

Relationships

Week 8

AEM 2850: R for Business Analytics
Cornell Dyson
Spring 2022

Acknowledgements: **Andrew Heiss**

Announcements

Reminders:

- Mini Project 1 due April 1 ([link](#))
- Local R install instructions posted ([link](#))

Questions before we get started?

Plan for today

The dangers of dual y-axes

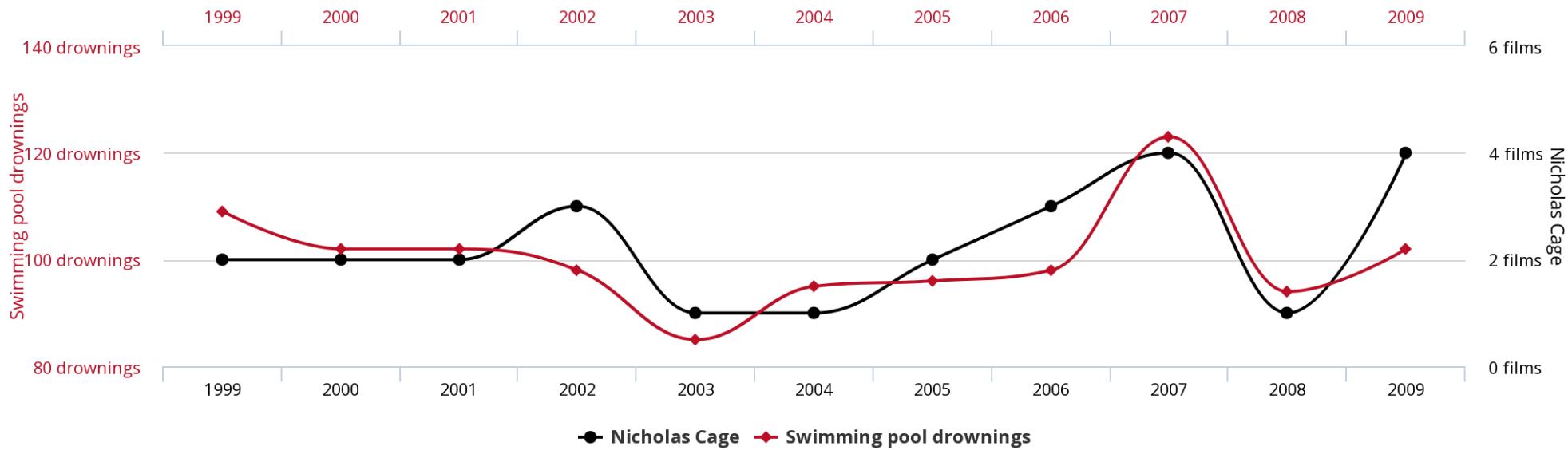
Visualizing correlations

Visualizing regressions

The dangers of dual y-axes

Oh no!

Number of people who drowned by falling into a pool
correlates with
Films Nicolas Cage appeared in



Source: Tyler Vigen's spurious correlations

tylervigen.com

Why not use double y-axes?

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

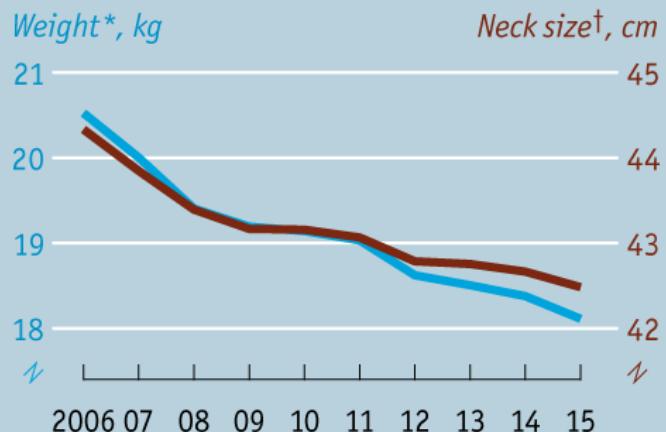
...you can force the two trends to line up however you want!

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



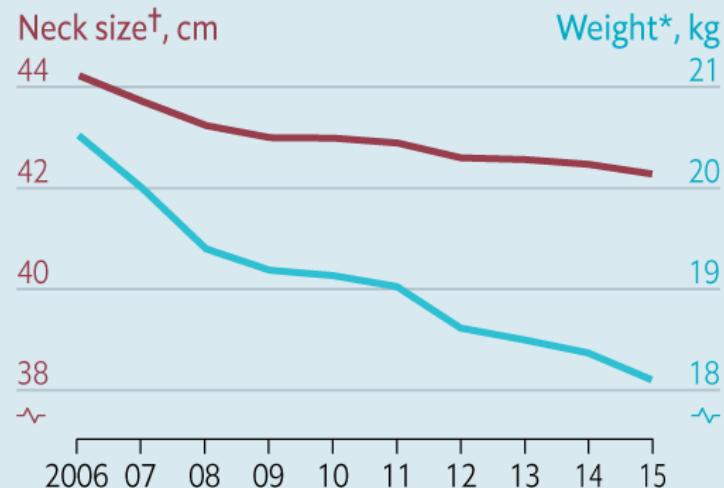
Sources: Kennel Club;
The Economist

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

Better

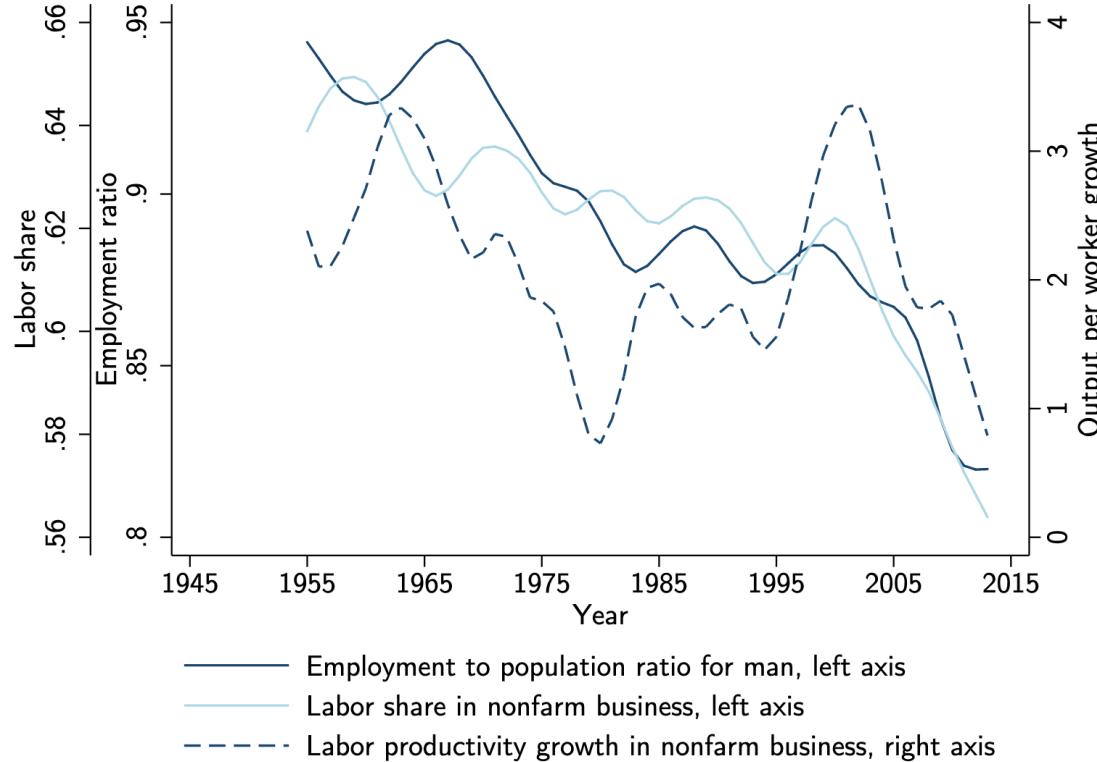
Fit as a butcher's dog

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



Sources: Kennel Club; *Where at least 50 are registered per year
The Economist †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 defensible?

When the two axes measure the same thing

How could we make this plot in R?

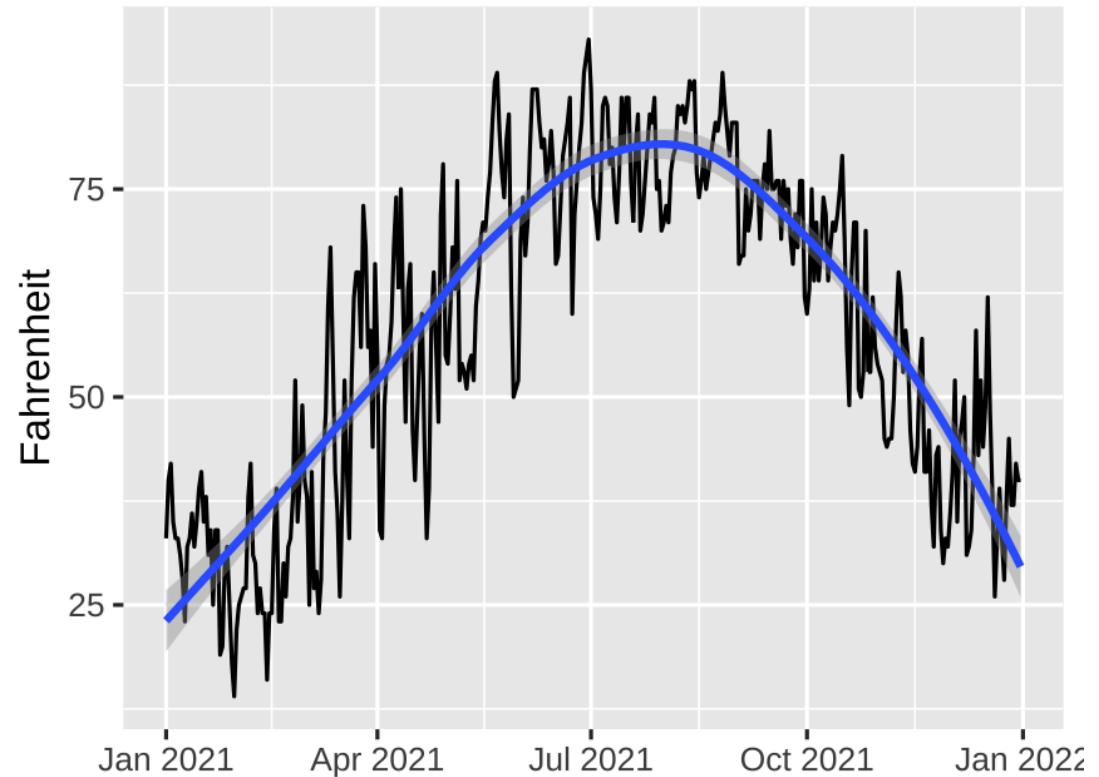
```
ithaca_weather <-  
  read_csv("data/ithaca-weather-2021.csv")  
  
ithaca_weather %>%  
  select(STATION, NAME, DATE, TMAX, SNOW)  
  
## # A tibble: 365 × 5  
##   STATION      NAME           DATE     TMAX   SNOW  
##   <chr>        <chr>          <date>    <dbl>  <dbl>  
## 1 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-01    33    0  
## 2 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-02    40    0  
## 3 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-03    42    0  
## 4 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-04    35   1.4  
## 5 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-05    33    0  
## 6 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-06    33    0  
## 7 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-07    31    0  
## 8 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-08    28    0  
## 9 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-09    23    0  
## 10 USC00304174 ITHACA CORNELL UNIVERSITY, NY US 2021-01-10   32    0  
## # ... with 355 more rows
```

Making the base plot in R

```
ggplot(ithaca_weather,  
       aes(x = DATE, y = TMAX)) +  
  geom_line() +  
  geom_smooth() +  
  labs(x = NULL, y = "Fahrenheit")
```

How could we add a second axis?

Do any functions come to mind?

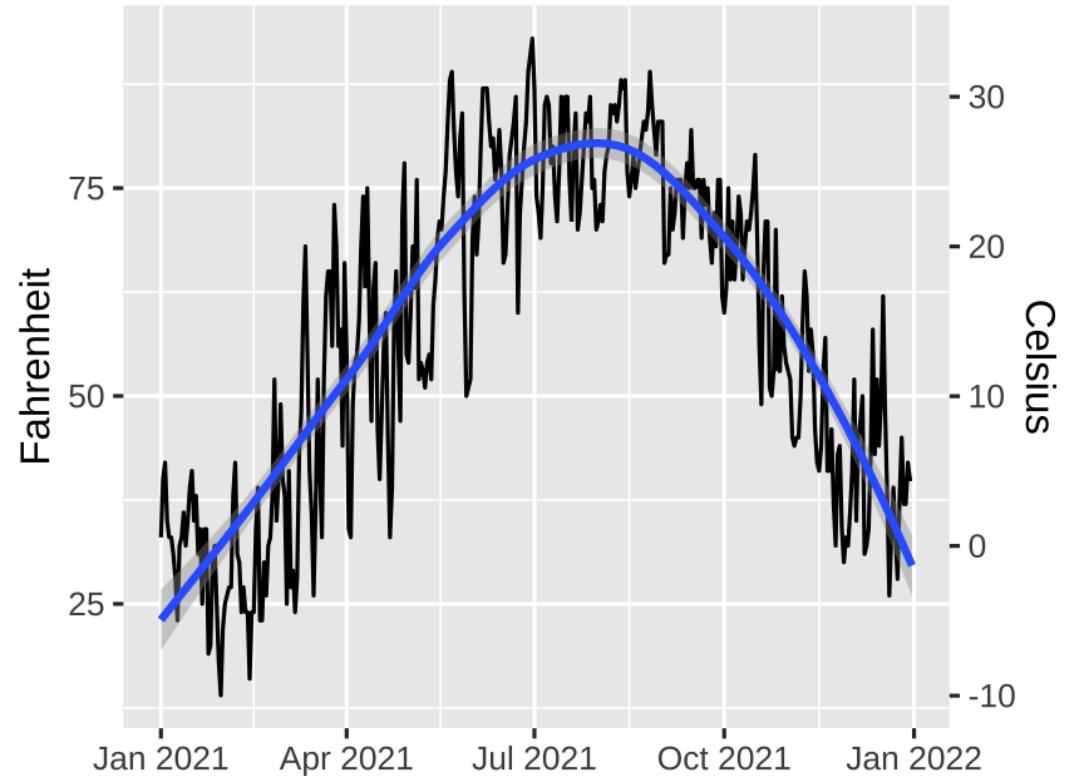


Adding a second scale in R

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

We provided this formula for the
transformation argument:

$$\text{Celsius} = (\text{Fahrenheit} - 32) * 5/9$$

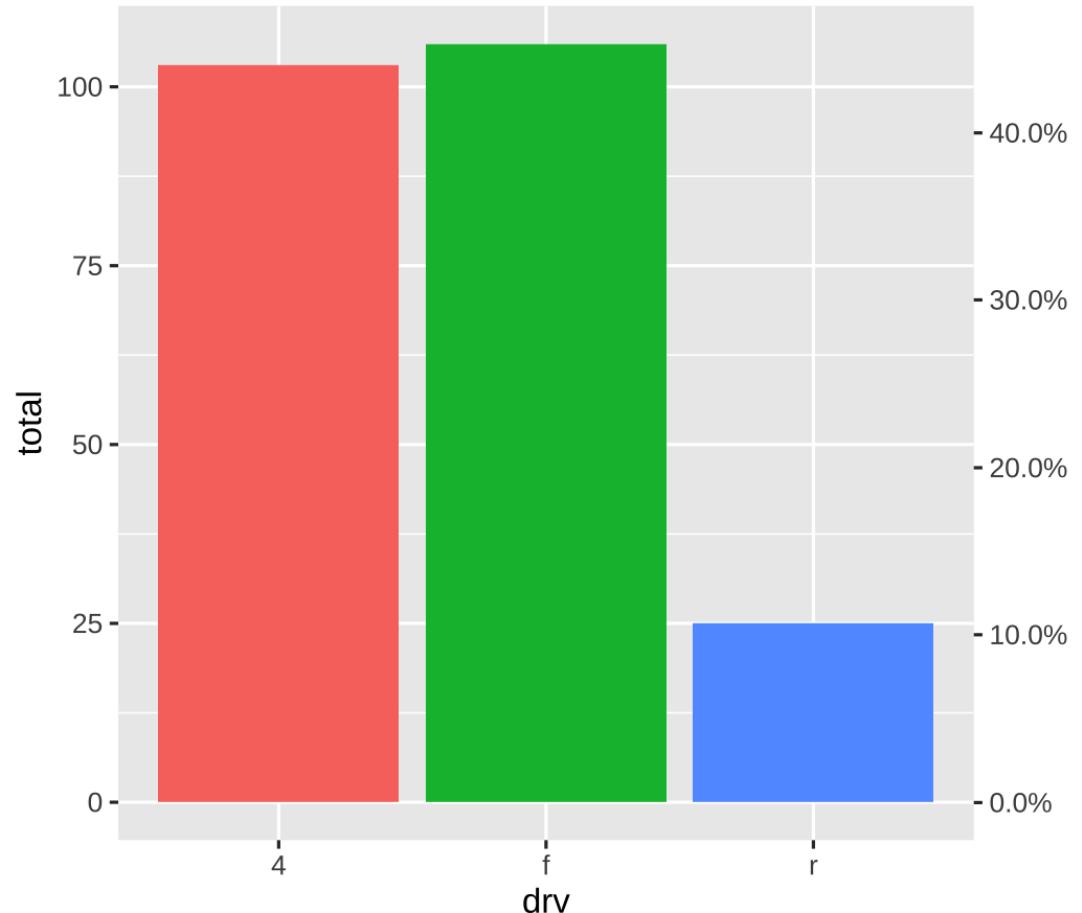


Adding a second scale in R

```
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 = "none")
```

This makes it a lot easier to see proportions with side-by-side bars!

Note: **total_cars** is not in **car_counts**



Alternative 1: Use multiple plots

Solar panels in Japan

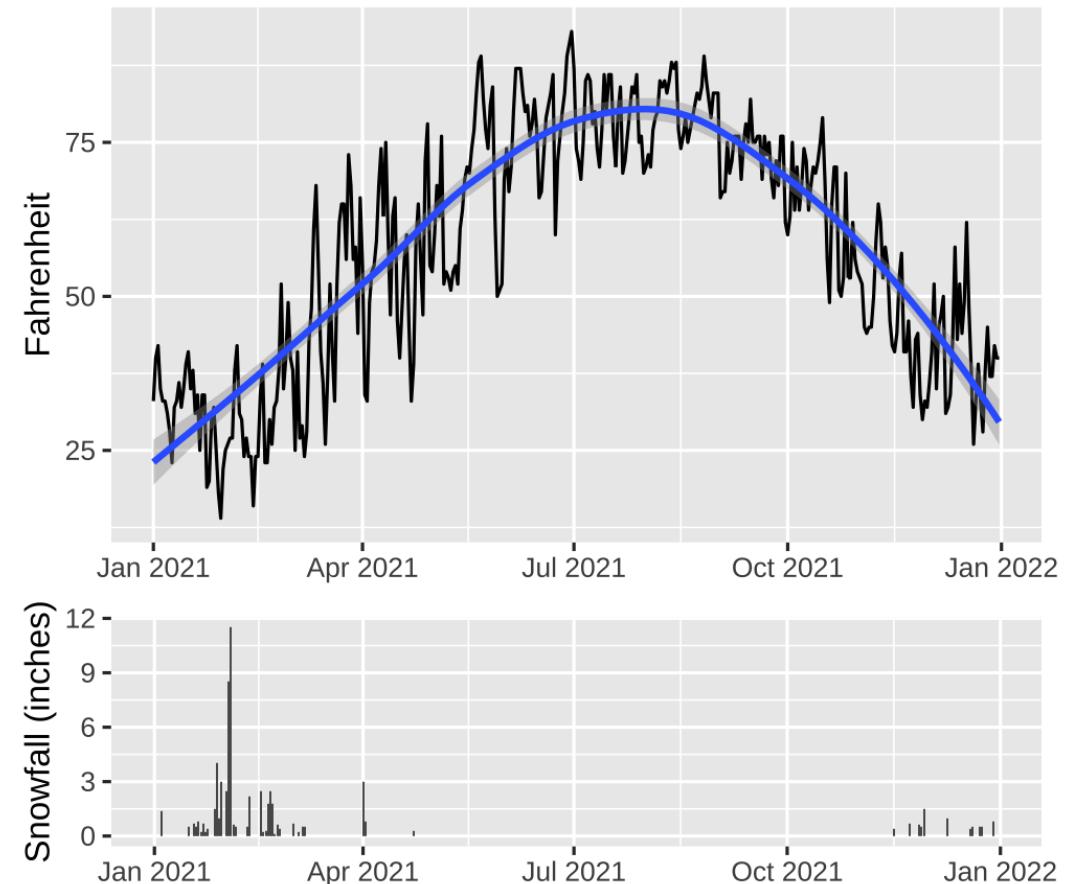
Alternative 1: Use multiple plots

```
library(patchwork)

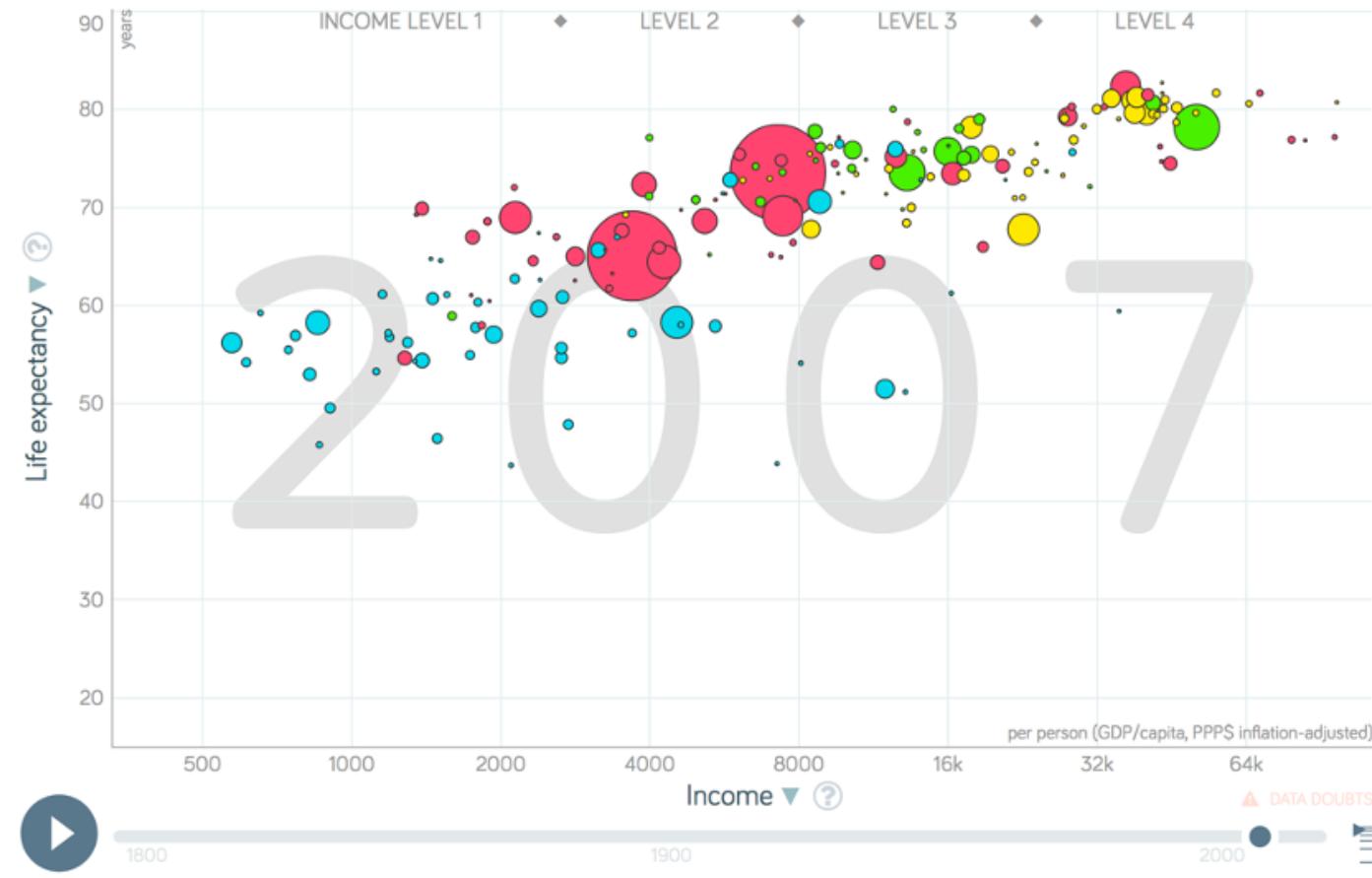
temp_plot <- ggplot(ithaca_weather,
                    aes(x = DATE,
                        y = TMAX)) +
  geom_line() + geom_smooth() +
  labs(x = NULL, y = "Fahrenheit")

snow_plot <- ggplot(ithaca_weather,
                     aes(x = DATE,
                         y = SNOW)) +
  geom_col() +
  labs(x = NULL, y = "Snowfall (inches)")

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



Alternative 2: Use another aesthetic



Visualizing correlations

What does "correlation" mean to you?

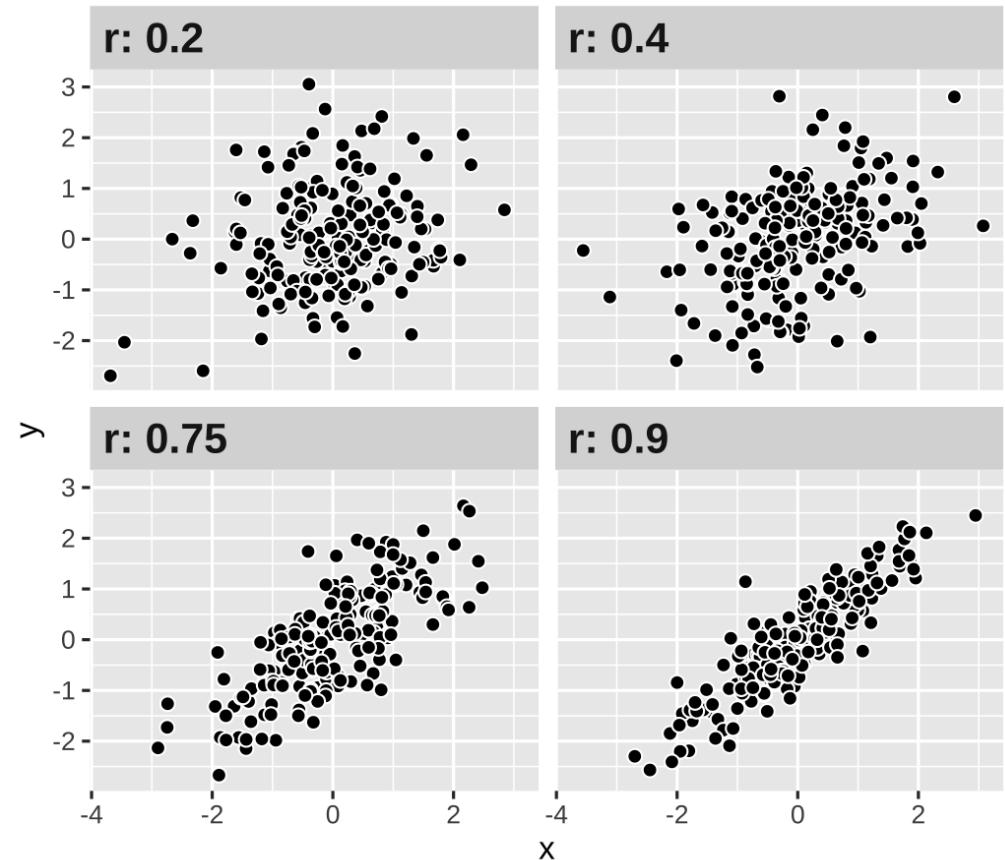
$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}$$

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

ρ	Rough meaning
$\pm 0.1\text{--}0.3$	Modest
$\pm 0.3\text{--}0.5$	Moderate
$\pm 0.5\text{--}0.8$	Strong
$\pm 0.8\text{--}0.9$	Very strong



Scatter plots

The humble scatter plot is often the best place to start when studying the association between two variables

Example: max and min temperature in Ithaca each day of the year

- Do you think they are highly correlated, somewhat correlated, or not at all correlated?
- What sign do you think this correlation has?
- How would you make a scatter plot of these data in R?

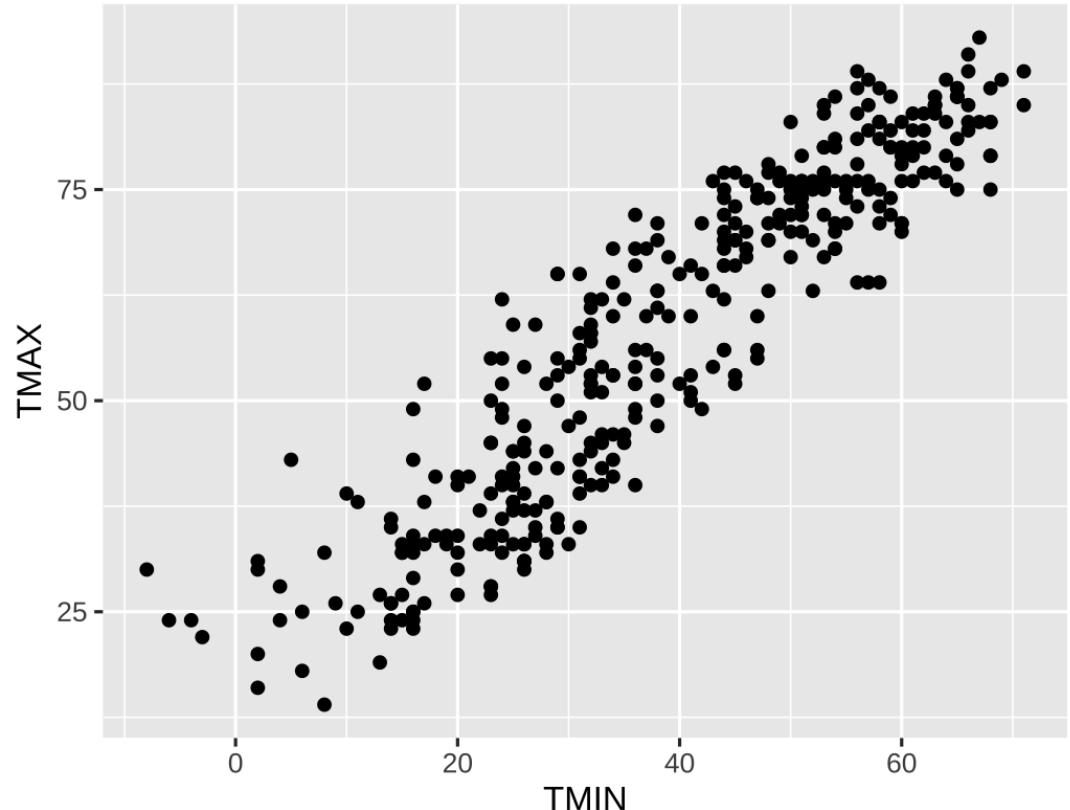
Scatter plots

```
ithaca_weather %>%
  ggplot(aes(x = TMIN, y = TMAX)) +
  geom_point()
```

```
cor(ithaca_weather$TMIN,
    ithaca_weather$TMAX) %>%
  round(., 3)
```

```
## [1] 0.919
```

What about min temp and snowfall?

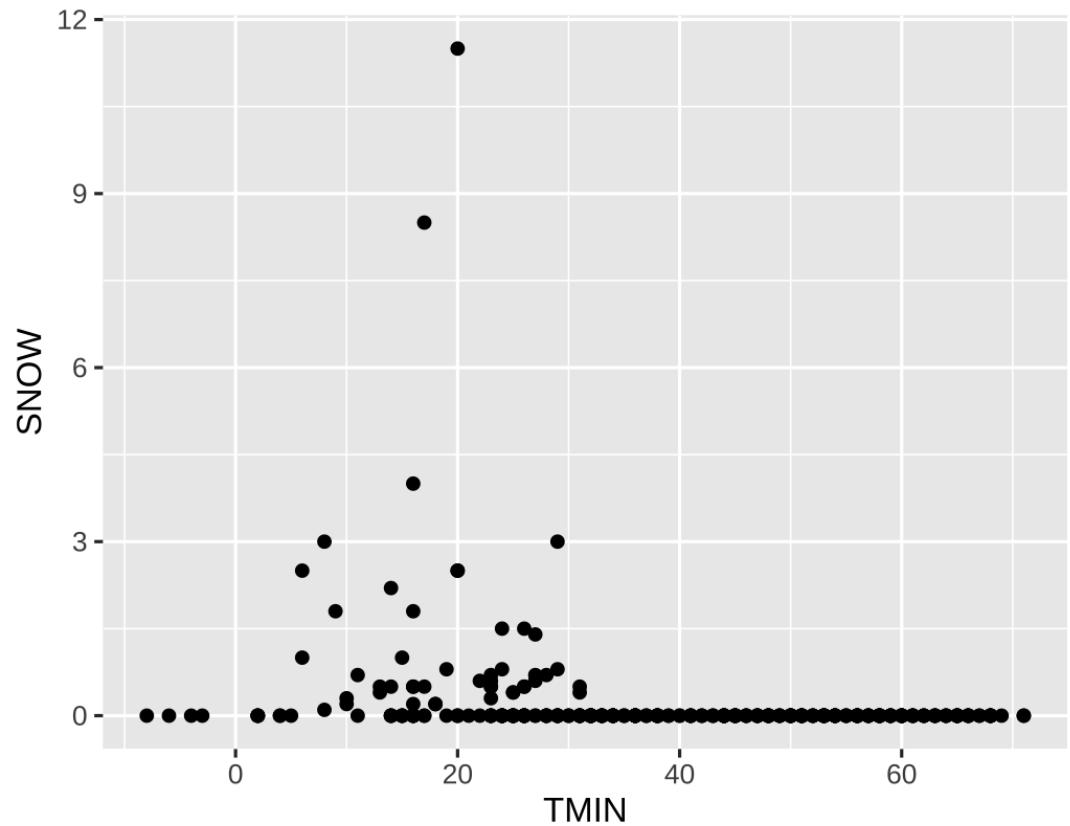


Scatter plots

```
ithaca_weather %>%  
  ggplot(aes(x = TMIN, y = SNOW)) +  
  geom_point()
```

```
cor(ithaca_weather$TMIN,  
    ithaca_weather$SNOW) %>%  
  round(., 3)
```

```
## [1] -0.239
```

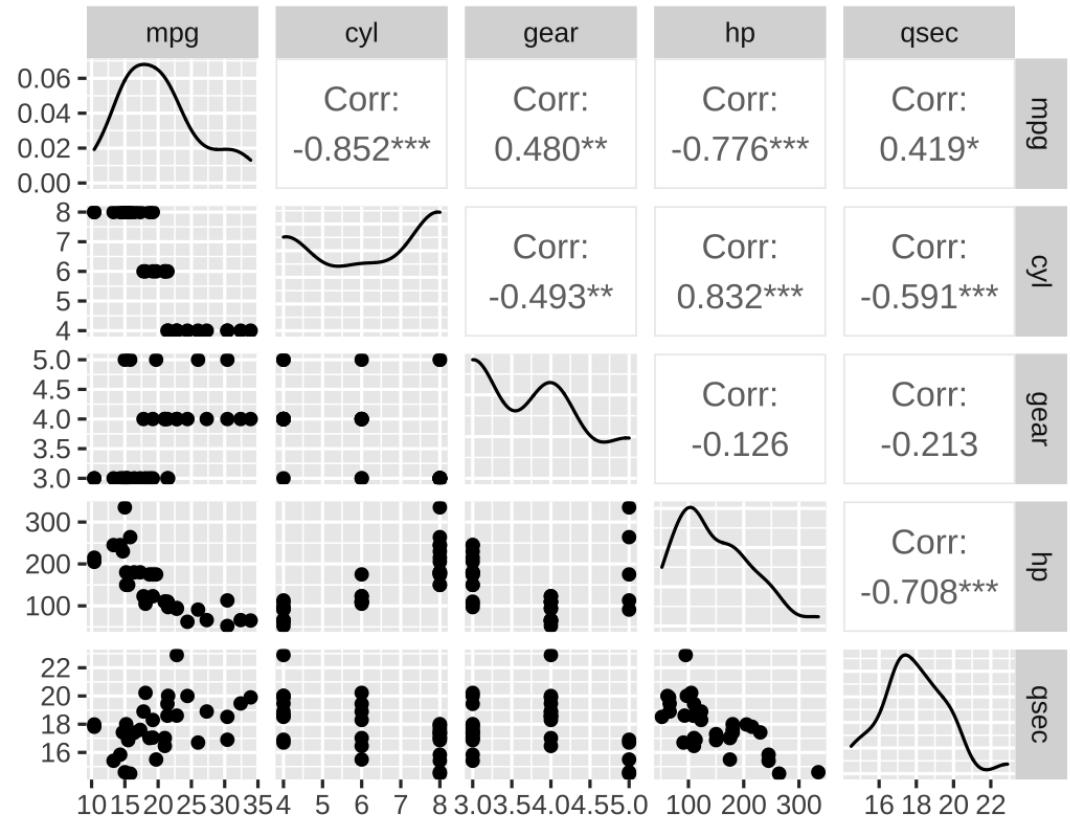


Scatter plot matrices

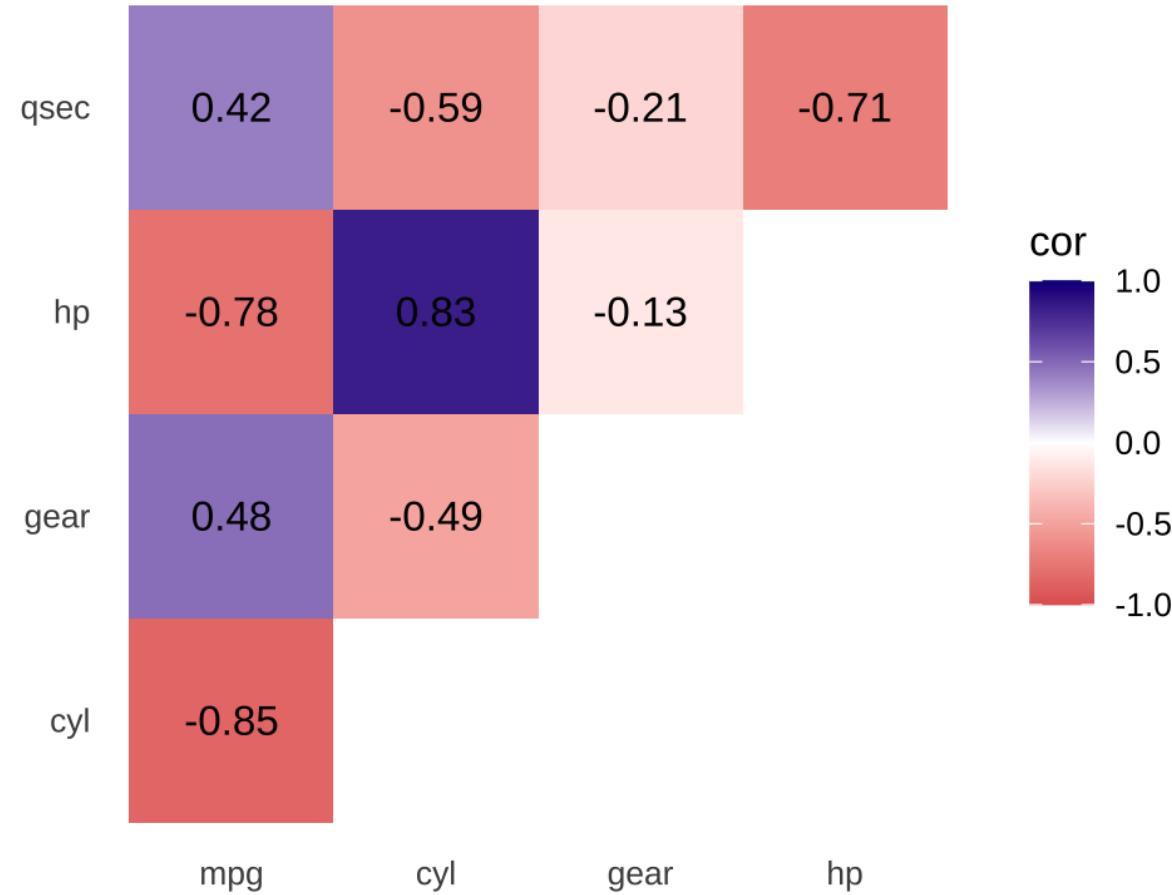
```
library(GGally)  
  
cars_smaller <- mtcars %>%  
  select(mpg, cyl, gear, hp, qsec)  
  
ggpairs(cars_smaller)
```

Scatter plots can be scaled up to matrices for several variables

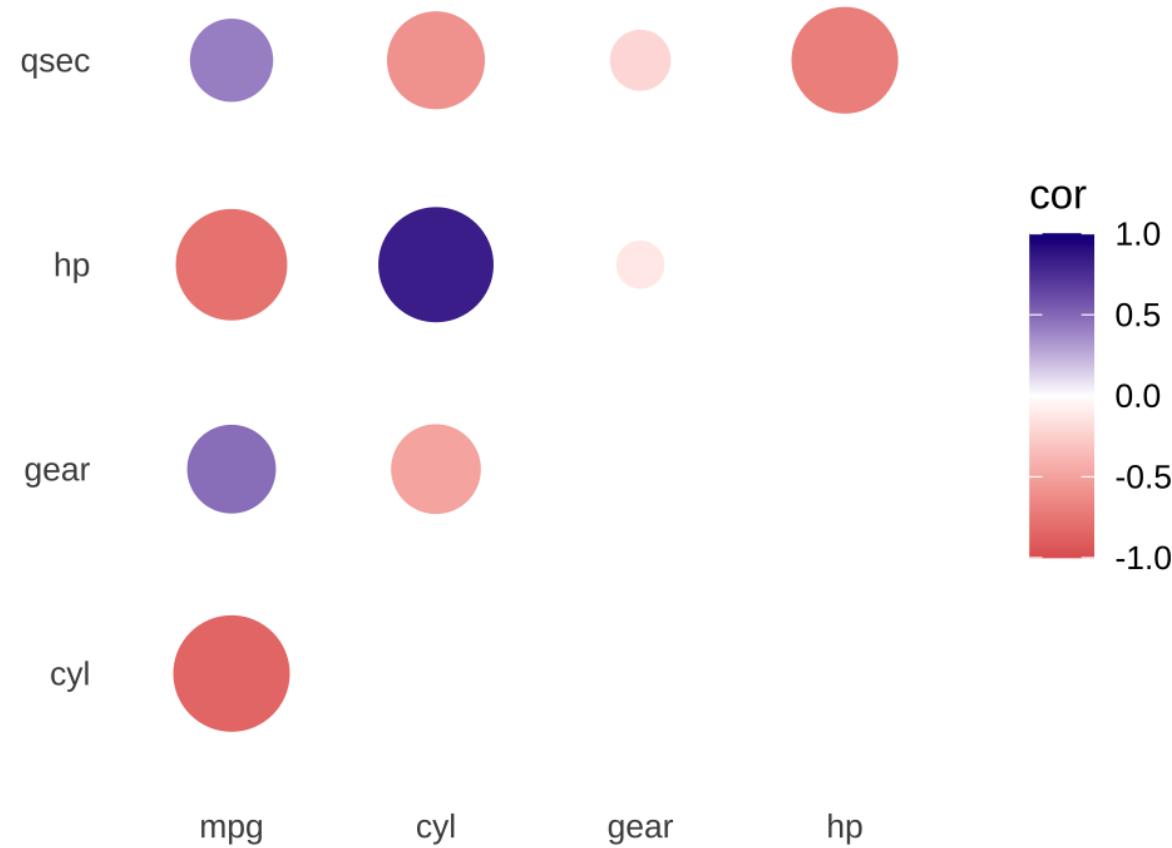
For many variables, correlograms or correlation coefficients often better



Correlograms: Heatmaps



Correlograms: Points



Visualizing regressions

Drawing lines with math

$$y = mx + b$$

y A number

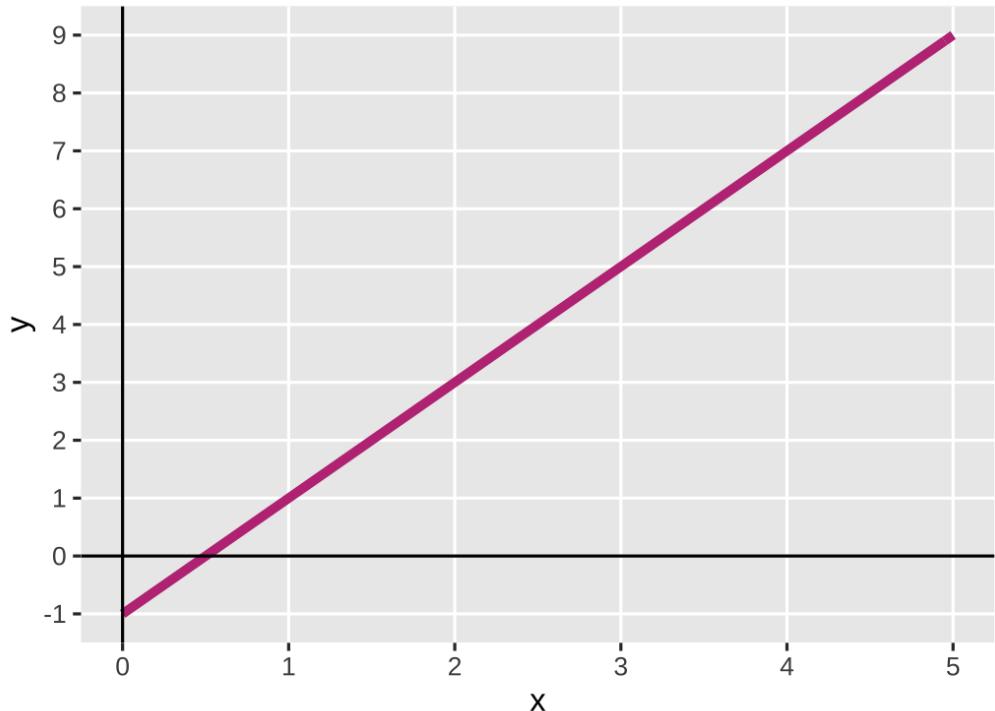
x A number

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

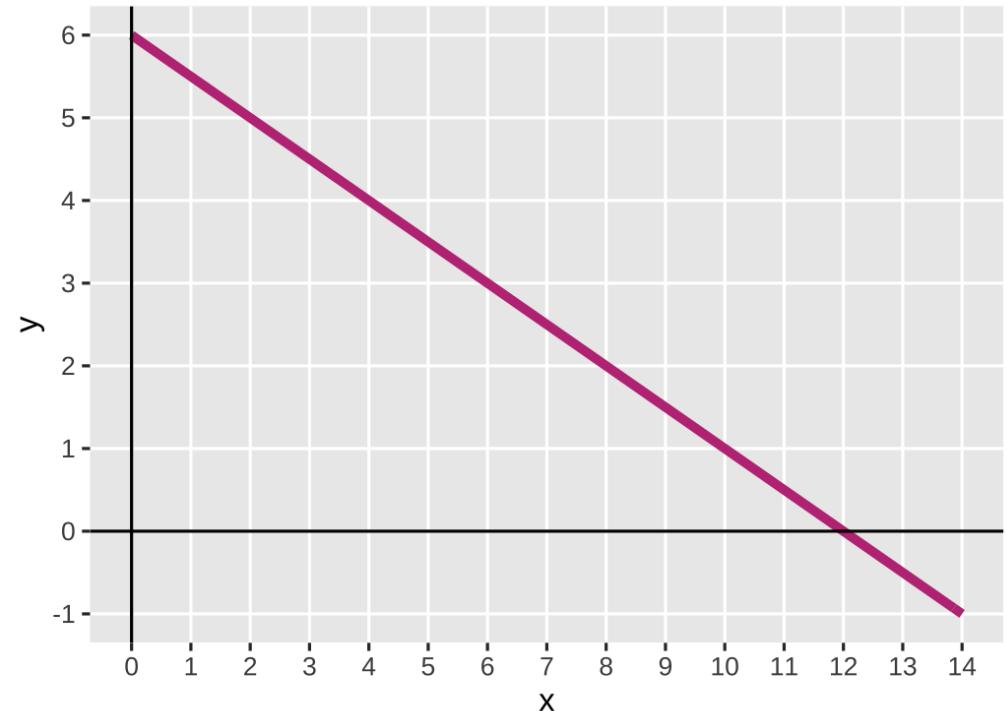
b y-intercept

Slopes and intercepts

$$y = 2x - 1$$



$$y = -0.5x + 6$$

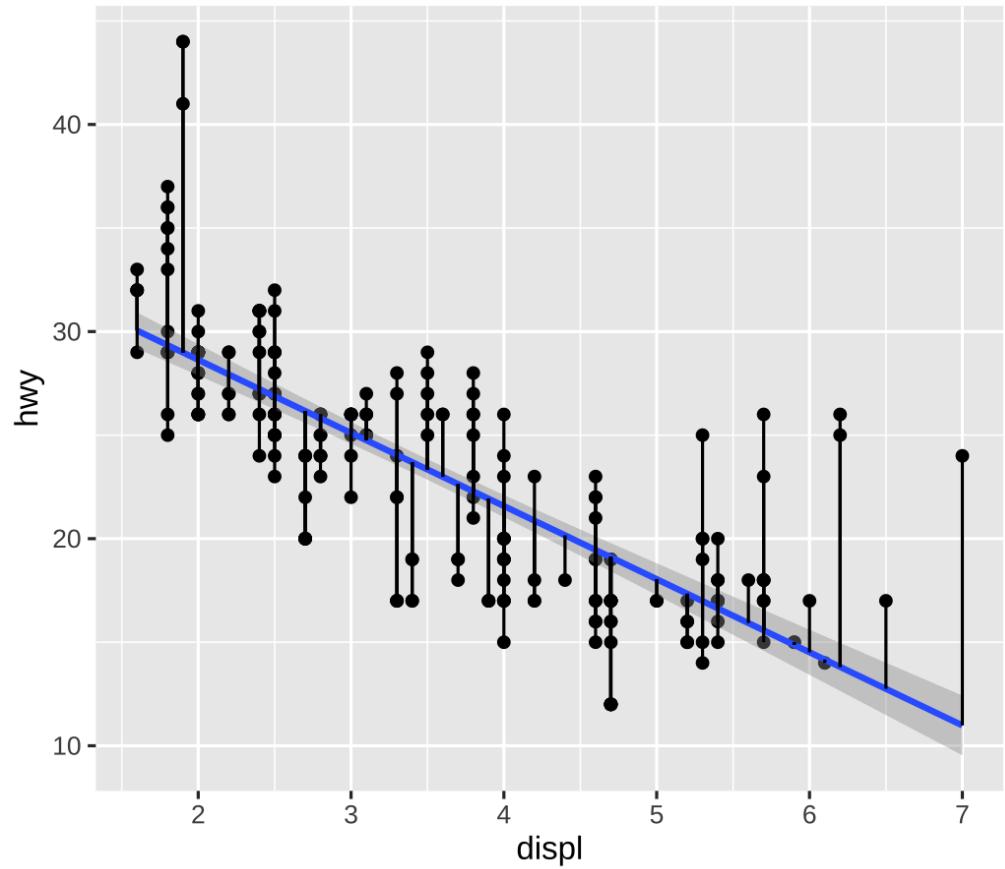
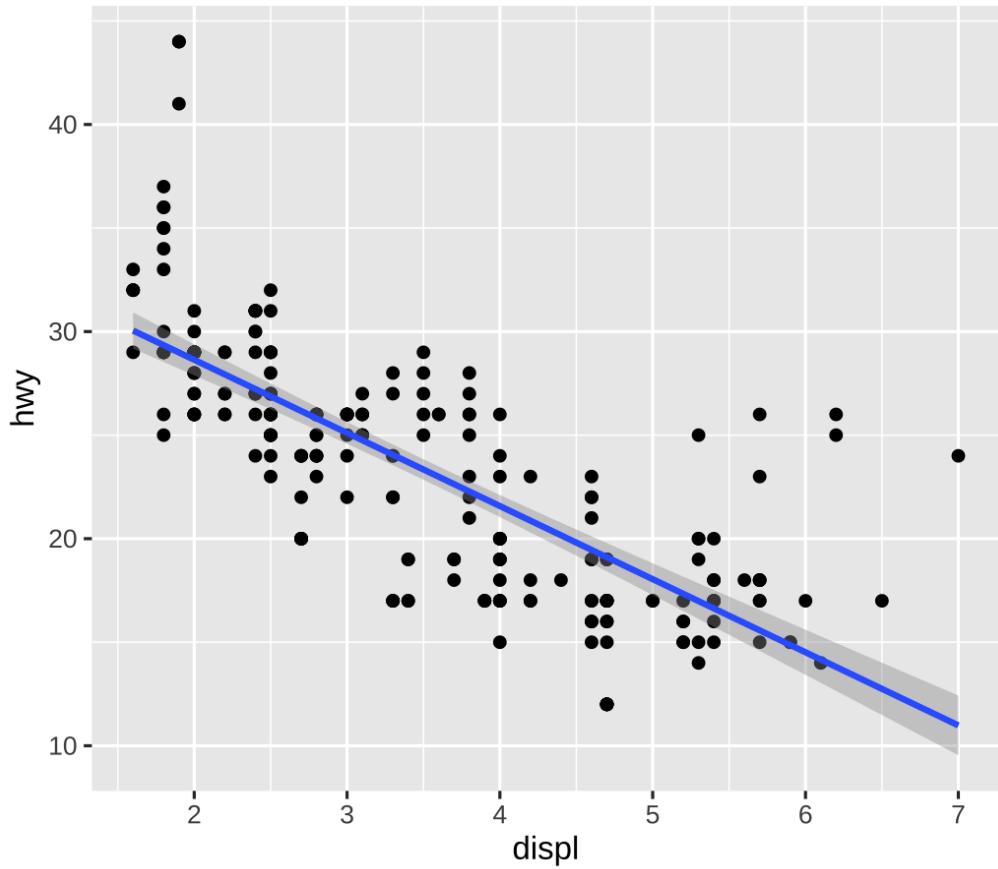


Drawing lines with stats

$$y = \beta_0 + \beta_1 x_1 + \varepsilon$$

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

Drawing lines with stats



Building models in R

Base R has some basic modeling tools:

```
name_of_model <- lm(<Y> ~ <X>, data = <DATA>) # estimate model  
summary(name_of_model) # see model details
```

The **broom** package provides helpful tools for tidying model output:

```
library(broom)  
  
# convert model estimates to a data frame for plotting  
tidy(name_of_model)  
  
# convert model diagnostics to a data frame  
glance(name_of_model)
```

Modeling displacement and highway MPG

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

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

Note how we didn't write anything for
the β_0 or ε terms

```
car_model
```

```
##  
## Call:  
## lm(formula = hwy ~ displ, data = mpg)  
##  
## Coefficients:  
## (Intercept)      displ  
##           35.698     -3.531
```

Modeling displacement and highway MPG

```
summary(car_model)

##
## Call:
## lm(formula = hwy ~ displ, data = mpg)
##
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -7.1039 -2.1646 -0.2242  2.0589 15.0105 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 35.6977    0.7204   49.55 <2e-16 ***
## displ       -3.5306    0.1945  -18.15 <2e-16 ***
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.836 on 232 degrees of freedom
## Multiple R-squared:  0.5868,    Adjusted R-squared:  0.585 
## F-statistic: 329.5 on 1 and 232 DF,  p-value: < 2.2e-16
```

Modeling displacement and highway MPG

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

```
## # A tibble: 2 × 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 × 12
##   r.squared adj.r.squared sigma statistic  p.value      df logLik     AIC     BIC
##   <dbl>        <dbl>    <dbl>     <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 0.587        0.585  3.84     329. 2.04e-46     1  -646. 1297. 1308.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Interpretation for a continuous variable

$$y = \beta_0 + \beta_1 x_1 + \varepsilon$$

A one unit increase in x_1 is *associated* with a β_1 change in y , on average

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

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

On average, a one unit increase in displacement is associated with 3.53 lower highway MPG

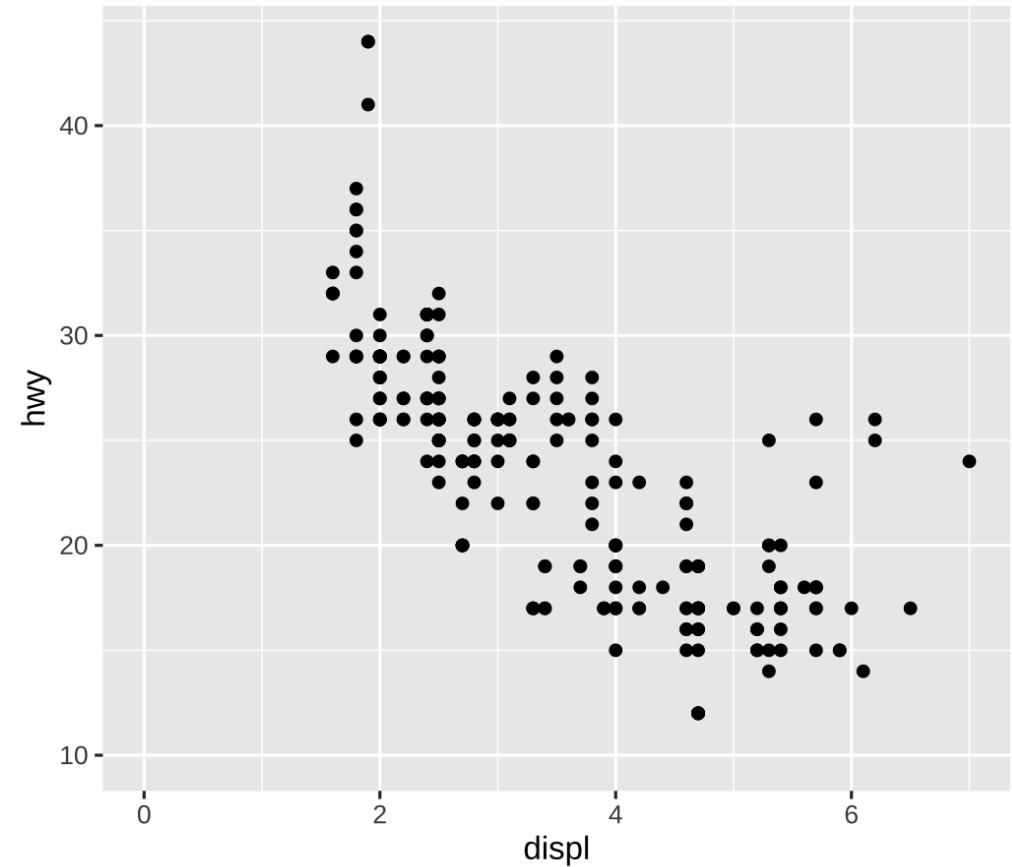
This is easy to visualize: it's a line!

Visualization of a continuous variable

```
tidy(car_model) %>%  
  select(term, estimate)
```

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

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

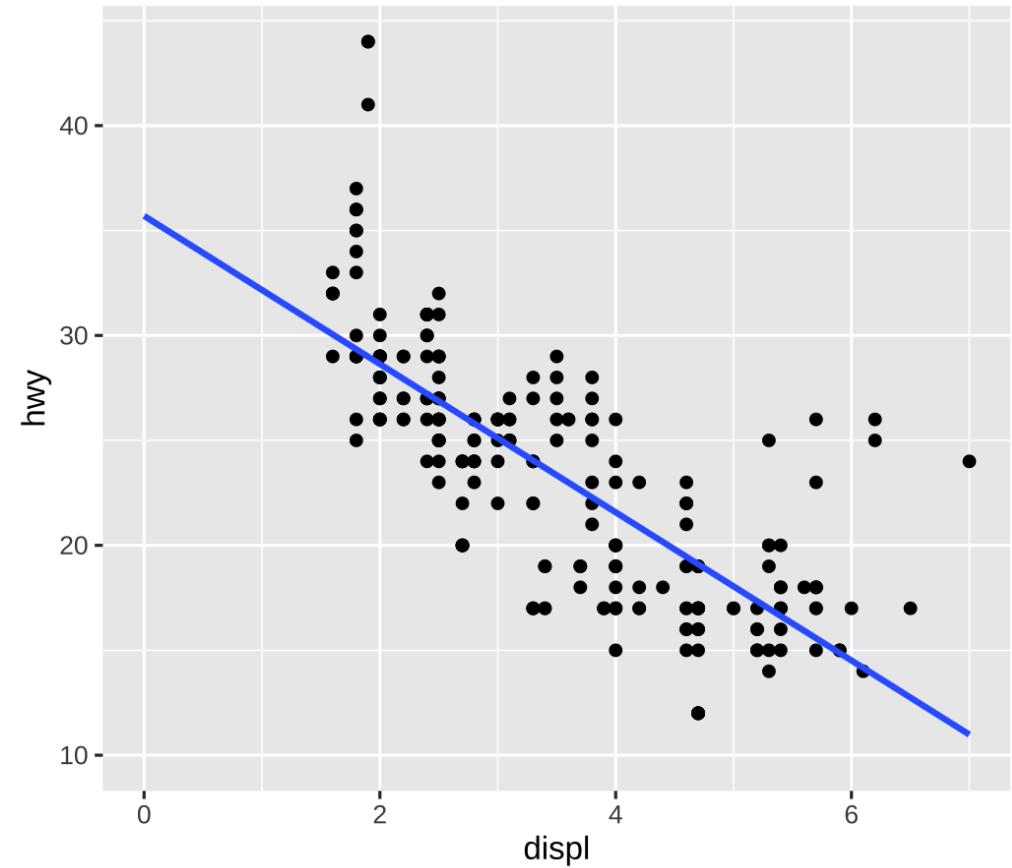


Visualization of a continuous variable

```
tidy(car_model) %>%  
  select(term, estimate)
```

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

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



Multiple regression

We're not limited to just one explanatory variable!

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

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

$$\widehat{\text{hwy}} = \widehat{\beta}_0 + \widehat{\beta}_1 \text{displ} + \widehat{\beta}_2 \text{cyl} + \widehat{\beta}_3 \text{drv:f} + \widehat{\beta}_4 \text{drv:r}$$

Modeling lots of things and MPG

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

```
## # A tibble: 5 × 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 drvf        5.04     0.513     9.83  3.07e-19    4.03      6.06
## 5 drvr        4.89     0.712     6.86  6.20e-11    3.48      6.29
```

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

Sliders and switches



Sliders and switches



Interpretation for continuous variables

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

Holding everything else constant, a one unit increase in x_n is associated with a β_n change in y , on average

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

On average, a one unit increase in displacement is associated with 1.12 lower highway MPG, holding everything else constant

Aside: for the earlier model we had said

- On average, a one unit increase in displacement is associated with 3.53 lower highway MPG

Interpretation for categorical variables

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

Holding everything else constant, y is β_n units larger (or smaller) for category x_n than for the omitted category, on average

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

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)

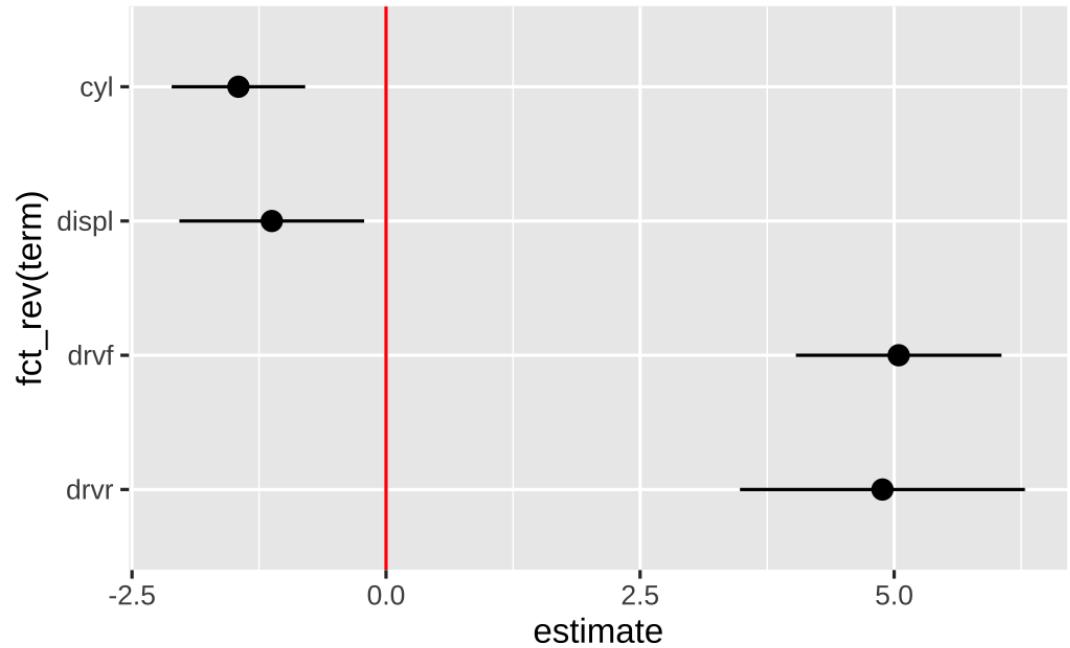
car_coefs <- tidy(car_model_big, conf.int = TRUE) %>%
  filter(term != "(Intercept)") # we can typically skip plotting the intercept, so remove it
car_coefs
```

```
## # A tibble: 4 × 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
## 3 drvf      5.04      0.513     9.83  3.07e-19     4.03     6.06
## 4 drvr      4.89      0.712     6.86  6.20e-11     3.48     6.29
```

Coefficient plots

Plot the point 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),      # create grid for displ
                        cyl = mean(mpg$cyl),          # hold cylinders at its mean
                        drv = "f")                  # drive: front-wheel

cars_new_data

## # A tibble: 51 × 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
## 7     2.6   5.89 f
## 8     2.7   5.89 f
## 9     2.8   5.89 f
## 10    2.9   5.89 f
## # ... with 41 more rows
```

Marginal effects plots

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

```
predicted_mpg <- augment(car_model_big,           # our estimated model
                           newdata = cars_new_data, # our new data for plotting
                           se_fit = TRUE)

head(predicted_mpg)

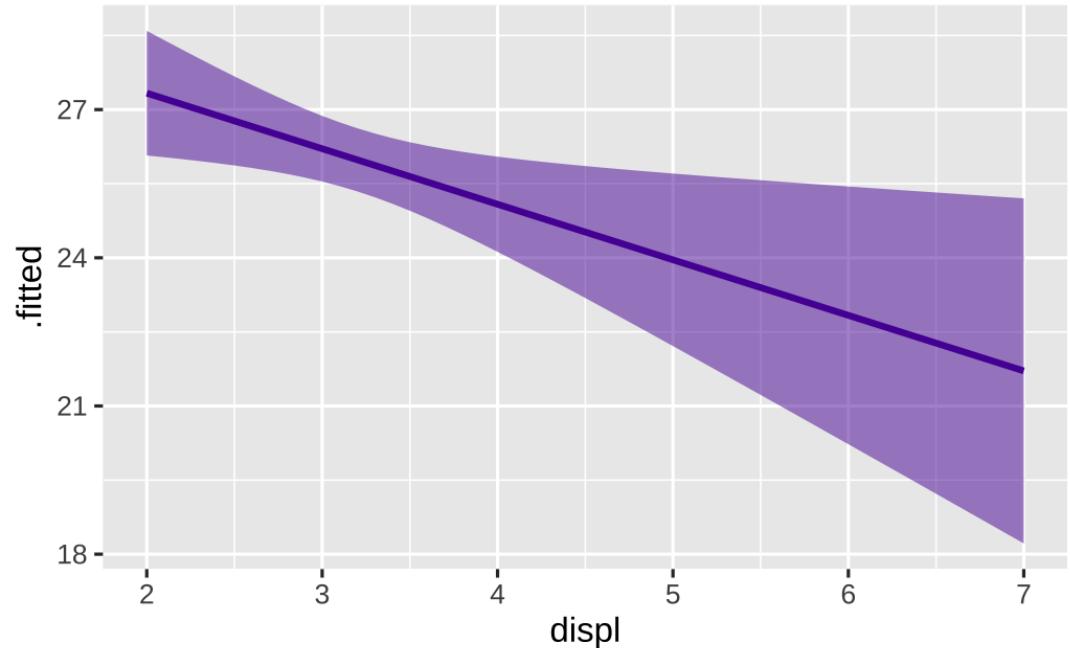
## # A tibble: 6 × 5
##   displ cyl drv   .fitted .se.fit
##   <dbl> <dbl> <chr>    <dbl>    <dbl>
## 1 2     5.89 f       27.3    0.644
## 2 2.1   5.89 f       27.2    0.604
## 3 2.2   5.89 f       27.1    0.566
## 4 2.3   5.89 f       27.0    0.529
## 5 2.4   5.89 f       26.9    0.494
## 6 2.5   5.89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

Marginal effects plots

Plot the fitted values for each row

We made predictions for front-wheel drive cars with cylinders held at their mean

```
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?

How would you approach this?

Multiple effects at once

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

The `expand_grid()` function comes in handy for this

`expand_grid()` creates a data frame with every possible combination of the variables you supply

Multiple effects at once

```
cars_new_data_fancy <- expand_grid(displ = seq(2, 7, by = 0.1),          # create grid for displ
                                    cyl = mean(mpg$cyl),           # hold cylinders at its mean
                                    drv = c("f", "r", "4"))       # drive: front-wheel

cars_new_data_fancy

## # A tibble: 153 × 3
##      displ   cyl   drv
##      <dbl> <dbl> <chr>
## 1     2.0    5.89 f
## 2     2.0    5.89 r
## 3     2.0    5.89 4
## 4     2.1    5.89 f
## 5     2.1    5.89 r
## 6     2.1    5.89 4
## 7     2.2    5.89 f
## 8     2.2    5.89 r
## 9     2.2    5.89 4
## 10    2.3    5.89 f
## ... with 143 more rows
```

Multiple effects at once

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

```
predicted_mpg_fancy <- augment(car_model_big,           # our estimated model
                                 newdata = cars_new_data_fancy, # our new data for plotting
                                 se_fit = TRUE)

head(predicted_mpg_fancy)
```

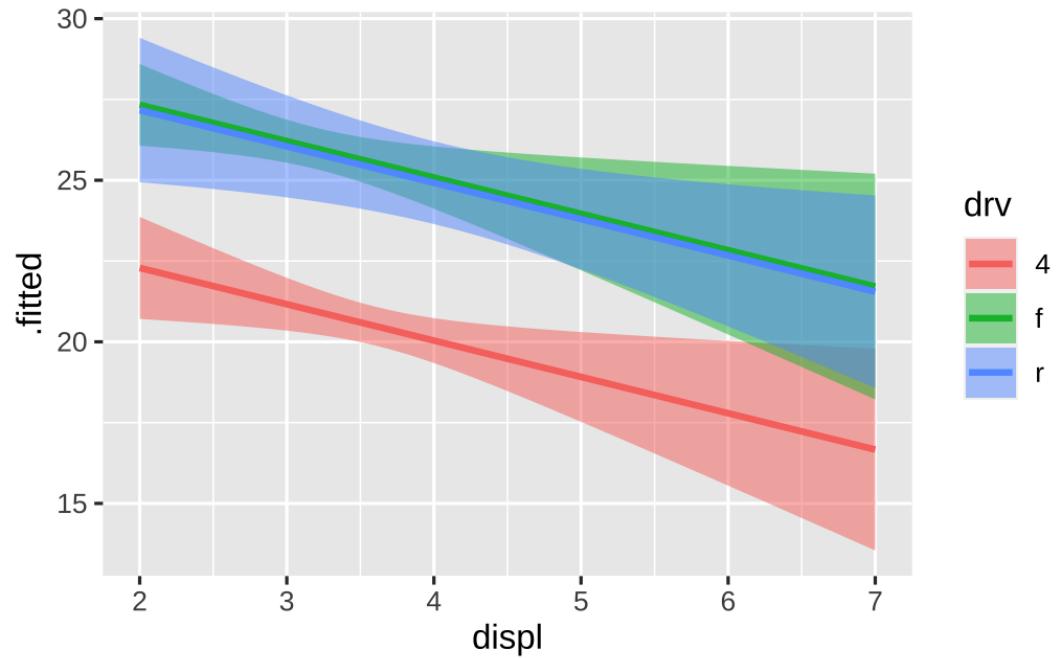
```
## # A tibble: 6 × 5
##   displ cyl drv   .fitted .se.fit
##   <dbl> <dbl> <chr>   <dbl>    <dbl>
## 1 2     5.89 f       27.3    0.644
## 2 2     5.89 r       27.2    1.14
## 3 2     5.89 4      22.3    0.805
## 4 2.1   5.89 f       27.2    0.604
## 5 2.1   5.89 r       27.1    1.10
## 6 2.1   5.89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

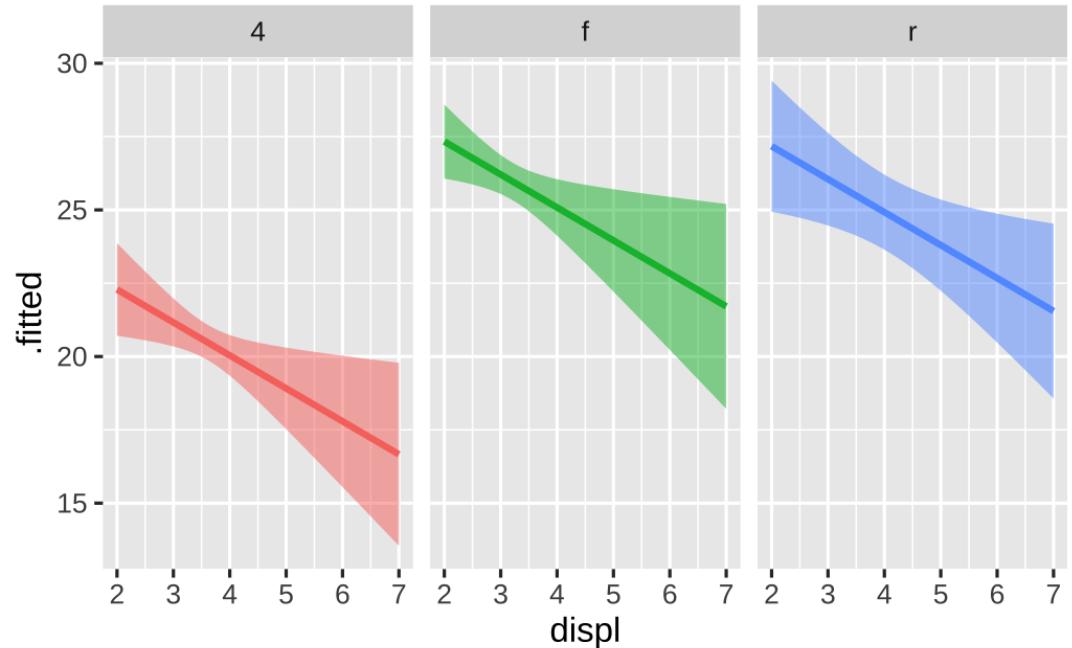


Multiple effects at once

Plot the fitted values for each row

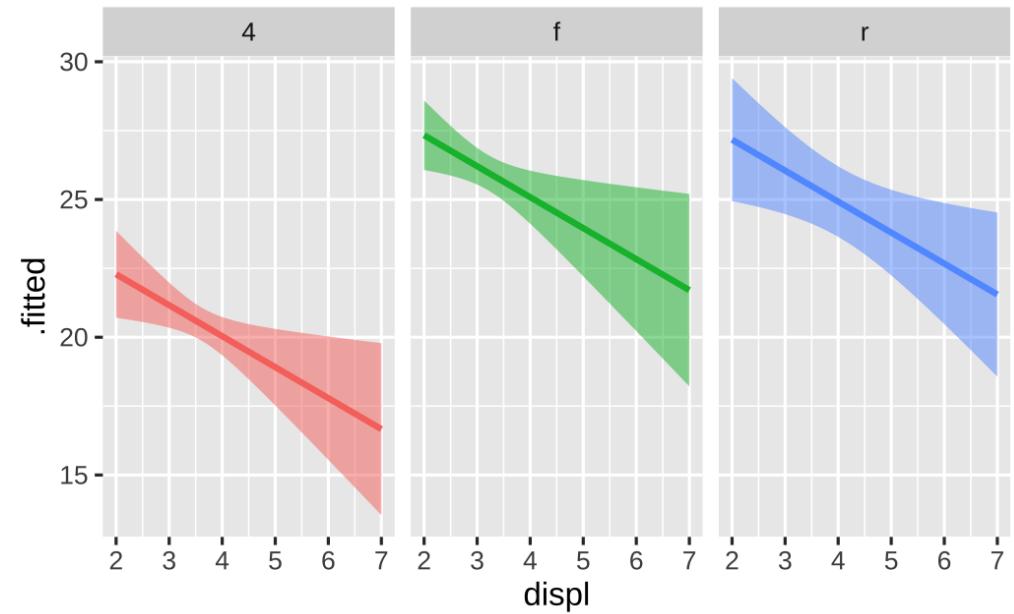
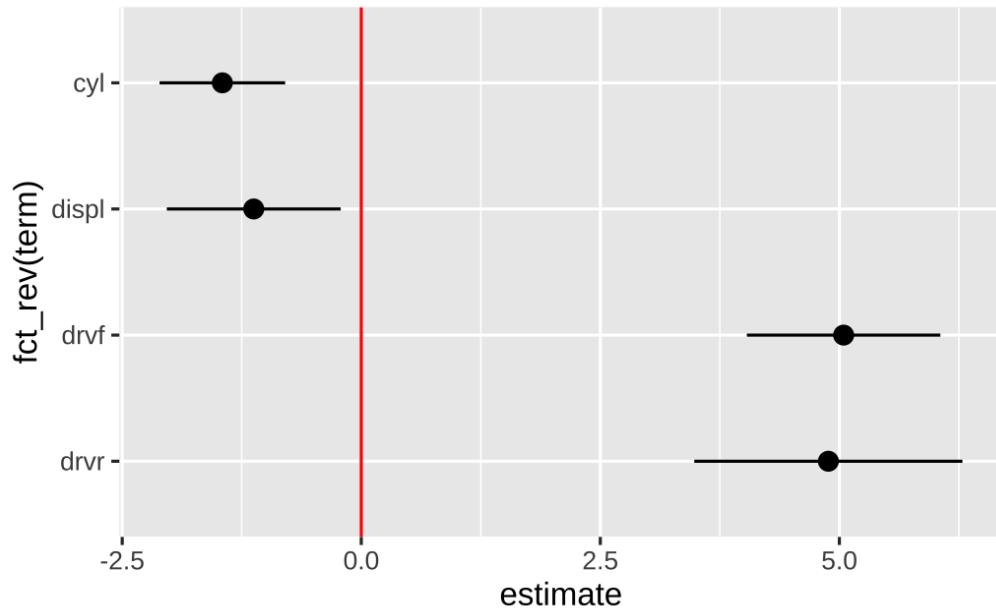
Cylinders held at their mean; colored/filled/faceted by drive

```
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 = "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

OLS with high-dimensional fixed effects

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

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

Bayesian models

Machine learning models

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