Lecture Note for Applied Econometrics

Yuta Toyama

Last updated: 2019/06/03

Contents

Chapter 1

Preface

Welcome to Applied Econometrics using R!

1.1 About this

This lecture note is maintained by Yuta Toyama.

1.2 Update: April 23, 2019

- Upload chapter 6 (review of statistics)

1.3 Update: April 16, 2019

- Update chapter 3 (basic programming)
- Upload chapter 4 (Data frame)
- Upload chapter 5 (Exercise 1)

1.4 Acknowledgement (as of April 16, 2019)

- Chapter 2 through 4 are largely based on Applied Statistics with `R`. https://daviddalpiaz.github.io/appliedstats/
- Chapter 6 is based on "Introduction to Econometrics with R". https://www.econometrics-with-r.org/index.html

Chapter 2

Introduction to the course

2.1 What is econometrics?

1. Estimating economic relationships
    1. Demand curve $\log(Q_t) = \alpha_0 + \alpha_1 P_t + \epsilon_t$
    2. Production function $Y_{it} = A_{it} K_{it}^{\alpha} L_{it}^{\beta}$
2. Testing economic theory
    - Does adverse selection exists in insurance markets?
    - Are consumers rational?
3. Determine the effect of a given intervention (causal inference)
    - What is the effect of increasing minimum wage on employment?
    - Do mergers increase the output price?
    - Does democracy cause economic growth? (a series of works by Acemoglu, Robinsohn, and their co-authors).
    - Effects of going to private colleges on your future earnings.
    - Note: Some questions may have underlying economic models, others may not.
4. Describe the data (prediction/forecasting)
    - How does the distribution of wage look like?
    - Relationship between electricity consumption and temperature (possibly nonlinear).
    - Related to machine learning (ML).

2.2 Why do we need to learn computation

1. Conduct statistical and empirical analysis using your own data set
    1. Construct the data set
    2. Describe the data
    3. Run regression or estimate an economic object
    4. Make tables and figures that show the results of your analysis.
2. Verify the econometric theory through numerical simulations.
    - Ex. Asymptotic theory considers the case when the sample size is large enough (i.e., $N \to \infty$)
        - Law of large numbers, central limit theorem
        - How well is the asymptotic approximation?
    - **Monte Carlo simulations**

- We will learn both aspects in this course.

2.3 Why do we use R?

- Many alternatives: Stata, Matlab, Python, etc…

1. Free software!!
   - Stata and Matlab are expensive.
   - Though you can use Matlab through the campus license from this April.
2. Good balance between flexibility in programming and easy-to-use for econometric analysis
   - Stata is easy to use for econometric analysis, but hard to write your own program.
   - Matlab is the opposite.
   - You can do everything with R, including data construction, regression analysis, and complicated structural estimation.
3. Many users
   - Popular in engineering.
   - Many packages being developed (especially important for recently popular tools.)

- Note: Python seems also good, though I have not used it before.

Chapter 3

Introduction of `R` and `R studio`

3.1 Getting Started

- You can use R/R studio in the PC room.
- However, I strongly recommend you install R/Studio in your laptop and bring it to the class.
- Install in the following order
  1. R: https://www.r-project.org/
  2. Rstudio: https://www.rstudio.com/
- Now open Rstudio.

3.2 Helps

- The RStudio team has developed a number of "cheatsheets" for working with both `R` and RStudio.
- This particular cheatsheet for Base `R` will summarize many of the concepts in this document.

3.3 Quick tour of Rstudio

- There are four panels
  1. Source: Write your own code here.
  2. Console:
  3. Environment/History:
  4. Files/Plots/Packages/Help:
- In the Source panel,
  - Write your own code.
  - Save your code in `.R` file
  - Click `Run` command to run your entire code.
- In the concole panel,
  - After clicking `Run` in the source panel, your code is evaluated.
  - You can directly type your code here to implement.

3.4 Basic Calculations

To get started, we'll use `R` like a simple calculator.

Addition, Subtraction, Multiplication and Division

Math	R	Result
$3+2$	`3 + 2`	5
$3-2$	`3 - 2`	1
$3 \cdot 2$	`3 * 2`	6
$3/2$	`3 / 2`	1.5

Exponents

Math	R	Result
3^2	`3 ^ 2`	9
$2^{(-3)}$	`2 ^ (-3)`	0.125
$100^{1/2}$	`100 ^ (1 / 2)`	10
$\sqrt{100}$	`sqrt(100)`	10

Mathematical Constants

Math	R	Result
π	`pi`	3.1415927
e	`exp(1)`	2.7182818

Logarithms

- Note that we will use ln and log interchangeably to mean the natural logarithm.
- There is no `ln()` in R, instead it uses `log()` to mean the natural logarithm.

Math	R	Result
$\log(e)$	`log(exp(1))`	1
$\log_{10}(1000)$	`log10(1000)`	3
$\log_2(8)$	`log2(8)`	3
$\log_4(16)$	`log(16, base = 4)`	2

Trigonometry

Math	R	Result
$\sin(\pi/2)$	`sin(pi / 2)`	1
$\cos(0)$	`cos(0)`	1

3.5 Getting Help

- In using R as a calculator, we have seen a number of functions: `sqrt()`, `exp()`, `log()` and `sin()`.
- To get documentation about a function in R, simply put a question mark in front of the function name

and RStudio will display the documentation, for example:

```
?log
?sin
?paste
?lm
```

3.6 Installing Packages

- One of the main strengths of `R` as an open-source project is its package system.
- To install a package, use the `install.packages()` function.
  - Think of this as buying a recipe book from the store, bringing it home, and putting it on your shelf.

```
install.packages("ggplot2")
```

- Once a package is installed, it must be loaded into your current `R` session before being used.
  - Think of this as taking the book off of the shelf and opening it up to read.

```
library(ggplot2)
```

- Once you close `R`, all the packages are closed and put back on the imaginary shelf.
- The next time you open `R`, you do not have to install the package again, but you do have to load any packages you intend to use by invoking `library()`.

Chapter 4

Data and Programming

4.1 Data Types

`R` has a number of basic data *types*.

- Numeric
  - Also known as Double. The default type when dealing with numbers.
  - Examples: `1`, `1.0`, `42.5`
- Logical
  - Two possible values: `TRUE` and `FALSE`
  - You can also use `T` and `F`, but this is *not* recommended.
  - `NA` is also considered logical.
- Character
  - Examples: `"a"`, `"Statistics"`, `"1 plus 2."`

4.2 Data Structures

- `R` also has a number of basic data *structures*.
- A data structure is either
  - homogeneous (all elements are of the same data type)
  - heterogeneous (elements can be of more than one data type).

Dimension	**Homogeneous**	**Heterogeneous**
1	Vector	List
2	Matrix	Data Frame
3+	Array	

4.3 Vectors

4.3.1 Basics of vectors

- Many operations in `R` make heavy use of **vectors**.
  - Vectors in `R` are indexed starting at `1`.
- The most common way to create a vector in `R` is using the `c()` function, which is short for "combine.""

```
c(1, 3, 5, 7, 8, 9)
```

```
## [1] 1 3 5 7 8 9
```

- If we would like to store this vector in a **variable** we can do so with the **assignment** operator `=`.
  - The variable `x` now holds the vector we just created, and we can access the vector by typing `x`.

```
x = c(1, 3, 5, 7, 8, 9)
x
```

```
## [1] 1 3 5 7 8 9
```

```
# The following does the same thing.
x <- c(1, 3, 5, 7, 8, 9)
x
```

```
## [1] 1 3 5 7 8 9
```

- The operator `=` and `<-` work as an assignment operator.
  - You can use both. This does not matter usually.
  - If you are interested in the weird cases where the difference matters, check out The R Inferno.
- In `R` code the line starting with `#` is `comment`, which is ignored when you run the fode.

- A vector based on a sequence of numbers.
- The quickest and easiest way to do this is with the `:` operator, which creates a sequence of integers between two specified integers.

```
(y = 1:100)
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
##  [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
##  [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
##  [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
```

- By putting parentheses around the assignment,
  - `R` both stores the vector in a variable called `y` and
  - automatically outputs `y` to the console.

4.3.2 Useful functions for creating vectors

- Use the `seq()` function for a more general sequence.

```
seq(from = 1.5, to = 4.2, by = 0.1)
```

```
##  [1] 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 3.0 3.1
## [18] 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0 4.1 4.2
```

- Here, the input labels `from`, `to`, and `by` are optional.

```
seq(1.5, 4.2, 0.1)
```

```
##  [1] 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 3.0 3.1
## [18] 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0 4.1 4.2
```

- The `rep()` function repeat a single value a number of times.

```
rep("A", times = 10)
```

```
##  [1] "A" "A" "A" "A" "A" "A" "A" "A" "A" "A"
```

- The `rep()` function can be used to repeat a vector some number of times.

```
rep(x, times = 3)
```

```
##  [1] 1 3 5 7 8 9 1 3 5 7 8 9 1 3 5 7 8 9
```

- We have now seen four different ways to create vectors:
  1. `c()`
  2. `:`
  3. `seq()`
  4. `rep()`
- They are often used together.

```
c(x, rep(seq(1, 9, 2), 3), c(1, 2, 3), 42, 2:4)
```

```
##  [1]  1  3  5  7  8  9  1  3  5  7  9  1  3  5  7  9  1  3  5  7  9  1  2
## [24]  3 42  2  3  4
```

- The length of a vector can be obtained with the `length()` function.

```
length(x)
```

```
## [1] 6
```

```
length(y)
```

```
## [1] 100
```

4.3.3 Subsetting

- Use square brackets, `[]`, to obtain a subset of a vector.
- We see that `x[1]` returns the first element.

```
x
```

```
## [1] 1 3 5 7 8 9
```

```
x[1]
```

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

```
x[3]
```

```
## [1] 5
```

- We can also exclude certain indexes, in this case the second element.

```
x[-2]
```

```
## [1] 1 5 7 8 9
```

- We can subset based on a vector of indices.

```
x[1:3]
```

```
## [1] 1 3 5
```

```
x[c(1,3,4)]
```

```
## [1] 1 5 7
```

- We could instead use a vector of logical values.

```
z = c(TRUE, TRUE, FALSE, TRUE, TRUE, FALSE)
z
```

```
## [1]  TRUE  TRUE FALSE  TRUE  TRUE FALSE
```

```
x[z]
```

```
## [1] 1 3 7 8
```

4.4 Vectorization

- One of the biggest strengths of R is its use of vectorized operations.
  - Frequently the lack of understanding of this concept leads of a belief that R is *slow*.
  - R is not the fastest language, but it has a reputation for being slower than it really is.)
- When a function like `log()` is called on a vector `x`, a vector is returned which has applied the function to each element of the vector `x`.

```
x = 1:10
x + 1
```

```
##  [1]  2  3  4  5  6  7  8  9 10 11
```

```
2 * x
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

```
2 ^ x
```

```
##  [1]    2    4    8   16   32   64  128  256  512 1024
```

```
sqrt(x)
```

```
##  [1] 1.000000 1.414214 1.732051 2.000000 2.236068 2.449490 2.645751
##  [8] 2.828427 3.000000 3.162278
```

```
log(x)
```

```
##  [1] 0.0000000 0.6931472 1.0986123 1.3862944 1.6094379 1.7917595 1.9459101
##  [8] 2.0794415 2.1972246 2.3025851
```

4.5 Logical Operators

Operator	Summary	Example	Result
`x < y`	x less than y	`3 < 42`	TRUE
`x > y`	x greater than y	`3 > 42`	FALSE
`x <= y`	x less than or equal to y	`3 <= 42`	TRUE
`x >= y`	x greater than or equal to y	`3 >= 42`	FALSE
`x == y`	xequal to y	`3 == 42`	FALSE
`x != y`	x not equal to y	`3 != 42`	TRUE
`!x`	not x	`!(3 > 42)`	TRUE
`x | y`	x or y	`(3 > 42) | TRUE`	TRUE
`x & y`	x and y	`(3 < 4) & ( 42 > 13)`	TRUE

- Logical operators are vectorized.

```
x = c(1, 3, 5, 7, 8, 9)
```

```
x > 3
```

```
## [1] FALSE FALSE  TRUE  TRUE  TRUE  TRUE
```

```
x < 3
```

```
## [1]  TRUE FALSE FALSE FALSE FALSE FALSE
```

```
x == 3
```

```
## [1] FALSE  TRUE FALSE FALSE FALSE FALSE
```

```
x != 3
```

```
## [1]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
```

```
x == 3 & x != 3
```

```
## [1] FALSE FALSE FALSE FALSE FALSE FALSE
```

```
x == 3 | x != 3
```

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE
```

- This is extremely useful for subsetting.

```
x[x > 3]
```

```
## [1] 5 7 8 9
```

```
x[x != 3]
```

```
## [1] 1 5 7 8 9
```

4.5.0.1 Short exercise

1. Create the vector $z = (1, 2, 1, 2, 1, 2)$, which has the same length as x.
2. Pick up the elements of x which corresponds to 1 in the vector z.

4.6 Matrices

4.6.1 Basics

- `R` can also be used for **matrix** calculations.
- Matrices have rows and columns containing a single data type.
- Matrices can be created using the `matrix` function.

```
x = 1:9
x
```

```
## [1] 1 2 3 4 5 6 7 8 9
```

```
X = matrix(x, nrow = 3, ncol = 3)
X
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
```

```
## [3,]    3    6    9
```

- We are using two different variables:
  - lower case `x`, which stores a vector and
  - capital `X`, which stores a matrix.
- By default the `matrix` function reorders a vector into columns, but we can also tell `R` to use rows instead.

```
Y = matrix(x, nrow = 3, ncol = 3, byrow = TRUE)
Y
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

- a matrix of a specified dimension where every element is the same, in this case `0`.

```
Z = matrix(0, 2, 4)
Z
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    0    0    0    0
## [2,]    0    0    0    0
```

- Matrices can be subsetted using square brackets, `[]`.
- However, since matrices are two-dimensional, we need to specify both a row and a column when subsetting.
- Here we get the element in the first row and the second column.

```
X
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
```

```
X[1, 2]
```

```
## [1] 4
```

- We could also subset an entire row or column.

```
X[1, ]
```

```
## [1] 1 4 7
```

```
X[, 2]
```

```
## [1] 4 5 6
```

- Matrices can also be created by combining vectors as columns, using `cbind`, or combining vectors as rows, using `rbind`.

```
x = 1:9
rev(x)
```

```
## [1] 9 8 7 6 5 4 3 2 1
```

```
rep(1, 9)
```

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

```
rbind(x, rev(x), rep(1, 9))
```

```
##   [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
## x    1    2    3    4    5    6    7    8    9
##      9    8    7    6    5    4    3    2    1
##      1    1    1    1    1    1    1    1    1
```

- When using `rbind` and `cbind` you can specify "argument" names that will be used as column names.

```
cbind(col_1 = x, col_2 = rev(x), col_3 = rep(1, 9))
```

```
##       col_1 col_2 col_3
##  [1,]     1     9     1
##  [2,]     2     8     1
##  [3,]     3     7     1
##  [4,]     4     6     1
##  [5,]     5     5     1
##  [6,]     6     4     1
##  [7,]     7     3     1
##  [8,]     8     2     1
##  [9,]     9     1     1
```

4.6.2 Matrix calculations

- Perform matrix calculations.

```
x = 1:9
y = 9:1
X = matrix(x, 3, 3)
Y = matrix(y, 3, 3)
X
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
```

```
Y
```

```
##      [,1] [,2] [,3]
## [1,]    9    6    3
## [2,]    8    5    2
## [3,]    7    4    1
```

```
X + Y
```

```
##      [,1] [,2] [,3]
## [1,]   10   10   10
## [2,]   10   10   10
## [3,]   10   10   10
```

```
X - Y
```

```
##      [,1] [,2] [,3]
## [1,]   -8   -2    4
## [2,]   -6    0    6
## [3,]   -4    2    8
```

```
X * Y
```

```
##      [,1] [,2] [,3]
## [1,]    9   24   21
## [2,]   16   25   16
## [3,]   21   24    9
```

```
X / Y
```

```
##           [,1]      [,2]     [,3]
## [1,] 0.1111111 0.6666667 2.333333
## [2,] 0.2500000 1.0000000 4.000000
## [3,] 0.4285714 1.5000000 9.000000
```

- Note that `X * Y` is **not** matrix multiplication.
- It is element by element multiplication. (Same for `X / Y`).
- Matrix multiplication uses `%*%`.
- `t()` which gives the transpose of a matrix

```
X %*% Y
```

```
##      [,1] [,2] [,3]
## [1,]   90   54   18
## [2,]  114   69   24
## [3,]  138   84   30
```

```
t(X)
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

- `solve()` which returns the inverse of a square matrix if it is invertible.

```
Z = matrix(c(9, 2, -3, 2, 4, -2, -3, -2, 16), 3, byrow = TRUE)
Z
```

```
##      [,1] [,2] [,3]
## [1,]    9    2   -3
## [2,]    2    4   -2
## [3,]   -3   -2   16
```

```
solve(Z)
```

```
##             [,1]        [,2]       [,3]
## [1,]  0.12931034 -0.05603448 0.01724138
## [2,] -0.05603448  0.29094828 0.02586207
## [3,]  0.01724138  0.02586207 0.06896552
```

- To verify that `solve(Z)` returns the inverse, we multiply it by `Z`.
  - We would expect this to return the identity matrix.
  - However we see that this is not the case due to some computational issues.
  - However, `R` also has the `all.equal()` function which checks for equality, with some small tolerance which accounts for some computational issues.

```
solve(Z) %*% Z
```

```
##                          [,1]                          [,2]
```

```
## [1,] 1.0000000000000000000000 -0.0000000000000006245005
## [2,] 0.0000000000000008326673  1.0000000000000022204460
## [3,] 0.0000000000000002775558  0.0000000000000000000000
##                              [,3]
## [1,] 0.0000000000000000000000
## [2,] 0.0000000000000005551115
## [3,] 1.0000000000000000000000
```

```
diag(3)
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

```
all.equal(solve(Z) %*% Z, diag(3))
```

```
## [1] TRUE
```

4.6.2.1 Exercise

- Solve the following simultanoues equations using matrix calculation

$$2x_1 + 3x_2 = 105x_1 + x_2 = 20$$

- Hint: You can write this as $Ax = y$ where A is the 2-times-2 matrix, x and y are vectors with the length of 2.

4.6.3 Getting information for matrix

- `R` has a number of matrix specific functions for obtaining dimension and summary information.

```
X = matrix(1:6, 2, 3)
X
```

```
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
```

```
dim(X)
```

```
## [1] 2 3
```

```
rowSums(X)
```

```
## [1]  9 12
```

```
colSums(X)
```

```
## [1]  3  7 11
```

```
rowMeans(X)
```

```
## [1] 3 4
```

```
colMeans(X)
```

```
## [1] 1.5 3.5 5.5
```

- The `diag()` function can be used in a number of ways. We can extract the diagonal of a matrix.

```
diag(Z)
```

```
## [1]  9  4 16
```

- Or create a matrix with specified elements on the diagonal. (And `0` on the off-diagonals.)

```
diag(1:5)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    2    0    0    0
## [3,]    0    0    3    0    0
## [4,]    0    0    0    4    0
## [5,]    0    0    0    0    5
```

- Or, lastly, create a square matrix of a certain dimension with `1` for every element of the diagonal and `0` for the off-diagonals.

```
diag(5)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    1    0    0    0
## [3,]    0    0    1    0    0
## [4,]    0    0    0    1    0
## [5,]    0    0    0    0    1
```

4.7 Lists

- A list is a one-dimensional heterogeneous data structure.
  - It is indexed like a vector with a single integer value,
  - but each element can contain an element of any type.

```
# creation
list(42, "Hello", TRUE)
```

```
## [[1]]
## [1] 42
##
## [[2]]
## [1] "Hello"
##
## [[3]]
## [1] TRUE
```

```
ex_list = list(
  a = c(1, 2, 3, 4),
  b = TRUE,
  c = "Hello!",
  d = function(arg = 42) {print("Hello World!")},
  e = diag(5)
)
```

- Lists can be subset using two syntaxes,
  1. the `$` operator, and
  2. square brackets `[]`.

```
# subsetting
ex_list$e
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    1    0    0    0
## [3,]    0    0    1    0    0
## [4,]    0    0    0    1    0
## [5,]    0    0    0    0    1
```

```
ex_list[1:2]
```

```
## $a
## [1] 1 2 3 4
##
## $b
## [1] TRUE
```

```
ex_list[1]
```

```
## $a
## [1] 1 2 3 4
```

```
ex_list[c("e", "a")]
```

```
## $e
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    1    0    0    0
## [3,]    0    0    1    0    0
## [4,]    0    0    0    1    0
## [5,]    0    0    0    0    1
##
## $a
## [1] 1 2 3 4
```

```
ex_list["e"]
```

```
## $e
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    1    0    0    0
## [3,]    0    0    1    0    0
## [4,]    0    0    0    1    0
## [5,]    0    0    0    0    1
```

```
ex_list$d
```

```
## function(arg = 42) {print("Hello World!")}
```

4.8 Data Frames

- We will talk about Dataframe in the next chapter.

4.9 Programming Basics -Control flow-

4.9.1 if/else

- The if/else syntax is:

```
if (...) {
  some R code
} else {
  more R code
}
```

- Example: To see whether x is large than y.

```
x = 1
y = 3
if (x > y) {
  z = x * y
  print("x is larger than y")
} else {
  z = x + 5 * y
  print("x is less than or equal to y")
}
```

```
## [1] "x is less than or equal to y"
```

```
z
```

```
## [1] 16
```

- `R` also has a special function `ifelse()`
  - It returns one of two specified values based on a conditional statement.

```
ifelse(4 > 3, 1, 0)
```

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

- The real power of `ifelse()` comes from its ability to be applied to vectors.

```
fib = c(1, 1, 2, 3, 5, 8, 13, 21)
ifelse(fib > 6, "Foo", "Bar")
```

```
## [1] "Bar" "Bar" "Bar" "Bar" "Bar" "Foo" "Foo" "Foo"
```

4.10 `for` loop

- A `for` loop repeats the same procedure for the specified number of times

```
x = 11:15
for (i in 1:5) {
  x[i] = x[i] * 2
}

x
```

```
## [1] 22 24 26 28 30
```

- Note that this `for` loop is very normal in many programming languages.
- In `R` we would not use a loop, instead we would simply use a vectorized operation.

  - `for` loop in `R` is known to be very slow.

```
x = 11:15
x = x * 2
x
```

```
## [1] 22 24 26 28 30
```

4.11 Functions

- To use a function,
  - you simply type its name,
  - followed by an open parenthesis,
  - then specify values of its arguments,
  - then finish with a closing parenthesis.
- An **argument** is a variable which is used in the body of the function.

```
# The following is just a demonstration, not the real function in R.
function_name(arg1 = 10, arg2 = 20)
```

- We can also write our own functions in `R`.
- Example: "standardize" variables

$$\frac{x - \bar{x}}{s}$$

- When writing a function, there are three thing you must do.
  1. Give the function a name. Preferably something that is short, but descriptive.
  2. Specify the arguments using `function()`
  3. Write the body of the function within curly braces, `{}`.

```
standardize = function(x) {
  m = mean(x)
  std = sd(x)
  result = (x - m) / std
  return(result)
}
```

- Here the name of the function is `standardize`,
- The function has a single argument `x` which is used in the body of function.
- Note that the output of the final line of the body is what is returned by the function.
- Let's test our function
- Take a random sample of size `n = 10` from a normal distribution with a mean of `2` and a standard deviation of `5`.

```
test_sample = rnorm(n = 10, mean = 2, sd = 5)
test_sample
```

```
##  [1] -2.53115782  5.62885917 -1.19781244  0.02842195 -4.35923091
##  [6]  6.95286927  8.44694256 10.14671116  7.56555571  7.33916341
```

```
standardize(x = test_sample)
```

```
##  [1] -1.2071441  0.3482042 -0.9530004 -0.7192728 -1.5555858  0.6005685
##  [7]  0.8853478  1.2093339  0.7173502  0.6741985
```

- The same function can be written more simply.

```
standardize = function(x) {
  (x - mean(x)) / sd(x)
}
```

- When specifying arguments, you can provide default arguments.

```
power_of_num = function(num, power = 2) {
  num ^ power
}
```

- Let's look at a number of ways that we could run this function to perform the operation `10^2` resulting in `100`.

```
power_of_num(10)
```

```
## [1] 100
```

```
power_of_num(10, 2)
```

```
## [1] 100
```

```
power_of_num(num = 10, power = 2)
```

```
## [1] 100
```

```
power_of_num(power = 2, num = 10)
```

```
## [1] 100
```

- Note that without using the argument names, the order matters. The following code will not evaluate to the same output as the previous example.

```
power_of_num(2, 10)
```

```
## [1] 1024
```

- Also, the following line of code would produce an error since arguments without a default value must be specified.

```
power_of_num(power = 5)
```

- To further illustrate a function with a default argument, we will write a function that calculates sample variance two ways.
- By default, the function will calculate the unbiased estimate of σ^2, which we will call s^2.

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x - \bar{x})^2$$

- It will also have the ability to return the biased estimate (based on maximum likelihood) which we will call $\hat{\sigma}^2$.

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x - \bar{x})^2$$

```
get_var = function(x, unbiased = TRUE) {

  if (unbiased == TRUE){
    n = length(x) - 1
```

```
  } else if (unbiased == FALSE){
    n = length(x)
   }

  (1 / n) * sum((x - mean(x)) ^ 2)
}
```

```
get_var(test_sample)
```

```
## [1] 27.52497
```

```
get_var(test_sample, unbiased = TRUE)
```

```
## [1] 27.52497
```

```
var(test_sample)
```

```
## [1] 27.52497
```

- We see the function is working as expected, and when returning the unbiased estimate it matches `R`'s built in function `var()`. Finally, let's examine the biased estimate of σ^2.

```
get_var(test_sample, unbiased = FALSE)
```

```
## [1] 24.77247
```

Chapter 5

Data frame

5.1 Introduction

- A **data frame** is the most common way that we store and interact with data in this course.

```
example_data = data.frame(x = c(1, 3, 5, 7, 9, 1, 3, 5, 7, 9),
                          y = c(rep("Hello", 9), "Goodbye"),
                          z = rep(c(TRUE, FALSE), 5))
```

- A data frame is a **list** of vectors.
  - Each vector must contain the same data type
  - The difference vectors can store different data types.

```
example_data
```

```
##    x       y     z
## 1  1   Hello  TRUE
## 2  3   Hello FALSE
## 3  5   Hello  TRUE
## 4  7   Hello FALSE
## 5  9   Hello  TRUE
## 6  1   Hello FALSE
## 7  3   Hello  TRUE
## 8  5   Hello FALSE
## 9  7   Hello  TRUE
## 10 9 Goodbye FALSE
```

- `write.csv` save (or export) the dataframe in `.csv` format.

5.2 Load csv file

- We can also import data from various file types in into `R`, as well as use data stored in packages.
- Read `csv` file into R.
  - `read.csv()` function as default
  - `read_csv()` function from the `readr` package. This is faster for larger data.

```
# install.packages("readr")
#library(readr)
```

```
#example_data_from_csv = read_csv("example-data.csv")
example_data_from_csv = read.csv("example-data.csv")
```

- Note: This particular line of code assumes that the file `example_data.csv` exists in your current working directory.
- The current working directory is the folder that you are working with. To see this, you type

```
getwd()
```

```
## [1] "C:/Users/Yuta/Dropbox/Teaching/2019S_Applied_Econometrics_JPN_ENG/Material_Github"
```

- If you want to set the working directory, use `setwd()` function

```
setwd(dir = "directory path" )
```

5.3 Examine dataframe

- Inside the `ggplot2` package is a dataset called `mpg`. By loading the package using the `library()` function, we can now access `mpg`.

```
library(ggplot2)
```

- Three things we would generally like to do with data:
  - Look at the raw data.
  - Understand the data. (Where did it come from? What are the variables? Etc.)
  - Visualize the data.
- To look at the data, we have two useful commands: `head()` and `str()`

```
head(mpg, n = 10)
```

```
## # A tibble: 10 x 11
##    manufacturer model displ  year   cyl trans drv     cty   hwy fl    cla~
##    <chr>        <chr> <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <ch>
##  1 audi         a4      1.8  1999     4 auto~ f        18    29 p     com~
##  2 audi         a4      1.8  1999     4 manu~ f        21    29 p     com~
##  3 audi         a4      2    2008     4 manu~ f        20    31 p     com~
##  4 audi         a4      2    2008     4 auto~ f        21    30 p     com~
##  5 audi         a4      2.8  1999     6 auto~ f        16    26 p     com~
##  6 audi         a4      2.8  1999     6 manu~ f        18    26 p     com~
##  7 audi         a4      3.1  2008     6 auto~ f        18    27 p     com~
##  8 audi         a4 q~   1.8  1999     4 manu~ 4        18    26 p     com~
##  9 audi         a4 q~   1.8  1999     4 auto~ 4        16    25 p     com~
## 10 audi         a4 q~   2    2008     4 manu~ 4        20    28 p     com~
```

- The function `str()` will display the "structure" of the data frame.
  - It will display the number of **observations** and **variables**, list the variables, give the type of each variable, and show some elements of each variable.
  - This information can also be found in the "Environment" window in RStudio.

```
str(mpg)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    234 obs. of  11 variables:
##  $ manufacturer: chr  "audi" "audi" "audi" "audi" ...
##  $ model       : chr  "a4" "a4" "a4" "a4" ...
##  $ displ       : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int  1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
```

```
##  $ cyl          : int  4 4 4 4 6 6 6 4 4 4 ...
##  $ trans        : chr  "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv          : chr  "f" "f" "f" "f" ...
##  $ cty          : int  18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy          : int  29 29 31 30 26 26 27 26 25 28 ...
##  $ fl           : chr  "p" "p" "p" "p" ...
##  $ class        : chr  "compact" "compact" "compact" "compact" ...
```

- `names()` function to obtain names of the variables in the dataset

```
names(mpg)
```

```
##  [1] "manufacturer" "model"        "displ"        "year"
##  [5] "cyl"          "trans"        "drv"          "cty"
##  [9] "hwy"          "fl"           "class"
```

- To access one of the variables **as a vector**, we use the `$` operator.

```
mpg$year
```

```
##   [1] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 2008 1999 1999 2008
##  [15] 2008 1999 2008 2008 2008 2008 2008 1999 2008 1999 1999 2008 2008 2008
##  [29] 2008 2008 1999 1999 1999 2008 1999 2008 2008 1999 1999 1999 1999 2008
##  [43] 2008 2008 1999 1999 2008 2008 2008 2008 1999 1999 2008 2008 2008 1999
##  [57] 1999 1999 2008 2008 2008 1999 2008 1999 2008 2008 2008 2008 2008 2008
##  [71] 1999 1999 2008 1999 1999 1999 2008 1999 1999 1999 2008 2008 1999 1999
##  [85] 1999 1999 1999 2008 1999 2008 1999 1999 2008 2008 1999 1999 2008 2008
##  [99] 2008 1999 1999 1999 1999 1999 2008 2008 2008 2008 1999 1999 2008 2008
## [113] 1999 1999 2008 1999 1999 2008 2008 2008 2008 2008 2008 2008 1999 1999
## [127] 2008 2008 2008 2008 1999 2008 2008 1999 1999 1999 2008 1999 2008 2008
## [141] 1999 1999 1999 2008 2008 2008 2008 1999 1999 2008 1999 1999 2008 2008
## [155] 1999 1999 1999 2008 2008 1999 1999 2008 2008 2008 2008 1999 1999 1999
## [169] 1999 2008 2008 2008 2008 1999 1999 1999 1999 2008 2008 1999 1999 2008
## [183] 2008 1999 1999 2008 1999 1999 2008 2008 1999 1999 2008 1999 1999 1999
## [197] 2008 2008 1999 2008 1999 1999 2008 1999 1999 2008 2008 1999 1999 2008
## [211] 2008 1999 1999 1999 1999 2008 2008 2008 2008 1999 1999 1999 1999 1999
## [225] 1999 2008 2008 1999 1999 2008 2008 1999 1999 2008
```

```
mpg$hwy
```

```
##   [1] 29 29 31 30 26 26 27 26 25 28 27 25 25 25 25 24 25 23 20 15 20 17 17
##  [24] 26 23 26 25 24 19 14 15 17 27 30 26 29 26 24 24 22 22 24 24 17 22 21
##  [47] 23 23 19 18 17 17 19 19 12 17 15 17 17 12 17 16 18 15 16 12 17 17 16
##  [70] 12 15 16 17 15 17 17 18 17 19 17 19 19 17 17 17 16 16 17 15 17 26 25
##  [93] 26 24 21 22 23 22 20 33 32 32 29 32 34 36 36 29 26 27 30 31 26 26 28
## [116] 26 29 28 27 24 24 24 22 19 20 17 12 19 18 14 15 18 18 15 17 16 18 17
## [139] 19 19 17 29 27 31 32 27 26 26 25 25 17 17 20 18 26 26 27 28 25 25 24
## [162] 27 25 26 23 26 26 26 26 25 27 25 27 20 20 19 17 20 17 29 27 31 31 26
## [185] 26 28 27 29 31 31 26 26 27 30 33 35 37 35 15 18 20 20 22 17 19 18 20
## [208] 29 26 29 29 24 44 29 26 29 29 29 29 23 24 44 41 29 26 28 29 29 29 28
## [231] 29 26 26 26
```

- We can use the `dim()`, `nrow()` and `ncol()` functions to obtain information about the dimension of the data frame.

```
dim(mpg)
```

```
## [1] 234  11
```

```
nrow(mpg)
```

```
## [1] 234
```

```
ncol(mpg)
```

```
## [1] 11
```

5.4 Subsetting data

- Subsetting data frames can work much like subsetting matrices using square brackets, `[,]`.
- Here, we find fuel efficient vehicles earning over 35 miles per gallon and only display `manufacturer`, `model` and `year`.

```
mpg[mpg$hwy > 35, c("manufacturer", "model", "year")]
```

```
## # A tibble: 6 x 3
##   manufacturer model       year
##   <chr>        <chr>      <int>
## 1 honda        civic       2008
## 2 honda        civic       2008
## 3 toyota       corolla     2008
## 4 volkswagen   jetta       1999
## 5 volkswagen   new beetle  1999
## 6 volkswagen   new beetle  1999
```

- An alternative would be to use the `subset()` function, which has a much more readable syntax.

```
subset(mpg, subset = hwy > 35, select = c("manufacturer", "model", "year"))
```

- Lastly, we could use the `filter` and `select` functions from the `dplyr` package which introduces the `%>%` operator from the `magrittr` package.

```
library(dplyr)
mpg %>%
filter(hwy > 35) %>%
select(manufacturer, model, year)
```

- I will give you an assignment about `dplyr` package in the `DataCamp` as a makeup lecture.

Chapter 6

Exercise 1

- Due date: April 22th (Monday) 11pm

Rules for Problem Sets

- If you are enrolled in Japanese class (i.e., Wednesday 2nd), you can use both Japanese and English to write your answer.
- Submit your solution through `CourseN@vi`.
- Submit both your answer and R script.
- Using `Rmarkdown` would be appreciated, though not mandatory.
  - `Rmarkdown` introduction in Japanese: https://kazutan.github.io/kazutanR/Rmd_intro.html
  - `Rmarkdown` introduction in English: https://rmarkdown.rstudio.com/articles_intro.html
- I might cover `Rmarkdown` in the course later.

6.1 Update (as of 10am, April 18th)

- Please calculate the standard deviation, not the variance in your simulation. The parameter you set when drawing the random number is the mean μ and the standard deviation σ in normal distribution.
- Use `seq` function to create the sequence of the sample sizes that you use in the simulation.

6.2 Question: Examine the law of large numbers through numerical simulations

Consider the random sample of $\{x_i\}_{i=1}^N$ drawn from the random variable X. The law of large numbers implies that

$$\frac{1}{N}\sum_{i=1}^{N} x_i \xrightarrow{p} E[X]$$

In other words, the sample mean converges to the population mean in probability as the sample size goes to infinity (i.e., $N \to \infty$).

Similary, the sample variance also converges to the population variance in probability

$$\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2 \xrightarrow{p} V[X]$$

(Note on 4/18) This implies that the sample standard deviation also converges to the population standard deviation

(This is an application of the law of large numbers, though it is a bit involved to prove this.)

The goal of this problem set is to demonstrate these two properties through numerical simulations. Here is what we are going to do:

1. For a certain sample size N, draw N random numbers from the normal distribution with known mean and standard deviation.
2. Calculate the sample mean and the sample variance for the "data" you draw.
3. Repeat this for many different sample sizes.
4. Examine to see whether the sample mean and **standard deviation** are getting closer to the true value, which you set when you draw the random numbers, as the sample size gets larger.

6.2.1 How to implement

I explain how to implement this in `R` step by step below.

1. Prepare a function like this
    1. There are two inputs: (1) a vector that contains the data $\{x_i\}_{i=1}^N$ and (2) the indicator of whether you calculate the mean or the standard deviation.

```
fun_something = function(firstinput, secondinput){

  # Two inputs: firstinput, secondinput
  # One output: output

  # Do something.

  return(output)

}
```

    2. Use if/else sentence. Example:

```
# "secondinput" is the name of the input variable in your function
if ( secondinput == "mean"){
  # calculate mean of the data (firstinput)

} else if ( secondinput == "sd"){
  # Calculate standard deviation of the data (firstinput)
}
```

    3. Use `return` function to define the output of the function.
2. Construct a vector that contains the sample size you want to use in your simulation. For example:

```
samplesize_vec = seq(from = 100, to = 100000, by = 100)
```

Here, let's try 100 different sample sizes that ranges from 100 to 100000.

3. Prepare two vectors that contain the result in the `forloop` below. Since we are trying 100 different sample sizes, let's create a vector with the length of 100.

```
# Hint:
# numeric(k) returns a zero vector with the length of k
# length( vector) returns the length of `vector`

# result_mean = ....
# result_sd = ....
```

4. To create the random draw from the normal distribution, use below

```
# You can choose the mean and the standard deviation as you like.
rnorm(n = 100, mean = 2, sd = 5)
```

4. Use `forloop` to calculate both mean and the standard deviation for each sample size. For example:

```
for (i in 1:length(samplesize_vec)){

  # Draw the random number

  # Calculate the mean using the function you construct.

  # Calculate the standard deviation using the function you construct.

}
```

5. Plot the result with `ggplot2`.
    1. Install the package if you have not done it yet.
    2. Load `ggplot2` by `library(ggplot2)`
    3. Use `qplot` command to make a figure

```
# Create plot and save it as the variable `plot1`
plot1 <- qplot(x = samplesize_vec, y = yourresult, geom = "line")

# print "plot1"
print(plot1)

# save the plot as PNG file
ggsave(file = "filename.png", plot = plot1)
```

6.2.2 What to submit

Your answer should include

1. The true value of mean and variance you choose in your simulation.
2. The plot that describes the relationship between the sample mean (variance) and the sample size.
3. Explain what the plots from your simulation indicate.

Chapter 7

A Review of Statistics

Acknowledgement: This chapter is largely based on chapter 3 of "Introduction to Econometrics with R". https://www.econometrics-with-r.org/index.html

The goal of this chapter is

1. Review of important concepts in statistics
    1. Estimation
    2. Hypothesis testing
2. Review of tools from probability theory
    1. Law of large numbers
    2. Central limit theorem

7.1 Estimation

- Estimator: A mapping from the sample data drawn from an unknown population to a certain feature in the population
- Example: Consider hourly earnings of college graduates Y .
- You want to estimate the mean of Y, defined as $E[Y] = \mu_y$
- Draw a random sample of n i.i.d. (identically and independently distributed) observations $Y_1, Y_2, \ldots, Y_N$
- How to estimate $E[Y]$ from the data?
- Idea 1: Sample mean

$$\bar{Y} = \frac{1}{n} \sum_{i=1}^{n} Y_i,$$

- Idea 2: Pick the first observation of the sample.
- Question: How can we say which is better?

7.1.1 Properties of the estimator

Consider the estimator $\hat{\mu}_N$ for the unknown parameter μ.

1. Unbiasdeness: The expectation of the estimator is the same as the true parameter in the population.

$$E[\hat{\mu}_N] = \mu$$

2. Consistency: The estimator converges to the true parameter in probability.

$$\forall \epsilon > 0, \lim_{N \to \infty} Prob(|\hat{\mu}_N - \mu| < \epsilon) = 1$$

   - Intuition: As the sample size gets larger, the estimator and the true parameter is close with probability one.
   - Note: a bit different from the usual convergence of the sequence.

7.1.2 Sample mean $\bar{Y}$ is unbiased and consistent

- Showing these two properties using mathmaetics is straightforward:
  - Unbiasedness: Take expectation.
  - Consistency: Law of large numbers.
- Let's examine these two properties using R.
- Step 1: Prepare a population. Here, I prepare income and age data from PUMS 5% sample of U.S. Census 2000.
  - PUMS: Public Use Microdata Sample
  - Download the example data here as a .csv file. Put this file in the same folder as your R script file.

```
# Use "readr" package
library(readr)
```

```
## Warning:     'readr'     3.5.3   R
```

```
pums2000 <- read_csv("data_pums_2000.csv")
```

```
## Parsed with column specification:
## cols(
##   AGE = col_double(),
##   INCTOT = col_double()
## )
```

- We treat this dataset as **population**.

```
pop <- as.vector(pums2000$INCTOT)
```

- *Population* mean and standard deviation

```
pop_mean = mean(pop)
pop_sd   = sd(pop)

# Average income in population
pop_mean
```

```
## [1] 30165.47
```

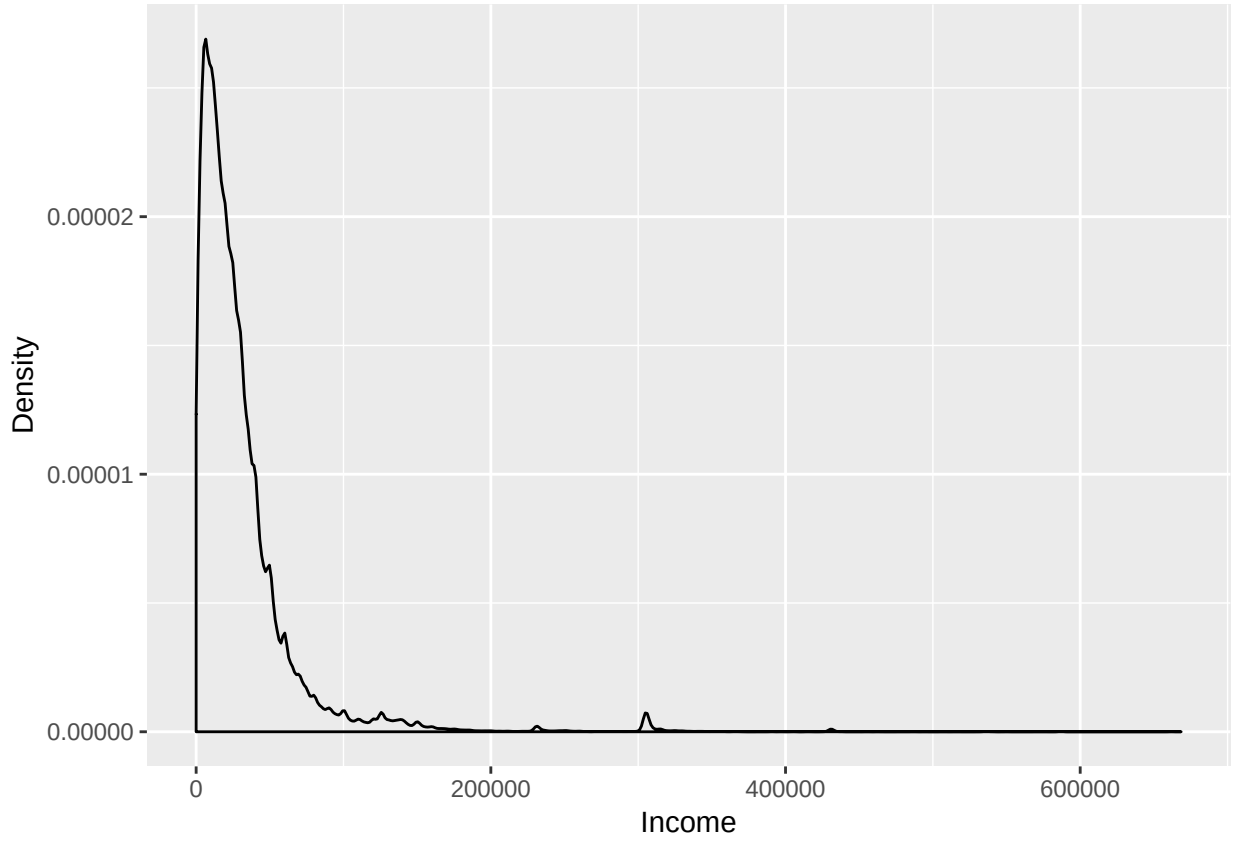
```
# Standard deviation of income in population
pop_sd
```

```
## [1] 38306.17
```

```
# income distribution in population
# Note that the unit is in USD.
library("ggplot2")
```

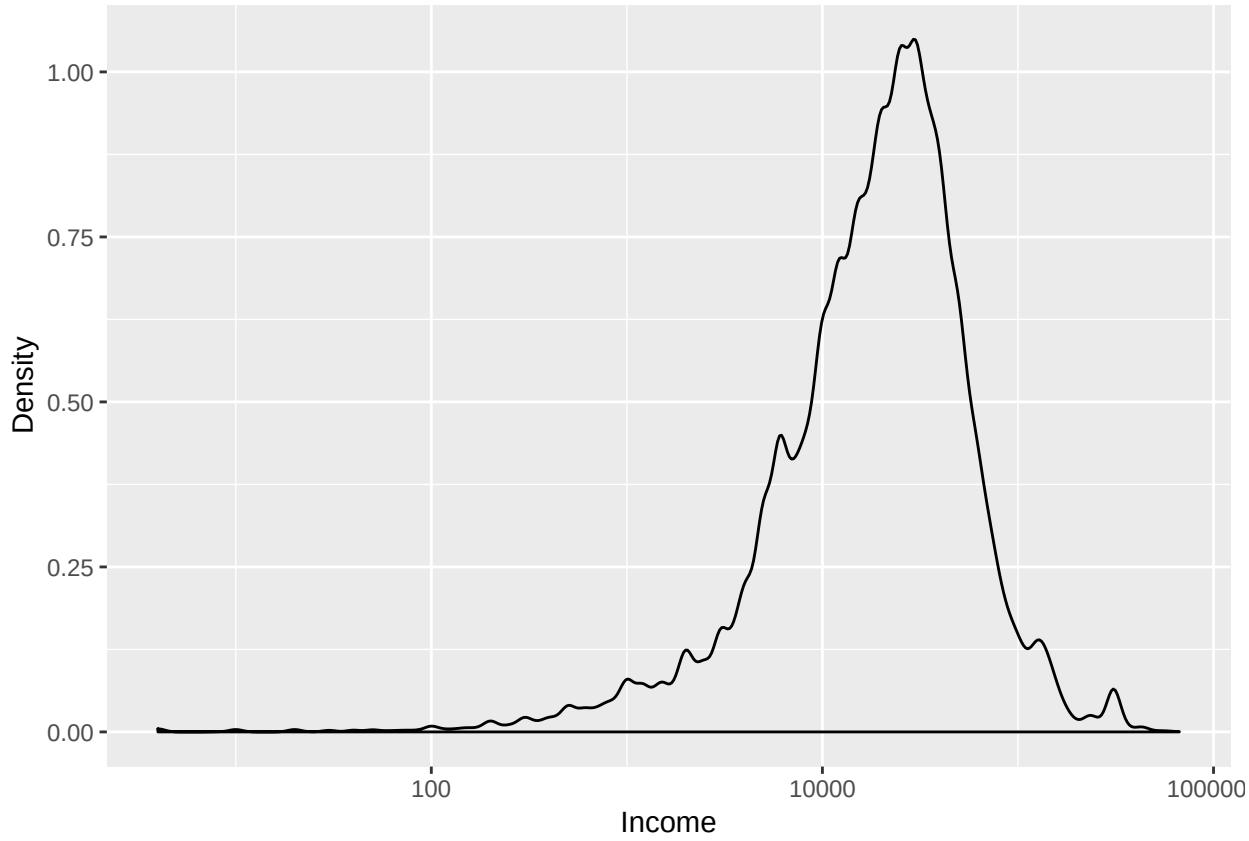
```
## Warning:     'ggplot2'     3.5.3   R
```

```
qplot(pop, geom = "density",
      xlab = "Income",
      ylab = "Density")
```



- The distribution has a long tail.
- Let's plot the distribution in *log* scale

```
# `log` option specifies which axis is represented in log scale.
qplot(pop, geom = "density",
      xlab = "Income",
      ylab = "Density",
      log = "x")
```



- Let's investigate how close the sample mean constucted from the random sample is to the true population mean.
- Step 1: Draw random sample*s* from this population and calculate $\bar{Y}$ for each sample.
  - Set the sample size N.
- Step 2: Repeat 2000 times. You now have 2000 sample means.

```
# Set the seed for the random number. This is needed to maintaine the reproducibility of the results.
set.seed(123)

# draw random sample of 100 observations from the variable pop
test <- sample(x = pop, size = 100)

# Use loop to repeat 2000 times.
Nsamples = 2000
result1 <- numeric(Nsamples)

for (i in 1:Nsamples ){

  test <- sample(x = pop, size = 100)
  result1[i] <- mean(test)

}

# Simple approach
result1 <- replicate(expr = mean(sample(x = pop, size = 10)), n = Nsamples)
result2 <- replicate(expr = mean(sample(x = pop, size = 100)), n = Nsamples)
result3 <- replicate(expr = mean(sample(x = pop, size = 500)), n = Nsamples)
```

```
# Create dataframe

result_data <- data.frame(  Ybar10 = result1,
                            Ybar100 = result2,
                            Ybar500 = result3)
```

- Step 3: See the distribution of those 2000 sample means.

```
# Use reshape library
# install.packages("reshape")
library("reshape")
```

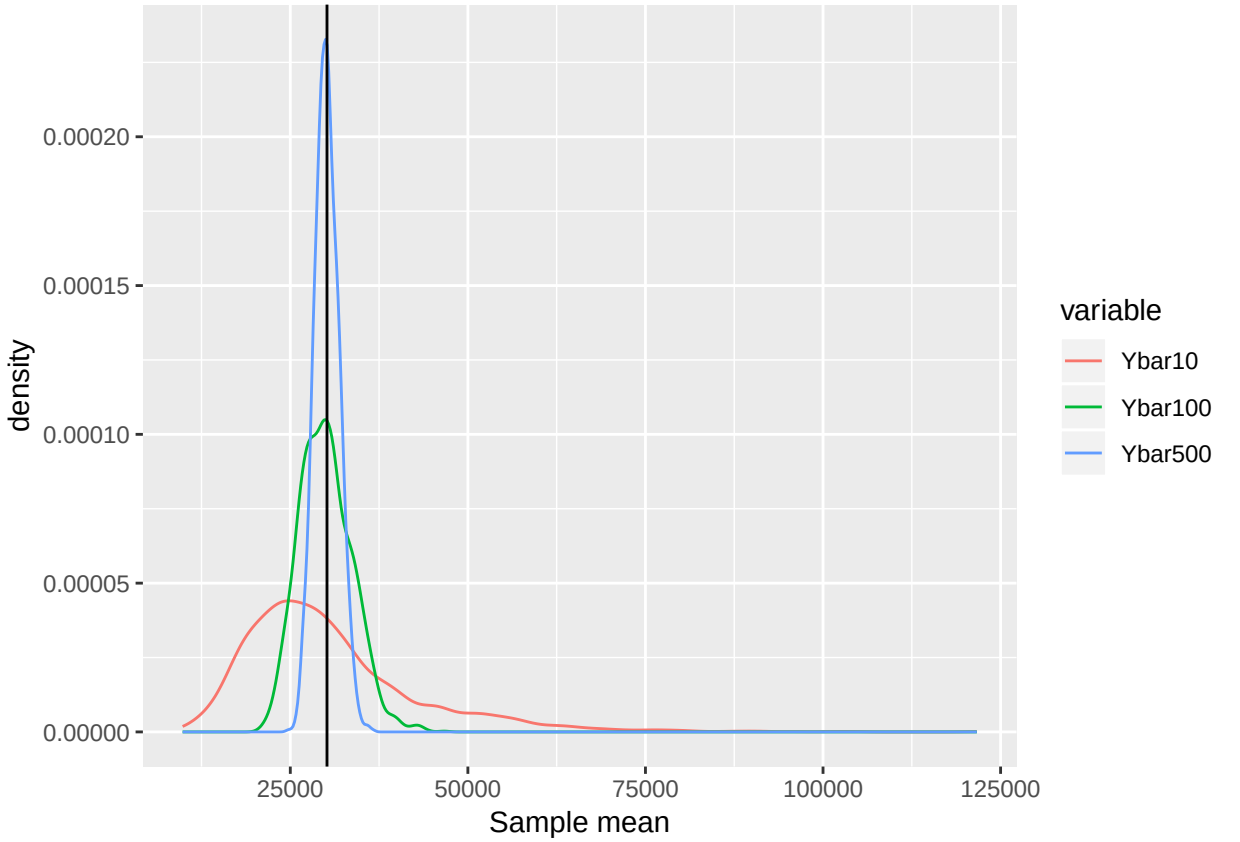
```
## Warning:     'reshape'     3.5.3   R
```

```
# Use "melt" to change the format of result_data
data_for_plot <- melt(data = result_data, variable.name = "Variable" )
```

```
## Using  as id variables
```

```
# Use "ggplot2" to create the figure.
# The variable `fig` contains the information about the figure
fig <-
  ggplot(data = data_for_plot) +
  xlab("Sample mean") +
  geom_line(aes(x = value, colour = variable ),   stat = "density" ) +
  geom_vline(xintercept=pop_mean ,colour="black")

# Display the figure
plot(fig)
```



- Observation 1: Regardless of the sample size, the average of the sample means is close to the population mean. **Unbiasdeness**
- Observation 2: As the sample size gets larger, the distribution is concentrated around the population mean. **Consistency (law of large numbers)**

7.2 Hypothesis Testing

7.2.1 Central limit theorem

- Cental limit theorem: Consider the i.i.d. sample of $Y_1, \cdots, Y_N$ drawn from the random variable Y with mean μ and variance σ^2. The following Z converges in distribution to the normal distribution.

$$Z = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} \frac{Y_i - \mu}{\sigma} \xrightarrow{d} N(0,1)$$

  In other words,

$$\lim_{N \to \infty} P\left(Z \leq z\right) = \Phi(z)$$

- The central limit theorem implies that if N is large **enough**, we can **approximate** the distribution of $\bar{Y}$ by the standard normal distribution with mean μ and variance σ^2/N **regardless of the underlying distribution of Y.**
- Let's examine this property through simulation!!
- Use the same example as before. Remember that the underlying income distribution is clearly NOT normal.

  - Population mean $\mu = 30165.4673315$ and standard deviation $\sigma = 38306.1712336$. Use these numbers.

```
# Set the seed for the random number
set.seed(124)

# define function for simulation
f_simu_CLT = function(Nsamples, samplesize, pop, pop_mean, pop_sd ){

  output = numeric(Nsamples)
  for (i in 1:Nsamples ){
    test <- sample(x = pop, size = samplesize)
    output[i] <- ( mean(test) - pop_mean ) / (pop_sd / sqrt(samplesize))
  }

  return(output)

}

# Comment: You can do better without using forloop. Let me know if you come with a good idea.

# Run simulation
Nsamples = 2000
result_CLT1 <- f_simu_CLT(Nsamples, 10, pop, pop_mean, pop_sd )
result_CLT2 <- f_simu_CLT(Nsamples, 100, pop, pop_mean, pop_sd )
result_CLT3 <- f_simu_CLT(Nsamples, 1000, pop, pop_mean, pop_sd )

# Random draw from standard normal distribution as comparison
result_stdnorm = rnorm(Nsamples)

# Create dataframe
result_CLT_data <- data.frame(  Ybar_standardized_10 = result_CLT1,
                            Ybar_standardized_100 = result_CLT2,
                            Ybar_standardized_1000 = result_CLT3,
                            Standard_Normal = result_stdnorm)

# Note: If you wanna quicky plot the density, type `plot(density(result1))`.
```

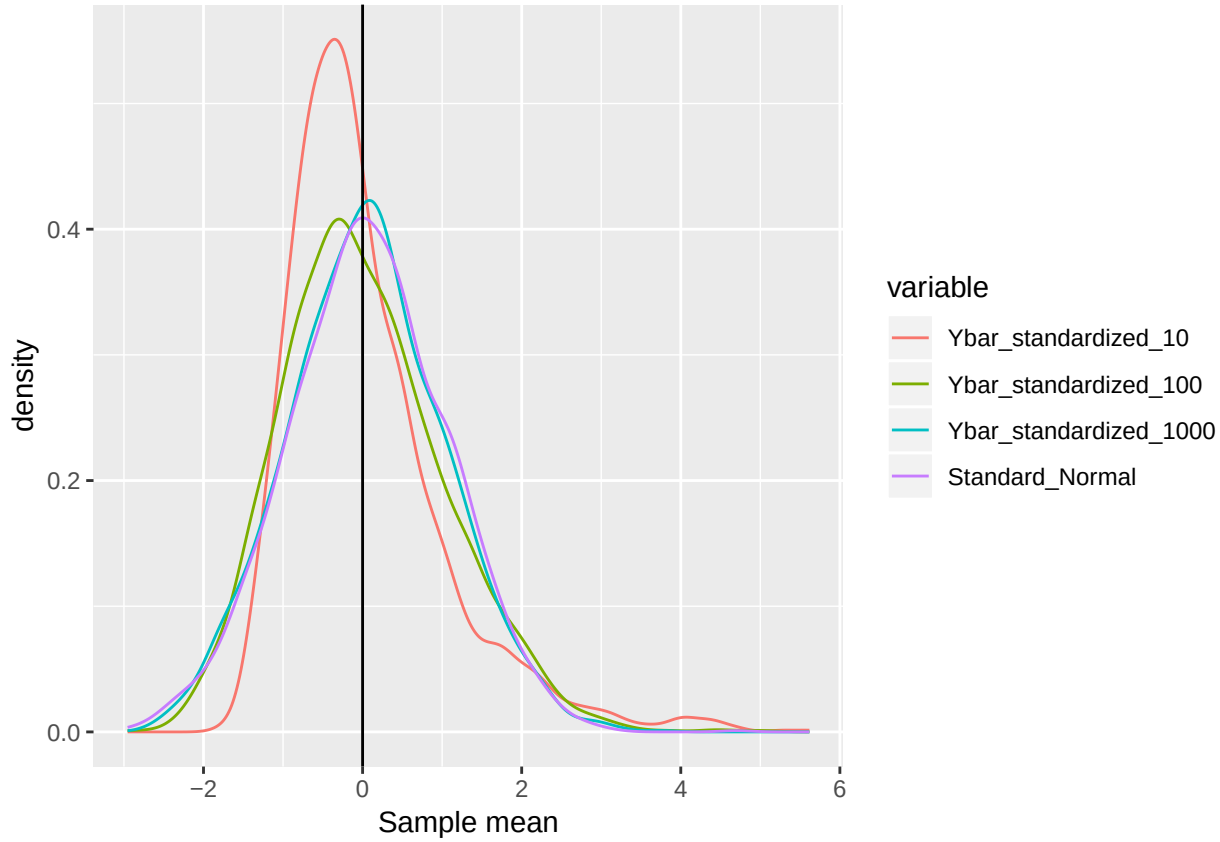
- Now take a look at the distribution.

```
# Use "melt" to change the format of result_data
data_for_plot <- melt(data = result_CLT_data, variable.name = "Variable" )
```

```
## Using  as id variables
```

```
# Use "ggplot2" to create the figure.
fig <-
  ggplot(data = data_for_plot) +
  xlab("Sample mean") +
  geom_line(aes(x = value, colour = variable ),   stat = "density" ) +
  geom_vline(xintercept=0 ,colour="black")

plot(fig)
```



- As the sample size grows, the distribution of Z converges to the standard normal distribution.

7.2.2 Hypothesis testing

To be added.

Chapter 8

Linear Regression 1: Theory

8.1 Regression framework

- Let Y_i be the dependent variable and X_{ik} be k-th explanatory variable.
  - We have K explantory variables (along with constant term)
  - i is an index for observations. $i = 1, \cdots, N$.
  - Data (sample): $\{Y_i, X_{i1}, \ldots, X_{iK}\}_{i=1}^N$
- **Linear regression model** is defined as

$$Y_i = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_K X_{Ki} + \epsilon_i$$

  - ϵ_i: error term (unobserved)
  - β: coefficients
- **Assumptions for Ordinaly Least Squares (OLS) estimation**
  1. Random sample: $\{Y_i, X_{i1}, \ldots, X_{iK}\}$ is i.i.d. drawn sample
     - i.i.d.: identically and independently distributed
  2. ϵ_i has zero conditional mean
$$E[\epsilon_i | X_{i1}, \ldots, X_{iK}] = 0$$
  3. Large outliers are unlikely: The random variable Y_i and X_{ik} have finite fourth moments.
  4. No perfect multicollinearity: There is no linear relationship betwen explanatory variables.
- OLS estimators are the minimizers of the sum of squared residuals:

$$\min_{\beta_0, \cdots, \beta_K} \frac{1}{N} \sum_{i=1}^{N} (Y_i - (\beta_0 + \beta_1 X_{i1} + \cdots + \beta_K X_{iK}))^2$$

- Using matrix notation, we have the following analytical formula for the OLS estimator

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

  where

$$\underbrace{X}_{N \times (K+1)} = \begin{pmatrix} 1 & X_{11} & \cdots & X_{1K} \\ \vdots & \vdots & & \vdots \\ 1 & X_{N1} & \cdots & X_{NK} \end{pmatrix}, \underbrace{Y}_{N \times 1} = \begin{pmatrix} Y_1 \\ \vdots \\ Y_N \end{pmatrix}, \underbrace{\beta}_{(K+1) \times 1} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_K \end{pmatrix}$$

8.2 Theoretical Properties of OLS estimator

- We briefly review theoretical properties of OLS estimator.

1. **Unbiasdness**: Conditional on the explantory variables X, the expectation of the OLS estimator $\hat{\beta}$ is equal to the true value β.
$$E[\hat{\beta}|X] = \beta$$
2. **Consistency**: As the sample size N goes to infinity, the OLS estimator $\hat{\beta}$ converges to β in probability
$$\hat{\beta} \xrightarrow{p} \beta$$
3. **Asymptotic normality**: Will talk this later

8.3 Interpretation and Specifications of Linear Regression Model

- Remember that
$$Y_i = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_K X_{Ki} + \epsilon_i$$
- The coefficient β_k captures the effect of X_k on Y **ceteris paribus (all things being equal)**
- Equivalently,
$$\frac{\partial Y}{\partial X_k} = \beta_k$$
if X_k is continuous random variable.
- If we can estimate β_k without bias, we can obtain **causal effect** of X_k on Y.
  - This is of course very difficult task. We will see this more later.
- We will see several specifications that are frequently used in empirical analysis. 1. Nonlinear term 1. log specification 2. dummy (categorical) variables 3. interaction terms

8.3.1 Nonlinear term

- We can capture non-linear relationship between Y and X in a linearly additive form
$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \beta_3 X_i^3 + \epsilon_i$$
- As long as the error term ϵ_i appreas in a additively linear way, we can estimate the coefficients by OLS.
  - Multicollinarity could be an issue if we have many polynomials (see later).
  - You can use other non-linear variables such as $log(x)$ and $\sqrt{x}$.

8.3.2 log specification

- We often use `log` variables in both dependent and independent variables.
- Using `log` changes the interpretation of the coefficient β in terms of scales.

Dependent variable	Explanatory variable	interpretation
Y	X	1 unit increase in X causes β units change in Y
$\log Y$	X	1 unit increase in X causes $100\beta\%$ incchangerease in Y
Y	$\log X$	1% increase in X causes $\beta/100$ unit change in Y
$\log Y$	$\log X$	1% increase in X causes $\beta\%$ change in Y

8.3.3 Dummy variable

- A dummy variable takes only 1 or 0. This is used to express qualititative information

- Example: Dummy variable for race

$$white_i = \begin{cases} 1 & if\ white \\ 0 & otherwise \end{cases}$$

- The coefficient on a dummy variable captures the difference of the outcome Y between categories
- Consider the linear regression

$$Y_i = \beta_0 + \beta_1 white_i + \epsilon_i$$

The coefficient β_1 captures the difference of Y between white and non-white people.

8.3.4 Interaction term

- You can add the interaction of two explanatory variables in the regression model.
- For example:

$$wage_i = \beta_0 + \beta_1 educ_i + \beta_2 white_i + \beta_3 educ_i \times white_i + \epsilon_i$$

where $wage_i$ is the earnings of person i and $educ_i$ is the years of schooling for person i.
- The effect of $educ_i$ is

$$\frac{\partial wage_i}{\partial educ_i} = \beta_1 + \beta_3 white_i,$$

- This allows for heterogenous effects of education across races.

8.4 Measures of Fit

- We often use R^2 as a measure of the model fit.
- Denote **the fitted value** as $\hat{y}_i$

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_{i1} + \cdots + \hat{\beta}_K X_{iK}$$

  - Also called prediction from the OLS regression.
- R^2 is defined as

$$R^2 = \frac{SSE}{TSS},$$

where

$$SSE = \sum_i (\hat{y}_i - \bar{y})^2,\ TSS = \sum_i (y_i - \bar{y})^2$$

- R^2 captures the fraction of the variation of Y explained by the regression model.
- Adding variables always (weakly) increases R^2.
- In a regression model with multiple explanatory variables, we often use **adjusted** R^2 that adjusts the number of explanatory variables

$$\bar{R}^2 = 1 - \frac{N-1}{N-(K+1)} \frac{SSR}{TSS}$$

where

$$SSR = \sum_i (\hat{y}_i - y_i)^2 (= \sum_i \hat{u}_i^2),$$

8.5 Statistical Inference

- Notice that the OLS estimators are **random variables**. They depend on the data, which are random variables drawn from some population distribution.
- We can conduct statistical inferences regarding those OLS estimators: 1. Hypothesis testing 2. Constructing confidence interval
- I first explain the sampling distribution of the OLS estimators.

8.5.1 Distribution of the OLS estimators based on asymptotic theory

- Deriving the exact (finite-sample) distribution of the OLS estimators is very hard.
  - The OLS estimators depend on the data Y_i, X_i in a complex way.
  - We typically do not know the distribution of Y and X.
- We rely on **asymptotic** argument. We approximate the sampling distribution of the OLS esimator based on the cental limit theorem.
- Under the OLS assumption, the OLS estimator has **asymptotic normality**

$$\sqrt{N}(\hat{\beta} - \beta) \xrightarrow{d} N(0, V)$$

where

$$\underbrace{V}_{(K+1)\times(K+1)} = E[\mathbf{x}_i'\mathbf{x}_i]^{-1} E[\mathbf{x}_i'\mathbf{x}_i\epsilon_i^2] E[\mathbf{x}_i'\mathbf{x}_i]^{-1}$$

and

$$\underbrace{\mathbf{x}_i}_{(K+1)\times 1} = \begin{pmatrix} 1 \\ X_{i1} \\ \vdots \\ X_{iK} \end{pmatrix}$$

- We can **approximate** the distribution of $\hat{\beta}$ by

$$\hat{\beta} \sim N(\beta, V/N)$$

- The above is joint distribution. Let V_{ij} be the (i, j) element of the matrix V.
- The individual coefficient β_k follows

$$\hat{\beta}_k \sim N(\beta_k, V_{kk}/N)$$

8.5.1.1 Estimation of Asymptotic Variance

- V is an unknown object. Need to be estimated.
- Consider the estimator $\hat{V}$ for V using sample analogues

$$\hat{V} = \left(\frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i'\mathbf{x}_i\right)^{-1} \left(\frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i'\mathbf{x}_i\hat{\epsilon}_i^2\right) \left(\frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i'\mathbf{x}_i\right)^{-1}$$

where $\hat{\epsilon}_i = y_i - (\hat{\beta}_0 + \cdots + \hat{\beta}_K X_{iK})$ is the residual.
- Technically speaking, $\hat{V}$ converges to V in probability. (Proof is out of the scope of this course)
- We often use the (asymptotic) **standard error** $SE(\hat{\beta}_k) = \sqrt{\hat{V}_{kk}/N}$.
- The standard error is an estimator for the standard deviation of the OLS estimator $\hat{\beta}_k$.

8.5.2 Hypothesis testing

- OLS estimator is the random variable.
- You might want to test a particular hypothesis regarding those coefficients.
  - Does x really affects y?
  - Is the production technology the constant returns to scale?
- Here I explain how to conduct hypothesis testing.
- Step 1: Consider the null hypothesis H_0 and the alternative hypothesis H_1

$$H_0 : \beta_1 = k, H_1 : \beta_1 \neq k$$

  where k is the known number you set by yourself.
- Step 2: Define **t-statistic** by

$$t_n = \frac{\hat{\beta}_1 - k}{SE(\hat{\beta}_1)}$$

- Step 3: We reject H_0 is at α-percent significance level if

$$|t_n| > C_{\alpha/2}$$

  where $C_{\alpha/2}$ is the $\alpha/2$ percentile of the standard normal distribution.
  - We say we **fail to reject** H_0 if the above does not hold.

8.5.2.1 Caveats on Hypothesis Testing

- We often say $\hat{\beta}$ is **statistically significant** at 5% level if $|t_n| > 1.96$ when we set $k = 0$.
- Arguing the statistical significance alone is not enough for argument in empirical analysis.
- Magnitude of the coefficient is also important.
- Case 1: Small but statistically significant coefficient.
  - As the sample size N gets large, the SE decreases.
- Case 2: Large but statistically insignificant coefficient.
  - The variable might have an important (economically meaningful) effect.
  - But you may not be able to estimate the effect precisely with the sample at your hand.

8.5.2.2 F test

- We often test a composite hypothesis that involves multiple parameters such as

$$H_0 : \beta_1 + \beta_2 = 0, \ H_1 : \beta_1 + \beta_2 \neq 0$$

- We use **F test** in such a case (to be added).

8.5.3 Confidence interval

- 95% confidence interval

$$CI_n = \left\{ k : |\frac{\hat{\beta}_1 - k}{SE(\hat{\beta}_1)}| \leq 1.96 \right\} = \left[\hat{\beta}_1 - 1.96 \times SE(\hat{\beta}_1), \hat{\beta}_1 + 1.96 \times SE(\hat{\beta}_1) \right]$$

- Interpretation: If you draw many samples (dataset) and construct the 95% CI for each sample, 95% of those CIs will include the true parameter.

8.5.4 Homoskedasticity vs Heteroskedasticity

- So far, we did not put any assumption on the variance of the error term ϵ_i.
- The error term ϵ_i has **heteroskedasticity** if $Var(u_i|X_i)$ depends on X_i.
- If not, we call ϵ_i has **homoskedasticity**.
- This has an important implication on the asymptotic variance.
- Remember the asymptotic variance

$$\underbrace{V}_{(K+1)\times(K+1)} = E[\mathbf{x}_i'\mathbf{x}_i]^{-1}E[\mathbf{x}_i'\mathbf{x}_i\epsilon_i^2]E[\mathbf{x}_i'\mathbf{x}_i]^{-1}$$

  Standard errors based on this is called **heteroskedasticity robust standard errors**/
- If homoskedasticity holds, then

$$V = E[\mathbf{x}_i'\mathbf{x}_i]^{-1}\sigma^2$$

  where $\sigma^2 = V(\epsilon_i)$.
- In many statistical packages (including R and Stata), the standard errors for the OLS estimators are calcualted under homoskedasticity assumption as a default.
- However, if the error has heteroskedasticity, the standard error under homoskedasticity assumption will be **underestimated**.
- In OLS, **we should always use heteroskedasticity robust standard error.**
  - We will see how to fix this in R.

Chapter 9

Linear Regression 2: Implementation in R

9.1 Implementation in R

9.1.1 Preliminary: packages

- We use the following packages:
  - `AER` :
  - `dplyr` : data manipulation
  - `stargazer` : output of regression results

```
# Install package if you have not done so
# install.packages("AER")
# install.packages("dplyr")
# install.packages("stargazer")
# install.packages("lmtest")

# load packages
library("AER")
```

```
##         car
##         carData
##         lmtest
##         zoo
##
##          : 'zoo'
##         'package:base'       :
##
##      as.Date, as.Date.numeric
##         sandwich
##         survival
```

```
library("dplyr")
```

```
##
##          : 'dplyr'

##         'package:car'       :
##
##      recode

##         'package:stats'       :
##
##      filter, lag

##         'package:base'       :
##
##      intersect, setdiff, setequal, union
```

```
library("stargazer")
```

```
##
## Please cite as:

##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.

##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
library("lmtest")
```

9.1.2 Empirical setting: Data from California School

- Question: How does the student-teacher ratio affects test scores?
- We use data from California school, which is included in `AER` package.
  - See here for the details: https://www.rdocumentation.org/packages/AER/versions/1.2-6/topics/CASchools

```
# load the the data set in the workspace
data(CASchools)
```

- Use `class()` function to see `CASchools` is `data.frame` object.

```
class(CASchools)
```

```
## [1] "data.frame"
```

- We take 2 steps for the analysis.
  - Step 1: Look at data (descriptive analysis)
  - Step 2: Run regression

9.1.3 Step 1: Descriptive analysis

- It is always important to grasp your data before running regression.
- `head()` function give you a first overview of the data.

```
head(CASchools)
```

```
##   district                          school  county grades students
## 1    75119              Sunol Glen Unified Alameda  KK-08      195
## 2    61499            Manzanita Elementary   Butte  KK-08      240
## 3    61549     Thermalito Union Elementary   Butte  KK-08     1550
## 4    61457 Golden Feather Union Elementary   Butte  KK-08      243
## 5    61523        Palermo Union Elementary   Butte  KK-08     1335
```

```
## 6    62042         Burrel Union Elementary  Fresno  KK-08      137
##   teachers calworks   lunch computer expenditure    income   english  read
## 1    10.90   0.5102  2.0408       67    6384.911 22.690001  0.000000 691.6
## 2    11.15  15.4167 47.9167      101    5099.381  9.824000  4.583333 660.5
## 3    82.90  55.0323 76.3226      169    5501.955  8.978000 30.000002 636.3
## 4    14.00  36.4754 77.0492       85    7101.831  8.978000  0.000000 651.9
## 5    71.50  33.1086 78.4270      171    5235.988  9.080333 13.857677 641.8
## 6     6.40  12.3188 86.9565       25    5580.147 10.415000 12.408759 605.7
##    math
## 1 690.0
## 2 661.9
## 3 650.9
## 4 643.5
## 5 639.9
## 6 605.4
```

- Alternatively, you can use `browse()` to see the entire dataset in browser window.

9.1.3.1 Create variables

- Create several variables that are needed for the analysis.
- We use `dplyr` for this purpose.

```
CASchools %>%
  mutate( STR = students / teachers ) %>%
  mutate( score = (read + math) / 2 ) -> CASchools
```

9.1.3.2 Descriptive statistics

- There are several ways to show descriptive statistics
- The standard one is to use `summary()` function

```
summary(CASchools)
```

```
##    district            school                  county      grades
##  Length:420         Length:420         Sonoma     : 29   KK-06: 61
##  Class :character   Class :character   Kern       : 27   KK-08:359
##  Mode  :character   Mode  :character   Los Angeles: 27
##                                        Tulare     : 24
##                                        San Diego  : 21
##                                        Santa Clara: 20
##                                        (Other)    :272
##     students          teachers          calworks          lunch
##  Min.   :   81.0   Min.   :   4.85   Min.   : 0.000   Min.   :  0.00
##  1st Qu.:  379.0   1st Qu.:  19.66   1st Qu.: 4.395   1st Qu.: 23.28
##  Median :  950.5   Median :  48.56   Median :10.520   Median : 41.75
##  Mean   : 2628.8   Mean   : 129.07   Mean   :13.246   Mean   : 44.71
##  3rd Qu.: 3008.0   3rd Qu.: 146.35   3rd Qu.:18.981   3rd Qu.: 66.86
##  Max.   :27176.0   Max.   :1429.00   Max.   :78.994   Max.   :100.00
##
##     computer       expenditure       income          english
##  Min.   :   0.0   Min.   :3926   Min.   : 5.335   Min.   : 0.000
##  1st Qu.:  46.0   1st Qu.:4906   1st Qu.:10.639   1st Qu.: 1.941
##  Median : 117.5   Median :5215   Median :13.728   Median : 8.778
```

```
##  Mean   : 303.4   Mean   :5312   Mean   :15.317   Mean   :15.768
##  3rd Qu.: 375.2   3rd Qu.:5601   3rd Qu.:17.629   3rd Qu.:22.970
##  Max.   :3324.0   Max.   :7712   Max.   :55.328   Max.   :85.540
##
##       read            math            STR            score
##  Min.   :604.5   Min.   :605.4   Min.   :14.00   Min.   :605.5
##  1st Qu.:640.4   1st Qu.:639.4   1st Qu.:18.58   1st Qu.:640.0
##  Median :655.8   Median :652.5   Median :19.72   Median :654.5
##  Mean   :655.0   Mean   :653.3   Mean   :19.64   Mean   :654.2
##  3rd Qu.:668.7   3rd Qu.:665.9   3rd Qu.:20.87   3rd Qu.:666.7
##  Max.   :704.0   Max.   :709.5   Max.   :25.80   Max.   :706.8
##
```

- This returns the desriptive statistics for all the variables in dataframe.
- You can combine this with `dplyr::select`

```
CASchools %>%
  select(STR, score) %>%
  summary()
```

```
##       STR            score
##  Min.   :14.00   Min.   :605.5
##  1st Qu.:18.58   1st Qu.:640.0
##  Median :19.72   Median :654.5
##  Mean   :19.64   Mean   :654.2
##  3rd Qu.:20.87   3rd Qu.:666.7
##  Max.   :25.80   Max.   :706.8
```

- You can do a bit lengthly thing manually like this.

```
# compute sample averages of STR and score
avg_STR <- mean(CASchools$STR)
avg_score <- mean(CASchools$score)

# compute sample standard deviations of STR and score
sd_STR <- sd(CASchools$STR)
sd_score <- sd(CASchools$score)

# set up a vector of percentiles and compute the quantiles
quantiles <- c(0.10, 0.25, 0.4, 0.5, 0.6, 0.75, 0.9)
quant_STR <- quantile(CASchools$STR, quantiles)
quant_score <- quantile(CASchools$score, quantiles)

# gather everything in a data.frame
DistributionSummary <- data.frame(Average = c(avg_STR, avg_score),
                                  StandardDeviation = c(sd_STR, sd_score),
                                  quantile = rbind(quant_STR, quant_score))

# print the summary to the console
DistributionSummary
```

```
##               Average StandardDeviation quantile.10. quantile.25.
## quant_STR    19.64043          1.891812      17.3486     18.58236
## quant_score 654.15655         19.053347     630.3950    640.05000
##             quantile.40. quantile.50. quantile.60. quantile.75.
## quant_STR       19.26618     19.72321      20.0783     20.87181
```

```
## quant_score    649.06999    654.45000     659.4000    666.66249
##             quantile.90.
## quant_STR       21.86741
## quant_score    678.85999
```

- My personal favorite is to use `stargazer` function.

```
stargazer(CASchools, type = "text")
```

```
##
## ================================================================================
## Statistic    N    Mean    St. Dev.     Min    Pctl(25)  Pctl(75)     Max
## --------------------------------------------------------------------------------
## students    420 2,628.793 3,913.105    81        379      3,008    27,176
## teachers    420  129.067   187.913    4.850    19.662    146.350  1,429.000
## calworks    420  13.246    11.455     0.000     4.395    18.981    78.994
## lunch       420  44.705    27.123     0.000    23.282    66.865    100.000
## computer    420  303.383   441.341      0        46       375.2     3,324
## expenditure 420 5,312.408  633.937  3,926.070 4,906.180 5,601.401 7,711.507
## income      420  15.317     7.226     5.335    10.639    17.629    55.328
## english     420  15.768    18.286       0        1.9      23.0       86
## read        420  654.970   20.108    604.500   640.400   668.725   704.000
## math        420  653.343   18.754      605      639.4     665.8      710
## STR         420  19.640     1.892    14.000    18.582    20.872    25.800
## score       420  654.157   19.053    605.550   640.050   666.662   706.750
## --------------------------------------------------------------------------------
```

- You can choose summary statistics you want to report.

```
CASchools %>%
  stargazer( type = "text", summary.stat = c("n", "p75", "sd") )
```

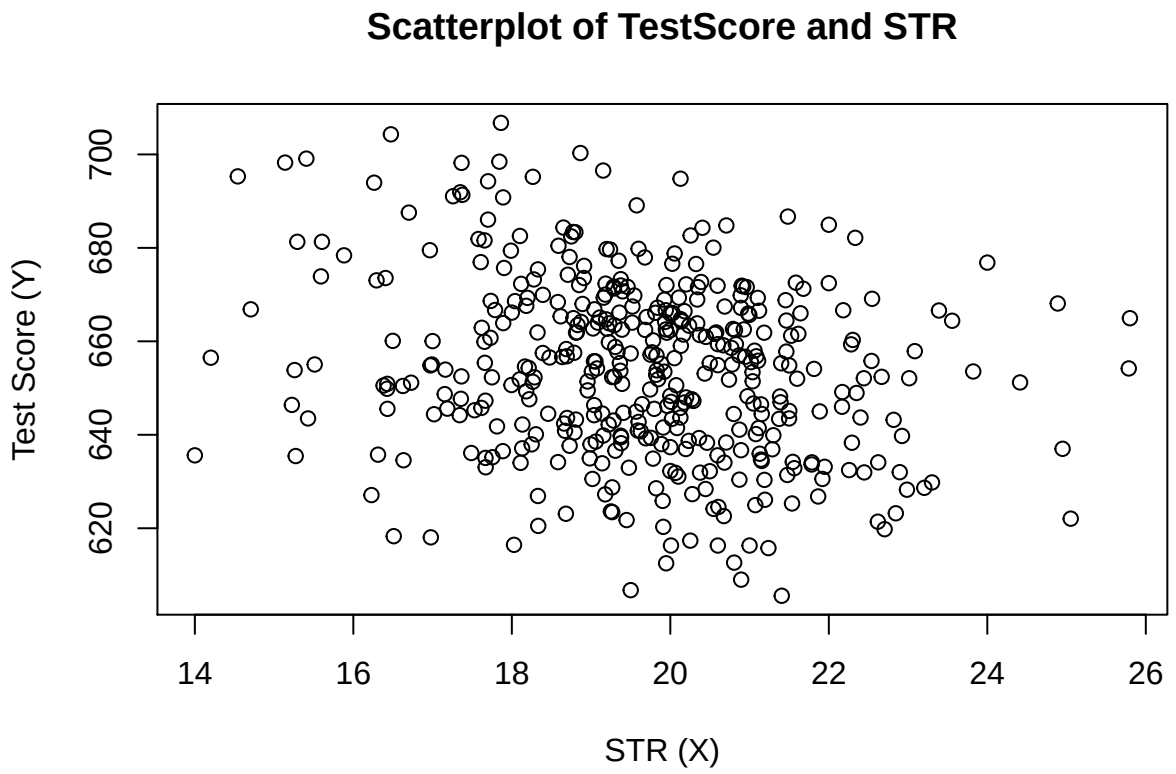
```
##
## ===================================
## Statistic    N  Pctl(75)  St. Dev.
## -----------------------------------
## students    420   3,008   3,913.105
## teachers    420  146.350   187.913
## calworks    420  18.981    11.455
## lunch       420  66.865    27.123
## computer    420   375.2    441.341
## expenditure 420 5,601.401  633.937
## income      420  17.629     7.226
## english     420   23.0     18.286
## read        420  668.725   20.108
## math        420   665.8    18.754
## STR         420  20.872     1.892
## score       420  666.662   19.053
## -----------------------------------
```

- See https://www.jakeruss.com/cheatsheets/stargazer/#the-default-summary-statistics-table for the details.
- We will use `stargazer` to report regression results.

9.1.3.3 Scatter plot

- Let's see how test score and student-teacher-ratio is correlated.

```
plot(score ~ STR,
     data = CASchools,
     main = "Scatterplot of TestScore and STR",
     xlab = "STR (X)",
     ylab = "Test Score (Y)")
```



- Use `cor()` to compute the correlation between two numeric vectors.

```
cor(CASchools$STR, CASchools$score)
```

```
## [1] -0.2263627
```

9.1.4 Step 2: Run regression

9.1.4.1 Simple linear regression

- We use `lm()` function to run linear regression
- First, consider the simple linear regression

$$score_i = \beta_0 + \beta_1 size_i + \epsilon_i$$

where $size_i$ is the class size (student-teacher-ratio).
  - From now on we call student-teacher-ratio (STR) class size.

- To run this regression, we use `lm`

```
# First, we rename the variable `STR`
CASchools %>%
  dplyr::rename( size = STR) -> CASchools

# Run regression and save results in the varaiable `model1_summary`
model1_summary <- lm( score ~ size, data = CASchools)

# See the results
summary(model1_summary)
```

```
##
## Call:
## lm(formula = score ~ size, data = CASchools)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.727 -14.251   0.483  12.822  48.540
##
## Coefficients:
##             Estimate Std. Error t value             Pr(>|t|)
## (Intercept) 698.9329     9.4675  73.825 < 0.0000000000000002 ***
## size         -2.2798     0.4798  -4.751           0.00000278 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.58 on 418 degrees of freedom
## Multiple R-squared:  0.05124,    Adjusted R-squared:  0.04897
## F-statistic: 22.58 on 1 and 418 DF,  p-value: 0.000002783
```

- Interpretations
  - An increase of one student per teacher leads to 2.2 point decrease in test scores.
  - p value is very small. The effect of the class size on test score is significant. Note: Be careful. These standard errors are NOT heteroskedasiticity robust. We will come back to this point soon.
  - $R^2 = 0.051$, implying that 5.1% of the variance of the dependent variable is explained by the model.
- You can add more variable in the regression (will see this soon)

9.1.4.2 Correction of Robust standard error

- We use `vcovHC()` function, a partof the package `sandwich`, to obtain the robust standard errors.
  - The package `sandwich` is automatically loaded if you load `AER` package.

```
# compute heteroskedasticity-robust standard errors
vcov <- vcovHC(model1_summary, type = "HC1")

# get standard error: the square root of the diagonal element in vcov
robust_se <- sqrt(diag(vcov))
robust_se
```

```
## (Intercept)        size
##  10.3643617   0.5194893
```

- Notice that robust standard errors are larger than the one we obtained from `lm`!

- How to combine the robust standard errors with the original summary? Use `coeftest()` from the package `lmtest`

```
# load `lmtest`
library(lmtest)

# Combine robust standard errors
coeftest(model1_summary, vcov. = vcov)
```

```
##
## t test of coefficients:
##
##              Estimate Std. Error t value              Pr(>|t|)
## (Intercept) 698.93295   10.36436 67.4362 < 0.0000000000000022 ***
## size         -2.27981    0.51949 -4.3886            0.00001447 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

9.1.4.3 Report by Stargazer

- `stargazer` is useful to show the regression result.

```
# load
library(stargazer)

# Create output by stargazer
stargazer::stargazer(model1_summary, type ="text")
```

```
##
## ================================================
##                         Dependent variable:
##                     ---------------------------
##                                score
## ------------------------------------------------
## size                          -2.280***
##                                (0.480)
##
## Constant                     698.933***
##                                (9.467)
##
## ------------------------------------------------
## Observations                    420
## R2                              0.051
## Adjusted R2                     0.049
## Residual Std. Error       18.581 (df = 418)
## F Statistic           22.575*** (df = 1; 418)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

- Use robust standard errors in stargazer output

```
# Prepare robust standard errors in list
rob_se <- list( sqrt(diag(vcovHC(model1_summary, type = "HC1") ) ) )

# generate regression table.
```

```
stargazer( model1_summary,
           se = rob_se,
           type = "text")
```

```
##
## ================================================
##                          Dependent variable:
##                      ---------------------------
##                                 score
## ------------------------------------------------
## size                           -2.280***
##                                 (0.519)
##
## Constant                      698.933***
##                                (10.364)
##
## ------------------------------------------------
## Observations                     420
## R2                               0.051
## Adjusted R2                      0.049
## Residual Std. Error       18.581 (df = 418)
## F Statistic           22.575*** (df = 1; 418)
## ================================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```

9.1.4.4 Full results

Taken from https://www.econometrics-with-r.org/7-6-analysis-of-the-test-score-data-set.html

```
# load the stargazer library

# estimate different model specifications
spec1 <- lm(score ~ size, data = CASchools)
spec2 <- lm(score ~ size + english, data = CASchools)
spec3 <- lm(score ~ size + english + lunch, data = CASchools)
spec4 <- lm(score ~ size + english + calworks, data = CASchools)
spec5 <- lm(score ~ size + english + lunch + calworks, data = CASchools)

# gather robust standard errors in a listh
rob_se <- list(sqrt(diag(vcovHC(spec1, type = "HC1"))),
               sqrt(diag(vcovHC(spec2, type = "HC1"))),
               sqrt(diag(vcovHC(spec3, type = "HC1"))),
               sqrt(diag(vcovHC(spec4, type = "HC1"))),
               sqrt(diag(vcovHC(spec5, type = "HC1"))))

# generate a LaTeX table using stargazer
stargazer(spec1, spec2, spec3, spec4, spec5,
          se = rob_se,
          digits = 3,
          header = F,
          column.labels = c("(I)", "(II)", "(III)", "(IV)", "(V)"),
          type ="text",
          keep.stat = c("N", "adj.rsq"))
```

```
##
## ========================================================================
##                                 Dependent variable:
##              ---------------------------------------------------------
##                                        score
##                  (I)         (II)       (III)        (IV)        (V)
##                  (1)         (2)         (3)         (4)         (5)
## ------------------------------------------------------------------------
## size          -2.280***   -1.101**   -0.998***   -1.308***   -1.014***
##                (0.519)     (0.433)     (0.270)     (0.339)     (0.269)
##
## english                   -0.650***   -0.122***   -0.488***   -0.130***
##                            (0.031)     (0.033)     (0.030)     (0.036)
##
## lunch                                 -0.547***               -0.529***
##                                        (0.024)                 (0.038)
##
## calworks                                          -0.790***    -0.048
##                                                    (0.068)     (0.059)
##
## Constant      698.933*** 686.032*** 700.150*** 697.999*** 700.392***
##                (10.364)   (8.728)     (5.568)     (6.920)     (5.537)
##
## ------------------------------------------------------------------------
## Observations    420         420         420         420         420
## Adjusted R2    0.049       0.424       0.773       0.626       0.773
## ========================================================================
## Note:                                       *p<0.1; **p<0.05; ***p<0.01
```

- The coefficient on the class size decreases as we add more explantory variables. Can you explain why? (Hint: omitted variable bias)

Chapter 10

Linear Regression 3: Discussions on OLS Assumptions

10.1 Introduction

- Remember that we have four assumptions in OLS estimation

1. Random sample: $\{Y_i, X_{i1}, \ldots, X_{iK}\}$ is i.i.d. drawn sample - i.i.d.: identically and independently distributed
2. ϵ_i has zero conditional mean
$$E[\epsilon_i | X_{i1}, \ldots, X_{iK}] = 0$$
   - This implies $Cov(X_{ik}, \epsilon_i) = 0$ for all k. (or $E[\epsilon_i X_{ik}] = 0$)
   - **No correlation between error term and explanatory variables.**
3. Large outliers are unlikely:
   - The random variable Y_i and X_{ik} have finite fourth moments.
4. No perfect multicollinearity:
   - There is no linear relationship betwen explanatory variables.

- The OLS estimator has ideal properties (consistency, asymptotic normality, unbiasdness) under these assumptions.
- In this chapter, we study the role of these assumptions.
- In particular, we focus on the following two assumptions
  1. No correlation between ϵ_{it} and X_{ik}
  2. No perfect multicollinearity

10.2 Endogeneity problem

- When $Cov(x_k, \epsilon) = 0$ does not hold, we have **endogeneity problem**
  - We call such x_k an **endogenous variable**.
- There are several cases in which we have endogeneity problem
  1. Omitted variable bias
  2. Measurement error
  3. Simultaneity
  4. Sample selection
- Here, I focus on the omitted variable bias.

10.2.1 Omitted variable bias

- Consider the wage regression equation (true model)

$$\begin{aligned} \log W_i = &\quad \beta_0 + \beta_1 S_i + \beta_2 A_i + u_i \\ E[u_i|S_i, A_i] = &\quad 0 \end{aligned}$$

  where W_i is wage, S_i is the years of schooling, and A_i is the ability.

- What we want to know is β_1, the effect of the schooling on the wage **holding other things fixed**. Also called the returns from education.
- An issue is that we do not often observe the ability of a person directly.
- Suppose that you omit A_i and run the following regression instead.

$$\log W_i = \alpha_0 + \alpha_1 S_i + v_i$$

  - Notice that $v_i = \beta_2 A_i + u_i$, so that S_i and v_i is likely to be correlated.

- The OLS estimator $\hat{\alpha}_1$ will have the bias:

$$E[\hat{\alpha}_1] = \beta_1 + \beta_2 \frac{Cov(S_i, A_i)}{Var(S_i)}$$

  – You can also say $\hat{\alpha}_1$ is not consistent for β_1, i.e.,

$$\hat{\alpha}_1 \xrightarrow{p} \beta_1 + \beta_2 \frac{Cov(S_i, A_i)}{Var(S_i)}$$

- This is known as **omitted variable bias formula**.
- Omitted variable bias depends on 1. The effect of the omitted variable (A_i here) on the dependent variable: β_2 2. Correlation between the omitted variable and the explanatory variable.
- This is super-important: You can make a guess regarding the direction and the magnitude of the bias!!
- This is crucial when you read an empirical paper and do am empirical exercise.
- Here is the summary table - x_1: included, x_2 omitted. β_2 is the coefficient on x_2.

	$Cov(x_1, x_2) > 0$	$Cov(x_1, x_2) < 0$
$\beta_2 > 0$	Positive bias	Negative bias
$\beta_2 < 0$	Negative bias	Positive bias

10.2.2 Correlation v.s. Causality

- Omitted variable bias is related to a well-known argument of "Correlation or Causality".
- Example: Does the education indeed affect your wage, or the unobserved ability affects both the ducation and the wage, leading to correlation between education and wage?
- See my lecture note from Intermediate Seminar (Fall 2018) for the details.

10.3 Multicollinearity issue

10.3.1 Perfect Multicollinearity

- If one of the explanatory variables is a linear combination of other variables, we have perfect multicollinearity.
- In this case, you cannot estimate all the coefficients.
- For example,

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 \cdot x_2 + \epsilon_i$$

  and $x_2 = 2x_1$.
- These explanatory variables are collinear. You are not able to estimate both β_1 and β_2.
- To see this, the above model can be written as

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 \cdot 2x_1 + \epsilon_i$$

  and this is the same as

$$y_i = \beta_0 + (\beta_1 + 2\beta_2) x_1 + \epsilon_i$$

- You can estimate the composite term $\beta_1 + 2\beta_2$ as a coefficient on x_1, but not β_1 and β_2 separately.

10.3.1.1 Some Intuition

- Intuitively speaking, the regression coefficients are estimated by capturing how the variation of the explanatory variable x affects the variation of the dependent variable y
- Since x_1 and x_2 are moving together completely, we cannot say how much the variation of y is due to x_1 or x_2, so that β_1 and β_2.

10.3.1.2 Dummy variable

- Consider the dummy variables that indicate male and famale.

$$male_i = \begin{cases} 1 & if\ male \\ 0 & if\ female \end{cases},\ female_i = \begin{cases} 1 & if\ female \\ 0 & if\ male \end{cases}$$

- If you put both male and female dummies into the regression,

$$y_i = \beta_0 + \beta_1 famale_i + \beta_2 male_i + \epsilon_i$$

- Since $male_i + famale_i = 1$ for all i, we have perfect multicolinarity.
- You should always omit the dummy variable of one of the groups in the linear regression.
- For example,

$$y_i = \beta_0 + \beta_1 famale_i + \epsilon_i$$

- In this case, β_1 is interpreted as the effect of being famale **in comparison with male**.
  - The omitted group is the basis for the comparison.

- You should the same thing when you deal with multiple groups such as

$$freshman_i = \begin{cases} 1 & if\ freshman \\ 0 & otherwise \end{cases}$$

$$sophomore_i = \begin{cases} 1 & if\ sophomore \\ 0 & otherwise \end{cases}$$

$$junior_i = \begin{cases} 1 & if\ junior \\ 0 & otherwise \end{cases}$$

$$senior_i = \begin{cases} 1 & if\ senior \\ 0 & otherwise \end{cases}$$

and

$$y_i = \beta_0 + \beta_1 freshman_i + \beta_2 sophomore_i + \beta_3 junior_i + \epsilon_i$$

10.3.2 Imperfect multicollinearity.

- Though not perfectly co-linear, the correlation between explanatory variables might be very high, which we call imperfect multicollinearity.
- How does this affect the OLS estimator?
- To see this, we consider the following simple model (with homoskedasticity)

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \epsilon_i, V(\epsilon_i) = \sigma_2$$

- You can show that the conditional variance (not asymptotic variance) is given by

$$V(\hat{\beta}_1|X) = \frac{\sigma^2}{N \cdot \hat{V}(x_{1i}) \cdot (1 - R_1^2)}$$

where $\hat{V}(x_{1i})$ is the sample variance

$$\hat{V}(x_{1i}) = \frac{1}{N} \sum (x_{1i} - \bar{x_1})^2$$

and R_1^2 is the R-squared in the following regression of x_2 on x_1.

$$x_{1i} = \pi_0 + \pi_1 x_{2i} + u_i$$

- You can see that the variance of the OLS estimator $\hat{\beta}_1$ is small if
  1. N is large (i.e., more observations!)
  2. $\hat{V}(x_{1i})$ is large (more variation in x_{1i}!)
  3. R_1^2 is small.
- Here, high R_1^2 means that x_{1i} is explained well by other variables in a linear way. – The extreme case is $R_1^2 = 1$, that is x_{1i} is the linear combination of other variables, implying perfect multicolinearity!!

10.4 Lesson for an empirical analysis

- We often say **the variation of the variable of interest is important in an empirical analysis**.
- This has two meanings:
  1. **exogenous** variation (i.e., uncorrelated with error term)
  2. large variance

- The former is a key for mean independence assumption.
- The latter is a key for precise estimation (smaller standard error).
- If we have more variation, the standard error of the OLS estimator is small, meaning that we can precisely estimate the coefficient.
- The variation of the variable **after controlling for other factors that affects** y is also crucial (corresponding to $1 - R_1^2$ above).
  - If you do not include other variables (say x_2 above), you will have omitted variable bias.
- To address research questions using data, it is important to find a good variation of the explanatory variable that you want to focus on. This is often called **identification strategy**.
  - Identification strategy is context-specific. To have a good identification strategy, you should be familiar with the background knowledge of your study.

Chapter 11

Exercise 2 (Problem Set 3)

- Due date: June 4th (Tue) 11pm.

11.1 Rules

- If you are enrolled in Japanese class (i.e., Wednesday 2nd), you can use both Japanese and English to write your answer.
- Submit your solution through `CourseN@vi`.
- **Important:** Submission format
- If you use Rmarkdown, please compile your Rmarkdown file into either "html" or "PDF" file and submit **both** the compiled file and a Rmarkdown file.
- If you do not use Rmarkdown, please submit the document file that contains your answer and R script file (.R file) **separately**, that is, you submit **two files**.

11.2 Question 1: Omitted Variable Bias

The goal of this question is to investigate the omitted variable bias through Monte Carlo simulations. Consider the following model

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i$$

You compare the sampling distribution of OLS estimates for β_1 with and without x_2 included in the regression. Here is the suggested procedure for this excercise.

1. Set the data generating process.
   - Set the parameters $\beta_0 = 1, \beta_1 = 2, \beta_2 = 1$
   - The explanatory variables (x_1, x_2) are i.i.d. drawn from the multivariate normal distribution

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim N\left(\begin{pmatrix} 3 \\ 4 \end{pmatrix}, \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \right)$$

   - The error term $\epsilon_i t$ is i.i.d. drawn from $N(0, 1)$
2. Draw the dataset $\{y_i, x_{i1}, x_{i2}\}_{i=1}^N$ with $N = 200$.
   - To draw the random numbers from the joint normal distribution, use `mvrnorm` function from `MASS` package.

3. Using the drawn dataset, regress y on x_1 and x_2 with constant term. Obtain the OLS estimate for β_1. Let's call this $\hat{\beta}_1^{long}$
4. Regress y on x_1 with constant term **by omitting** x_2 and obtain the OLS estimate for β_1. Let's call this $\hat{\beta}_1^{short}$
5. Repeat step 2 to 4 for 500 times and obtain $\hat{\beta}_1^{long}$ and $\hat{\beta}_1^{short}$ for each drawn sample.
6. Plot the distribution of $\hat{\beta}_1^{long}$ and $\hat{\beta}_1^{short}$ across samples.

Please answer the following questions using your simulation results.

- (1) Show the sampling distribution for $\hat{\beta}_1^{long}$ and $\hat{\beta}_1^{short}$.
- (2) Are these estimates biased? If biased, is the magnitude of bias consistent with theory?
- (3) We set $Cov(x_1, x_2) = 1$ above. Repeat the same simulation with $Cov(x_1, x_2) = 0$. How does the result would change?

11.3 Question 2: Empirical Analysis using Data from Washington(2008, AER)

Acknowledgement: This exercise is based on the material from Econ 281 "Introductory Applied Econometrics" in Winter 2017 taught by Daley Kutzman at Northwestern University

This exercise uses the data from Ebonya Washington's paper, "Female Socialization: How Daughters Affect Their Legislator Father's Voting on Women's Issues," published in *American Economic Review* in 2008. This paper studies whether having a daughter affects legislator's voting on women's issues.

11.3.1 Preliminary: data cleaning

You can find the file "data_PS3_basic.dta" that is available at the journal website. This file is in Stata format. You can use `read.dta` function included in `foreign` packages.

```
# Example:
library(foreign)
mydata <- read.dta("c:/mydata.dta")
```

The original dataset contains data from the 105th to 108th U.S. Congress. We only use the observation from the 105th congress. The variable `congress` indicates this information. Use `filter` function in `dplyr` to subtract observations from the 105th.

The dataset contains many variables, some of which are not used in this exercise. Keep the following variables in the final dataset (Hint: use `select` function in `dplyr`).

Name	Description
aauw	AAUW score
totchi	Total number of children
ngirls	Number of daughters
party	Political party. Democrats if 1, Republicans if 2, and Independent if 3.
famale	Female dummy variable
white	White dummy variable
srvlng	Years of service
age	Age
demvote	State democratic vote share in most recent presidential election
medinc	District median income

Name	Description
perf	Female proportion of district voting age population
perw	White proportion of total district population
perhs	High school graduate proportion of district population age 25
percol	College graduate proportion of district population age 25
perur	Urban proportion of total district population
moredef	State proportion who favor more defense spending
stateabb	State abbreviation
district	id for electoral district

You can find the detailed description of each variable in the original paper. The main variable in this analysis is `AAUW`, a score created by the American Association of University Women (AAUW). For each congress, AAUW selects pieces of legislation in the areas of education, equality, and reproductive rights. The AAUW keeps track of how each legislator voted on these pieces of legislation and whether their vote aligned with the AAUW's position. The legislator's score is equal to the proportion of these votes made in agreement with the AAUW.

11.3.2 Questions

1. Report summary statistics of the following variables in the dataset: political party, age, race, gender, AAUW score, the number of children, and the number of daughters.
2. Estimate the following linear regression models using `lm` command. Do not forget to correct the standard errors! Report your regression results in a table.

$$
\begin{aligned}
aauw_i &= \beta_0 + \beta_1 ngirls_i + \epsilon_i \\
aauw_i &= \beta_0 + \beta_1 ngirls_i + \beta_2 totchi + \epsilon_i \\
aauw_i &= \beta_0 + \beta_1 ngirls_i + \beta_2 totchi + \beta_3 famale_i + \beta_4 repub_i + \epsilon_i
\end{aligned}
$$

   - All the variables used in the above specifications are in the dataset except for $repub_i$. $repub_i$ takes 1 if the legislator i is affiliated with the Republican party.
   - **Important** Never put the raw output from `lm` command shown in R console into your answer! Please prepare a table for regression results as if you write a report or a paper. **If you copy and paste the raw output from `lm` command, you will get 0 points for the empirical exercise part of this problem set.**
3. Compare the OLS estimates of β_1 across the above three specifications. Discuss what explains the difference (if any) of the estimate across three specifications?
4. Consider the third specification (with 3 controls in addition to $ngirls_i$). Conditional on the number of children and other variables, do you think $ngrils_i$ is plausibly exogenous (i.e., uncorrelated with the error term)? Discuss.
5. It is possible that the effects of having daughters might be different for female and male legislators. Estimate a regression model that allow for heterogenous effects of daughters for male and female. Discuss whether this story is true or not.

Chapter 12

Instrumental Variable 1: Framework

12.1 Introduction: Endogeneity Problem and its Solution

- When $Cov(x_k, \epsilon) = 0$ does not hold, we have **endogeneity problem**
  - We call such x_k an **endogenous variable**.
- In this chapter, I introduce an **instrumental variable** estimation method, a solution to this issue.
- The lecture plan
  1. More on endogeneity issues
  2. Framework
  3. Implementation in R
  4. Examples

12.2 Examples of Endogeneity Problem

- Here, I explain a bit more about endogeneity problems.
  1. Omitted variable bias
  2. Measurement error
  3. Simultaneity

12.2.1 More on Omitted Variable Bias

- Remember the wage regression equation (true model)

$$\begin{aligned} \log W_i = &\quad \beta_0 + \beta_1 S_i + \beta_2 A_i + u_i \\ E[u_i | S_i, A_i] = &\quad 0 \end{aligned}$$

  where W_i is wage, S_i is the years of schooling, and A_i is the ability.
- Suppose that you omit A_i and run the following regression instead.

$$\log W_i = \alpha_0 + \alpha_1 S_i + v_i$$

  Notice that $v_i = \beta_2 A_i + u_i$, so that S_i and v_i is likely to be correlated.
- You might want to add more and more additional variables to capture the effect of ability.
  - Test scores, GPA, SAT scores, etc…
- However, can you make sure that S_i is indeed exogenous after adding many control variables?
- Multivariate regression cannot deal with the presence of **unobserved heterogeneity** that matters both in wage and years of schooling.

12.2.2 Measurement error

- Measurement error in variables
  - Reporting error, respondent does not understand the question, etc…
- Consider the regression

$$y_i = \beta_0 + \beta_1 x_i^* + \epsilon_i$$

- Here, we only observe x_i with error:

$$x_i = x_i^* + e_i$$

  where e_i is measurement error.
  - e_i is independent from ϵ_i and x_i^* (called classical measurement error)
  - You can think e_i as a noise added to the data.
- The regression equation is

$$\begin{aligned} y_i &= \beta_0 + \beta_1(x_i - e_i) + \epsilon_i \\ &= \beta_0 + \beta_1 x_i + (\epsilon_i - \beta_1 e_i) \end{aligned}$$

- Then we have correlation between x_i and the error $\epsilon_i - \beta_1 e_i$, violating the mean independence assumption

12.2.3 Simultaneity (or reverse causality)

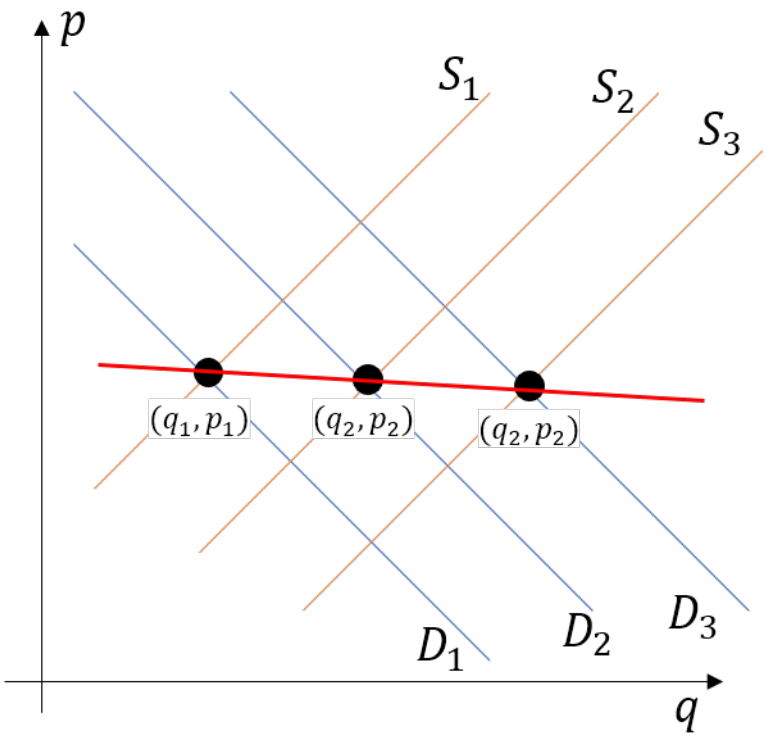
- Dependent variable and explanatory variable (endogenous variable) are determined simultaneously.
- Consider the demand and supply curve

$$\begin{aligned} q^d &= \beta_0^d + \beta_1^d p + \beta_2^d x + u^d \\ q^s &= \beta_0^s + \beta_1^s p + \beta_2^s z + u^s \end{aligned}$$

- The equilibrium price and quantity are determined by $q^d = q^s$.
- In this case,

$$p = \frac{(\beta_2^s z - \beta_2^d z) + (\beta_0^s - \beta_0^d) + (u^s - u^d)}{\beta_1^d - \beta_1^s}$$

  implying the correlation between the price and the error term.
- Putting this differently, the data points we observed is the intersection of these supply and demand curves.



- How can we distinguish demand and supply?

12.3 Idea of IV Regression

- Let's start with a simple case.

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i,$$

  and $Cov(x_i, \epsilon_i) \neq 0$.

- Now, we consider another variable z_i, which we call **instrumental variable (IV)**.

- Instrumental variable z_i should satisfies the following two conditions:

  1. **Independence**: $Cov(z_i, \epsilon_i) = 0$. No correlation between IV and error.
  2. **Relevance**: $Cov(z_i, x_i) \neq 0$. There should be correlation between IV and endogenous variable x_i.

- Idea: Use the variation of x_i **induced by instrument** z_i to estimate the direct (causal) effect of x_i on y_i, that is β_1!.

- More on this:

  1. Intuitively, the OLS estimator captures the correlation between x and y.
  2. If there is no correlation between x and ϵ, it captures the causal effect β_1.
  3. If not, the OLS estimator captures both direct and indirect effect, the latter of which is bias.
  4. Now, let's capture the variation of x due to instrument z,
     - Such a variation should exist under **relevance** assumption.
     - Such a variation should not be correlated with the error under **independence assumption**
  5. By looking at the correlation between such variation and y, you can get the causal effect β_1.

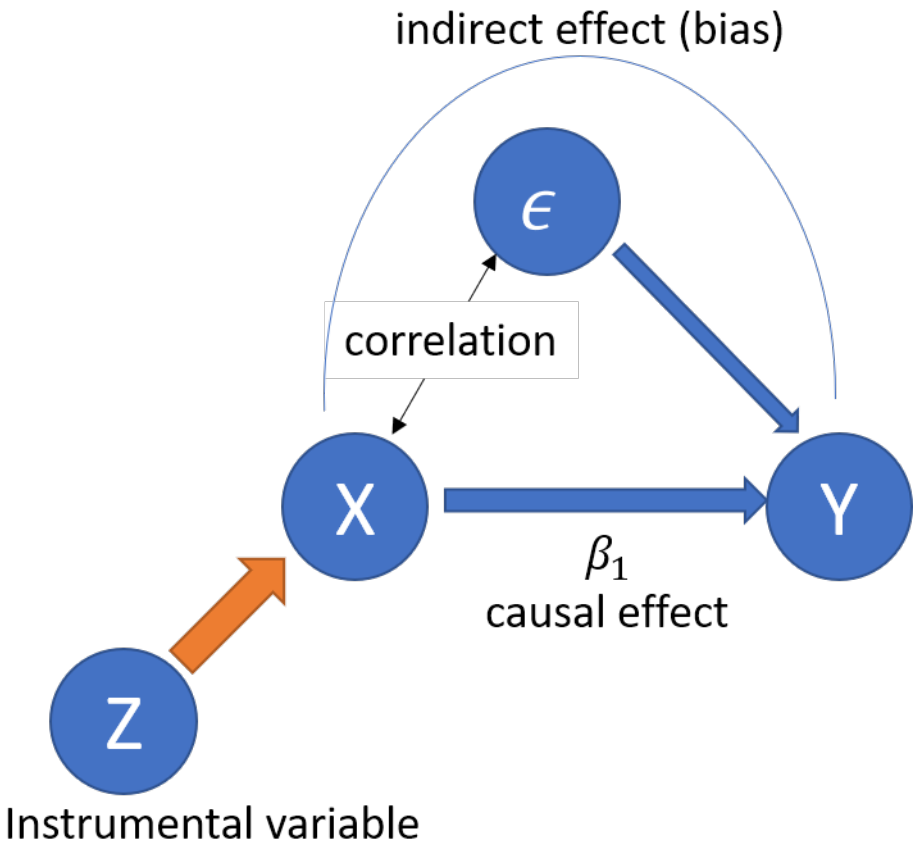


Figure 12.1: Idea IV

12.4 Formal Framework and Estimation

12.4.1 Model

- We now introduce a general framework with multiple endogenous variables and multiple instruments.
- Consider the model

$$Y_i = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_K X_{Ki} + \beta_{K+1} W_{1i} + \cdots + \beta_{K+R} W_{Ri} + u_i,$$

  with $i = 1, \ldots, n$ is the general instrumental variables regression model where
  - Y_i is the dependent variable
  - $\beta_0, \ldots, \beta_{K+R}$ are $1 + K + R$ unknown regression coefficients
  - $X_{1i}, \ldots, X_{Ki}$ are K endogenous regressors: $Cov(X_{ki}, u_i) \neq 0$ for all k.
  - $W_{1i}, \ldots, W_{Ri}$ are R exogenous regressors which are uncorrelated with u_i. $Cov(W_{ri}, u_i) = 0$ for all r.
  - u_i is the error term
  - $Z_{1i}, \ldots, Z_{Mi}$ are M instrumental variables
- I will discuss conditions for valid instruments later.

12.4.2 Estimation by Two Stage Least Squares (2SLS)

- We can estimate the above model by **Two Stage Least Squares (2SLS)**
- Step 1: **First-stage regression(s)**
  - Run an OLS regression for each of the endogenous variables $(X_{1i}, \ldots, X_{ki})$ on all instrumental variables $(Z_{1i}, \ldots, Z_{mi})$, all exogenous variables $(W_{1i}, \ldots, W_{ri})$ and an intercept.
  - Compute the fitted values $(\widehat{X}_{1i}, \ldots, \widehat{X}_{ki})$.
- Step 2: **Second-stage regression**
  - Regress the dependent variable Y_i on **the predicted values** of all endogenous regressors $(\widehat{X}_{1i}, \ldots, \widehat{X}_{ki})$, all exogenous variables $(W_{1i}, \ldots, W_{ri})$ and an intercept using OLS.
  - This gives $\widehat{\beta}_0^{TSLS}, \ldots, \widehat{\beta}_{k+r}^{TSLS}$, the 2SLS estimates of the model coefficients.

12.4.2.1 Intuition

- Why does this work? Let's go back to the simple example with 1 endogenous variable and 1 IV.
- In the first stage, we estimate

$$x_i = \pi_0 + \pi_1 z_i + v_i$$

  by OLS and obtain the fitted value $\widehat{x}_i = \widehat{\pi}_0 + \widehat{\pi}_1 z_i$.
- In the second stage, we estimate

$$y_i = \beta_0 + \beta_1 \widehat{x}_i + u_i$$

- Since $\widehat{x}_i$ depends only on z_i, which is uncorrelated with u_i, the second stage can estimate β_1 without bias.
- Can you see the importance of both independence and relevance asssumption here? (More formal discussion later)

12.4.3 Conditions for Valid IVs in a general framework

12.4.3.1 Necessary condition

- Depending on the number of IVs, we have three cases

1. Over-identification: $M > K$
2. Just identification] $M = K$
3. Under-identification $M < K$

- The necessary condition is $M \geq K$.
  - We should have more IVs than endogenous variables!!

12.4.3.2 Sufficient condition

- How about sufficiency?
- In a general framework, the sufficient condition for valid instruments is given as follows.
  1. **Independence**: $Cov(Z_{mi}, \epsilon_i) = 0$ for all m.
  2. **Relevance**: In the second stage regression, the variables

$$\left(\widehat{X}_{1i}, \ldots, \widehat{X}_{ki}, W_{1i}, \ldots, W_{ri}, 1\right)$$

     are not perfectly multicollinear.
- What does the relevance condition mean?
- In the simple example above, The first stage is

$$x_i = \pi_0 + \pi_1 z_i + v_i$$

  and the second stage is

$$y_i = \beta_0 + \beta_1 \widehat{x}_i + u_i$$

- The second stage would have perfect multicollinarity if $\pi_1 = 0$ (i.e., $\widehat{x}_i = \pi_0$).
- Back to the general case, the first stage for X_k can be written as

$$X_{ki} = \pi_0 + \pi_1 Z_{1i} + \cdots + \pi_M Z_{Mi} + \pi_{M+1} W_{1i} + \cdots + \pi_{M+R} W_{Ri}$$

  and one of $\pi_1, \cdots, \pi_M$ should be non-zero.
- Intuitively speaking, **the instruments should be correlated with endogenous variables after controlling for exogenous variables**

12.5 Check Instrument Validity

12.5.1 Relevance

- Instruments are **weak** if those instruments explain little variation in the endogenous variables.
- Weak instruments lead to
  1. imprecise estimates (higher standard errors)
  2. The asymptotic distribution would deviate from a normal distribution even if we have a large sample.
- Here is a rule of thumb to check the relevance conditions.
- Consider the case with one endogenous variable X_{1i}.
- The first stage regression

$$X_k = \pi_0 + \pi_1 Z_{1i} + \cdots + \pi_M Z_{Mi} + \pi_{M+1} W_{1i} + \cdots + \pi_{M+R} W_{Ri}$$

- And test the null hypothesis

$$H_0 : \pi_1 = \cdots = \pi_M = 0$$
$$H_1 : otherwise$$

  - This is F test (test of joint hypothesis)
- If we can reject this, we can say no concern for weak instruments.
- A rule of thumbs is that the F statistic should be larger than 10.
-

12.5.2 Independence (Instrument exogeneity)

- Arguing for independence is hard and a key in empirical analysis.
- Justification of this assumption depends on a context, institutional features, etc…
- We will see this through examples in the next chapter.

Chapter 13

Instrumental Variable 2: Implementation in R

13.1 Example 1: Wage regression

- Use dataset "Mroz", cross-sectional labor force participation data that accompany "Introductory Econometrics" by Wooldridge.
  - Original data from *"The Sensitivity of an Empirical Model of Married Women's Hours of Work to Economic and Statistical Assumptions"* by Thomas Mroz published in *Econometrica* in 1987.
  - Detailed description of data: https://www.rdocumentation.org/packages/npsf/versions/0.4.2/topics/mroz

```
library("foreign")

# You might get a message "cannot read factor labels from Stata 5 files", but you do not have to worry
data <- read.dta("MROZ.DTA")
```

```
## Warning in read.dta("MROZ.DTA"): cannot read factor labels from Stata 5
## files
```

- Describe data

```
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
stargazer(data, type = "text")
```

```
##
## ===============================================================
## Statistic  N     Mean     St. Dev.   Min   Pctl(25) Pctl(75)  Max
## ---------------------------------------------------------------
## inlf      753   0.568      0.496      0       0        1       1
## hours     753  740.576    871.314     0       0      1,516   4,950
## kidslt6   753   0.238      0.524      0       0        0       3
## kidsge6   753   1.353      1.320      0       0        2       8
```

```
## age       753   42.538     8.073      30      36       49      60
## educ      753   12.287     2.280      5       12       13      17
## wage      428   4.178      3.310    0.128   2.263    4.971   25.000
## repwage   753   1.850      2.420    0.000   0.000    3.580   9.980
## hushrs    753 2,267.271   595.567    175    1,928    2,553   5,010
## husage    753   45.121     8.059      30      38       52      60
## huseduc   753   12.491     3.021      3       11       15      17
## huswage   753   7.482      4.231    0.412   4.788    9.167   40.509
## faminc    753 23,080.600 12,190.200 1,500   15,428   28,200  96,000
## mtr       753   0.679      0.083    0.442   0.622    0.721   0.942
## motheduc  753   9.251      3.367      0       7        12      17
## fatheduc  753   8.809      3.572      0       7        12      17
## unem      753   8.624      3.115      3      7.5       11      14
## city      753   0.643      0.480      0       0        1       1
## exper     753   10.631     8.069      0       4        15      45
## nwifeinc  753   20.129     11.635   -0.029  13.025   24.466  96.000
## lwage     428   1.190      0.723    -2.054  0.817    1.604   3.219
## expersq   753  178.039    249.631     0       16       225    2,025
## ----------------------------------------------------------------------
```

- Consider the wage regression

$$\log(w_i) = \beta_0 + \beta_1 educ_i + \beta_2 exper_i + \beta_3 exper_i^2 + \epsilon_i$$

- We assume that $exper_i$ is exogenous but $educ_i$ is endogenous.
- As an instrument for $educ_i$, we use the years of schooling for his or her father and mother, which we call $fathereduc_i$ and $mothereduc_i$.
- Discussion on these IVs will be later.

```
library("AER")
```

```
##        car
##        carData
##        lmtest
##        zoo
##
##         : 'zoo'
##        'package:base'       :
##
##      as.Date, as.Date.numeric
##        sandwich
##        survival
```

```
library("dplyr")
```

```
##
##         : 'dplyr'
##        'package:car'       :
##
##      recode
##        'package:stats'       :
##
```

```
##      filter, lag
##
##         'package:base'       :
##
##      intersect, setdiff, setequal, union
```

```
# data cleaning
data %>%
  select(lwage, educ, exper, expersq, motheduc, fatheduc) %>%
  filter( is.na(lwage) == 0 ) -> data

# OLS regression
result_OLS <- lm( lwage ~ educ + exper + expersq, data = data)

# IV regression using fathereduc and mothereduc
result_IV <- ivreg(lwage ~ educ + exper + expersq | fatheduc + motheduc + exper + expersq, data = data)

# Robust standard errors
# gather robust standard errors in a list
rob_se <- list(sqrt(diag(vcovHC(result_OLS, type = "HC1"))),
               sqrt(diag(vcovHC(result_IV, type = "HC1"))))

# Show result
stargazer(result_OLS, result_IV, type ="text", se = rob_se)
```

```
##
## ===============================================================
##                                        Dependent variable:
##                                    -----------------------------
##                                               lwage
##                                          OLS           instrumental
##                                                          variable
##                                          (1)               (2)
## ---------------------------------------------------------------
## educ                                   0.107***           0.061*
##                                         (0.013)           (0.033)
##
## exper                                  0.042***           0.044***
##                                         (0.015)           (0.016)
##
## expersq                                 -0.001*           -0.001**
##                                        (0.0004)           (0.0004)
##
## Constant                               -0.522***           0.048
##                                         (0.202)           (0.430)
##
## ---------------------------------------------------------------
## Observations                              428               428
## R2                                       0.157             0.136
## Adjusted R2                              0.151             0.130
## Residual Std. Error (df = 424)           0.666             0.675
## F Statistic                       26.286*** (df = 3; 424)
## ===============================================================
## Note:                                      *p<0.1; **p<0.05; ***p<0.01
```

- How about the first stage? You should always check this!!

```r
# First stage regression

result_1st <- lm(educ ~ motheduc + fatheduc + exper + expersq, data = data)

# F test
linearHypothesis(result_1st,
                 c("fatheduc = 0", "motheduc = 0" ),
                 vcov = vcovHC, type = "HC1")
```

```
## Linear hypothesis test
##
## Hypothesis:
## fatheduc = 0
## motheduc = 0
##
## Model 1: restricted model
## Model 2: educ ~ motheduc + fatheduc + exper + expersq
##
## Note: Coefficient covariance matrix supplied.
##
##   Res.Df Df      F                Pr(>F)
## 1    425
## 2    423  2 48.644 < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

13.1.1 Discussion on IV

- Labor economists have used family background variables as IVs for education.
- Relevance: OK from the first stage regression.
- Independence: A bit suspicious. Parents' education would be correlated with child's ability through quality of nurturing at an early age.
- Still, we can see that these IVs can mitigate (though may not eliminate completely) the omitted variable bias.
- Discussion on the validity of instruments is crucial in empirical research.

13.2 Example 2: Estimation of the Demand for Cigaretts

- Demand model is a building block in many branches of Economics.
- For example, health economics is concerned with the study of how health-affecting behavior of individuals is influenced by the health-care system and regulation policy.
- Smoking is a prominent example as it is related to many illnesses and negative externalities.
- It is plausible that cigarette consumption can be reduced by taxing cigarettes more heavily.
- Question: how much taxes must be increased to reach a certain reduction in cigarette consumption? -> Need to know **price elasticity of demand** for cigaretts.
- Use `CigarrettesSW` in the package `AER`.
- a panel data set that contains observations on cigarette consumption and several economic indicators for all 48 continental federal states of the U.S. from 1985 to 1995.

- What is **panel data**? The data involves both time series and cross-sectional information.
  - The variable is denoted as y_{it}, which indexed by individual i and time t.
  - Cross section data y_i: information for a particular individual i (e.g., income for person i).
  - Time series data y_t: information for a particular time period (e.g., GDP in year y)
  - Panel data y_{it}: income of person i in year t.
- We will see more on panel data later in this course. For now, we use the panel data as just cross-sectional data (**pooled cross-sections**)

```
# load the data set and get an overview
library(AER)
data("CigarettesSW")
summary(CigarettesSW)
```

```
##      state      year         cpi          population
##  AL     : 2   1985:48   Min.   :1.076   Min.   :  478447
##  AR     : 2   1995:48   1st Qu.:1.076   1st Qu.: 1622606
##  AZ     : 2             Median :1.300   Median : 3697472
##  CA     : 2             Mean   :1.300   Mean   : 5168866
##  CO     : 2             3rd Qu.:1.524   3rd Qu.: 5901500
##  CT     : 2             Max.   :1.524   Max.   :31493524
##  (Other):84
##      packs             income               tax            price
##  Min.   : 49.27   Min.   :  6887097   Min.   :18.00   Min.   : 84.97
##  1st Qu.: 92.45   1st Qu.: 25520384   1st Qu.:31.00   1st Qu.:102.71
##  Median :110.16   Median : 61661644   Median :37.00   Median :137.72
##  Mean   :109.18   Mean   : 99878736   Mean   :42.68   Mean   :143.45
##  3rd Qu.:123.52   3rd Qu.:127313964   3rd Qu.:50.88   3rd Qu.:176.15
##  Max.   :197.99   Max.   :771470144   Max.   :99.00   Max.   :240.85
##
##       taxs
##  Min.   : 21.27
##  1st Qu.: 34.77
##  Median : 41.05
##  Mean   : 48.33
##  3rd Qu.: 59.48
##  Max.   :112.63
##
```

- Consider the following model

$$\log(Q_{it}) = \beta_0 + \beta_1 \log(P_{it}) + \beta_2 \log(income_{it}) + u_{it}$$

  where
  - Q_{it} is the number of packs per capita in state i in year t,
  - P_{it} is the after-tax average real price per pack of cigarettes, and
  - $income_{it}$ is the real income per capita. This is demand shifter.
- As an IV for the price, we use the followings:
  - $SalesTax_{it}$: the proportion of taxes on cigarettes arising from the general sales tax.
    - Relevant as it is included in the after-tax price
    - Exogenous(indepndent) since the sales tax does not influence demand directly, but indirectly through the price.
  - $CigTax_{it}$: the cigarett-specific taxes

```
library(dplyr)
CigarettesSW %>%
```

```
  mutate( rincome = (income / population) / cpi) %>%
  mutate( rprice  = price / cpi ) %>%
  mutate( salestax = (taxs - tax) / cpi ) %>%
  mutate( cigtax = tax/cpi ) -> Cigdata
```

- Let's run the regressions

```
cig_ols <- lm(log(packs) ~ log(rprice) + log(rincome) ,  data = Cigdata)
#coeftest(cig_ols, vcov = vcovHC, type = "HC1")


cig_ivreg <- ivreg(log(packs) ~ log(rprice) + log(rincome)  |
                     log(rincome) +  salestax +  cigtax, data = Cigdata)
#coeftest(cig_ivreg, vcov = vcovHC, type = "HC1")

# Robust standard errors
rob_se <- list(sqrt(diag(vcovHC(cig_ols, type = "HC1"))),
               sqrt(diag(vcovHC(cig_ivreg, type = "HC1"))))

# Show result
stargazer(cig_ols, cig_ivreg, type ="text", se = rob_se)
```

```
##
## ===============================================================
##                                       Dependent variable:
##                                   ----------------------------
##                                          log(packs)
##                                     OLS           instrumental
##                                                     variable
##                                     (1)               (2)
## ---------------------------------------------------------------
## log(rprice)                      -1.334***         -1.229***
##                                   (0.154)           (0.155)
##
## log(rincome)                      0.318**           0.257*
##                                   (0.154)           (0.153)
##
## Constant                         10.067***         9.736***
##                                   (0.502)           (0.514)
##
## ---------------------------------------------------------------
## Observations                        96                96
## R2                                 0.552             0.549
## Adjusted R2                        0.542             0.539
## Residual Std. Error (df = 93)      0.165             0.165
## F Statistic                  57.185*** (df = 2; 93)
## ===============================================================
## Note:                                *p<0.1; **p<0.05; ***p<0.01
```

- The first stage regression

```
# First stage regression

result_1st <- lm(log(rprice) ~ log(rincome) + log(rincome) + salestax + cigtax , data= Cigdata)
```

```
# F test
linearHypothesis(result_1st,
                 c("salestax = 0", "cigtax = 0" ),
                 vcov = vcovHC, type = "HC1")
```

```
## Linear hypothesis test
##
## Hypothesis:
## salestax = 0
## cigtax = 0
##
## Model 1: restricted model
## Model 2: log(rprice) ~ log(rincome) + log(rincome) + salestax + cigtax
##
## Note: Coefficient covariance matrix supplied.
##
##   Res.Df Df      F                Pr(>F)
## 1     94
## 2     92  2 127.77 < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

13.3 Example 3: Effects of Turnout on Partisan Voting

- THOMAS G. HANSFORD and BRAD T. GOMEZ "Estimating the Electoral Effects of Voter Turnout" The American Political Science Review Vol. 104, No. 2 (May 2010), pp. 268-288
  - Link: https://www.cambridge.org/core/journals/american-political-science-review/article/estimating-the-electoral-effects-of-voter-turnout/8A880C28E79BE770A5CA1A9BB6CF933C
- Here, we will see a simplified version of their analysis.
- The dataset is here

```
library(readr)
```

```
## Warning:     'readr'     3.5.3   R
```

```
HGdata <- read_csv("HansfordGomez_Data.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double()
## )
```

```
## See spec(...) for full column specifications.
```

```
stargazer::stargazer(as.data.frame(HGdata), type="text")
```

```
##
## ================================================================================================
## Statistic            N       Mean       St. Dev.      Min    Pctl(25)  Pctl(75)     Max
## ------------------------------------------------------------------------------------------------
## Year              27,401  1,973.972     16.111      1,948    1,960     1,988     2,000
## FIPS_County       27,401 29,985.500   13,081.250    4,001   20,013    39,157    56,045
## Turnout           27,401   65.562       10.514     20.366   58.477    72.613    100.000
## Closing2          27,401   23.053       13.042       0        11        30        125
## Literacy          27,401    0.058        0.234       0        0         0         1
```

```
## PollTax           27,401    0.001        0.023        0        0         0         1
## Motor             27,401    0.211        0.408        0        0         0         1
## GubElection       27,401    0.434        0.496        0        0         1         1
## SenElection       27,401    0.680        0.467        0        0         1         1
## GOP_Inc           27,401    0.501        0.500        0        0         1         1
## Yr52              27,401    0.071        0.258        0        0         0         1
## Yr56              27,401    0.071        0.258        0        0         0         1
## Yr60              27,401    0.071        0.258        0        0         0         1
## Yr64              27,401    0.071        0.258        0        0         0         1
## Yr68              27,401    0.071        0.258        0        0         0         1
## Yr72              27,401    0.071        0.258        0        0         0         1
## Yr76              27,401    0.071        0.258        0        0         0         1
## Yr80              27,401    0.071        0.258        0        0         0         1
## Yr84              27,401    0.072        0.258        0        0         0         1
## Yr88              27,401    0.072        0.258        0        0         0         1
## Yr92              27,401    0.072        0.258        0        0         0         1
## Yr96              27,401    0.072        0.258        0        0         0         1
## Yr2000            27,401    0.070        0.256        0        0         0         1
## DNormPrcp_KRIG    27,401    0.005        0.208     -0.419   -0.093     0.001     2.627
## GOPIT             27,401   33.282       34.066        0        0      66.3       100
## DemVoteShare2_3MA 27,401   44.250       10.606     10.145   37.006    50.996    88.982
## DemVoteShare2     27,401   43.622       12.415      6.420   34.954    51.858    97.669
## RainGOPI          27,401    0.007        0.142       -0     -0.03         0         2
## TO_DVS23MA        27,401  2,886.877    792.530    473.161 2,321.025 3,384.772 8,526.616
## Rain_DVS23MA      27,401    0.355       10.188     -25.054  -4.019     0.028    144.257
## dph               27,401    0.021        0.145        0        0         0         1
## dvph              27,401    0.018        0.133        0        0         0         1
## rph               27,401    0.025        0.155        0        0         0         1
## rvph              27,401    0.025        0.155        0        0         0         1
## state_del         27,401    0.037        0.187     -0.821   -0.090     0.172     0.619
## dph_StateVAP      27,401 77,525.150   597,474.000     0        0         0     6,150,988
## dvph_StateVAP     27,401 63,138.400   663,707.600     0        0         0     12,700,000
## rph_StateVAP      27,401 243,707.900 1,720,659.000    0        0         0     18,300,000
## rvph_StateVAP     27,401 142,166.500 1,071,445.000    0        0         0     12,800,000
## State_DVS_lag     27,401   46.896        8.317     22.035   40.767    52.197    80.872
## State_DVS_lag2    27,401  2,268.381    786.199    485.533 1,661.934 2,724.515 6,540.244
## ------------------------------------------------------------------------------------------
```

- Data description:

Name	Description
Year	Election Year
FIPS_County	FIPS County Code
Turnout	Turnout as Pcnt VAP
Closing2	Days between registration closing date and election
Literacy	Literacy Test
PollTax	Poll Tax
Motor	Motor Voter
GubElection	Gubernatorial Election in State
SenElection	U.S. Senate Election in State
GOP_Inc	Republican Incumbent
Yr52	1952 Dummy
Yr56	1956 Dummy
Yr60	1960 Dummy

Name	Description
Yr64	1964 Dummy
Yr68	1968 Dummy
Yr72	1972 Dummy
Yr76	1976 Dummy
Yr80	1980 Dummy
Yr84	1984 Dummy
Yr88	1988 Dummy
Yr92	1992 Dummy
Yr96	1996 Dummy
Yr2000	2000 Dummy
DNormPrcp_KRIG	Election day rainfall - differenced from normal rain for the day
GOPIT	Turnout x Republican Incumbent
DemVoteShare_A	Partisan composition measure = 3 election moving avg. of Dem Vote Share
DemVoteShare2	Democratic Pres Candidate's Vote Share
RainGOPI	Rainfall measure x Republican Incumbent
TO_DVS3MA	Turnout x Partisan Composition measure
Rain_DVS3MA	Rainfall measure x Partisan composition measure
dph	=1 if home state of Dem pres candidate
dvph	=1 if home state of Dem vice pres candidate
rph	=1 if home state of Rep pres candidate
rvph	=1 if home state of Rep vice pres candidate
state_del	avg common space score for the House delegation
dph_StateVAP	dph*State voting age population
dvph_StateVAP	dvph*State voting age population
rph_StateVAP	rph*State voting age population
rvph_StateVAP	rvph*State voting age population
State_DVS_lag	State-wide Dem vote share, lagged one election
State_DVS_lag2	State_DVS_lag squared

- Consider the following regression

$$DemoShare_{it} = \beta_0 + \beta_1 Turnout_{it} + u_t + u_{it}$$

  where
  - $Demoshare_{it}$: Two-party vote share for Democrat candidate in county i in the presidential election in year t
  - $Turnout_{it}$: Turnout rate in county i in the presidential election in year t
  - u_t: **Year fixed effects**. Time dummies for each presidential election year
- As an IV, we use the rainfall measure denoted by `DNormPrcp_KRIG`

```
# You can do this, but it is tedious.
hg_ols <- lm( DemVoteShare2 ~ Turnout + Yr52 + Yr56 + Yr60 + Yr64 + Yr68 +
                Yr72 + Yr76 + Yr80 + Yr84 + Yr88 + Yr92 + Yr96 + Yr2000,  data = HGdata)
#coeftest(hg_ols, vcov = vcovHC, type = "HC1")

# By using "factor(Year)" as an explanatory variable, the regression automatically incorporates the dum
hg_ols <- lm( DemVoteShare2 ~ Turnout + factor(Year)   ,  data = HGdata)
#coeftest(hg_ols, vcov = vcovHC, type = "HC1")

# Iv regression
hg_ivreg <- ivreg( DemVoteShare2 ~ Turnout + factor(Year) |
                     factor(Year) + DNormPrcp_KRIG, data = HGdata)
#coeftest(hg_ivreg, vcov = vcovHC, type = "HC1")
```

```r
# Robust standard errors
rob_se <- list(sqrt(diag(vcovHC(hg_ols, type = "HC1"))),
               sqrt(diag(vcovHC(hg_ivreg, type = "HC1"))))

# Show result
stargazer(hg_ols, hg_ivreg, type ="text", se = rob_se)
```

```
##
## ===============================================================================
##                                          Dependent variable:
##                                     ----------------------------------------
##                                              DemVoteShare2
##                                            OLS             instrumental
##                                                              variable
##                                            (1)                (2)
## -------------------------------------------------------------------------------
## Turnout                                  -0.157***           0.363**
##                                           (0.008)            (0.178)
##
## factor(Year)1952                        -10.215***          -15.832***
##                                           (0.374)            (1.987)
##
## factor(Year)1956                         -8.756***          -13.656***
##                                           (0.357)            (1.746)
##
## factor(Year)1960                         -3.862***          -11.094***
##                                           (0.351)            (2.531)
##
## factor(Year)1964                         10.851***           6.837***
##                                           (0.337)            (1.441)
##
## factor(Year)1968                         -6.477***          -8.514***
##                                           (0.356)            (0.811)
##
## factor(Year)1972                        -13.749***          -16.473***
##                                           (0.335)            (1.022)
##
## factor(Year)1976                          -0.367            -2.111***
##                                           (0.337)            (0.715)
##
## factor(Year)1980                        -10.346***          -11.696***
##                                           (0.356)            (0.620)
##
## factor(Year)1984                        -13.134***          -13.515***
##                                           (0.352)            (0.404)
##
## factor(Year)1988                         -5.712***          -4.951***
##                                           (0.349)            (0.447)
##
## factor(Year)1992                          -0.327            -1.008**
##                                           (0.362)            (0.469)
##
## factor(Year)1996                         -1.193***           0.811
```

```
##                                          (0.377)             (0.782)
##
## factor(Year)2000                        -9.013***           -8.130***
##                                          (0.386)             (0.465)
##
## Constant                                59.085***           26.910**
##                                          (0.560)             (11.024)
##
## ------------------------------------------------------------------------
## Observations                             27,401               27,401
## R2                                        0.281               0.130
## Adjusted R2                               0.280               0.130
## Residual Std. Error (df = 27386)          10.533              11.582
## F Statistic                      763.153*** (df = 14; 27386)
## ========================================================================
## Note:                                        *p<0.1; **p<0.05; ***p<0.01
```

```r
# First stage regression
hg_1st <- lm(Turnout ~  factor(Year) + DNormPrcp_KRIG, data= HGdata)

# F test
linearHypothesis(hg_1st,
                 c("DNormPrcp_KRIG = 0" ),
                 vcov = vcovHC, type = "HC1")
```

```
## Linear hypothesis test
##
## Hypothesis:
## DNormPrcp_KRIG = 0
##
## Model 1: restricted model
## Model 2: Turnout ~ factor(Year) + DNormPrcp_KRIG
##
## Note: Coefficient covariance matrix supplied.
##
##   Res.Df Df      F           Pr(>F)
## 1  27387
## 2  27386  1 44.029 0.00000000003296 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- We will see this paper more in excercise 4 (due on July 9th).

Chapter 14

Exercise 3 (Problem Set 4)

- Due date: June 18th (Tue) 11pm.

14.1 Rules

- If you are enrolled in Japanese class (i.e., Wednesday 2nd), you can use both Japanese and English to write your answer.
- Submit your solution through `CourseN@vi`.
- **Important:** Submission format
- If you use Rmarkdown, please compile your Rmarkdown file into either "html" or "PDF" file and submit **both** the compiled file and a Rmarkdown file.
- If you do not use Rmarkdown, please submit the document file that contains your answer and R script file (.R file) **separately**, that is, you submit **two files**.

14.2 Question: Demand Estimation

- You might want to refer the lecture note in my other class where I covered the same topic.
- We use the dataset from Stephen Ryan (2012) "The Costs of Environmental Regulation in a Concentrated Industry", Econometrica, Issue 3, 1019-1061, 2012
- This is the data for Portland cement in the US. The data is panel for 22 regions (roughly corresponding to the US state) from 1981 to 1999.
- Ryan (2012) studies the effects of environmental regulation on dynamic competition among cement producers.
- Graduate-level knowledge about econometrics and industrial organization is needed to understand the whole paper. Rather we focus on a part of his analysis, that is demand estimation.
- The dataset is here.
- The original dataset contains many variable. Here, we pick a subset of the variable we use in the analysis.

```
library(readr)
```

```
## Warning:     'readr'     3.5.3   R
```

```
library(dplyr)
```

```
##
##          : 'dplyr'
##         'package:stats'       :
##
##      filter, lag
##         'package:base'       :
##
##      intersect, setdiff, setequal, union
```

```
library(stargazer)
```

```
##
## Please cite as:
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
RyanData <- readr::read_csv("cementDec2009.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   region = col_character()
## )
## See spec(...) for full column specifications.
```

```
RyanData %>%
  dplyr::select(year, shipped, price, wage96, coal96, elec96, population, gas96) -> RyanData

stargazer::stargazer(as.data.frame(RyanData), type = "text")
```

```
##
## ================================================================================================
## Statistic    N       Mean         St. Dev.       Min     Pctl(25)     Pctl(75)      Max
## ------------------------------------------------------------------------------------------------
## year        483   1,989.907         5.537       1,981     1,985        1,995       1,999
## shipped     483   2,850.592      1,575.068       186      1,717.5      3,689.5      10,262
## price       483     67.935         13.731      39.351     58.644       74.892      138.992
## wage96      483     31.679          4.377      20.139     28.695       34.475       44.338
## coal96      483     26.904          8.243      15.880     19.190       34.200       42.330
## elec96      483     5.706           1.023       4.230      4.750        6.740        7.600
## population 483 10,297,214.000 7,410,527.000 689,584 4,692,603.0 12,000,000 33,100,000
## gas96       483     6.263           2.266       3.701      5.041        6.810        24.304
## ------------------------------------------------------------------------------------------------
```

- The data description

Name	Description
shipped	Quantity of cement shipped (measured in thousands of tons per year)
price	Price (measured in dollars per ton).
wage96	wage in dollars per hour for skilled manufacturing workers, and taken from County Business Patterns

Name	Description
coal96	Coal price (dollars per ton)
elec96	electricity price (dollars per kilowatt hour for electricity)
populatio	the total populations of the states covered by a regional market.
gas96	Gas price (dollars per thousand cubic feet for gas.)

- Note that all price are adjusted to 1996 constant dollars.

14.2.1 Questions

1. Estimate the demand model for the following two specifications with and without instruments.

$$\log(Q_{jt}) = \beta_0 + \beta_1 \log(P_{jt}) + \epsilon_{it}$$
$$\log(Q_{jt}) = \beta_0 + \beta_1 \log(P_{jt}) + \beta_2 \log(population_{it}) + \epsilon_{it}$$

   where Q_{jt}: quantity, P_{jt}: price, $population_{jt}$: population As an instrument for price, you use wage, coal price, electricity price, and gas price. The variable $\log(population_{jt})$ is treated as exogenous.

2. Discuss the results above. In particular, explain (1) the importance of adding population as a control variable, and (2) the difference between results with and without IVs.

3. Discuss the validity of those instruments.

4. Instead of log-log specification, consider the following linear specification (with population as a control variable)

$$Q_{jt} = \beta_0 + \beta_1 P_{jt} + \beta_2 \log(population_{it}) + \epsilon_{it}.$$

   Estimate this specification using instruments. Do not forget checking the 1st stage.

5. The price elasticity of demand is defined as

$$\frac{\partial Q_{jt}}{\partial P_{jt}} \frac{P_{jt}}{Q_{jt}}$$

   If we use the log-log specification, the coefficient on the price is the elasticity and it is constant across markets and time. Using the estimation result with linear specification, calculate the price elasticity for each observation (i.e., market-year pair). Report the summary statistics of the price elasticity across markets and time. Compare the result with the one from log-log specification that includes population as a control variable.