# Gov 50: 11. Summarizing Bivariate Relationships

Matthew Blackwell

Harvard University

# Roadmap

1. Z-scores and standardization
2. Correlation
3. Writing our own functions

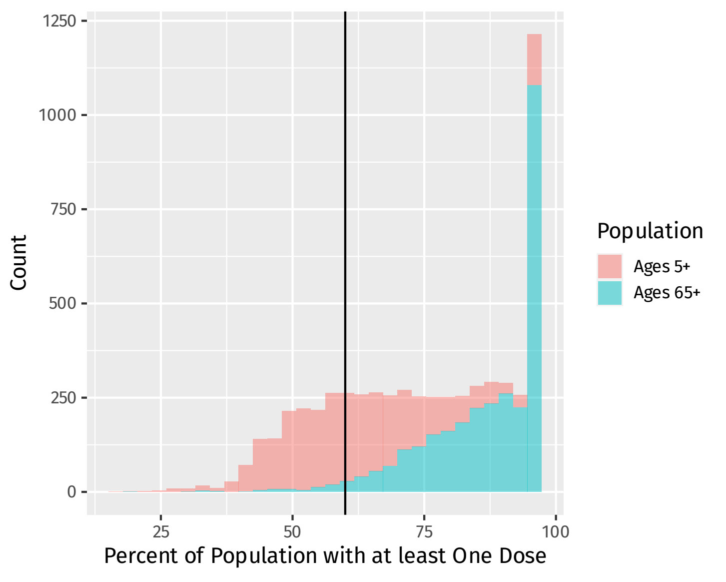
# 1/ Z-scores and standardization

# COVID vaccination rates and votes

```r
library(tidyverse)
library(gov50data)
covid_votes
```

```
## # A tibble: 3,114 x 8
##    fips  county state one_dose_5plus_pct one_dose_65plus_pct
##    <chr> <chr>  <chr>              <dbl>               <dbl>
##  1 26039 Crawf~ MI                  55.7                77.3
##  2 40015 Caddo~ OK                  83.3                95
##  3 17007 Boone~ IL                  71.1                94.5
##  4 12055 Highl~ FL                  68.9                93.7
##  5 34029 Ocean~ NJ                  71                  95
##  6 01067 Henry~ AL                  58.5                85.5
##  7 27037 Dakot~ MN                  81                  95
##  8 27115 Pine ~ MN                  56.5                85
##  9 51750 Radfo~ VA                  41.5                73.8
## 10 22009 Avoye~ LA                  59.7                80.1
## # i 3,104 more rows
## # i 3 more variables: booster_5plus_pct <dbl>,
## #   dem_pct_2000 <dbl>, dem_pct_2020 <dbl>
```

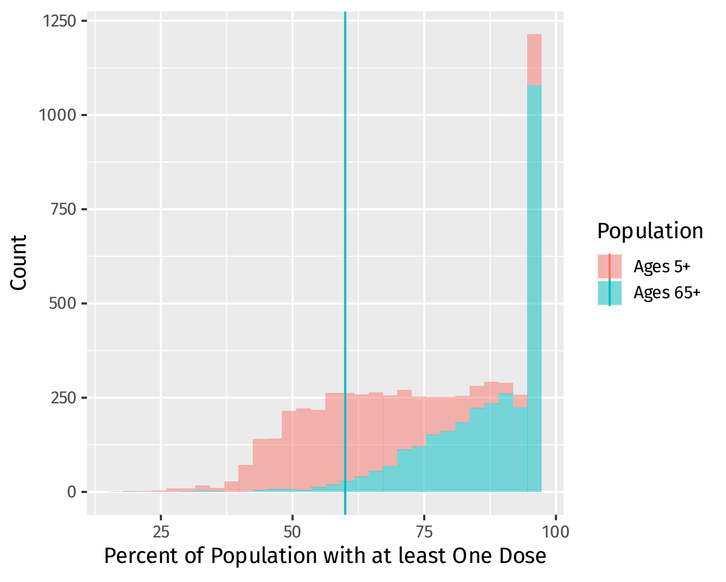
# Is 60% vaccinated a lot?



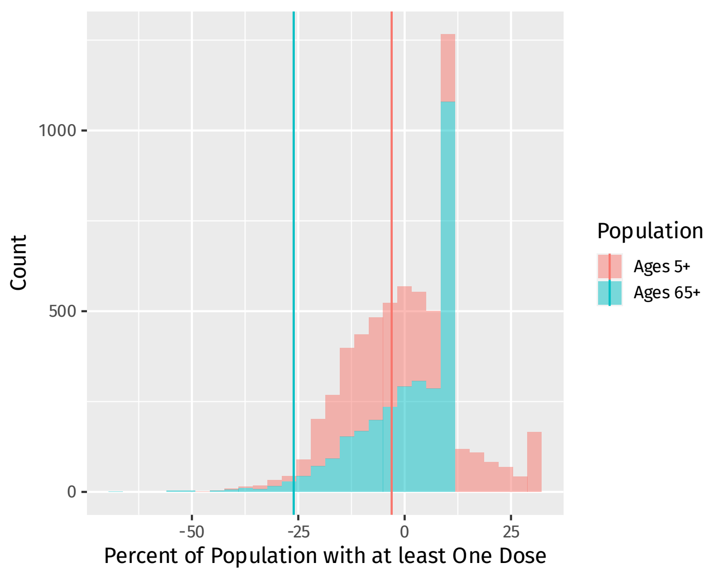
# How large is large?

- How large 60% vaccinated is depends on the distribution!
    - Clear to see from the histogram
    - Middling for the 5+ group, but very low for the 65+ group.
- Can we transform the values of our variables to be **common units**?
- Yes, with two transformations:
    - **Centering**: subtract the mean of the variable from each value.
    - **Scaling**: dividing deviations from the mean by the standard deviation.

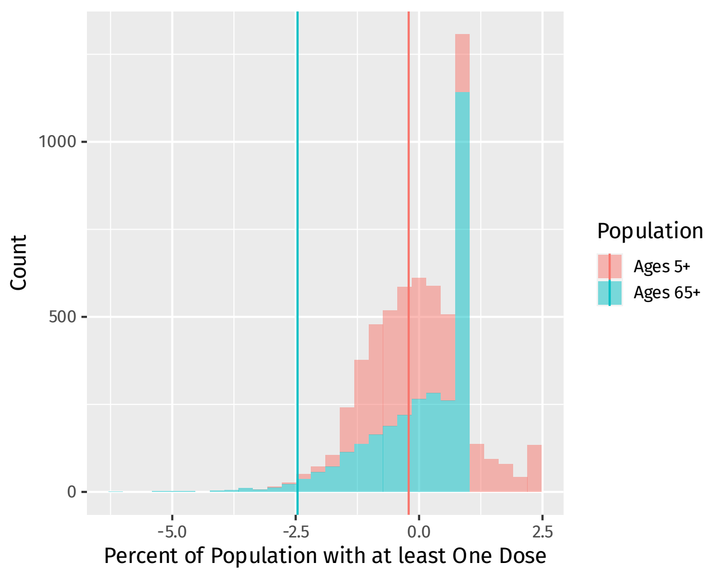
# Original distributions



# Centered distributions



# Centered and scaled distributions



# Z-scores

- Centering tells us immediately if a value is above or below the mean.
- Scaling tells us how many standard deviations away from the mean it is.
- Combine them with the **z-score** transformation:

$$\text{z-score of } x_i = \frac{x_i - \text{mean of } x}{\text{standard deviation of } x}$$

- Useful heuristic: data more than 3 SDs away from mean are rare.

# z-score example

```
covid_votes |>
  mutate(one_dose_centered = one_dose_5plus_pct -
           mean(one_dose_5plus_pct, na.rm = TRUE)) |>
  select(fips:state, one_dose_5plus_pct, one_dose_centered)
```

```
## # A tibble: 3,114 x 5
##    fips  county   state one_dose_5plus_pct one_dose_centered
##    <chr> <chr>    <chr>              <dbl>             <dbl>
##  1 26039 Crawfor~ MI                  55.7             -7.35
##  2 40015 Caddo C~ OK                  83.3             20.2
##  3 17007 Boone C~ IL                  71.1              8.05
##  4 12055 Highlan~ FL                  68.9              5.85
##  5 34029 Ocean C~ NJ                  71                7.95
##  6 01067 Henry C~ AL                  58.5             -4.55
##  7 27037 Dakota ~ MN                  81               17.9
##  8 27115 Pine Co~ MN                  56.5             -6.55
##  9 51750 Radford~ VA                  41.5            -21.6
## 10 22009 Avoyell~ LA                  59.7             -3.35
## # i 3,104 more rows
```

# z-score example

```
covid_votes |>
  mutate(
    one_dose_z =
      (one_dose_5plus_pct - mean(one_dose_5plus_pct, na.rm = TRUE)) /
      sd(one_dose_5plus_pct, na.rm = TRUE))   |>
  select(fips:state, one_dose_5plus_pct, one_dose_z)
```

```
## # A tibble: 3,114 x 5
##    fips  county           state one_dose_5plus_pct one_dose_z
##    <chr> <chr>            <chr>              <dbl>      <dbl>
##  1 26039 Crawford County  MI                  55.7     -0.508
##  2 40015 Caddo County     OK                  83.3      1.40
##  3 17007 Boone County     IL                  71.1      0.556
##  4 12055 Highlands Coun~  FL                  68.9      0.404
##  5 34029 Ocean County     NJ                  71        0.549
##  6 01067 Henry County     AL                  58.5     -0.314
##  7 27037 Dakota County    MN                  81        1.24
##  8 27115 Pine County      MN                  56.5     -0.452
##  9 51750 Radford city     VA                  41.5     -1.49
## 10 22009 Avoyelles Pari~  LA                  59.7     -0.231
## # i 3,104 more rows
```

# 2/ Correlation

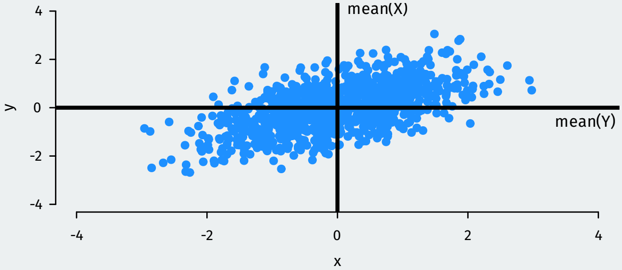
# Correlation

- How do variables move together on average?
- When $x_i$ is big, what is $y_i$ likely to be?
  - Positive correlation: when $x_i$ is big, $y_i$ is also big
  - Negative correlation: when $x_i$ is big, $y_i$ is small
  - High magnitude of correlation: data cluster tightly around a line.
- The technical definition of the **correlation coefficient**:

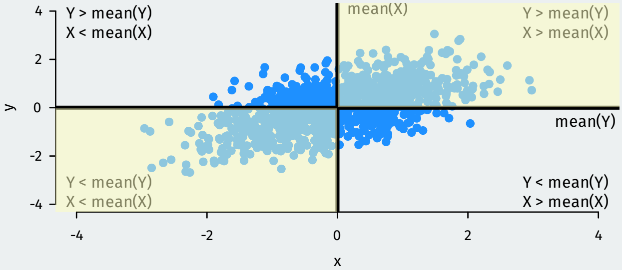
$$\frac{1}{n-1}\sum_{i=1}^{n}\left[(\text{z-score for } x_i) \times (\text{z-score for } y_i)\right]$$

- Interpretation:
  - Correlation is between -1 and 1
  - Correlation of 0 means no linear association.
  - Positive correlations $\rightsquigarrow$ positive associations.
  - Negative correlations $\rightsquigarrow$ negative associations.
  - Closer to -1 or 1 means stronger association.

# Correlation intuition

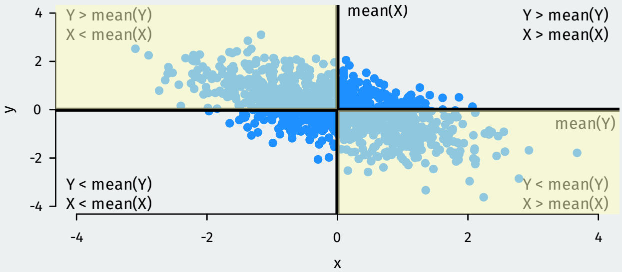


# Correlation intuition



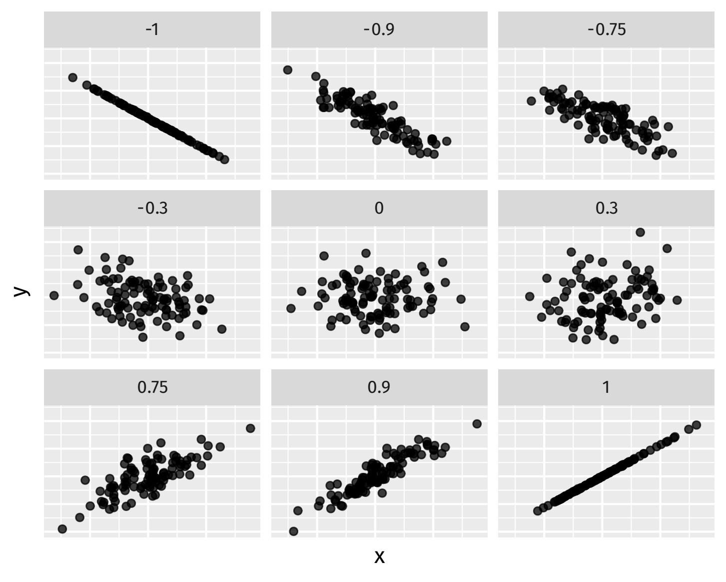

- Large values of $X$ tend to occur with large values of $Y$:
  - $(\text{z-score for } x_i) \times (\text{z-score for } y_i) = (\text{pos. num.}) \times (\text{pos. num}) = +$
- Small values of $X$ tend to occur with small values of $Y$:
  - $(\text{z-score for } x_i) \times (\text{z-score for } y_i) = (\text{neg. num.}) \times (\text{neg. num}) = +$
- If these dominate $\rightsquigarrow$ positive correlation.

# Correlation intuition




- Large values of $X$ tend to occur with small values of $Y$:
    - (z-score for $x_i$) $\times$ (z-score for $y_i$) $=$ (pos. num.) $\times$ (neg. num) $= -$
- Small values of $X$ tend to occur with large values of $Y$:
    - (z-score for $x_i$) $\times$ (z-score for $y_i$) $=$ (neg. num.) $\times$ (pos. num) $= -$
- If these dominate $\rightsquigarrow$ negative correlation.

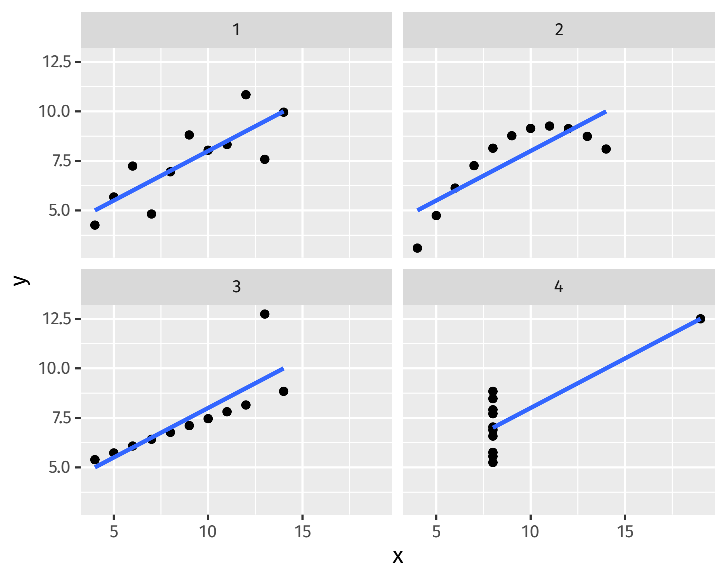
# Correlation examples



# Properties of correlation coefficient

- Correlation measures **linear** association.
- Order doesn't matter: `cor(x,y) = cor(y,x)`
- Not affected by changes of scale:
  - `cor(x,y) = cor(ax+b, cy+d)`
  - Celsius vs. Fahreneheit; dollars vs. pesos; cm vs. in.

# All 4 relationships have 0.816 correlation



# Correlation in R

Use the `cor()` function:

```
cor(covid_votes$one_dose_5plus_pct, covid_votes$dem_pct_2020)
```

```
## [1] NA
```

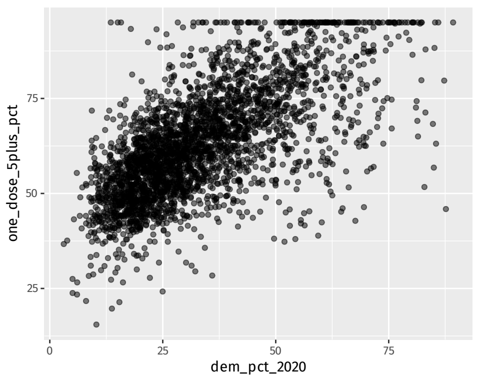
Missing values: set the `use = "pairwise"` → available case analysis

```
cor(covid_votes$one_dose_5plus_pct, covid_votes$dem_pct_2020,
    use = "pairwise")
```

```
## [1] 0.666
```

# Comparing correlations

```r
covid_votes |>
  ggplot(aes(x = dem_pct_2020, y = one_dose_5plus_pct)) +
  geom_point(alpha = 0.5)
```

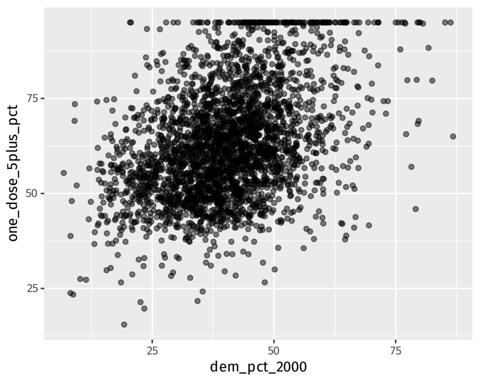



```r
cor(covid_votes$one_dose_5plus_pct, covid_votes$dem_pct_2020,
    use = "pairwise")
```

```
## [1] 0.666
```

# Comparing correlations

```
covid_votes |>
  ggplot(aes(x = dem_pct_2000, y = one_dose_5plus_pct)) +
  geom_point(alpha = 0.5)
```

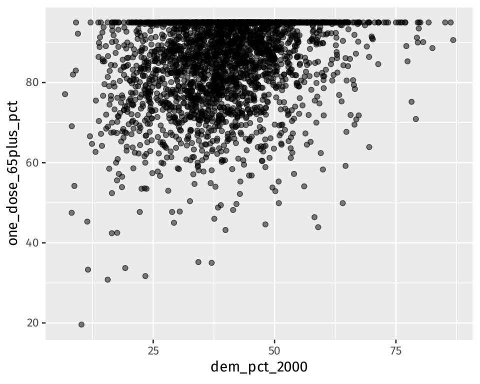



```
cor(covid_votes$one_dose_5plus_pct, covid_votes$dem_pct_2000,
    use = "pairwise")
```

```
## [1] 0.394
```

# Comparing correlations

```
covid_votes |>
  ggplot(aes(x = dem_pct_2000, y = one_dose_65plus_pct)) +
  geom_point(alpha = 0.5)
```




```
cor(covid_votes$one_dose_65plus_pct, covid_votes$dem_pct_2000,
    use = "pairwise")
```

```
## [1] 0.263
```

# 3/ Writing our own functions

# Why write functions?

Copy-pasting code tedious and prone to failure:

```
covid_votes |>
  mutate(
    one_dose_5p_z =
      (one_dose_5plus_pct - mean(one_dose_5plus_pct, na.rm = TRUE)) /
      sd(one_dose_5plus_pct, na.rm = TRUE),
    one_dose_65p_z =
      (one_dose_65plus_pct - mean(one_dose_65plus_pct, na.rm = TRUE)) /
      sd(one_dose_65plus_pct, na.rm = TRUE),
    booster_z =
      (booster_5plus_pct - mean(booster_5plus_pct, na.rm = TRUE)) /
      sd(booster_5plus_pct, na.rm = TRUE),
    dem_pct_2000_z =
      (dem_pct_2000 - mean(dem_pct_2000, na.rm = TRUE)) /
      sd(dem_pct_2000, na.rm = TRUE),
    dem_pct_2020_z =
      (dem_pct_2020 - mean(dem_pct_2020, na.rm = TRUE)) /
      sd(dem_pct_2020, na.rm = TRUE)
  )
```

# Writing a new function

Notice that all of the mutations follow the same template:

```
(█ - mean(█, na.rm = TRUE)) / sd(█, na.rm = TRUE)
```

Only one thing varies: the column of data, represented with █

# Components of a function

We create functions like so:

```
name <- function(arguments) {
  body
}
```

Three components:

1. **Name**: the name of the function that we'll use to call it. Maybe `z_score`?
2. **Arguments**: things that we want to vary across calls of our function. We'll use x.
3. **Body**: the code that the function performs.

# Our first function

Convert our template to a function:

```
z_score <- function(x) {
  (x - mean(x, na.rm = TRUE)) / sd(x, na.rm = TRUE)
}
```

Check that it seems to work:

```
z_score(c(1,2, 3, 4, 5))
```

```
## [1] -1.265 -0.632  0.000  0.632  1.265
```

# Cleaning up our code

```
covid_votes |>
  mutate(
    one_dose_5p_z = z_score(one_dose_5plus_pct),
    one_dose_65p_z = z_score(one_dose_65plus_pct),
    booster_z = z_score(booster_5plus_pct),
    dem_pct_2000_z = z_score(dem_pct_2000),
    dem_pct_2020_z = z_score(dem_pct_2020)
  )
```

# across() function

If we want to replace our variables with z-scores, we can use the across() function to perform many mutations at once:

```
covid_votes |>
  mutate(across(one_dose_5plus_pct:dem_pct_2020, z_score))
```

```
## # A tibble: 3,114 x 8
##    fips  county state one_dose_5plus_pct one_dose_65plus_pct
##    <chr> <chr>  <chr>              <dbl>               <dbl>
##  1 26039 Crawf~ MI                -0.508              -0.829
##  2 40015 Caddo~ OK                 1.40                0.843
##  3 17007 Boone~ IL                 0.556               0.795
##  4 12055 Highl~ FL                 0.404               0.720
##  5 34029 Ocean~ NJ                 0.549               0.843
##  6 01067 Henry~ AL                -0.314              -0.0545
##  7 27037 Dakot~ MN                 1.24                0.843
##  8 27115 Pine ~ MN                -0.452              -0.102
##  9 51750 Radfo~ VA                -1.49               -1.16
## 10 22009 Avoye~ LA                -0.231              -0.564
## # i 3,104 more rows
## # i 3 more variables: booster_5plus_pct <dbl>,
## #   dem_pct_2000 <dbl>, dem_pct_2020 <dbl>
```

# Alternative approach

We could also target all the numeric variables:

```
covid_votes |>
  mutate(across(where(is.numeric), z_score))
```

```
## # A tibble: 3,114 x 8
##    fips  county state one_dose_5plus_pct one_dose_65plus_pct
##    <chr> <chr>  <chr>              <dbl>               <dbl>
##  1 26039 Crawf~ MI                -0.508             -0.829
##  2 40015 Caddo~ OK                 1.40               0.843
##  3 17007 Boone~ IL                 0.556              0.795
##  4 12055 Highl~ FL                 0.404              0.720
##  5 34029 Ocean~ NJ                 0.549              0.843
##  6 01067 Henry~ AL                -0.314             -0.0545
##  7 27037 Dakot~ MN                 1.24               0.843
##  8 27115 Pine ~ MN                -0.452             -0.102
##  9 51750 Radfo~ VA                -1.49              -1.16
## 10 22009 Avoye~ LA                -0.231             -0.564
## # i 3,104 more rows
## # i 3 more variables: booster_5plus_pct <dbl>,
## #   dem_pct_2000 <dbl>, dem_pct_2020 <dbl>
```

# Alternative approach

We could also target only the first dose variables:

```
covid_votes |>
  mutate(across(starts_with("one_dose"), z_score))
```

```
## # A tibble: 3,114 x 8
##    fips  county state one_dose_5plus_pct one_dose_65plus_pct
##    <chr> <chr>  <chr>              <dbl>               <dbl>
##  1 26039 Crawf~ MI                -0.508              -0.829
##  2 40015 Caddo~ OK                 1.40                0.843
##  3 17007 Boone~ IL                 0.556               0.795
##  4 12055 Highl~ FL                 0.404               0.720
##  5 34029 Ocean~ NJ                 0.549               0.843
##  6 01067 Henry~ AL                -0.314              -0.0545
##  7 27037 Dakot~ MN                 1.24                0.843
##  8 27115 Pine ~ MN                -0.452              -0.102
##  9 51750 Radfo~ VA                -1.49               -1.16
## 10 22009 Avoye~ LA                -0.231              -0.564
## # i 3,104 more rows
## # i 3 more variables: booster_5plus_pct <dbl>,
## #   dem_pct_2000 <dbl>, dem_pct_2020 <dbl>
```

# Adding arguments to our function

What if we want to be able to control `na.rm` in the calls to `mean()` and `sd()` in our `z_score` function? Add an argument!

```
z_score2 <- function(x, na.rm = FALSE) {
  (x - mean(x, na.rm = na.rm)) / sd(x, na.rm = na.rm)
}
```

```
head(z_score2(covid_votes$one_dose_5plus_pct))
```

```
## [1] NA NA NA NA NA NA
```

```
head(z_score2(covid_votes$one_dose_5plus_pct, na.rm = TRUE))
```

```
## [1] -0.508  1.398  0.556  0.404  0.549 -0.314
```