Microeconometrics (Causal Inference)
Week 2 - Linear regression (OLS) and bootstrapping

Joshua D. Merfeld
KDI School of Public Policy and Management

2023-09-12

What are we doing today?

- This week we will review linear regression (OLS) and its assumptions
    - We will also discuss how to interpret the results of a regression

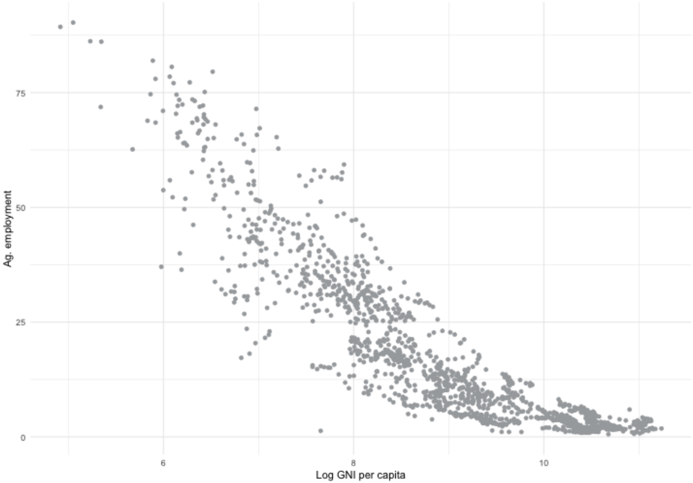
- Some of this will be review, but there will be some new stuff, too
    - For example, we will discuss bootstrapping on Thursday

What are we doing today?

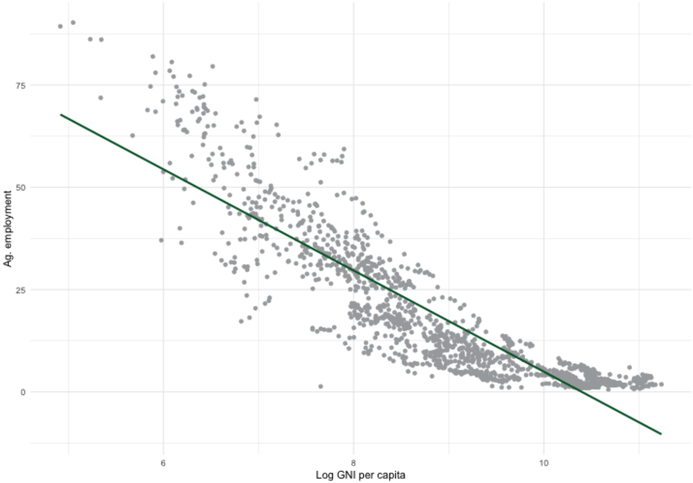
- I will be going through some econometric theory this week

- But we will also be using some data throughout the week
  - This data comes from the World Development Indicators database
  - I have uploaded the csv file to this week's GitHub folder

Fitting lines to data



Which line best fits the data?



Ordinary Least Squares (OLS)

- OLS is a method for fitting a line to data
  - It is the most common method for fitting lines to data

- You have an outcome, let's call it y_i (where y is the outcome and i is an individual)
- You also have a set of covariates, let's call them x_{ik} (where k denotes different covariates)

- The relationship might look something like this, assuming linearity:

$$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \cdots + \beta_k x_{i,k} + \epsilon_i \tag{1}$$

Ordinary Least Squares (OLS)

- OLS is about finding the **average** relationship between Y and a set of covariates, X
  - The fit will never be perfect, which is why we have ϵ_i
  - ϵ_i is the error term and shows that individuals will deviate from the average relationship
  - For example, Bill Gates is a high-school drop out but is still a billionaire; this is deviation from an average relationship between education and income

Minimizing the sum of squared errors

- In order to find a “line of best fit”, we need an objective function
  - We want to minimize the distance between the line and the data, but how do we define that distance?

- We can define the distance as the sum of squared errors (SSE)
  - The SSE is the sum of the squared difference between the actual value and the predicted value
  - The predicted value is the value on the line of best fit

- In other words:

$$\text{SSE} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{2}$$

Minimizing the sum of squared errors

$$\text{SSE} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3}$$

- $\hat{y}_i$ is the predicted value of y_i for individual i
- But how do we find $\hat{y}_i$?
  - We need to find the line of best fit
  - We need to find the values of $\beta_0, \beta_1, \ldots, \beta_k$ that minimize the SSE.
- We can do this by noting that $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i,1} + \hat{\beta}_2 x_{i,2} + \cdots + \hat{\beta}_k x_{i,k}$:

$$\text{SSE} = \sum_{i=1}^{n} (y_i - \beta_0 - \hat{\beta}_1 x_{i,1} - \hat{\beta}_2 x_{i,2} - \cdots - \hat{\beta}_k x_{i,k})^2 \tag{4}$$

In matrix form

- Let's rewrite the equation in matrix form:
- $\mathbf{Y}$ is a vector of outcomes, $\mathbf{X}$ is a matrix of covariates, and β is a vector of coefficients:

$$\text{SSE} = (\mathbf{Y} - \mathbf{X}\beta)'(\mathbf{Y} - \mathbf{X}\beta) \tag{5}$$

- It turns out that this becomes an optimization problem
- We know the $\mathbf{Y}$ and $\mathbf{X}$, so we want to minimize this with respect to β:

$$\min_{\beta}(\mathbf{Y} - \mathbf{X}\beta)'(\mathbf{Y} - \mathbf{X}\beta) \tag{6}$$

The solution

- We know the solution to this:

$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \tag{7}$$

- Seer this formula into your brain!

A problem: uncertainty

- We have a solution, but we have a problem: uncertainty
  - We generally do not have the population, which means $\hat{\beta}$ is just a sample estimate of the population parameters β
  - Note the use of hats to denote sample estimates

- We need to know how uncertain we are about our estimates
  - We need to know how much $\hat{\beta}$ will vary from sample to sample
  - Before we do this, we need to do a little bit of background on probability and statistics

Two important theorems

- There are two important theorems that we need to know about
    - The Law of Large Numbers
    - The Central Limit Theorem

- Let's go through these one at a time

Law of Large Numbers (LLN)

- The LLN says that as the sample size increases, the sample mean will converge to the population mean
  - In other words, as $n \to \infty$, $\bar{x} \to \mu$
    - where n is the sample size, $\bar{x}$ is the sample mean, μ is the true population mean
- You don't have to take my word for this, though. Let's see it in action.

LLN as an example

Let's create a "population" of 100,000 random numbers from a normal distribution with mean 0 and standard deviation 1

```r
# note that rnorm is a "random" number generator, so we need to set a seed to make sure we get the same results each time
set.seed(1304697)
# NOTE: just set the seed once, at the top of your script. Then run everything and you will get reproducible results
population <- rnorm(100000, mean = 0, sd = 1)

mean(population)
```

```
## [1] -0.003674513
```

LLN as an example

- So we know the true mean is $\mu = -0.0036745$
- Let’s see what happens as we take a sample of size 10:

```
sample <- population[sample(1:length(population), 10, replace = FALSE)]

mean(sample)
```

```
## [1] -0.07407166
```

- Let’s see what happens as we take a sample of size 100:

```
sample <- population[sample(1:length(population), 100, replace = FALSE)]

mean(sample)
```

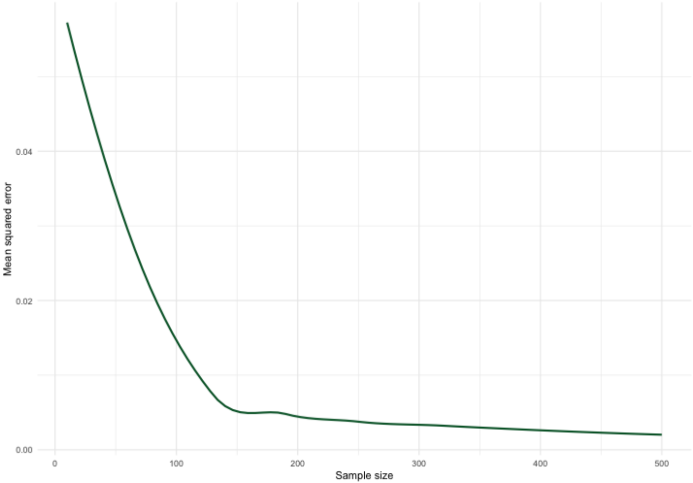
```
## [1] -0.01106315
```

Suppose we did this a bunch of times. . .

- Let's say we took a sample of size 10, 100 times
    - We would get 100 different sample means
    - We can calculate how "far" each sample mean is from the true mean
    - We can do the same thing for a bunch of different sample sizes

- Let's then plot the average mean squared error (MSE) for each sample size
    - The MSE is the average squared difference between the sample mean and the true mean
    - In other words, it is the average of $(\bar{x} - \mu)^2$

The results


Mean squared error
0.00
0.02
0.04
0
100
200
300
400
500
Sample size

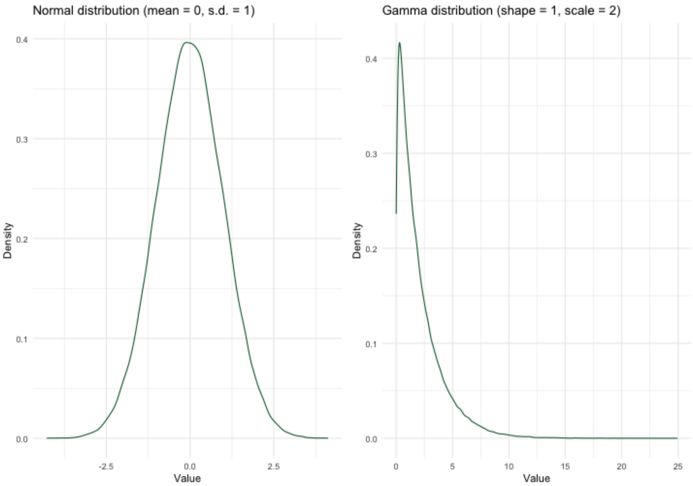
Non-normal distributions

- This doesn't just work with normal distributions
- Here's an example with a more skewed distribution
- Let's create a "population" of 100,000 random numbers from a gamma distribution with a shape (k) of 1 and a scale (θ) of 2

```r
population2 <- rgamma(100000, shape = 1, scale = 2)
mean(population2)
```

```
## [1] 2.005826
```

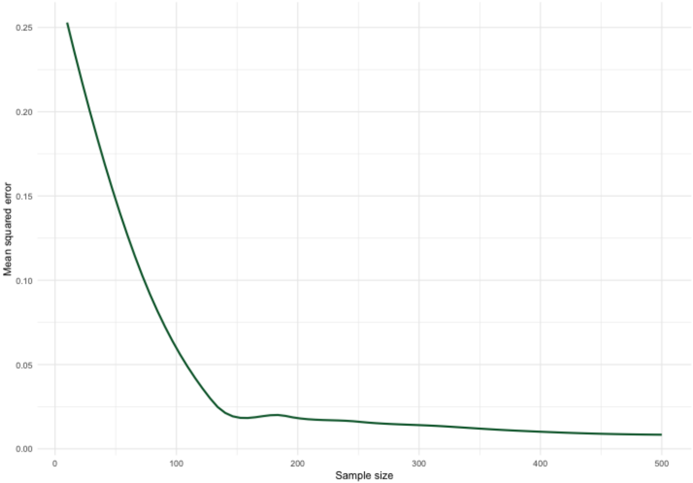
The two populations



Normal distribution (mean = 0, s.d. = 1)

Gamma distribution (shape = 1, scale = 2)

The results


0.25
0.20
0.15
0.10
0.05
0.00
Mean squared error
0
100
200
300
400
500
Sample size

The takeaway from the LLN

- The main takeaway from the LLN is that as the sample size increases, the sample mean will converge to the population mean
  - In other words, as $n \rightarrow \infty$, $\bar{x} \rightarrow \mu$

- This holds for many different distributions!
  - Something we will return to is that this also holds for the distribution of $\hat{\beta}$

The Central Limit Theorem

- The LLN is about the **mean**
    - The Central Limit Theorem (CLT) is about the **distribution**

- The CLT says that as the sample size increases, the distribution of the sample mean will converge to a normal distribution
    - In other words, as $n \rightarrow \infty$, $\bar{x} \sim N(\mu, \sqrt{\sigma^2/n})$
    - where μ is the true population mean and σ^2 is the true population variance

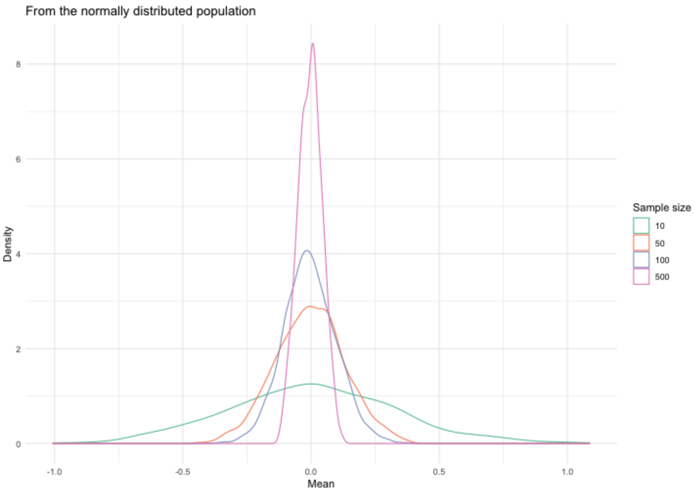
The CLT, empirically

- We can do something similar to what we did with the LLN
  - Instead of looking at means, though we will look at *distributions of the mean*
  - In other words, we will density functions of the sample means

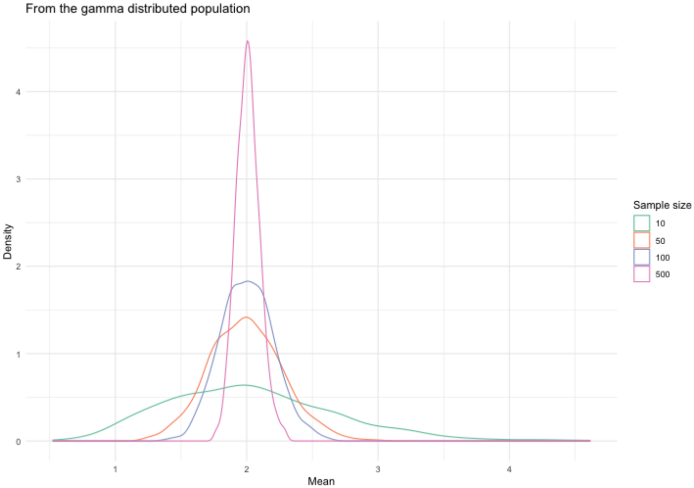
- I am going to take a sample and find the mean
  - Then I'm going to do it again
  - And again
  - 1,000 times

- Then I will plot the density of the sample means, for four separate sample sizes (10, 50, 100, and 500)

The CLT with four sample sizes: 10, 50, 100, and 500 and 1,000 replications



The CLT with four sample sizes: 10, 50, 100, and 500 and 1,000 replications



The CLT lets us quantify uncertainty

- The key thing about the CLT is that the math behind us *lets us quantify the uncertainty!*
- For sample means, the CLT says that the standard error of the mean is:
  - $\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}}$
    - where σ is the population standard deviation and n is the sample size
- Of course, we don't know σ, but we can estimate it!

The CLT lets us quantify uncertainty

- We can estimate σ with s
  - s is the sample standard deviation
  - So we can estimate the standard error of the mean as:
    - $SE_{\bar{x}} = \frac{s}{\sqrt{n}}$
- This “standard error of the mean” is the standard deviation of the distribution of the sample mean
  - That distribution is called the sampling distribution of the mean

Uncertainty in OLS

- Let's go back to our (population) regression equation:

$$\mathbf{Y} = \mathbf{X}\beta + \boldsymbol{\epsilon} \tag{8}$$

- What drives uncertainty here?
  - The **error term**, ϵ
  - Note that an error term is a *population* parameter
  - The sample analog is the **residual**

Uncertainty in OLS

- We can see that the error term is responsible for uncertainty by thinking about the deviation of estimated regression coefficients ($\hat{\beta}$) from the true regression coefficients (β)

$$\boldsymbol{Y} = \beta\mathbf{X} + \boldsymbol{\epsilon} \qquad\qquad \hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{Y}$$

$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'(\beta\mathbf{X} + \boldsymbol{\epsilon})$$

$$\hat{\beta} = \beta + (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{\epsilon} \tag{9}$$

- β is fixed, but ϵ is random
  - So $\hat{\beta}$ is random, too
  - That randomness is driven by the error term, $\boldsymbol{\epsilon}$

Variance of $\hat{\beta}$

- Variance is about how much a random variable varies from its mean
  - It's a second-order moment: it's about how much a random variable varies from its mean (true value!)
  - Note how we use expectations here, since we are talking about population parameters

$$Var(\hat{\beta}|\boldsymbol{X}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbb{E}(\epsilon'\epsilon|\boldsymbol{X})\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \tag{10}$$

- Well that doesn't look very good, does it?
  - We can simplify this... with an assumption on $\mathbb{E}(\epsilon'\epsilon|\boldsymbol{X})$, which we can denote $\boldsymbol{\Omega}$

Homoskedasticity

- If we are willing to assume structure on the error term, we can simplify the variance

Assumption: Homoskedasticity

$\mathbf{\Omega} = \sigma^2\mathbf{I}$, where $\mathbf{I}$ is the identity matrix (ones on the diagonal and zeros elsewhere).

This simplifies the variance to:

$$Var(\hat{\beta}_{homoskedasticity}) = \sigma^2(\mathbf{X}'\mathbf{X})^{-1} \tag{11}$$

Homoskedasticity

- What, exactly, is homoskedasticity?
- It means that the variance of the error term is constant
  - In other words, the variance of the error term does not depend on the covariates
  - This is a very strong assumption and is often violated in practice

Estimating the variance under homoskedasticity

- Even though it may not be reasonable, let's assume homoskedasticity
  - Then, the variance of $\hat{\beta}_{homoskedasticity}$ is:

$$\sigma^2(\mathbf{X}'\mathbf{X})^{-1} \tag{12}$$

- There's a problem, though: we don't know σ^2
  - We can estimate it, just like with the mean
  - We can estimate it with the residual sum of squares (RSS)
    - The RSS is the sum of the squared *residuals*

Estimating the variance under homoskedasticity

- We can estimate σ^2 as $\frac{RSS}{n-k-1}$, where n is the sample size and k is the number of covariates
  - The extra 1 is due to the intercept
- We estimate the variance as:

$$
\begin{aligned}
Var(\hat{\beta}_{homoskedasticity}) &= \hat{\sigma}^2(\mathbf{X}'\mathbf{X})^{-1} && (13)\\
&= \frac{RSS}{n-k-1}(\mathbf{X}'\mathbf{X})^{-1} && (14)\\
&= \frac{\sum_1^n (y_i-\hat{y}_i)^2}{n-k-1}(\mathbf{X}'\mathbf{X})^{-1} && (15)
\end{aligned}
$$

What does this equation tell us?

$$Var(\hat{\beta}_{homoskedasticity}) = \frac{\sum_1^n (y_i - \hat{y}_i)^2}{n - k - 1} (\mathbf{X}'\mathbf{X})^{-1} \tag{16}$$

- There are a couple things to point out here
  - First, the variance of $\hat{\beta}$ is a function of the variance of the error term
    - The more unexplained variation in the outcome, the larger the variance of our estimated coefficients
  - Second, the variance of $\hat{\beta}$ is also a function of the variance of the covariates
    - The more variation in the covariates, the smaller the variance of our estimated coefficients

Why do we care about variance?

- We care about variance because it allows us to do hypothesis testing
  - We can test whether or not a coefficient is different from zero (or any other value)
  - We can test whether or not two coefficients are different from each other

- We only have a *sample*, which means we cannot say anything with certainty
  - We can only say something with a certain degree of confidence

- Taking into account variance allows us to say something about *the population from which the sample is drawn*
  - We generally don't care about the sample itself, only what it tells us about something larger

Confidence intervals

- Suppose we have an estimated coefficient, $\hat{\beta}$
- Let's create a simulation exercise with a "population"

```
# some variable, x, randomly distributed between 0 and 5
x <- runif(100000, min = 0, max = 5)
# y is a function of x, plus some random error
y <- 3*x + rnorm(100000, mean = 0, sd = 1)
# put it into a data frame
df <- as_tibble(cbind(y, x))

# what is the "true" value of beta in our population of 100,000 people?
lm(y ~ x, data = df)
beta <- coef(lm(y ~ x, data = df))[2]
beta
```

Confidence intervals

```r
# what is the "true" value of beta in our population of 100,000 people?
# I'm ignoring uncertainty for now
lm(y ~ x, data = df)
```

```
##
## Call:
## lm(formula = y ~ x, data = df)
##
## Coefficients:
## (Intercept)            x
##   -0.006007     3.003164
```

```r
beta <- coef(lm(y ~ x, data = df))[2]
beta
```

```
##        x
## 3.003164
```

Confidence intervals

Now let's take a sample of just 50. In a smaller sample, the coefficient can be quite different sometimes.

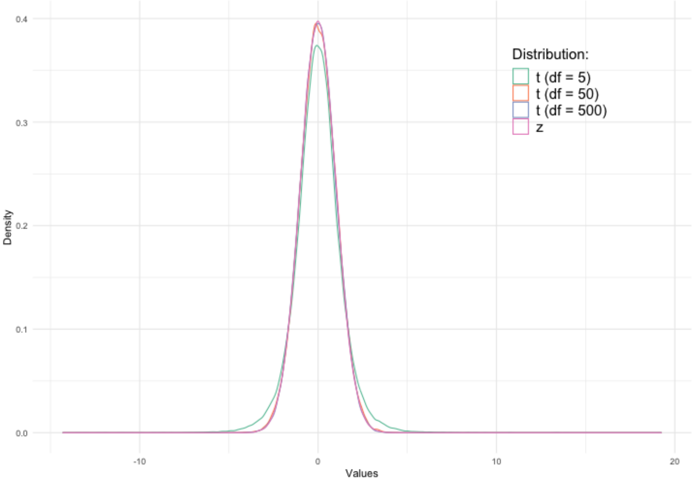
```r
sample <- df[sample(1:nrow(df), 50, replace = FALSE),]
summary(lm(y ~ x, data = sample))
```

```
##
## Call:
## lm(formula = y ~ x, data = sample)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.95124 -0.67324  0.05356  0.61053  2.52386
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.3640     0.3214   1.133    0.263
## x             2.9222     0.1045  27.971   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.026 on 48 degrees of freedom
## Multiple R-squared:  0.9422, Adjusted R-squared:  0.941
## F-statistic: 782.4 on 1 and 48 DF,  p-value: < 2.2e-16
```

Confidence intervals

- We want to put a confidence interval around our estimate of $\hat{\beta}$
  - This will give us some idea of what the "true" value is (we know it here, but we generally do not!)
- You hopefully remember two separate distributions from earlier classes:
  - z distribution
  - t distribution
- In theory, you should always use a t distribution if *you do not know the true population* σ^2 (which we usually don't)
  - We will always use t distributions

Different distributions



Critical values

- I'll assume you all remember critical values from earlier metrics classes
  - If not, ask now!
  - Confidence level (e.g. 0.05 for 95% confidence): α

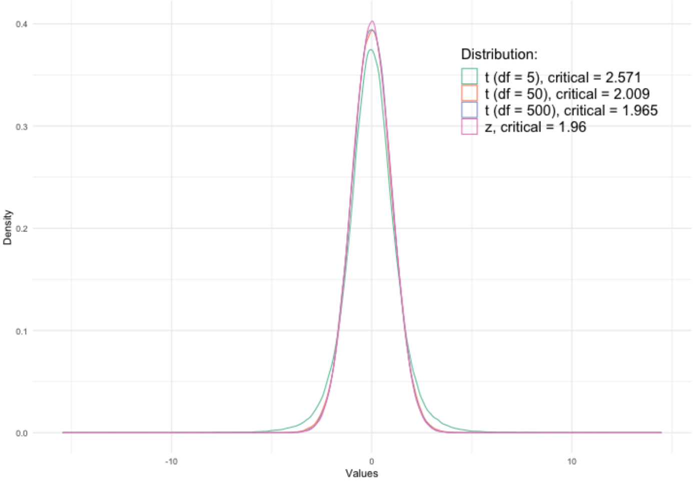
Critical values

- I'll assume you all remember critical values from earlier metrics classes
  - If not, ask now!
  - Confidence level (e.g. 0.05 for 95% confidence): α
- We construct our CI as:

$$CI = \left(\hat{\beta} - t_{n-k-1}^{1-\frac{\alpha}{2}} \times \sqrt{Var(\hat{\beta})},\ \ \hat{\beta} + t_{n-k-1}^{1-\frac{\alpha}{2}} \times \sqrt{Var(\hat{\beta})}\right) \tag{17}$$

- At smaller sample sizes, the t distribution has fatter tails
  - This means that we need to be more uncertain about the true value of β and the critical value can be quite different

Critical values for the different distributions

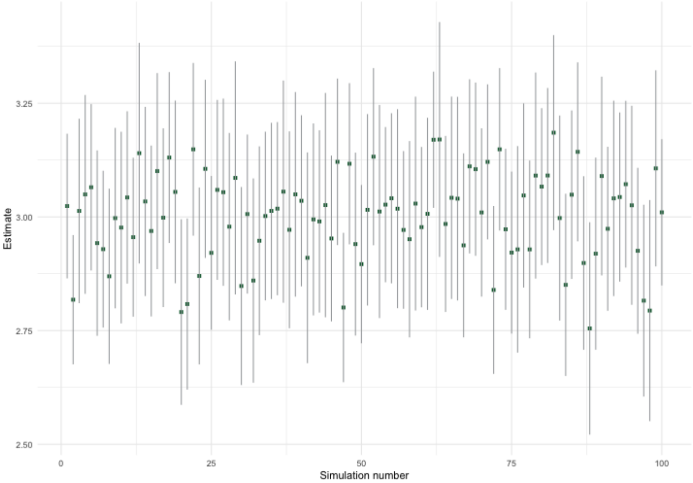


Understanding the confidence interval

- Let's go back to our example with the sample of 50
- What, exactly, does the confidence interval represent?

- Suppose we took a bunch of samples of size 50, let's say 100 of them
  - We would get a bunch of different estimates of $\hat{\beta}$
  - Let's plot them along with the confidence intervals

Degrees of freedom: $n - k - 1 = 50 - 1 - 1 = 48$



```
## [1] "Coverage rate is defined as the proportion of samples for which the CI contains the true value: 0.94"
```

Confidence intervals

- Confidence intervals are a frequentist idea
  - They are about the probability of the true value being in the interval

- We usually can't know for sure whether or not the true value is in the interval
  - The probability is only about "averages" over many samples

- We can change our confidence level:
  - $\alpha = 0.01$ (confidence level of 99%) gives us a wider interval
  - $\alpha = 0.10$ (confidence level of 90%) gives us a narrower interval

Hypothesis testing

- We can also explicitly test whether or not a coefficient is different from some value
  - What is the most common value we are interested in?

Hypothesis testing

- We can also explicitly test whether or not a coefficient is different from some value
  - What is the most common value we are interested in?
  - Zero!
- We can do this with a t-test
  - Null hypothesis (H_0): what we assume is true
  - Alternative hypothesis (H_1): not the null
- Common example with a regression coefficient:
  - $H_0 : \beta = 0$
  - $H_1 : \beta \neq 0$
- NOTE: We are testing for the POPULATION parameter, not the sample statistics

Hypothesis testing

- t-test:

$$\hat{t} = \frac{\hat{\beta} - H_0}{SE(\hat{\beta})}, \tag{18}$$

  where $SE(\hat{\beta})$ is the standard error of $\hat{\beta}$, or $\sqrt{Var(\hat{\beta})}$. I use $\hat{t}$ here to underline that this comes from our sample statistics.

- We can then compare this to a critical value
  - If $|\hat{t}| > t_{n-k-1}^{1-\frac{\alpha}{2}}$, then we reject the null
  - If $|\hat{t}| < t_{n-k-1}^{1-\frac{\alpha}{2}}$, then we fail to reject the null

Type 1 and type 2 errors

- Type 1 error: we reject the null when it is true
  - α is the probability of a type 1 error

- Type 2 error: we fail to reject the null when it is false
  - We do not know the probability of a type 2 error

Interpreting the output

```r
# using feols (from fixest package) instead of lm now
feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data)
```

```
## OLS estimation, Dep. Var.: agemployment
## Observations: 1,420
## Standard-errors: IID
##                   Estimate Std. Error   t value  Pr(>|t|)
## (Intercept)     105.201328   2.731881  38.50875 < 2.2e-16 ***
## log(gnipc)      -10.112581   0.239763 -42.17745 < 2.2e-16 ***
## lfp               0.241674   0.015230  15.86875 < 2.2e-16 ***
## log(population)   0.376890   0.129339   2.91398 0.0036246 **
## urbanpop         -0.208143   0.016205 -12.84437 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 7.33605   Adj. R2: 0.834605
```

Something really nice about fixest (feols)

```r
# using feols instead of lm now
reg1 <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data)
reg2 <- feols(servicesemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data)
etable(reg1, reg2,
        se.below = TRUE, se.row = FALSE,
        digits = 3,
        digits.stats = 3,
        fitstat = c("r2", "ar2", "f", "n")
        )
```

```
##                         reg1               reg2
## Dependent Var.: agemployment servicesemployment
##
## Constant          105.2***            -7.05**
##                     (2.73)             (2.18)
## log(gnipc)        -10.1***             7.35***
##                     (0.240)            (0.191)
## lfp                 0.242***          -0.048***
##                     (0.015)            (0.012)
## log(population)     0.377**           -0.873***
##                     (0.129)            (0.103)
## urbanpop           -0.208***           0.267***
##                     (0.016)            (0.013)
## _______________ ____________ __________________
## R2                     0.835              0.849
## Adj. R2                0.835              0.849
## F-test               1,791.1            1,995.2
## Observations           1,420              1,420
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Some important statistics

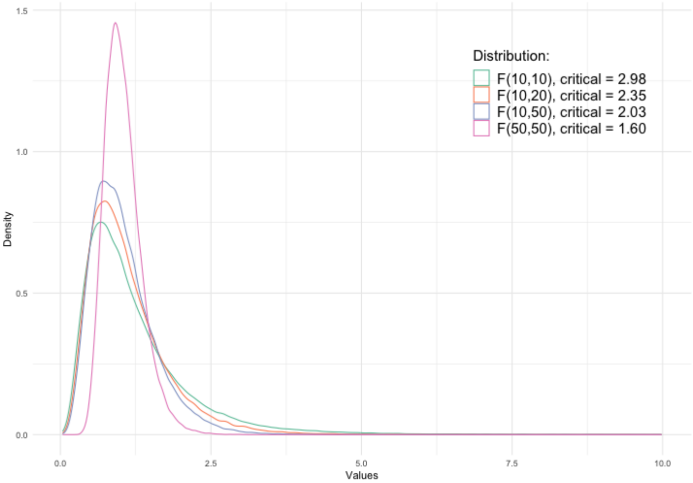
- R^2: the proportion of variation in the outcome explained by the model
  - R^2 is always between 0 and 1
  - R^2 is a measure of *fit*, but you don't really want to use it to judge how "good" your model is
  - Adding more variables to your model can NEVER decrease R^2
- Adjusted R^2 ($\bar{R}^2$): penalizes R^2 for adding more variables
  - Adjusted R^2 is always between 0 and 1
  - Adjusted R^2 is a measure of *fit*, but you don't really want to use it to judge how "good" your model is
  - Adding more variables to your model can decrease adjusted R^2

$$\bar{R}^2 = 1 - (1 - R^2)\frac{n-1}{n-k-1} \tag{19}$$

F tests

- The F distribution is important in linear regression
  - It is the ratio of two chi-squared distributions, so it has two different degrees of freedom
  - We sometimes refer to these as the "numerator" and "denominator" degrees of freedom
  - An F statistic is always positive
- In the previous regression, the F at the bottom is a test of whether or not all of the coefficients are equal to zero
  - This is a joint test of the coefficients
  - The null hypothesis is that *all* of the coefficients are equal to zero
  - The alternative hypothesis is that *at least one* of the coefficients is not equal to zero
- In practice, we usually just use the p-value because of the fact that critical values of F distributions can be quite different depending on the d.f.

F distributions with different degrees of freedom



F distributions and hypothesis testing

- t-tests are the workhorse for inference on a *single* coefficient
- But what if we want to test multiple coefficients at once?
  - F-tests
- Consider the following regression:

$$y_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \beta_3 x_{3,i} + \epsilon_i \tag{20}$$

- How might we think about testing whether $\beta_1 = \beta_2 = 0$?

F distributions and hypothesis testing

$$y_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \beta_3 x_{3,i} + \epsilon_i \tag{21}$$

- If we assume $\beta_1 = \beta_2 = 0$, that is the same as estimating:

$$y_i = \tilde{\beta}_0 + \tilde{\beta}_1 x_{1,i} + \tilde{\epsilon}_i \tag{22}$$

- Intuitively, we want to know whether the first specification is "better" than the second. We can calculate the estimated variance of the residuals in each specification:

$$\hat{\sigma}^2 = \sum_1^n (y_i - \hat{y}_i)^2 \tag{23}$$

$$\tilde{\sigma}^2 = \sum_1^n (y_i - \tilde{y}_i)^2 \tag{24}$$

F distributions and hypothesis testing

$$\hat{\sigma}^2 = \sum_1^n (y_i - \hat{y}_i)^2 \tag{25}$$

$$\tilde{\sigma}^2 = \sum_1^n (y_i - \tilde{y}_i)^2 \tag{26}$$

- We can then calculate the F statistic:

$$F = \frac{(\tilde{\sigma}^2 - \hat{\sigma}^2)/q}{\hat{\sigma}^2/(n-k-1)}, \tag{27}$$

where q is the number of restrictions we are testing. In this case, $q = 2$ (two restricted coefficients).

F distributions and hypothesis testing

$$F = \frac{(\tilde{\sigma}^2 - \hat{\sigma}^2)/q}{\hat{\sigma}^2/(n-k-1)} \tag{28}$$

- The numerator ($(\tilde{\sigma}^2 - \hat{\sigma}^2)/q$) is distributed as χ^2 with q degrees of freedom
- The denominator ($\hat{\sigma}^2/(n-k-1)$) is distributed as χ^2 with $n-k-1$ degrees of freedom
- Their ratio is distributed as F with q and $n-k-1$ degrees of freedom:

$$F = \frac{(\tilde{\sigma}^2 - \hat{\sigma}^2)/q}{\hat{\sigma}^2/(n-k-1)} \sim F(q, n-k-1) \tag{29}$$

- Now you see why they are sometimes referred to as “numerator” and “denominator” degrees of freedom!

F-test by hand - note the relationship with the t-test for a single coefficient!

```
# full specification
full <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data)
# restricted
restricted <- feols(agemployment ~ log(gnipc) + lfp + urbanpop, data = data)
sigma_full <- sum((full$resid)^2) # sigma for full
sigma_restricted <- sum((restricted$resid)^2) # sigma for restricted
F <- ((sigma_restricted - sigma_full)/1)/(sigma_full/(full$nobs - 4 - 1)) # calculate F statistic
F
```

```
## [1] 8.49128
```

```
full
```

```
## OLS estimation, Dep. Var.: agemployment
## Observations: 1,420
## Standard-errors: IID
##                   Estimate Std. Error   t value  Pr(>|t|)
## (Intercept)     105.201328   2.731881  38.50875 < 2.2e-16 ***
## log(gnipc)      -10.112581   0.239763 -42.17745 < 2.2e-16 ***
## lfp               0.241674   0.015230  15.86875 < 2.2e-16 ***
## log(population)   0.376890   0.129339   2.91398 0.0036246 **
## urbanpop         -0.208143   0.016205 -12.84437 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 7.33605   Adj. R2: 0.834605
```

- For a *single* coefficient, $F = t^2$

We've been assuming homoskedasticity

- In the previous examples, we've been assuming *homoskedasticity*

$$\mathbb{E}(\epsilon'\epsilon|\boldsymbol{X}) = \boldsymbol{\Omega} = \sigma^2\mathbf{I} \tag{30}$$

- In practice, this assumption is unlikely to be true
  - You can test for it, but it's not really worth it. Just assume it's not true.
  - In other words, assume $\boldsymbol{\Omega} \neq \sigma^2\mathbf{I}$
- **Homoskedasticity** can take many forms
  - It simply means that the variance is not constant across observations
  - Let's talk about a general estimator under heteroskedasticity and another one under clustering

Sandwich estimators

- There is a broad class of estimators that are called “sandwich estimators” because of their form
- The baseline *heteroskedasticity-consistent* standard errors are:

$$Var(\hat{\beta}) = (\mathbf{X}'\mathbf{X})^{-1} \left(\sum_i^n X_i X_i' \sigma_i^2 \right) (\mathbf{X}'\mathbf{X})^{-1} \tag{31}$$

- Of course we don’t know σ_i^2, so we calculate a feasible estimate using the residuals:

$$Var(\hat{\beta})^{HC0} = (\mathbf{X}'\mathbf{X})^{-1} \left(\sum_i^n X_i X_i' \hat{e_i}^2 \right) (\mathbf{X}'\mathbf{X})^{-1} \tag{32}$$

- This is sometimes referred to as $HC0$ standard errors, or **Eicker-White** standard errors

Bias in HC0

- But there's a problem: it turns out that $\sum_i \hat{e_i}^2$ is a biased estimator for σ_i^2!
  - It's biased towards zero ("attenuated") so the standard errors are a bit too small

- The next step is to use a scaling factor to "correct" the standard errrors
  - This is sometimes referred to as $HC1$ standard errors

$$Var(\hat{\beta})^{HC1} = \frac{n}{n-k-1}(\mathbf{X'X})^{-1}\left(\sum_i^n X_i X_i' \hat{e_i}^2\right)(\mathbf{X'X})^{-1} \tag{33}$$

- This is a bit ad hoc, but is generally preferred (Hanson, 2022).
  - **This is currently the default in Stata (with the option *robust*)**

Two more. . .

- To confuse you more, there are two additional alternatives:

$$Var(\hat{\beta})^{HC2} = (\mathbf{X}'\mathbf{X})^{-1}\left(\sum_i^n (1-h_{ii})^{-1} X_i X_i' \hat{e}_i{}^2\right)(\mathbf{X}'\mathbf{X})^{-1} \quad (34)$$

$$Var(\hat{\beta})^{HC3} = (\mathbf{X}'\mathbf{X})^{-1}\left(\sum_i^n (1-h_{ii})^{-2} X_i X_i' \hat{e}_i{}^2\right)(\mathbf{X}'\mathbf{X})^{-1} \quad (35)$$

- h_{ii} has to do with a diagonal element of an orthogonal projection matrix. We're going to ignore this for now.

- These are sometimes referred to as $HC2$ and $HC3$ standard errors

- When I refer to HC standard errors, I will be talking about $HC1$ unless I say otherwise

Clustering

- One of the most common types of heteroskedasticity is clustering
  - This is when the error term is correlated within groups
  - For example, if we have data on students within schools, the error term for students in the same school is likely to be correlated

- Let K denote the number of clusters and k a specific cluster
  - We can then calculate the cluster-robust variance estimator as:

$$Var(\hat{\beta})^{cluster} = (\mathbf{X}'\mathbf{X})^{-1} \sum_{k=1}^{K} \left(\sum_{i=1}^{n_k} X_{ik} \hat{e}_{ik} \right) \left(\sum_{l=1}^{n_k} X_{lk} \hat{e}_{lk} \right)' (\mathbf{X}'\mathbf{X})^{-1} \tag{36}$$

- The basic intuition is that the more correlated the observations in a cluster is, the larger the variance of the estimated coefficients
  - You can actually show this with some algebra and some assumptions
  - This *intra-cluster correlation* is often referred to as ρ

Clustering

- Suppose you have no clustering at all. Then your effective sample size is n.

- Suppose all of your observations are perfectly correlated with the cluster ($\rho = 1$). Then your effective sample size is just the number of clusters.

- Of course, neither assumption is likely to ever be true. We are usually somewhere in between, but hopefully this helps with intuition.

- All of these alternative standard error (variance) calculations only affect the standard errors. **Coefficients do not change.**

Clustering and standard errors

```r
reg1 <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data) # assume iid
reg2 <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data, cluster = "country") # clustering
etable(reg1, reg2,
        se.below = TRUE, se.row = FALSE,
        digits = 3, digits.stats = 3,
        fitstat = c("r2", "ar2", "f", "n")
        )
```

```
##                         reg1         reg2
## Dependent Var.: agemployment agemployment
##
## Constant          105.2***     105.2***
##                    (2.73)       (9.40)
## log(gnipc)        -10.1***     -10.1***
##                    (0.240)      (0.713)
## lfp                 0.242***     0.242***
##                    (0.015)      (0.041)
## log(population)     0.377**      0.377
##                    (0.129)      (0.462)
## urbanpop          -0.208***    -0.208***
##                    (0.016)      (0.047)
## _______________ ____________ ____________
## R2                     0.835        0.835
## Adj. R2                0.835        0.835
## F-test               1,791.1        178.5
## Observations           1,420        1,420
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

What level do we cluster at?

- This is a hard question to answer in many cases
  - With panel data, we usually cluster at the unit level (e.g. individuals)
  - In the previous example, this was the country level

- Always cluster at the level of randomization
  - If the data is collected using a complex survey design, we cluster at the the "enumeration area" level

- **We do not know the "correct" level at which to cluster except in rare situations.**

Some complications

- What if the size of clusters is quite different?
  - Variance is calculated as the sum of errors within clusters, so different cluster sizes means differences in variance
  - This is basically heteroskedasticity
  - The key takeaway from this is that smaller samples may show bias (finite-sample bias)
  - Which leads to. . .
- What if K is small, i.e. we have relatively few clusters?
  - One way to think about it is that our effective sample size is closer to K than n
  - Unless we are willing to assume normality of the error term (which we already know we can't), small samples can be problematic
  - What to do? We will return to this in the next section on bootstrapping.

Before moving on... the regression "assumptions"

Have you seen these before?

1. **Linearity:** the relationship between the outcome and covariates is linear
2. **Full rank:** the matrix $\mathbf{X}$ has full rank
   - Another way to say this: no perfect collinearity
3. **Exogeneity:** the error term is uncorrelated with the covariates
4. **Homoskedasticity:** the variance of the error term is constant
5. **Normality:** the error term is normally distributed

Linearity

1. **Linearity:** the relationship between the outcome and covariates is linear - Regardless of the “true” relationship, OLS will always give you the best linear approximation

Full rank of X

2. **Full rank:** the matrix $\mathbf{X}$ has full rank, or no perfect collinearity - This is required to estimate the regression. - If there is perfect collinearity, then the matrix $\mathbf{X}'\mathbf{X}$ is not invertible - Stata and R will automatically drop variables to make this happen

Full rank of X

3. **Exogeneity:** the error term is uncorrelated with the covariates - Ignore this for now. We'll come back to it.

Homoskedasticity

4. **Homoskedasticity:** the variance of the error term is constant - Obviously not required! We've already talked about this. There are corrections.

Normality

5. **Normality:** the error term is normally distributed - This is not required for OLS! - The only time this is important is with small samples. - With small samples, we need to assume normality to do hypothesis testing. - With large samples, we can use the central limit theorem to get around non-normality. - A note: the more “non-normal” the error term is, the larger the sample size we need for appropriate inferences.

Bootstrapping (resampling)

- It turns out that there is an alternative to calculting standard errors using formulas
  - It's called bootstrapping
  - It's a resampling method
    - I'll abuse terminology and will generally refer to all of these as "bootstrapping"

- Bootstrapping originated with Efron (1970)
  - Lots of work since then
  - There are many different types of bootstrapping

Resampling *with replacement*

- Bootstrapping is about sampling from your data
  - Something you might not be used to: it is sampling *with replacement*
- Imagine a vector with 10 elements: $\{1,2,3,4,5,6,7,8,9,10\}$
  - Imagine we sample a randomly selected element. Let's say we get 5.
  - Now, we want to take another randomly selected element. The question: do we put the 5 back in or not?
  - With replacement: put the 5 back in
  - Without replacement: don't put the 5 back in
- Bootstrapping is *with replacement*
  - Put the 5 back in!

Resampling *with replacement*

```
## # A tibble: 10 x 2
##    country  year
##    <chr>   <dbl>
##  1 AUT      2003
##  2 BGD      2003
##  3 BEL      2003
##  4 BEN      2003
##  5 BRA      2003
##  6 BGR      2003
##  7 KHM      2003
##  8 CAN      2003
##  9 CHL      2003
## 10 COL      2003
```

Resampling 10 observations *with replacement*

- The key is that we sample rows – i.e. observations. We do not sample columns. We take an observation and all its values.

```
## # A tibble: 10 x 2
##    country  year
##    <chr>   <dbl>
##  1 CHL      2003
##  2 COL      2003
##  3 KHM      2003
##  4 BEN      2003
##  5 AUT      2003
##  6 AUT      2003
##  7 BRA      2003
##  8 CAN      2003
##  9 BEN      2003
## 10 CAN      2003
```

Bootstrapping to estimate variance

- Suppose we are interested in the variance of a coefficient estimate, $\hat{\beta}_1$
    - We can use bootstrapping to estimate the variance of $\hat{\beta}_1$
    - Keeping it simple, here is our regression:

$$y_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \beta_3 x_{3,i} + \beta_4 x_{4,i} + \epsilon_i \tag{37}$$

```r
feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data)
```

```
## OLS estimation, Dep. Var.: agemployment
## Observations: 1,420
## Standard-errors: IID
##                   Estimate Std. Error   t value  Pr(>|t|)
## (Intercept)     105.201328   2.731881  38.50875 < 2.2e-16 ***
## log(gnipc)      -10.112581   0.239763 -42.17745 < 2.2e-16 ***
## lfp               0.241674   0.015230  15.86875 < 2.2e-16 ***
## log(population)   0.376890   0.129339   2.91398 0.0036246 **
## urbanpop         -0.208143   0.016205 -12.84437 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 7.33605   Adj. R2: 0.834605
```

Bootstrapping to estimate variance

- Imagine a single bootstrap sample, which we will denote with b
  - We are going to call the estimate we are interested in $\hat{\theta}_b$ (which in this case is a coefficient)
- Let's calculate the variance using our bootstrap as follows:

$$Var(\hat{\theta}) = \frac{1}{B-1}\sum_{b=1}^{B}(\hat{\theta}_b - \bar{\theta})^2, \tag{38}$$

where

$$\bar{\theta} = \frac{1}{B}\sum_{b=1}^{B}\hat{\theta}_b \tag{39}$$

Bootstrapping to estimate variance

- In words:
  - We take a bunch of bootstrap samples (more is better, but we'll do 1,000)
  - We estimate the regression in each bootstrap sample and save the coefficient we are interested in
  - After all 1,000 bootstrap samples, we calculate the variance as the mean (almost) of the squared differences between each bootstrap estimate and the mean of the bootstrap estimates

Bootstrapping to estimate variance

```r
# empty container to hold the estimates
bootvec <- c()
# 1000 samples
for (b in 1:1000){
  # take a sample with replacement (same size as data)
  sample <- data[sample(1:nrow(data), nrow(data), replace = TRUE),]
  # estimate regression
  reg <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = sample)
  # add coefficient to vector
  bootvec <- c(bootvec, reg$coefficients[2])
}
# find variance
var <- sum((bootvec - mean(bootvec))^2)/999
# estimated standard error is the square root
sqrt(var)
```

```
## [1] 0.2373331
```

Bootstrapping to estimate variance

- The estimated standard error is the square root of the variance, and it is quite close to the standard error we got from the regression
- In reality, there's no reason to do this for a single coefficient like this
    - But it's a good way to understand the intuition behind bootstrapping
- Where this becomes really powerful is where there is no closed-form solution for the variance
    - For example, what if we wanted to calculate $\frac{\hat{\beta}_2}{\hat{\beta}_3}$?
    - To be clear, this is a non-sense example, but it's just to illustrate the point

Our non-sense example

```r
# empty container to hold the estimates
bootvec <- c()
# 1000 samples
for (b in 1:1000){
  # take a sample with replacement (same size as data)
  sample <- data[sample(1:nrow(data), nrow(data), replace = TRUE),]
  # estimate regression
  reg <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = sample)
  # add coefficient to vector
  bootvec <- c(bootvec, reg$coefficients[2]/reg$coefficients[3])
}
# find variance
var <- sum((bootvec - mean(bootvec))^2)/999
# what's the mean?
mean(bootvec)
```

```
## [1] -41.87397
```

```r
# estimated standard error is the square root
sqrt(var)
```

```
## [1] 2.857473
```

Clustered bootstrap

- There's a problem with the previous example: we are assuming that the observations are independent

- What if we think there is clustering?
  - We can use a clustered bootstrap, where we randomly sample CLUSTERS, not observations
  - In this case, clusters are countries, so we will randomly sample countries

Clustered bootstrap

```r
feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data, cluster = "country")
```

```
## OLS estimation, Dep. Var.: agemployment
## Observations: 1,420
## Standard-errors: Clustered (country)
##                   Estimate Std. Error    t value   Pr(>|t|)
## (Intercept)     105.201328   9.400891  11.190570  < 2.2e-16 ***
## log(gnipc)      -10.112581   0.713338 -14.176414  < 2.2e-16 ***
## lfp               0.241674   0.041227   5.861974 3.0986e-08 ***
## log(population)   0.376890   0.462084   0.815631 4.1609e-01
## urbanpop         -0.208143   0.047444  -4.387083 2.2344e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 7.33605   Adj. R2: 0.834605
```

Bootstrapping to estimate variance

```r
# empty container to hold the estimates
bootvec <- c()
# get list of countries
countries <- unique(data$country)
# This takes longer to run, so just 500 as an example (try to use more)
for (b in 1:500){
  # take a sample with replacement (same size as data)
  samplecountries <- countries[sample(1:length(countries), length(countries), replace = TRUE)]
  # go through sample countries and get data, appending
  sample <- c()
  for (c in samplecountries){
    sample <- rbind(sample, data %>% filter(data$country==paste0(c)))
  }
  # estimate regression
  reg <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = sample)
  # add coefficient to vector
  bootvec <- c(bootvec, reg$coefficients[2])
}
```

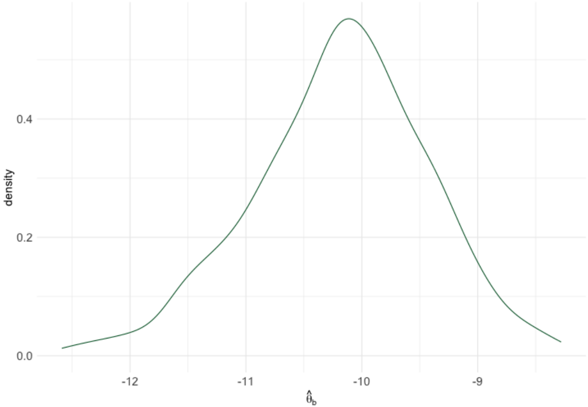
```
## [1] "Standard error: 0.759"
```

Using the bootstrap vector for confidence intervals

- We can use our bootstrap vector to calculate confidence intervals
  - We can use the 2.5th and 97.5th percentiles of the bootstrap vector to get a 95% confidence interval:

$$CI = \left(\hat{\theta}_{\alpha/2}, \hat{\theta}_{1-\alpha/2}\right) \tag{40}$$

Using the bootstrap vector for confidence intervals



```
## [1] "Mean: -10.2"

## [1] "Lower: -11.9"

## [1] "Upper: -8.75"
```

Non-parametric

- This type of bootstrap is non-parametric
  - That is, we are not assuming anything about the distribution of the data
  - The previous kernel density plot wasn't normally distributed, it was skewed, etc.
  - This flexibility is one of the main advantages of bootstrapping

- The main downside to bootstrapping is computation
  - The previous example takes about 5 minutes to run on my computer with just 500 samples (and it's a good computer)
  - More elaborate estimation can take much longer

An actual example from one of my papers

- Brummund and Merfeld (2022)
  - We want to estimate marginal revenue product of labor (MRPL)
- Here is the specification:

$$\begin{aligned}\ln R_{iht} = \alpha_h + I(Farm = 1) \times (\sum_j \beta_j \ln \gamma_{jiht} + \frac{1}{2}\sum_j \sum_k \beta_{jk} \ln \gamma_{jiht} \ln \gamma_{kiht} + \delta C_{iht} \\ + D_{dt} + \eta_m) + \sum_j \beta_j \ln \gamma_{jiht} + \frac{1}{2}\sum_j \sum_k \beta_{jk} \ln \gamma_{jiht} \ln \gamma_{kiht} + \delta C_{iht} + D_{dt} + \eta_m \\ + I(Farm = 1) + \varepsilon_{iht}\end{aligned} \tag{41}$$

An actual example from one of my papers

- This is what we need to estimate, using coefficients from the previous equation:

$$\frac{\partial R^f}{\partial L^f} = \frac{\hat{R}^f_{iht}}{L^f_{iht}} \left[\beta^f_L + \beta^f_{LL} \log L^f_{iht} + \beta_{LA} \log A_{iht} + \beta_{LF} \log F_{iht} \right] \tag{42}$$

$$\frac{\partial R^{nf}}{\partial L^{nf}} = \frac{\hat{R}^{nf}_{iht}}{L^{nf}_{iht}} \left[\beta^{nf}_L + \beta^{nf}_{LL} \log L^{nf}_{iht} + \beta_{LC} \log C_{iht} \right] \tag{43}$$

- It's not possble to get closed-form solutions for the variance of these
    - So what did we do? Bootstrapped!
    - Household panel data, so clustered bootstrap at the household level

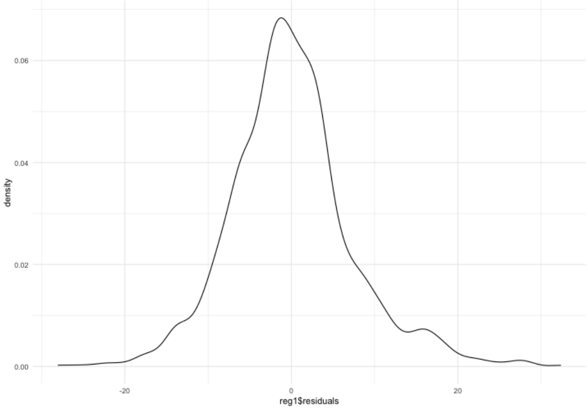
The parametric bootstrap

- There is another type of bootstrap called the parametric bootstrap
  - This is where we assume a distribution for the data
  - We then sample from that distribution

- Essentially, we draw random values from the assumed distribution and add these to the original data
  - We then re-estimate the model
  - We do this many times
  - We then calculate the variance of the coefficient of interest

The parametric bootstrap

```
reg1 <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data)
ggplot() +
  geom_density(aes(x = reg1$residuals)) +
  theme_minimal()
```



The parametric bootstrap

- Seems *relatively* normal. Let's assume a normal distribution!

```
reg1 <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data)
dist <- fitdistr(reg1$residuals, "normal")
dist
```

```
##        mean             sd
##   6.012568e-13   7.336050e+00
##  (1.946786e-01) (1.376585e-01)
```

The parametric bootstrap

```r
# NOTE: This assume homoskedasticity!
# Heteroskedasticity would require allowing the variance to change. This is a simple example.
reg1 <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = data)
dist <- fitdistr(reg1$residuals, "normal")

bootvec <- c()
for (b in 1:1000){
  brep <- data
  # draw from distribution
  brep$agemployment <- brep$agemployment + rnorm(nrow(brep), mean = dist$estimate[1], sd = dist$estimate[2])
  # re-estimate
  reg1 <- feols(agemployment ~ log(gnipc) + lfp + log(population) + urbanpop, data = brep)
  # add to container
  bootvec <- c(bootvec, reg1$coefficients[2])
}
# coefficient was -10.112581
mean(bootvec)
```

```
## [1] -10.11209
```

```r
# standard error of homoskedastic model was 0.239763
sd(bootvec)
```

```
## [1] 0.2392727
```

The parametric bootstrap in my work

- The parametric bootstrap is actually integral to some of the work I do with the UN and World Bank on estimating outcomes

- Small area estimation:

$$y_{isa} = \beta_0 + \beta X_{isa} + \upsilon_{sa} + \epsilon_{isa},$$

where υ_{isa} is a *random effect*. We use a parametric bootstrap to draw from TWO distributions: - The distribution of the random effect - The distribution of the error term

Class exercise

- I'd like you all to try bootstrapping!
  - This is a useful exercise because it lets you practice quite a few important skills in R

- The task:
  - Download `bootstrapdata` from GitHub. This data has just one variable in it: `x`
  - Calculate the 75th percentile of `x` using the function `quantile()`
  - Find the 95% confidence interval for the 75th percentile using the (non-parametric) bootstrap

- I'll give you 10 minutes to work on this. I'll be around to help.
  - One way to do this on next slide. But don't look at it!

Class exercise solution (one of many)

```
bootstrapdata <- read_csv("bootstrapdata.csv")
quantile(bootstrapdata$x, 0.75)
```

```
##      75%
## 2.690306
```

```
bootvec <- c()
for (b in 1:1000){
  # random sample
  sample <- bootstrapdata$x[sample(1:nrow(bootstrapdata), nrow(bootstrapdata), replace = TRUE)]
  # add quantile to container
  bootvec <- c(bootvec, quantile(sample, 0.75))
}
# 2.5th percentile
quantile(bootvec, 0.025)
```

```
##     2.5%
## 2.488951
```

```
# 97.5th percentile
quantile(bootvec, 0.975)
```

```
##    97.5%
## 2.998896
```