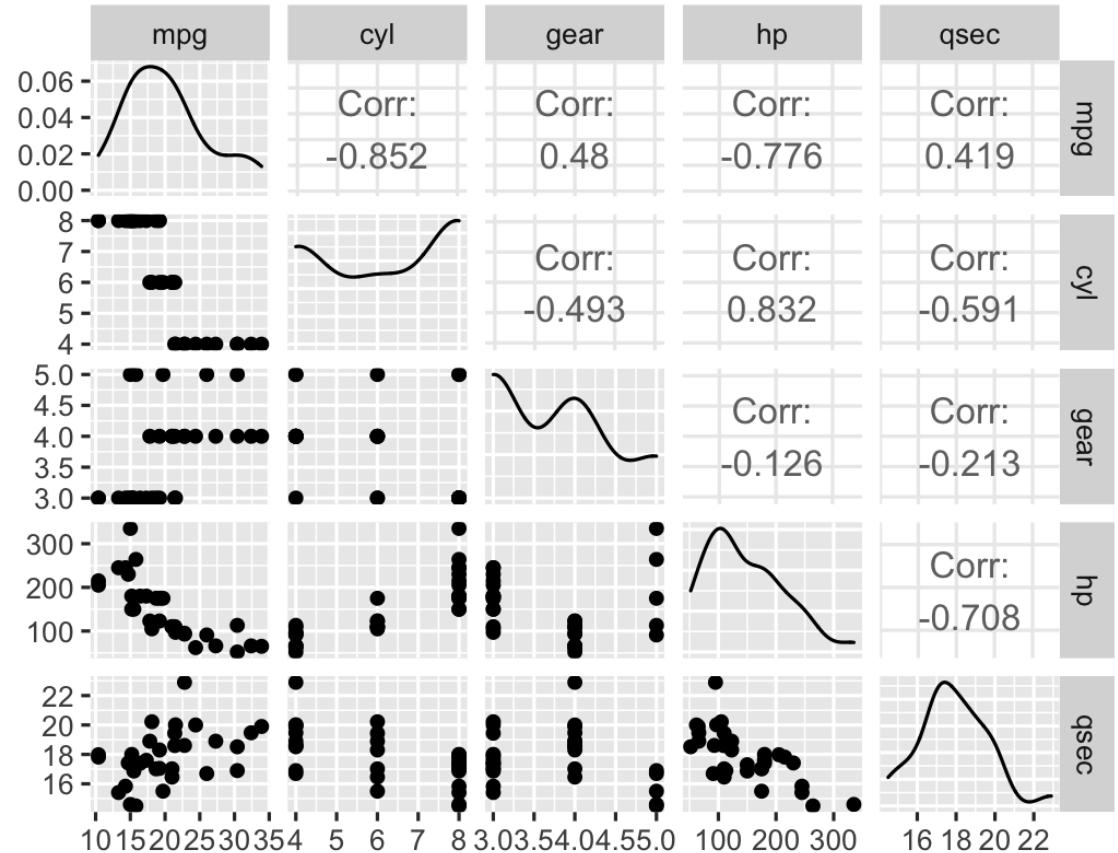
# Correlation values

<b>r</b>	<b>Rough meaning</b>
$\pm 0.1\text{--}0.3$	Modest
$\pm 0.3\text{--}0.5$	Moderate
$\pm 0.5\text{--}0.8$	Strong
$\pm 0.8\text{--}0.9$	Very strong

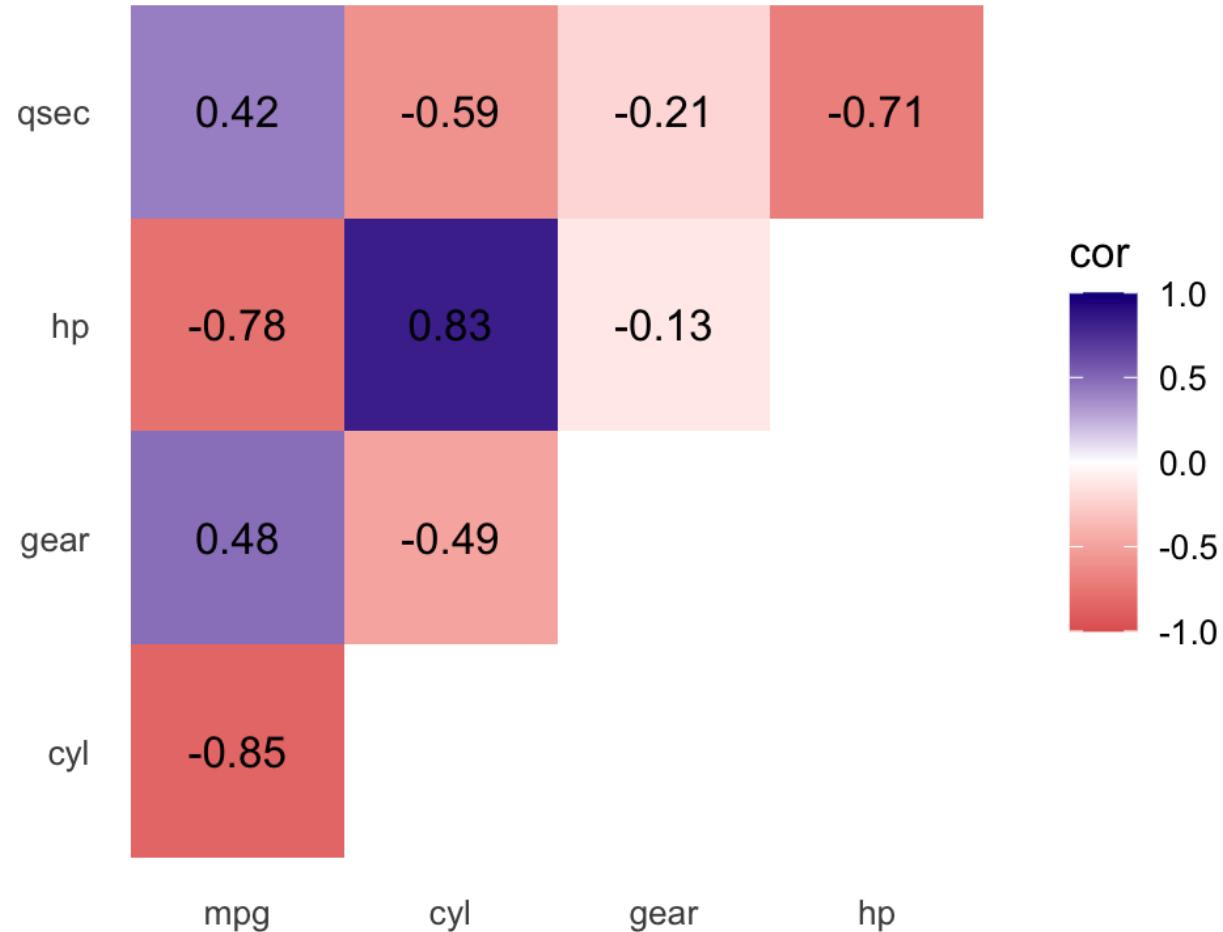


# Scatterplot matrices

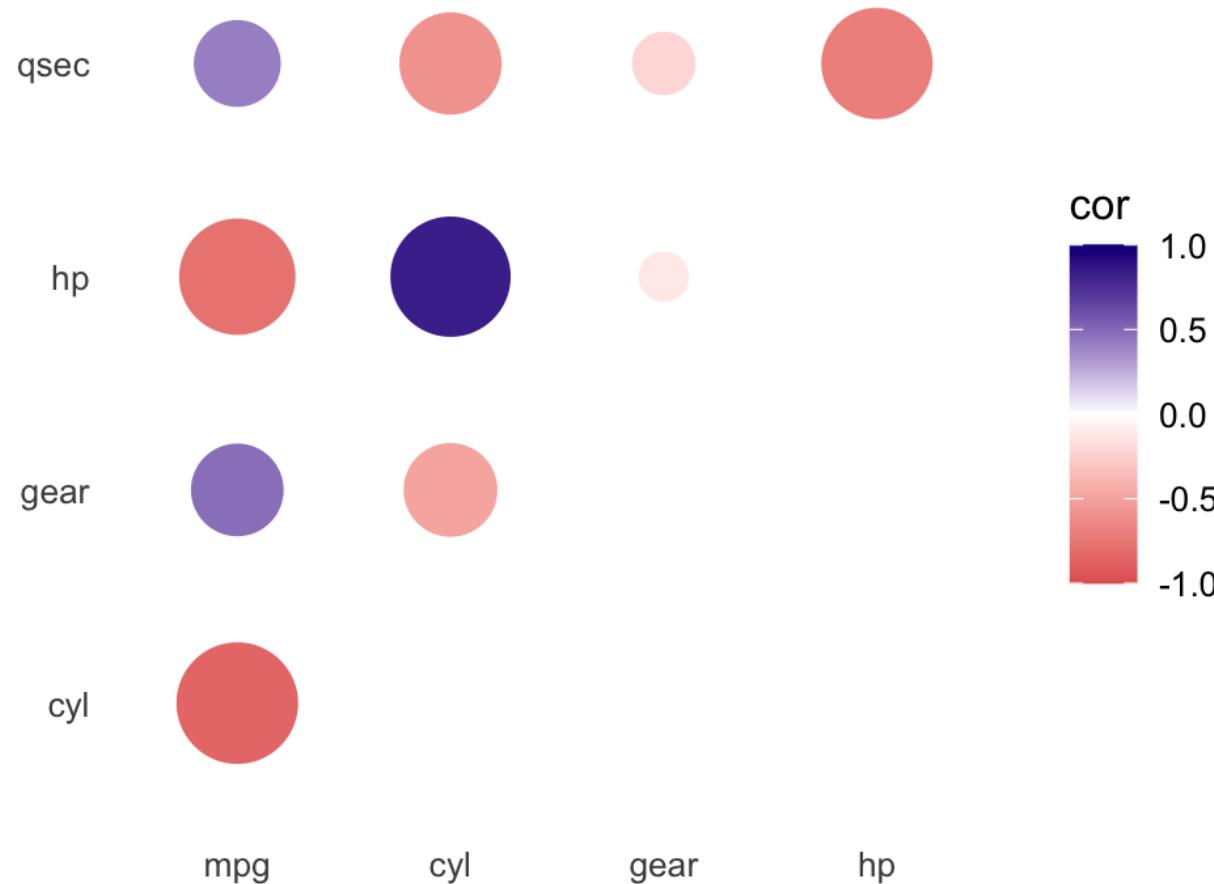
```
library(GGally)  
  
cars_smaller <- mtcars %>%  
  select(mpg, cyl, gear, hp, qsec)  
  
ggpairs(cars_smaller)
```



# Correlograms: Heatmaps

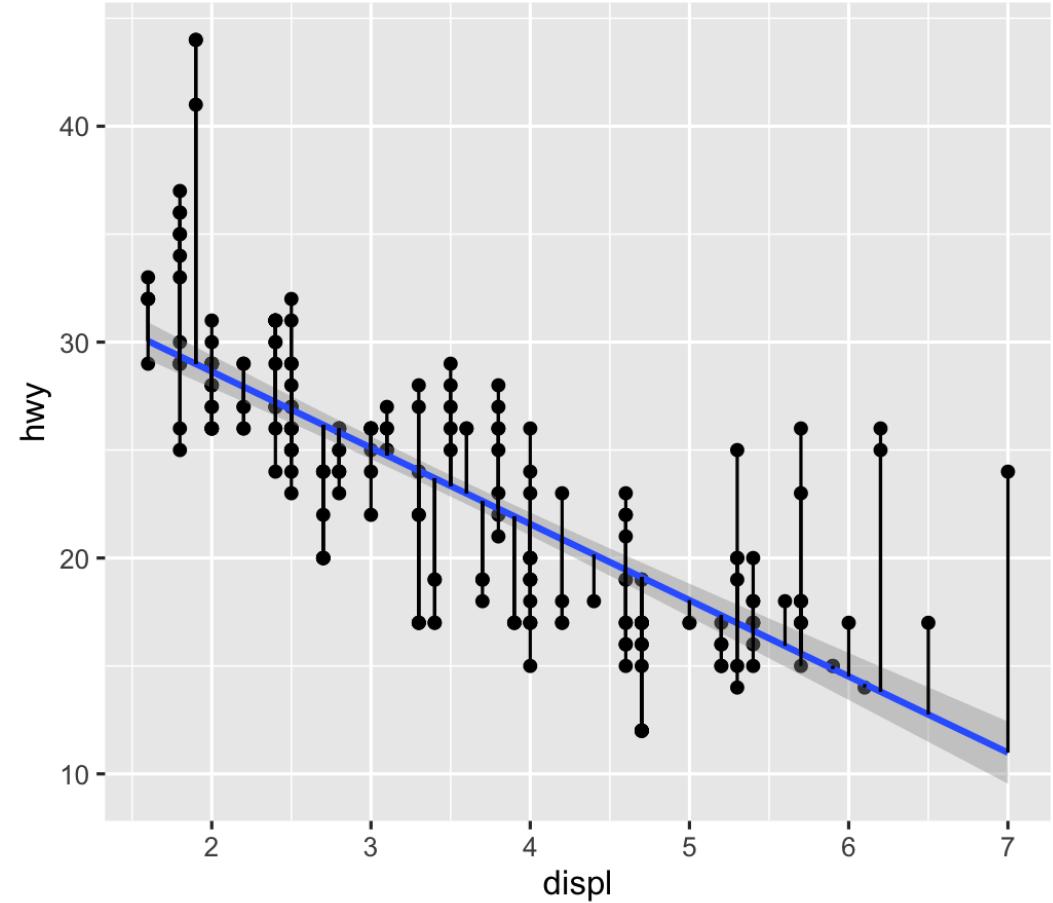
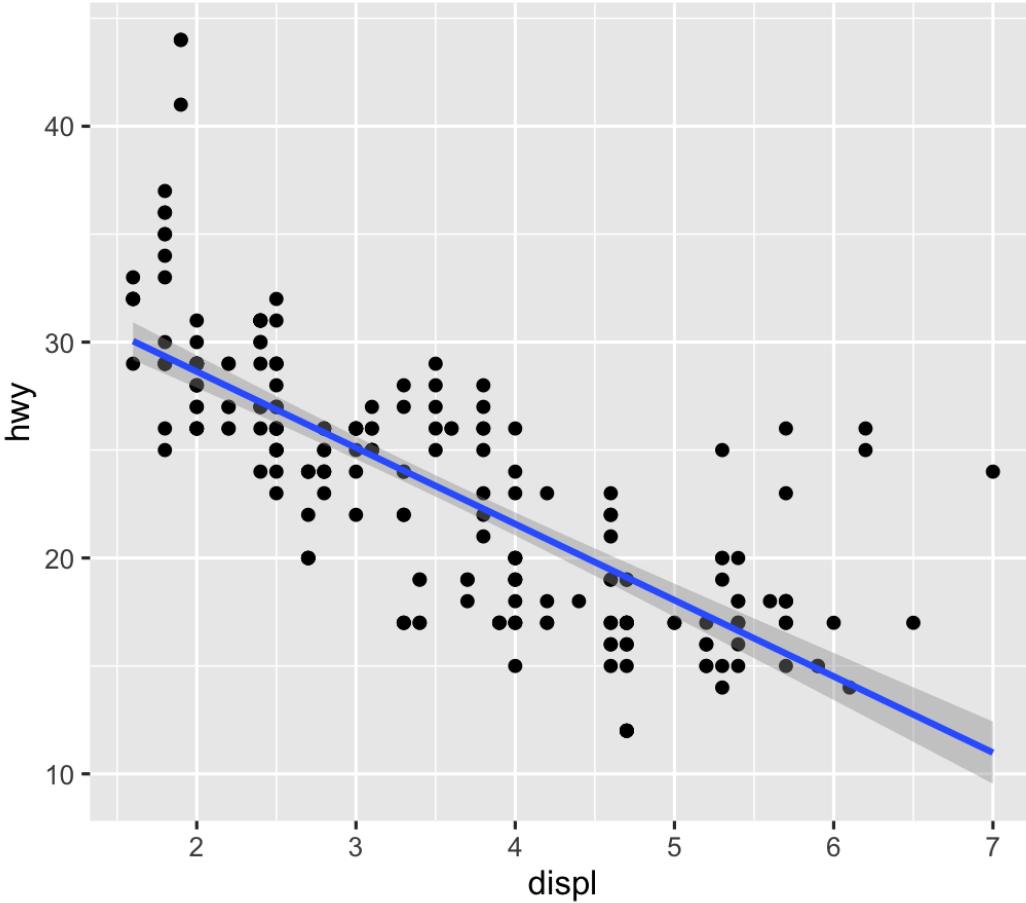


# Correlograms: Points



# Visualizing regressions

# Drawing lines



# Drawing lines with math

$$y = mx + b$$

---

$y$  A number

$x$  A number

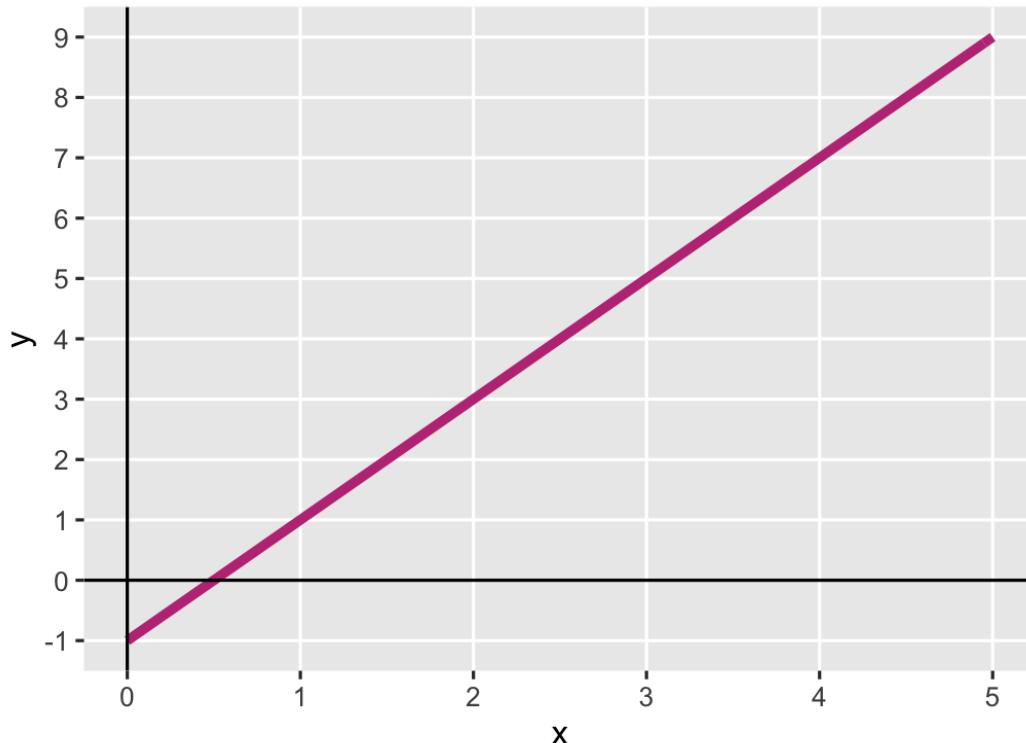
$m$  Slope ( $\frac{\text{rise}}{\text{run}}$ )

$b$  y-intercept

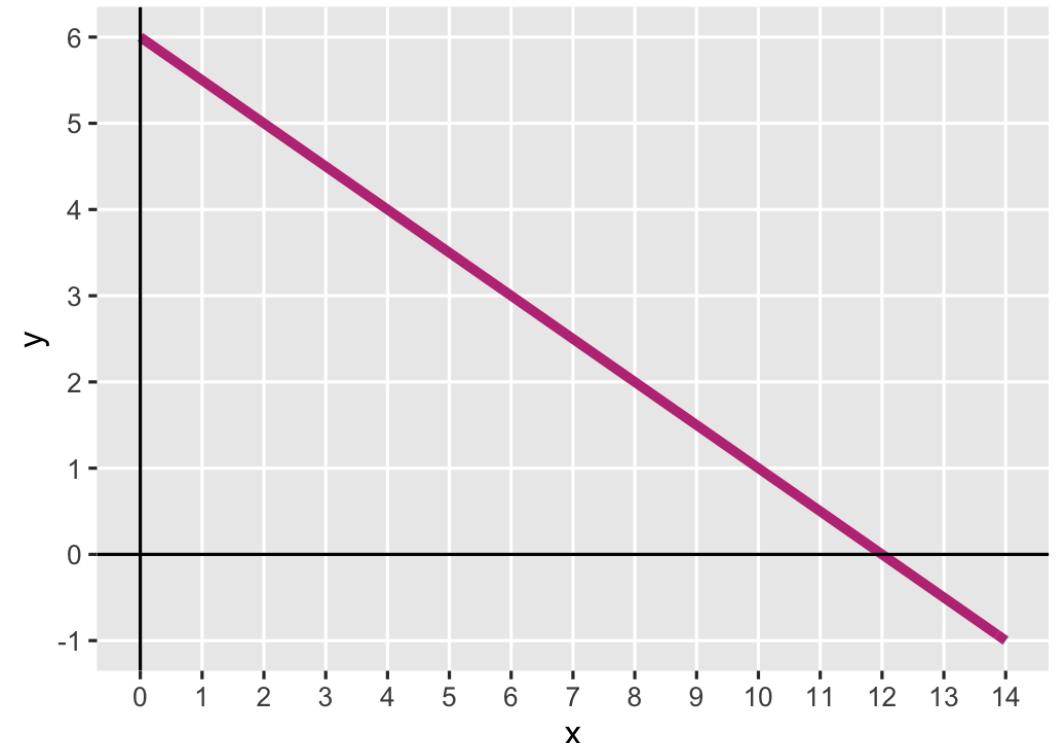
---

# 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)
$m$	$\beta_1$	Slope
$b$	$\beta_0$	y-intercept
	$\varepsilon$	Error (residuals)

# Building models in R

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

summary(name_of_model) # See model details
```

```
library(broom)

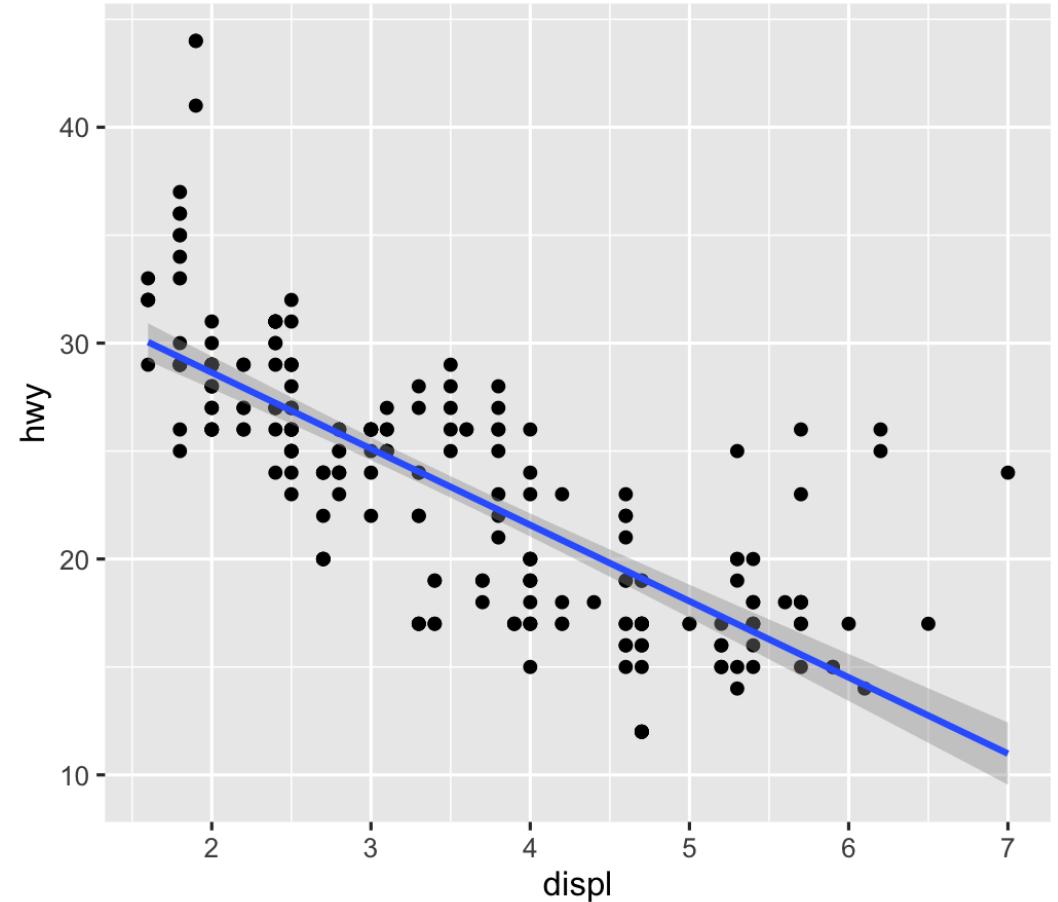
# Convert model results to a data frame for plotting
tidy(name_of_model)

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

# Modeling displacement and MPG

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

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



# Modeling displacement and MPG

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

```
## # A tibble: 2 x 7
##   term      estimate std.error statistic p.value conf.low conf.high
##   <chr>      <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)  35.7     0.720     49.6  2.12e-125    34.3     37.1
## 2 displ       -3.53     0.195    -18.2  2.04e- 46   -3.91    -3.15
```

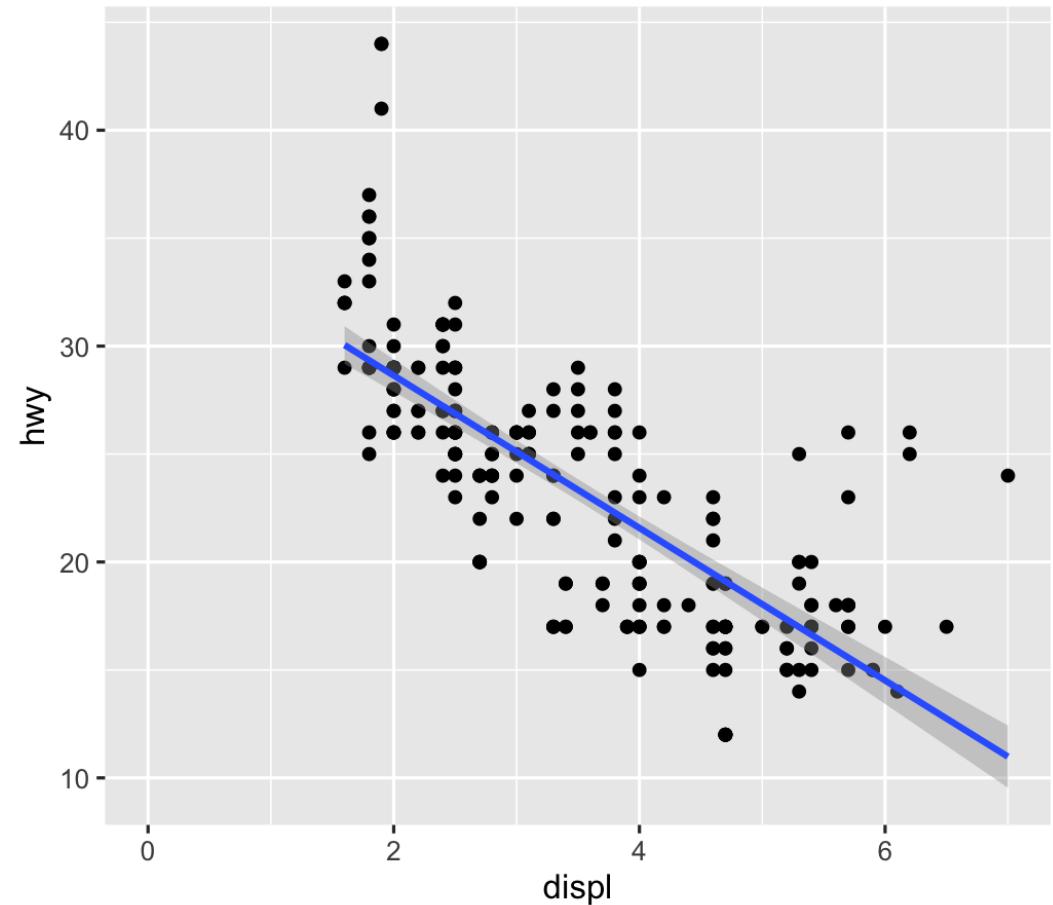
```
glance(car_model)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic p.value    df logLik     AIC     BIC
##   <dbl>        <dbl> <dbl>     <dbl>     <dbl> <int> <dbl> <dbl> <dbl>
## 1     0.587       0.585  3.84     329.  2.04e-46     2  -646. 1297. 1308.
## # ... with 2 more variables: deviance <dbl>, df.residual <int>
```

# Translating results to math

```
## # A tibble: 2 x 2
##   term      estimate
##   <chr>      <dbl>
## 1 (Intercept) 35.7
## 2 displ     -3.53
```

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



# Template for single variables

A one unit increase in X is *associated with*  
a  $\beta_1$  increase (or decrease) in Y, on average

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

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

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)
```

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

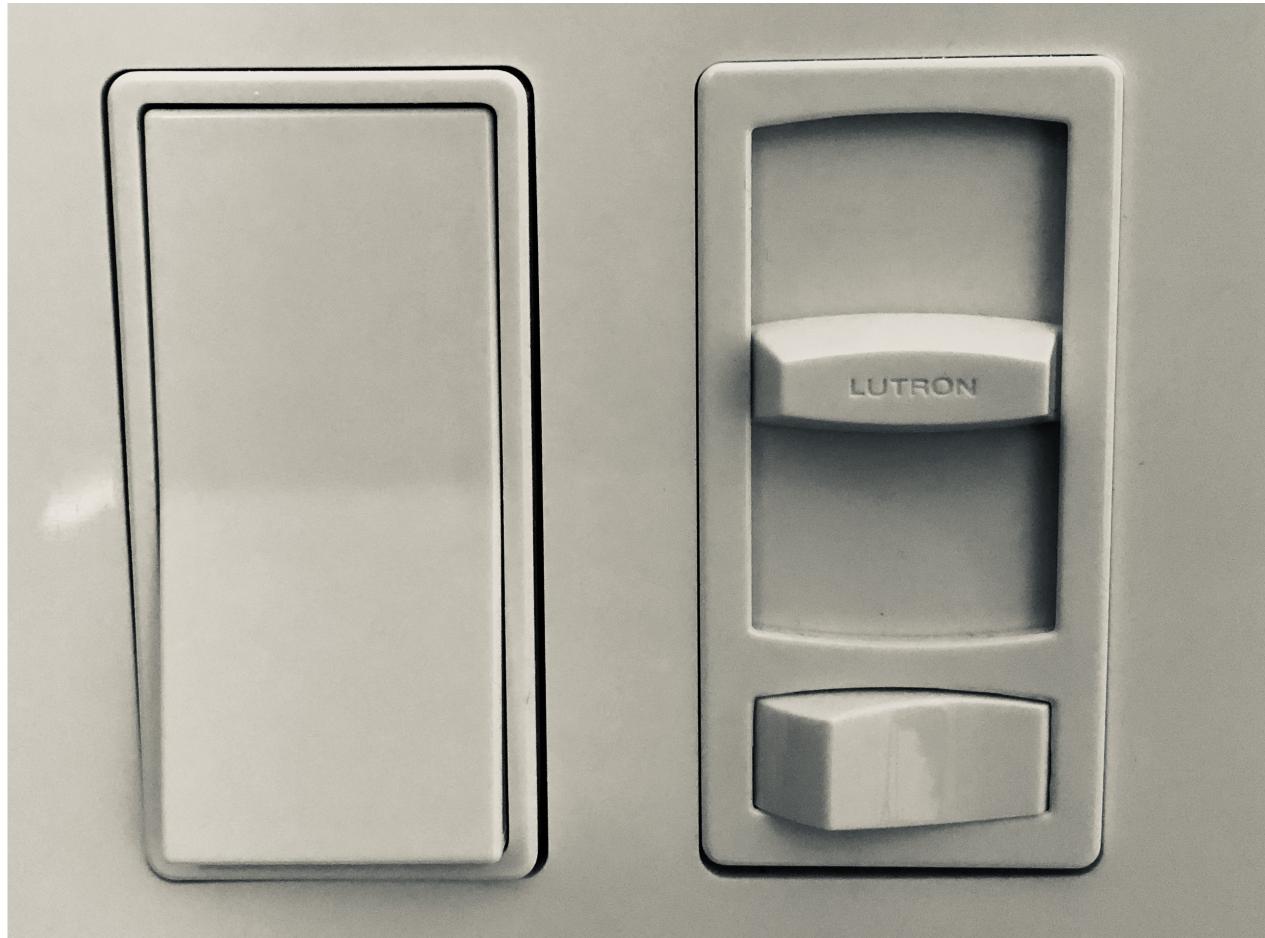
# Modeling lots of things and MPG

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

```
## # A tibble: 5 x 7
##   term      estimate std.error statistic p.value conf.low conf.high
##   <chr>     <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept) 33.1      1.03     32.1  9.49e-87    31.1     35.1
## 2 displ     -1.12      0.461    -2.44  1.56e- 2    -2.03    -0.215
## 3 cyl       -1.45      0.333    -4.36  1.99e- 5    -2.11    -0.796
## 4 drv:f      5.04      0.513     9.83  3.07e-19     4.03     6.06
## 5 drv:r      4.89      0.712     6.86  6.20e-11     3.48     6.29
```

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

# 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  $\beta_n$  increase (or decrease) in Y, on average*

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

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  $\beta_n$  units larger (or smaller) in  $X_n$ , compared to  $X_{omitted}$ , on average*

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

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

# Good luck visualizing 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()`

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

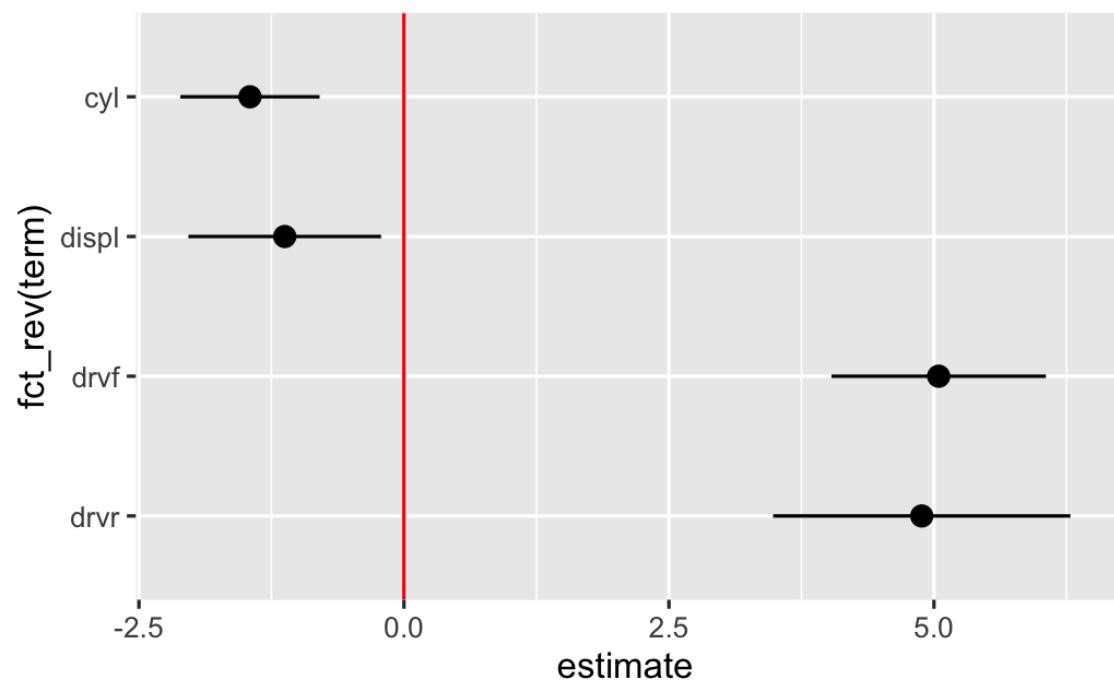
car_coefs <- tidy(car_model_big, conf.int = TRUE) %>%
  filter(term != "(Intercept)") # We can typically skip plotting the intercept, so remove it
car_coefs
```

```
## # A tibble: 4 x 7
##   term estimate std.error statistic p.value conf.low conf.high
##   <chr>    <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1 displ    -1.12     0.461    -2.44 1.56e- 2    -2.03    -0.215
## 2 cyl      -1.45     0.333    -4.36 1.99e- 5    -2.11    -0.796
## 3drvf     5.04      0.513     9.83 3.07e-19     4.03     6.06
## 4drvrv    4.89      0.712     6.86 6.20e-11     3.48     6.29
```

# 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

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

```
## # A tibble: 6 x 3  
##   displ   cyl  drv  
##   <dbl> <dbl> <chr>  
## 1     2    5.89 f  
## 2     2.1   5.89 f  
## 3     2.2   5.89 f  
## 4     2.3   5.89 f  
## 5     2.4   5.89 f  
## 6     2.5   5.89 f
```

# Marginal effects plots

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

```
predicted_mpg <- augment(car_model_big, newdata = cars_new_data)

head(predicted_mpg)
```

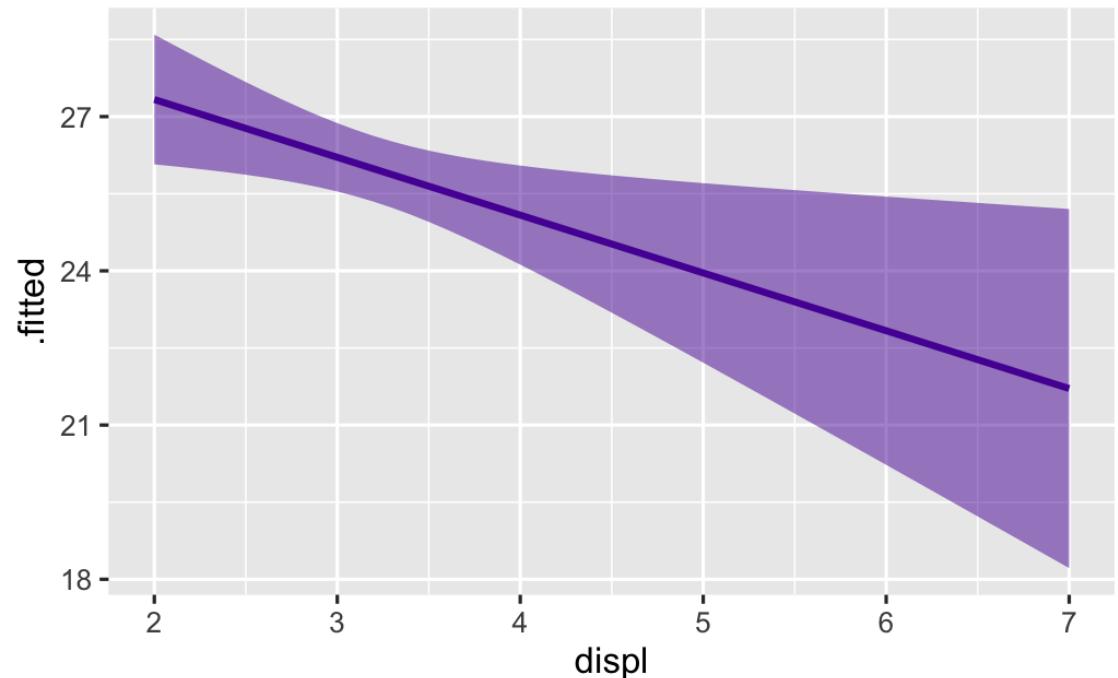
```
## # A tibble: 6 x 5
##   displ cyl drv .fitted .se.fit
##   <dbl> <dbl> <chr>   <dbl>    <dbl>
## 1     2   5.89 f       27.3    0.644
## 2   2.1   5.89 f       27.2    0.604
## 3   2.2   5.89 f       27.1    0.566
## 4   2.3   5.89 f       27.0    0.529
## 5   2.4   5.89 f       26.9    0.494
## 6   2.5   5.89 f       26.8    0.460
```

# 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

```
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)
```

```
## # A tibble: 6 x 3  
##   displ   cyl   drv  
##   <dbl> <dbl> <chr>  
## 1     2     5.89 f  
## 2     2     5.89 r  
## 3     2     5.89 4  
## 4    2.1     5.89 f  
## 5    2.1     5.89 r  
## 6    2.1     5.89 4
```

# Multiple effects at once

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

```
predicted_mpg_fancy <- augment(car_model_big, newdata = cars_new_data_fancy)

head(predicted_mpg_fancy)
```

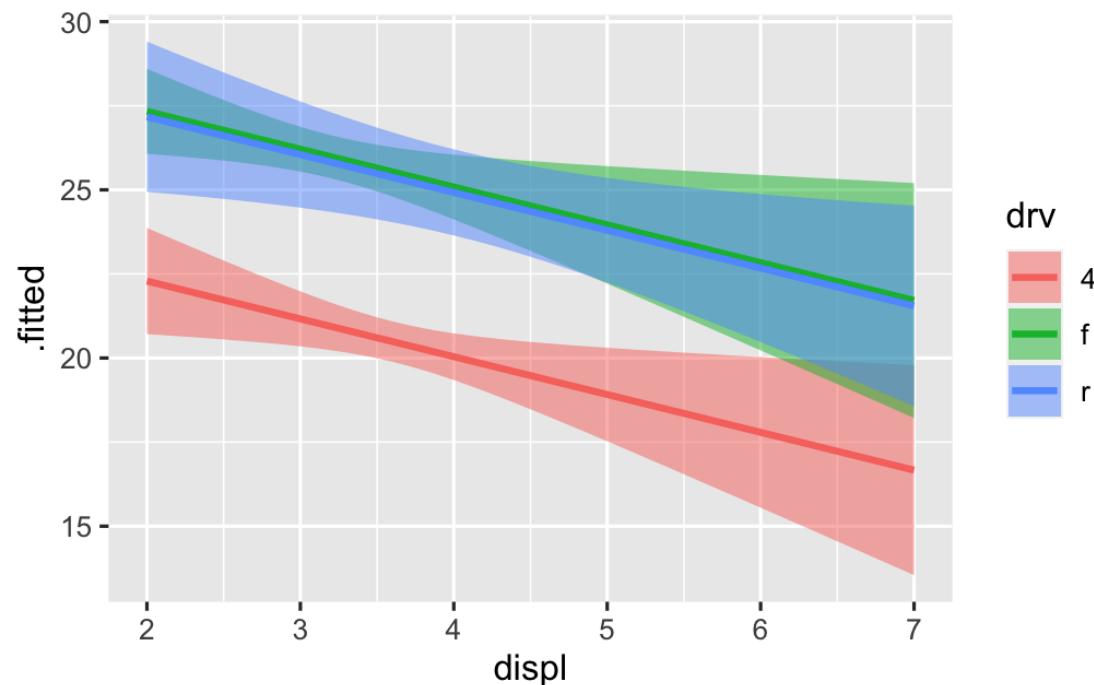
```
## # A tibble: 6 x 5
##   displ cyl drv   .fitted .se.fit
##   <dbl> <dbl> <chr>    <dbl>    <dbl>
## 1  2     5.89 f        27.3    0.644
## 2  2     5.89 r        27.2    1.14 
## 3  2     5.89 4       22.3    0.805
## 4  2.1   5.89 f        27.2    0.604
## 5  2.1   5.89 r        27.1    1.10 
## 6  2.1   5.89 4       22.2    0.763
```

# 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), size = 1)
```

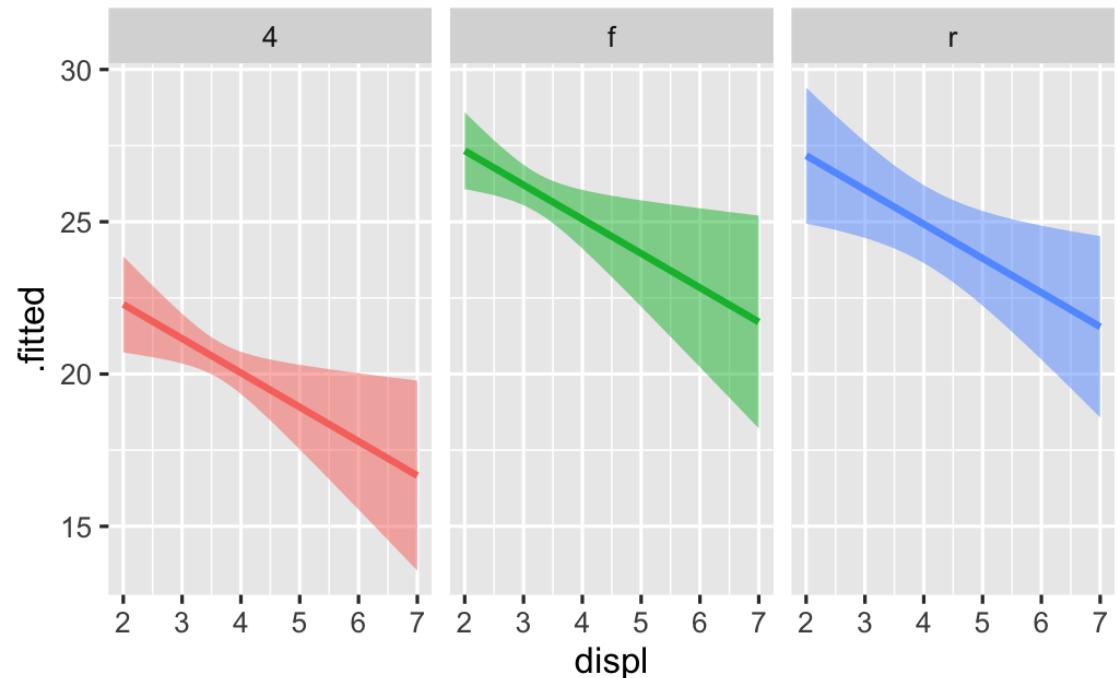


# Multiple effects at once

Plot the fitted values for each row

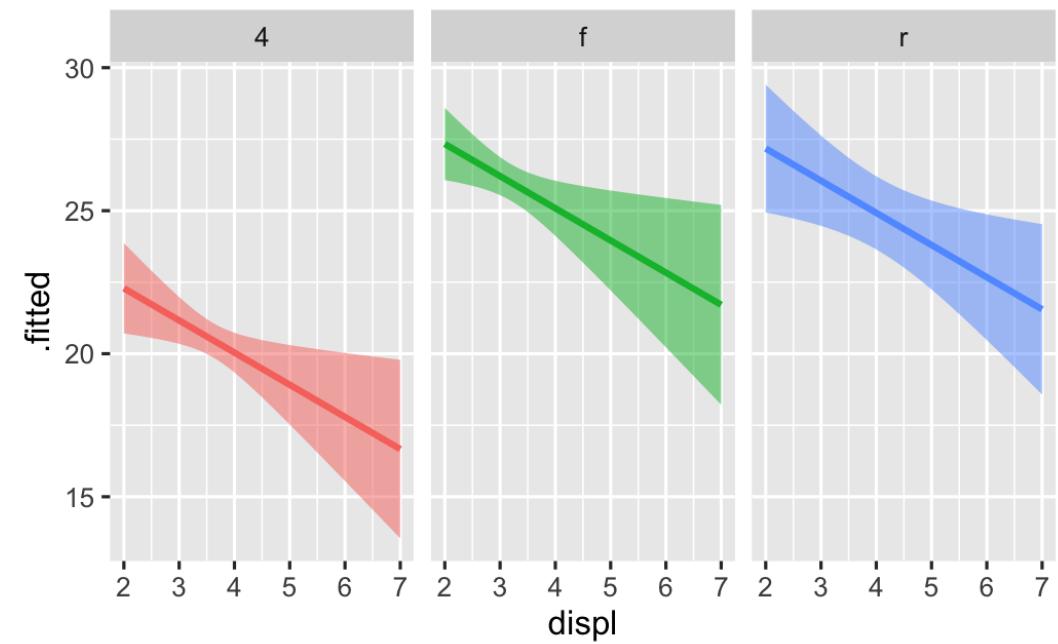
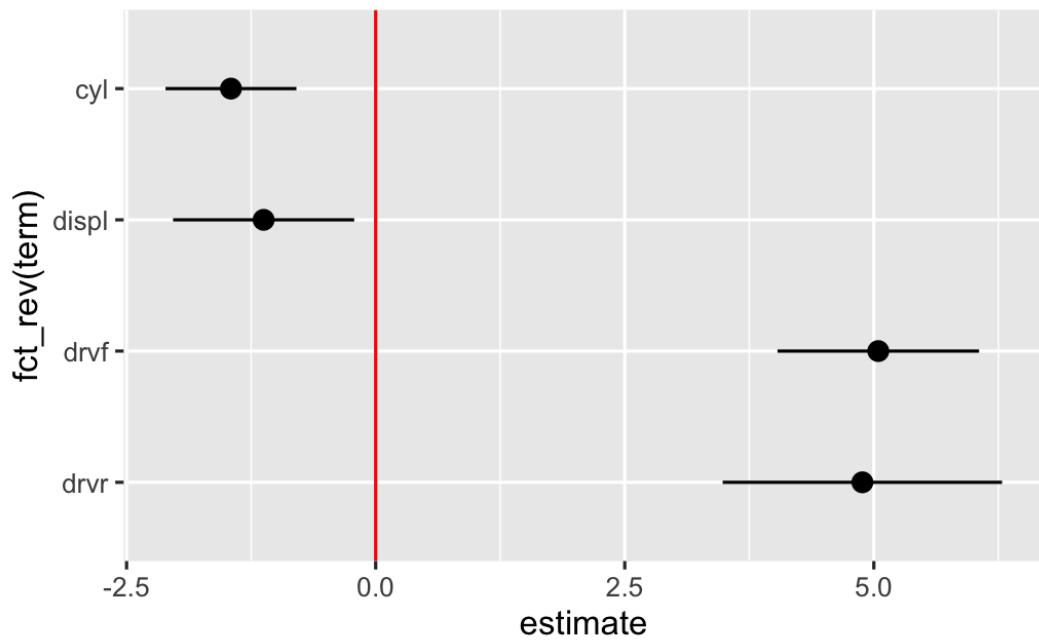
Cylinders held at their mean; colored/filled/facettted by drive

```
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), size = 1) +  
  guides(fill = FALSE, color = FALSE) +  
  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!