R-workshops

Mehtap Hisarciklilar

2025-01-20

Table of contents

Welcome! **7**

I Seminar 1 (21 January 2025) **8**

1 Introduction to R **9**
1.1 R, R Studio and Quarto . . . 9
1.2 File Organisation . . . 10
1.3 Getting Help . . . 11

2 Basics of R **12**
2.1 Using R as a calculator . . . 12
2.1.1 Basic Operators . . . 13
2.1.2 Order of operators . . . 13
2.2 Storing information in objects . . . 14
2.2.1 Naming of objects . . . 15
2.2.2 Naming conventions . . . 15
2.2.3 Removing objects . . . 16
2.2.4 Example of using variables . . . 16
2.3 Datatypes in R . . . 17
2.3.1 Numeric . . . 17
2.3.2 Logical . . . 17
2.3.3 Characters . . . 18
2.3.4 Checking data type classes . . . 18

II Seminar 2 (28 January 2025) **20**

3 Data Management in R **21**
3.1 Packages and libraries . . . 21
3.2 Functions . . . 21
3.2.1 Basic Functions . . . 23
3.3 Scripts . . . 23

4 Importing Data into R **25**
4.1 Example 1: Crime data . . . 25
4.1.1 Task 1 . . . 25
4.1.2 Task 2 . . . 26
4.1.3 Task 3 . . . 27
4.1.4 Task 4 . . . 27
4.1.5 Task 5 . . . 28
4.1.6 Task 6 . . . 29
4.1.7 Task 7 . . . 30
4.1.8 Further Exercises . . . 31

5 Introduction to Regression Analysis **32**
5.1 Example 1: Crime data . . . 32
5.1.1 Task 1 . . . 32
5.1.2 Task 2 . . . 32
5.1.3 Task 3 . . . 33
5.1.4 Task 4 . . . 34
5.1.5 Task 5 . . . 35
5.1.6 Task 6 . . . 37
5.2 Example 2: Wage data . . . 38
5.2.1 Task 1 . . . 38
5.2.2 Task 2 . . . 38
5.2.3 Task 3 . . . 39
5.2.4 Task 4 . . . 40
5.2.5 Task 5 . . . 41
5.2.6 Task 6 . . . 45
5.2.7 Task 7 . . . 48
5.2.8 Task 8 . . . 49
5.2.9 Task 9 . . . 51
5.2.10 Task 10 . . . 52
5.2.11 Task 11 . . . 52
5.2.12 Task 12 . . . 56
5.2.13 Task 13 . . . 57
5.2.14 A Gentle Introduction to dplyr library . . . 58
5.3 Further Exercises . . . 60
5.3.1 Tasks . . . 61

III Seminar 3 (4 February 2025) **62**

6 Multiple Regression and Diagnostic Checks **63**
6.1 Example: wage data . . . 63
6.1.1 Task 1 . . . 63

6.1.2 Task 2 . . . 63
6.1.3 Task 3 . . . 64
6.1.4 Task 4 . . . 67
6.1.5 Task 5 . . . 68
6.1.6 Task 6 . . . 68
6.1.7 Task 7 . . . 69

IV Seminar 4 (11 February 2025) 70

7 Introduction to Time Series Analysis 71
7.1 Example: GAP Sales data . . . 71
7.1.1 Task 1 . . . 72
7.1.2 Task 2 . . . 72
7.1.3 Task 3 . . . 73
7.1.4 Task 4 . . . 76
7.1.5 Task 5 . . . 79
7.1.6 Task 6 . . . 83
7.1.7 Task 7 . . . 84
7.1.8 Task 8 . . . 89
7.1.9 Task 9 . . . 91

V Seminar 5 (18 February 2025) 94

8 Unit Root (Non-stationary Time Series): Augmented Dickerm(y-Fuller Test 95
8.1 Example: Pepper Price . . . 95
8.1.1 Task 1 . . . 97
8.1.2 Task 2 . . . 97
8.1.3 Task 3 . . . 98
8.1.4 Task 4 . . . 100
8.1.5 Task 5 . . . 102
8.1.6 Task 6 . . . 103

9 Cointegration: Engle-Granger Test 105
9.1 Example: Pepper Price . . . 106
9.1.1 Task 1 . . . 106
9.1.2 Task 2 . . . 108

VI Seminar 6 (25 February 2025) 110

10 Data Visualisation Using `ggplot2` 111
10.1 Example 1: Scatter plot with wage data . . . 111
10.1.1 Task 1 . . . 111
10.1.2 Task 2 . . . 111
10.1.3 Task 3 . . . 114
10.2 Example 2: Histogram of error term using wage2 data . . . 118
10.2.1 Task 4 . . . 118
10.3 Example 3: Line plots using Gap_sales data . . . 120

11 Introduction to Panel Data Analysis 125
11.1 Example: European Countries Gasoline Consumption Data . . . 125
11.1.1 Task 1 . . . 126
11.1.2 Task 2 . . . 126
11.1.3 Task 3 . . . 126
11.1.4 Task 4 . . . 128
11.1.5 Task 5 . . . 133
11.1.6 Task 6 . . . 134
11.1.7 Task 7 . . . 135
11.1.8 Task 8 . . . 137
11.1.9 Task 9 . . . 138

VII Seminar 7 and 8 (4 and 11 March 2025) 140

12 Panel Data Models 141
12.1 Least Squares Dummy Variables (LSDV) Approach . . . 142
12.1.1 Task 1 . . . 142
12.1.2 Task 2 . . . 143
12.1.3 Task 3 . . . 144
12.1.4 Task 4 . . . 145
12.1.5 Task 5 . . . 147
12.1.6 Task 6 . . . 148
12.1.7 Task 7 . . . 150
12.1.8 Task 8 . . . 151
12.1.9 Task 9 . . . 152
12.1.10 Task 10 . . . 154
12.1.11 Task 11 . . . 155

13 Panel Data Misspecification Tests 157
13.1 Testing for Autocorrelation . . . 158
13.1.1 Task 1 . . . 158

13.1.2 Task 2 . . . 159
13.1.3 Task 3 . . . 161
13.1.4 Task 4 . . . 162
13.2 Heteroscedasticity . . . 163
13.2.1 Task 5 . . . 163
13.2.2 Task 6 . . . 163

VIIISeminar 9 (18 March 2025) 167

14 Endogeneity and Instrumental Variable Estimation 168
14.1 Case I: 2SLS with one endogenous, one exogenous variable . . . 170
14.1.1 Task 1 . . . 170
14.1.2 Task 2 . . . 171
14.1.3 Task 3 . . . 171
14.1.4 Task 4 . . . 172
14.1.5 Task 5 . . . 174
14.1.6 Task 6 . . . 175
14.2 Case II: 2SLS with one endogenous and multiple exogenous variables . . . 177
14.2.1 Task 7 . . . 177
14.2.2 Task 8 . . . 179
14.2.3 Task 9 . . . 180
14.2.4 Task 10 . . . 180
14.2.5 Task 11 . . . 182
14.2.6 Guidance . . . 182
14.3 Further Reading . . . 183

IX Seminar 10 (25 March 2025) 184

15 IV Estimation: The Role of Institutions in Economic Growth 185
15.0.1 Task 1 . . . 187
15.0.2 Task 2 . . . 187
15.0.3 Task 3 . . . 189
15.0.4 Task 4 . . . 189
15.0.5 Task 5 . . . 190
15.0.6 Task 6 . . . 191

X References 193

References 194

Welcome!

This workbook is created for the seminar sessions of

6036ECN Further Econometrics module.

It is written using Quarto on RStudio by

Mehtap Hisarciklilar

Part I

Seminar 1 (21 January 2025)

1 Introduction to R

1.1 R, R Studio and Quarto

R is a very powerful statistical software that is becoming increasingly popular. Being able to do data analysis using R will very likely increase your employability.

Warning: R is not like other apps that you have used! It requires coding. You will need to attend the seminar sessions and practice regularly. There will be a lot of struggle, but the result is worth it.

R, as a programming language, is like any other language: the more you use it, the better you will get. Therefore, make sure to attend the lectures & seminars and engage with the module material. Otherwise, you will struggle to catch-up.

- I list below three apps that you will need to work with this module's material. I recommend installing these on your computers. Alternatively, you may use Coventry University's Appsanywhere platform to get access.
- We will be using **R** as the statistical analysis tool in this module. For R documentations, support and download links, visit the R Project for Statistical Computing. R is freely available for Linux, MacOS and Windows. Please download the version that matches your computer's operating system.
- To facilitate your work with R, I highly recommend to download and install the integrated development environment (IDE) **RStudio Desktop** from posit. This platform will make it easier for you to write and run R code.
- A final package that I highly recommend you to install is a publishing system, **Quarto**. You may use Quarto to produce documents in various formats (such as HTML, MS Word, PDF, PowerPoint, etc) while integrating your R code and output. You will easily have the option to change the format of your output as you desire. I will be using Quarto to produce R worksheets for this module. Please visit Quarto for further information and download.
  - Once you download Quarto, you will have access to it through RStudio.

RStudio has four main windows, that often have more than just one purpose. Figure 1.1 provides a brief description of each RStudio window. We will use all of them during the module, but the most important ones will be the console and the editor pane.

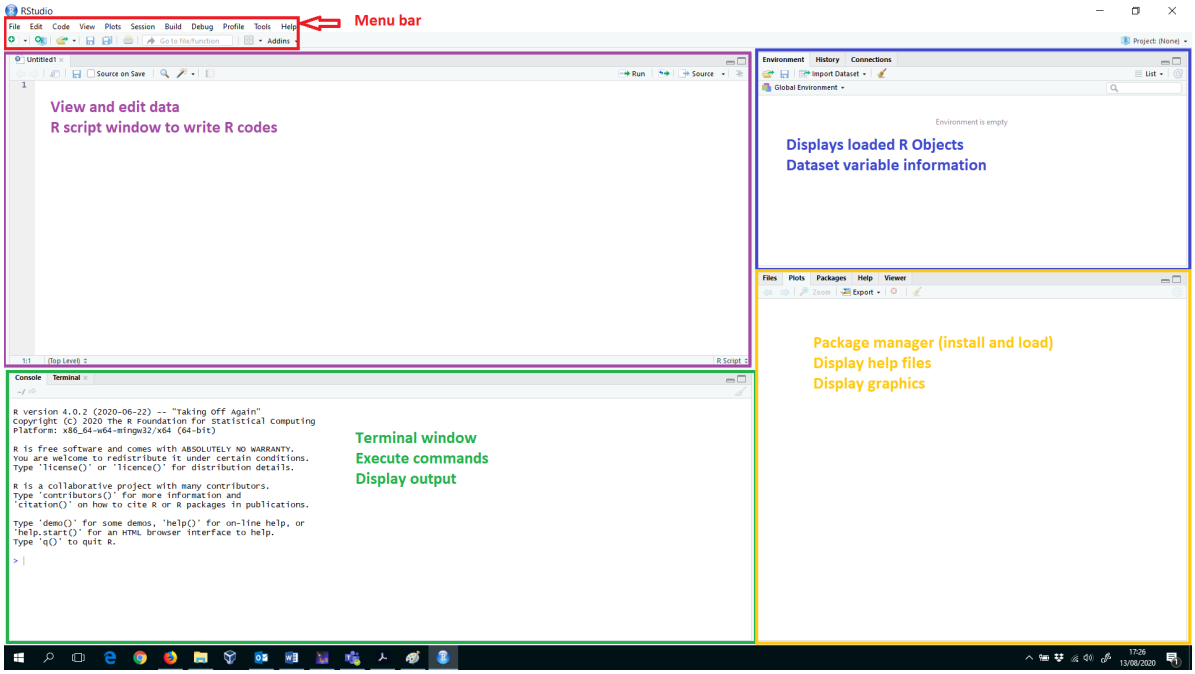


Figure 1.1: RStudio windows and their functions

1.2 File Organisation

- Create a folder for this module. This folder should include all module material you download from Aula or other platforms. Group files in sub-folders in a way that you can locate them easily. So for example, `6036ECN-Further-Econometrics` may be the name of the folder and then you may have sub-folders such as `Lecture-Slides`, `R-workshops`, etc.
- You should have one folder for R-workshops. I recommend naming this folder as `R-workshops` and within that folder, create sub-folders as we progress in the module.
- Please note that my `R-workshops` folder is located on my `desktop`. Hence, I will refer to the folder as `~/Desktop/R-workshops`. You will need to modify this depending on where you locate your files.
- If you are using the computers in the lab, it may be best if you create a folder on your OneDrive account as you can easily access this at home and on-campus.
- Before working on the data, set your working directory. R will save all files in there and, if you want to open a dataset, R will also look in there first. Select the folder you have created for R workshops.

- Use `setwd(the_address_you_would_like_to_locate_your_work)` in the console to choose your work directory. You may alternatively do this through the menu:

Session –> Set Working Directory –> Choose Directory

You will see the console printing this action, which may help you to remember how to use the console next time.

- If you are unsure of in which folder your work is, type `getwd()` in the console and R will print the current location you are at.

1.3 Getting Help

If you should ever struggle with some of R's commands, a look into R's help-files can be very helpful. To access the help file, you have to type into the console window `?` and then the command name. For example, if you want to know more about the command `getwd()`, type the following:

```
?getwd()
```

2 Basics of R

2.1 Using R as a calculator

You may use R as a calculator. Some examples are given below.

```
# Addition
5 + 4
```

```
[1] 9
```

```
# Subtraction
5 - 4
```

```
[1] 1
```

```
# Multiplication
3 * 6
```

```
[1] 18
```

```
# Division
10 / 2
```

```
[1] 5
```

```
# Exponents
2^3
```

```
[1] 8
```

```
# Modulo
5 %% 2
```

```
[1] 1
```

2.1.1 Basic Operators

Operator	Description
Arithmetic	
+	Addition
-	Subtraction
*	Multiplication
/	Division
^ or **	Exponential
%%	Modulus
% / %	Integer Division
Logic	
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Exactly equal to
!=	Not equal to
!x	Not x
x | y	x OR y
x & y	x AND y

2.1.2 Order of operators

- Parenthesis
- Multiplication / division
- Addition / subtraction
- Multiplication has the same importance as division. Similarly, addition and subtraction are at the same level. When we need to decide between the two, we apply the operation that shows first from the left to the right.
- Use of parentheses makes it easier to perform the correct operation

- Can you guess the result of the following operation?

  $-$ 8 / 2 * (2 + 2)

```
8 / 2 * ( 2 + 2)
```

```
[1] 16
```

```
8 / 2 * 2 + 2
```

```
[1] 10
```

```
100 * 2 + 50 / 2
```

```
[1] 225
```

```
(100 * 2) + (50 / 2)
```

```
[1] 225
```

2.2 Storing information in objects

R lets you save data by storing it inside an R `object`. An object is a name that you can use to call up stored data.

```
a <- 5

a
```

```
[1] 5
```

```
a + 2
```

```
[1] 7
```

In the example above, we store value of 5 under object `a`. We then call the value stored under `a` and sum it with 2.

Note the use of `<` together with `-` . This representation (`<-`) resembles a backward pointing arrow, and it assigns the value 2 to the object `a`.

```
b_vector <- 1:6
b_vector
```

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

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

In the above example, we create a vector, whose elements are numbers from 1 to 6 and store it under b_vector.

When you create an object, the object will appear in the environment pane of RStudio (on the top right-hand-side of the R screen). This pane will show you all of the objects you've created since opening RStudio.

2.2.1 Naming of objects

Note the following;

- An object name cannot start with a number (for example, `2var` or `2_var`)
- A name cannot use some special symbols, like `^`, `!`, `$`, `@`, `+`, `-`, `/`, or `*` . You may use `_`
- R is case-sensitive, so `name` and `Name` will refer to different objects
- R will overwrite any previous information stored in an object without asking your confirmation. So, be careful while making changes.
- You can see which object names you have already used by calling the function `ls`:

```
ls()
```

```
[1] "a"        "b_vector"
```

```
## [1] "a"        "b_vector"
```

2.2.2 Naming conventions

You may see the following styles for naming of variables:

- Camel case

Camel case variable naming is common in Javascipt. However, it is considered as bad practise in R. Try to avoid this kind of naming.

```
bankAccount = 100
```

- Use of dots

dot is used in variable names by many R users. However, try to avoid this too because base R uses dots in function names (`contrib.url()`) and class names (`data.frame`). Avoiding dot in your variable names will help you avoid confusion, particularly in the initial stages of your learning!

```
bank.account = 100
```

- Snake case

Use of snake case is considered to be good practice. Try to follow this approach.

```
bank_account = 100
```

Note that you may find different users of R having a preference towards different styles. The recommendations above are from the "Tidyverse style guide", which is available from https://style.tidyverse.org.

Start your variable names with a lower case and reserve the capital letter start for function names!

2.2.3 Removing objects

You will see that the `Environment` window can quickly get over-crowded while working interactively. You may remove the objects that you no longer need. by `rm(object_name )`

```
rm(a)
```

2.2.4 Example of using variables

Let us calculate the module mark for a student who got 65% from coursework and 53% from exam. The weights for the coursework and exam are, respectively, 25% and 75%.

```r
# let's calculate module mark for a student
coursework <- 65
exam <- 53
module_mark <- coursework * 0.25 + exam * 0.75

print(module_mark)
```

```
[1] 56
```

2.3 Datatypes in R

2.3.1 Numeric

Decimal numbers and integers are part of the numeric class in R.

2.3.1.1 Decimal (floating point values)

```r
decimal_number <- 2.2
```

2.3.1.2 Integer

```r
i <- 5
```

2.3.2 Logical

Boolean values (TRUE and FALSE) are part of the logical class in R. These are written in capital letters.

```r
t <- TRUE
f <- FALSE
```

```r
t
```

```
[1] TRUE
```

```
f
```

```
[1] FALSE
```

2.3.3 Characters

Text (string) values are known as characters in R. You may use single or double quotation to create a text (string).

```
message <- "hello all!"
print(message)
```

```
[1] "hello all!"
```

```
an_other_message <- 'how are you?'
print(an_other_message)
```

```
[1] "how are you?"
```

2.3.4 Checking data type classes

We can use the `class()` function to check the data type of a variable:

```
class(decimal_number)
```

```
[1] "numeric"
```

```
class(i)
```

```
[1] "numeric"
```

```
class(t)
```

```
[1] "logical"
```

```r
class(f)
```

```
[1] "logical"
```

```r
class(message)
```

```
[1] "character"
```

Part II

Seminar 2 (28 January 2025)

3 Data Management in R

3.1 Packages and libraries

In order to access specialised data analysis tools in R, we will need to install some R packages.

"An R **package** is a collection of functions, data, and documentation that extends the capabilities of base R. Using packages is key to the successful use of R." (Wickham, Cetinkaya-Rundel, and Grolemund, n.d.)

We will start by installing the `tidyverse` package

```
#install.packages("tidyverse")
```

To install `tidyverse`, type the above line of code in the console, and then press enter to run it. R will download the packages from CRAN and install them on to your computer.

Once installed, you may use this package after loading it with the `library()` function.

```
#library(tidyverse)
```

You see above a list of packages that come with `tidyverse`.

You may update `tidyverse` by running

```
#tidyverse_update()
```

3.2 Functions

You may identify functions with the `()` after the function name. For example, `ls()` that we used above.

Functions may also take *arguments*. The data that we pass into the function is called the function's *argument*. The argument can be raw data, an R object, or even the results of another R function.

```r
# round a number
round(4.5218)
```

```
[1] 5
```

```r
## 5

# calculate the factorial
factorial(3)
```

```
[1] 6
```

```r
## 6

# calculate the mean of values from 1 to 6:
mean(1:6)
```

```
[1] 3.5
```

```r
## 3.5

round(mean(1:6))
```

```
[1] 4
```

```r
## 4
```

Many R functions take multiple arguments that help them do their job. You can give a function as many arguments as you like as long as you separate each argument with a comma.

To see which arguments a function can take, you may type `args` in parenthesis after function name:

```r
args(round)
```

```
function (x, digits = 0, ...)
NULL
```

```
## function (x, digits = 0)
## NULL

round(3.1415, digits = 2)
```

```
[1] 3.14
```

```
## 3.14
```

3.2.1 Basic Functions

Function	Description
?() or **help()**	Access the documentation and help file for a particular function
install.packages()	Download and install an R package
library()	Loads an R package into the working environment
setwd()	Set the working directory
getwd()	Get the working directory
c()	Create a vector
as.numeric()	Converts an object to a numeric vector
as.logical()	Converts an object to a logical vector
as.character()	Converts and object to a character vector
mode()	Returns the type of the object
sum()	Returns the sum of all input values
length()	Returns the lenght of the obejct
mean()	Returns the arithmetic mean of the vector
median()	Returns the median of the vector
sample()	Returns a specificed size of elements from the object
replicate()	Repeats an expression a specific number of times
hist()	Creates a histogram of given data values

3.3 Scripts

You can create a draft of your code as you go by using an R *script*. An R script is just a plain text file that you save R code in. You can open an R script in RStudio using the menu bar:

File –> New File –> R Script

We will write and edit R code in a script. This will help create a reproducible record of your work. When you're finished for the day, you can save your script and then use it to rerun your entire analysis the next day.

To save a script, click the scripts pane, and then go to *File –> Save As* in the menu bar.

- You can automatically execute a line of code in a script by clicking the Run button on the top right of the pane. R will run whichever line of code your cursor is on.
- If you have a whole section highlighted, R will run the highlighted code.
- You can run the entire script by clicking the Source button.
- You can use Control + Return in your keyboard as a shortcut for the Run button. On Macs, that would be Command + Return.

4 Importing Data into R

4.1 Example 1: Crime data

The example and instructions provided in this section is taken from (Riegler 2022).

The following exercise gives you a hands-on introduction to basic operations in R using a real-world data set. It begins with importing a MS-Excel data set into R and asks you to perform some basic operations to familiarise yourself with some of the commands that will be relevant for the coursework and in subsequent computer classes.

Please download the Excel data set called `crime.xls` from Aula and save it into a drive of your choice. This is a data set that contains crime levels and other socio-economic information on 46 cities across the US for the year 1982. The full version of the data set can be accessed at http://fmwww.bc.edu/ec-p/data/wooldridge/datasets.list.html. The variables are defined as follows:

Variable	Definition
pop	actual population in number
crimes	total number of crimes
unem	unemployment rate (%)
officers	number of police officers
pcinc	per capita income, $
area	land area, square miles
lawexpc	law enforcement expenditure per capita, $

From here on, you need to open a R script to save all your commands to be able to replicate your results:

4.1.1 Task 1

4.1.1.1 Task

Import the Excel data set into R.

4.1.1.2 Guidance

The native data format of R is .Rdata, however, you can also open other formats, such as .xlsx, .csv, etc. Non-native data formats have to be imported rather than just opened. Before we can we can import Excel spreadsheets directly into R, we have to activate a R-library first.

You can either use the package manager window (in the right bottom corner of the R screen) and tick the box next to the package name or you type the following into the terminal window (in the left bottom of the R screen)

```
library(readxl)
```

This line loads the necessary `readxl` library. But you will probably receive an error message when you run the above line. This is because we first need to install the `read_excel` package. (Note that you will need to type the below line without the pound (hashtag) sign at the beginning of the line).

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

There are two ways to import:

1. Through command line:

   ```
   crime <- read_excel("./assets/data/crime.xls")
   ```

   In the above line, we import the dataset with the `read_excel` function and store it under the name `crime`. Notice how the new `crime` data is added as an object in the R environment.

2. Through menu:

   File –> Import Dataset –> From Excel

Don't forget to tick the "First Row as Names" box if it is not ticked!

4.1.2 Task 2

4.1.2.1 Task

View the dataset in R's data viewer.

4.1.2.2 Guidance

To open the data viewer, use the `View` function.

```
# View(crime)
```

Note that the first letter of `View` is capitalised.

4.1.3 Task 3

4.1.3.1 Task

View the first few (six) entries of the crime data to get a feeling of what the values look like.

4.1.3.2 Guidance

Use the `head` function

```
head(crime)
```

```
# A tibble: 6 x 7
     pop crimes  unem officers pcinc  area lawexpc
   <dbl>  <dbl> <dbl>    <dbl> <dbl> <dbl>   <dbl>
1 229528  17136  8.20      326  8532  44.6    851.
2 814054  75654  8.10     1621  7551 375      875.
3 374974  31352  9         633  8343  49.8   1122.
4 176496  15698 12.6       245  7592  74      744.
5 288446  31202 12.6       504  7558  97.3    974.
6 122768  16806 13.9       186  6411  55.3    762.
```

4.1.4 Task 4

4.1.4.1 Task

Label the variables using the definitions given above.

4.1.4.2 Guidance

You have to attach a variable label to each variable. There is already a library available which facilitates the allocation of labels to variables. First, we need to install the package!

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

```
Loading required package: maditr
```

```
To modify variables or add new variables:
             let(mtcars, new_var = 42, new_var2 = new_var*hp) %>% head()
```

```
Use 'expss_output_rnotebook()' to display tables inside R Notebooks.
 To return to the console output, use 'expss_output_default()'.
```

```
crime <- apply_labels(crime,
                      pop = "actual population in number",
                      crimes = "total number of crimes",
                      unem = "unemployment rate (%)",
                      officers = "number of police officers",
                      pcinc = "per capita income, $",
                      area = "land area, square miles",
                      lawexpc = "law enforcement expenditure per capita, $"
                      )
```

4.1.5 Task 5

4.1.5.1 Task

Create a new variable which measures the population density for each city.

4.1.5.2 Guidance

To generate a new variable and add it to the existing crime dataset, we use the following command:

```
crime$popdens <- crime$pop / crime$area
```

You may wonder why we add `crime$` in front of every variable. The reason is that R can store more than one data frame, matrix, list, vector etc., at the same time, so the prefix `crime$` is necessary to avoid ambiguity and ensure that we are working with variables in the `crime` data. Think of `crime$` as an address where e.g. the variable `pop` stays. If you have loaded another data frame that contains a `pop` variable, R would know that we want to use the variable from the crime dataset and not from the other data frame. There are library packages that can facilitate the process, however, we will cover them later in the module.

Note that the newly created population density variable is now labelled as the original population variable (`pop`). Let's update the label with the method we introduced in Task 4. Note that we do not need to call the library again, as it is already called.

```
crime <- apply_labels(crime,
                      popdens = "population density: pop / area")
```

4.1.6 Task 6

4.1.6.1 Task

Sort the data with respect to the population density of each city.

4.1.6.2 Guidance

Sorting data is a useful action to get a general feeling for the data, e.g. are there any outliers in the dataset? Are there any unusual patterns?

To change the order of the rows in a data frame, we will apply the `order` function. We first rank all observations with respect to the population density and store this information in a vector called `rank`. The rank vector contains indices that we can use to sort the crime data frame. Below, we save the sorted data under a new name, `crime.popdens1`

```
rank <- order(crime$popdens)
crime.popdens1 <- crime[rank,]
```

Let's see the result (note. how the population density variable is now sorted from the smallest to the largest):

```
head(crime.popdens1)
```

```
# A tibble: 6 x 8
  pop        crimes     unem       officers   pcinc      area  lawexpc popdens
  <labelled> <labelled> <labelled> <labelled> <labelled> <lab> <label> <labelle>
1 425093     38195       4.7        767       7991       604.0  570.00  703.7964
2 268887     14537       5.5        400       7704       255.9  570.63 1050.7503
3 462657     34736      10.4        937       7585       352.0  582.56 1314.3665
4 451397     45503      10.4       1145       7480       316.4 1054.17 1426.6656
5 412661     47128       8.3        719       7336       258.5  554.70 1596.3675
6 173630     18915       8.7        366       7409       100.5  827.16 1727.6617
```

```
# you may alternatively use
# View(crime.popdens1)
```

This procedure sorts the data from the smallest to the largest value. To sort the data from the largest to the smallest number, we set the order argument decreasing to TRUE.

```
crime.popdens2 <- crime[order(crime$popdens, decreasing = TRUE),]
head(crime.popdens2)
```

```
# A tibble: 6 x 8
  pop        crimes     unem       officers   pcinc      area  lawexpc popdens
  <labelled> <labelled> <labelled> <labelled> <labelled> <lab> <label> <labelle>
1  708287     68598       8.4       1971       9265        46.4 1050.00 15264.806
2  334414     36172      15.4       1166       4525        24.1 1139.32 13876.099
3  365506     52901      12.3        979       6084        34.3  714.00 10656.152
4 1181868    152962      20.3       4092       6251       135.6 1483.52  8715.840
5  360493     28592      16.9       1034       5929        41.8  749.44  8624.235
6  158533     15233      11.3        408       6169        18.9  661.50  8387.990
```

Have you observed a slight difference in the way we sorted the data? We can save some time and space by merging the two steps into one line, however, it is sometimes easier to understand a command if it is split into separate stages.

4.1.7 Task 7

4.1.7.1 Task

What is the minimum and maximum value for population density in the crime data?

4.1.7.2 Guidance

The minimum and maximum values can be produced by generating standard descriptive statistics of the variables.

```
summary(crime$popdens)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  703.8  2797.1  4236.8  4967.5  7052.2 15264.8
```

Before you finish, save the dataset under a new name. Never overwrite your original data!

```
save(crime, file = "./assets/data/crime_v2.Rdata")
```

The above command tells R to use the crime dataset and save it as `crime_v2.Rdata`. Rdata is an R specific format. R can also save data in .csv format, that can be opened with any text editor or spreadsheet software:

```
write.csv(crime, file = "./assets/data/crime_v2.csv", row.names = TRUE)
```

Now you are ready to answer the following questions on your own:

4.1.8 Further Exercises

1. Find the minimum and maximum number of police officers in the data set.
2. Create a new variable which measures the crime rate per 1,000 of population.
3. Is the city with the highest number of police officers also the city with the highest crime density?
4. How many crimes occurred in the richest city?
5. Is the richest city also the one with the highest number of police officers?
6. What is the average unemployment rate across these 46 U.S. cities?
7. Does the city with the highest unemployment rate also have the highest crime level?

5 Introduction to Regression Analysis

5.1 Example 1: Crime data

The example and instructions provided in this section is taken from (Riegler 2022).

Suppose you are examining the relationship between number of crimes and number of police officers. Below, we will generate descriptive statistics, create a scatter plot and see how we estimate OLS regression.

We will use the crime data set, which is already saved in `Rdata` format.

5.1.1 Task 1

Open `crime_v2.Rdata` (if it not already open). You may so this trough the menu or the command line using the `load()` function:

```
load("~/Desktop/R-workshops/assets/data/crime_v2.Rdata")
```

5.1.2 Task 2

Check the summary statistics for `crimes` and `officers` variables.

```
summary(crime[, c("crimes", "officers")])
```

```
     crimes          officers     
 Min.   :  5276   Min.   : 109.0  
 1st Qu.: 19658   1st Qu.: 402.8  
 Median : 32518   Median : 694.5  
 Mean   : 38123   Mean   : 902.1  
 3rd Qu.: 49434   3rd Qu.:1212.0  
 Max.   :152962   Max.   :4092.0  
```

```r
# Standard deviation for the 'crimes' variable
sd_crimes <- sd(crime$crimes, na.rm = TRUE)

# Standard deviation for the 'officers' variable
sd_officers <- sd(crime$officers, na.rm = TRUE)

# Print the results
sd_crimes
```

```
[1] 27660.3
```

```r
sd_officers
```

```
[1] 721.7255
```

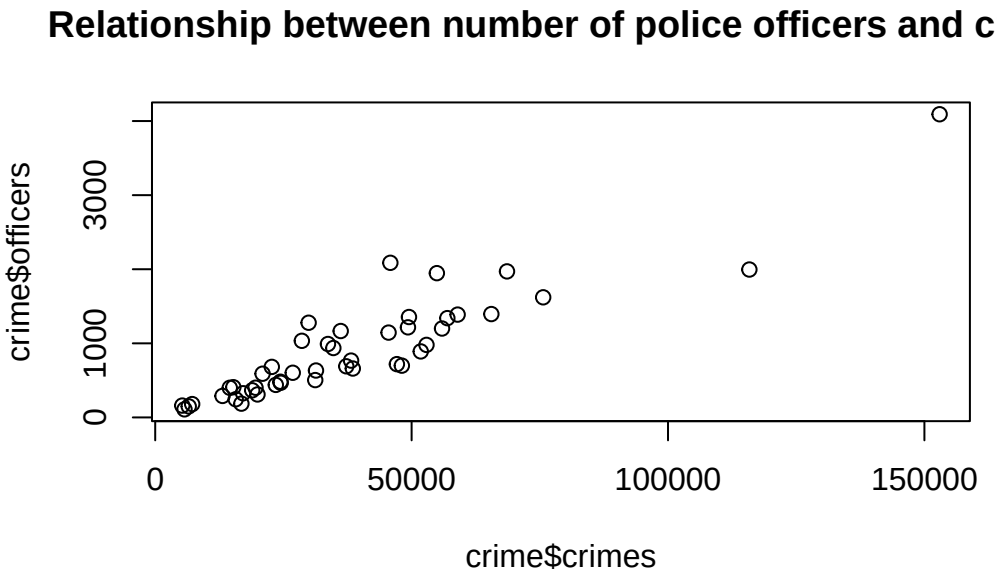
Note the `na.rm = TRUE` above. This argument ensures that nay `NA` values are removed before the calculation.

5.1.3 Task 3

In addition to checking summary statistics, it is always wise to visualise your data before getting into more complicated modelling.

For this task, generate a scatter plot with number of crimes on the y-axis and the number of police officers on the x-axis.

```r
plot(crime$officers~crime$crimes,
     main = "Relationship between number of police officers and crime")
```



5.1.4 Task 4

Calculate the Covariance and the Correlation Coefficient between number of crimes and number of police officers. Comment on their values.

5.1.4.1 Guidance

A scatter plot is a good start for identifying relationships between two variables, but it is not sufficient to identify accurately how strong the relationship is between `crimes` and `officers`. There are two numerical statistics, that provide information about the relationship between two variables: The Covariance and the Correlation Coefficient.

To produce a Covariance matrix, use the following command:

```
cov(crime$officers, crime$crimes)
```

```
[1] 18212436
```

The result is: 18,212,436! This number may appear to be too large but the value we obtain as covariance depends on the measuremetn levels of the variables. This measure (i.e. the covariance) does not provide any information on how strong this relationship between `crimes` and `officers` is. It only reveals that there is a positive relationship between the number of police officers and the number of crimes committed.

Instead of using the covariance, we can use a *standardised covariance* - the *correlation coefficient*. To calculate the correlation matrix, we only have to adjust slightly the covariance command.

```
cor(crime$officers, crime$crimes)
```

```
[1] 0.9123032
```

The correlation coefficient between number of officers and crimes is 0.91. We conclude that there is a strong positive relationship between our two variables.

5.1.5 Task 5

Regress the number of police officers on crimes and comment on:

- the sign and magnitude of the regression coefficients
- the goodness of fit of the estimated model.

```
lm(officers ~ crimes, data = crime)
```

```
Call:
lm(formula = officers ~ crimes, data = crime)

Coefficients:
(Intercept)       crimes
    -5.4183       0.0238
```

We can save this estimation under an object (please note that we use `model_1` below, but you may give any name as long as it satisfies the naming conventions):

```
model_1 <- lm(officers ~ crimes, data = crime)
# display the model
model_1
```

```
Call:
lm(formula = officers ~ crimes, data = crime)

Coefficients:
(Intercept)       crimes
    -5.4183       0.0238
```

Although we see the estimated coefficients in the above output we do not have information about the other statistics that we need to proceed. We use the `summary()` function below:

```
summary(model_1)
```

```
Call:
lm(formula = officers ~ crimes, data = crime)

Residuals:
    Min      1Q  Median      3Q     Max
-756.64 -153.71  -25.75   89.64 1000.97

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.418291  75.587257  -0.072    0.943
crimes       0.023804   0.001611  14.777   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 298.9 on 44 degrees of freedom
Multiple R-squared:  0.8323,    Adjusted R-squared:  0.8285
F-statistic: 218.4 on 1 and 44 DF,  p-value: < 2.2e-16
```

The intercept term is not statistically significant.

The `crimes` variable is statistically significant at 0.1%. (Note the significance codes in the output).

The slope coefficient states that for every additional crime, we observe on average of 0.024 more police officers. Using more reader-friendly numbers, we can also infer that for every 1,000 additional crimes committed within a city, 24 more police officers are employed. Note how the latter way of phrasing makes more sense.

R^2 is the measure that provides information on the overall goodness of fit of the model. In this case it is 0.83. This means that 83% of the variation in police officers can be explained with the variation in number of crimes committed. Our estimated model has a good degree of explanatory power.

Looking at the F-statistic (218.4 with a p-value of almost zero), we can conclude that the model, overall, is statistically significant.

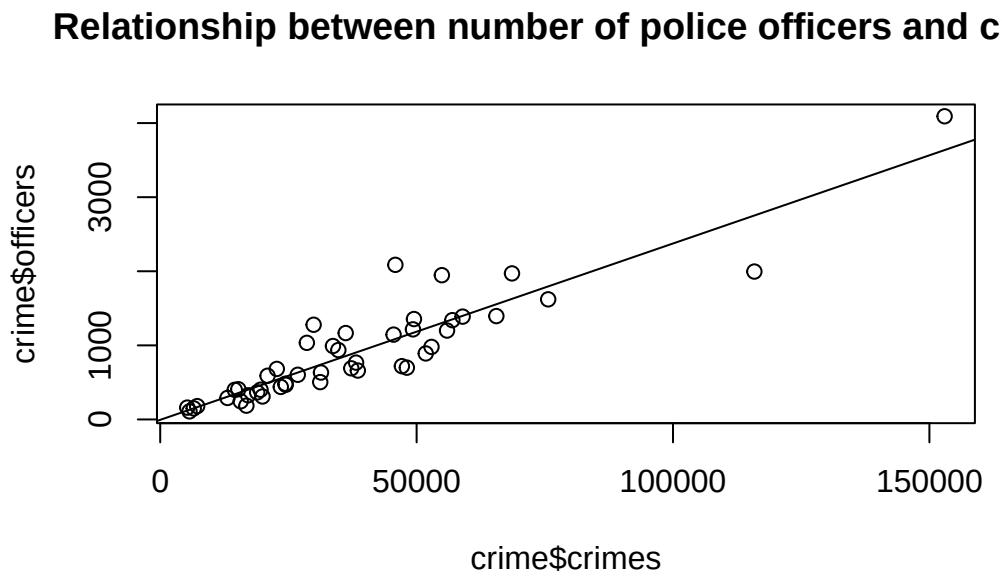
5.1.6 Task 6

Add a regression line to the scatter plot you created in Task 3.

5.1.6.1 Guidance

To add a regression line to the plot, we have to use the previously saved regression object `model_1` and add it to the previous scatter plot.

```
plot(crime$officers~crime$crimes,
     main = "Relationship between number of police officers and crime")
abline(model_1)
```



5.2 Example 2: Wage data

5.2.1 Task 1

5.2.1.1 Task

Import `wage.xls` data into R and view the first few rows of the data to have an idea about the contents of the variables, and then save the data in R format.

5.2.1.2 Guidance

Use `read_excel()` and `head()` functions.

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

# Import Excel data
wage2 <- read_excel("./assets/data/wage2.xls", sheet = "wage2")
```

```
head(wage2)
```

```
# A tibble: 6 x 15
   wage hours    IQ   KWW  educ exper tenure   age married south urban  sibs
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl> <dbl>   <dbl> <dbl> <dbl> <dbl>
1   769    40    93    35    12    11      2    31       1     0     1     1
2   808    50   119    41    18    11     16    37       1     0     1     1
3   825    40   108    46    14    11      9    33       1     0     1     1
4   650    40    96    32    12    13      7    32       1     0     1     4
5   562    40    74    27    11    14      5    34       1     0     1    10
6  1400    40   116    43    16    14      2    35       1     0     1     1
# i 3 more variables: brthord <dbl>, meduc <dbl>, feduc <dbl>
```

```
# Save data in R format
save(wage2, file = "./assets/data/wage2.Rdata")
```

5.2.2 Task 2

5.2.2.1 Task

Label variable `educ` as "years of schooling" and `exper` as "years of experience".

5.2.2.2 Guidance

We will need the `expss` package to label the variables. The installation and calling of the package is deactivated below since we already have done these steps above. After running the below command check the changes in the data from the Environment window on the top-right.

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

```
Loading required package: maditr
```

```
To aggregate all non-grouping columns: take_all(mtcars, mean, by = am)
```

```
Use 'expss_output_viewer()' to display tables in the RStudio Viewer.
 To return to the console output, use 'expss_output_default()'.
```

```
wage2 <- apply_labels(wage2,
                      educ = "years of schooling",
                      exper = "years of experince")
```

5.2.3 Task 3

5.2.3.1 Task

Generate two new variables: hourly wage and logarithmic wage.

5.2.3.2 Guidance

```
# Generate new variables
wage2$hourly_wage <- wage2$wage / wage2$hours
wage2$ln_wage <- log(wage2$wage)
```

5.2.4 Task 4

5.2.4.1 Task

Check the summary statistics for (i) the `wage` variable, (ii) for all variables.

5.2.4.2 Guidance

We will use the `summary()` function.

```
# Summary statistics for the wage variable only
summary(wage2$wage)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  115.0   669.0   905.0   957.9  1160.0  3078.0
```

```
# Summary statistics for all variables in wage2 data
summary(wage2)
```

```
      wage            hours             IQ            KWW
 Min.   : 115.0   Min.   :20.00   Min.   : 50.0   Min.   :12.00
 1st Qu.: 669.0   1st Qu.:40.00   1st Qu.: 92.0   1st Qu.:31.00
 Median : 905.0   Median :40.00   Median :102.0   Median :37.00
 Mean   : 957.9   Mean   :43.93   Mean   :101.3   Mean   :35.74
 3rd Qu.:1160.0   3rd Qu.:48.00   3rd Qu.:112.0   3rd Qu.:41.00
 Max.   :3078.0   Max.   :80.00   Max.   :145.0   Max.   :56.00

      educ           exper          tenure           age
 Min.   : 9.00   Min.   : 1.00   Min.   : 0.000   Min.   :28.00
 1st Qu.:12.00   1st Qu.: 8.00   1st Qu.: 3.000   1st Qu.:30.00
 Median :12.00   Median :11.00   Median : 7.000   Median :33.00
 Mean   :13.47   Mean   :11.56   Mean   : 7.234   Mean   :33.08
 3rd Qu.:16.00   3rd Qu.:15.00   3rd Qu.:11.000   3rd Qu.:36.00
 Max.   :18.00   Max.   :23.00   Max.   :22.000   Max.   :38.00

   married          south           urban            sibs
 Min.   :0.000   Min.   :0.0000   Min.   :0.0000   Min.   : 0.000
 1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 1.000
 Median :1.000   Median :0.0000   Median :1.0000   Median : 2.000
 Mean   :0.893   Mean   :0.3412   Mean   :0.7176   Mean   : 2.941
```

```
3rd Qu.:1.000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.: 4.000
Max.   :1.000   Max.   :1.0000   Max.   :1.0000   Max.   :14.000

   brthord          meduc           feduc        hourly_wage
Min.   : 1.000   Min.   : 0.00   Min.   : 0.00   Min.   :  2.30
1st Qu.: 1.000   1st Qu.: 8.00   1st Qu.: 8.00   1st Qu.: 15.07
Median : 2.000   Median :12.00   Median :10.00   Median : 21.02
Mean   : 2.277   Mean   :10.68   Mean   :10.22   Mean   : 22.32
3rd Qu.: 3.000   3rd Qu.:12.00   3rd Qu.:12.00   3rd Qu.: 27.70
Max.   :10.000   Max.   :18.00   Max.   :18.00   Max.   :102.60
NA's   :83       NA's   :78      NA's   :194
   ln_wage
Min.   :4.745
1st Qu.:6.506
Median :6.808
Mean   :6.779
3rd Qu.:7.056
Max.   :8.032
```

5.2.5 Task 5

5.2.5.1 Task

Calculate the correlation coefficient between wage and education.

5.2.5.2 Guidance

We can calculate the correlation coefficients using the `cor()` function. In the first example below, the correlation coefficient is reported as a single number, while in the second example, we get a correlation matrix.

In most of empirical work, we are usually interested with pairwise correlations among all variables. Hence, we may use the correlation matrix to check the binary correlations among all variables in our sample. This is provided in the third example below.

The `"use = complete.obs"` added to the commands below asks R to handle missing values by casewise deletion.

```
# Correlation
cor(wage2$wage, wage2$educ)
```

```
[1] 0.3271087
```

```r
# Correlation
cor(wage2[, c("wage", "educ")], use = "complete.obs")
```

```
          wage      educ
wage 1.0000000 0.3271087
educ 0.3271087 1.0000000
```

```r
cor(wage2, use = "pairwise.complete.obs")
```

```
                  wage        hours           IQ          KWW        educ
wage         1.000000000 -0.009504302  0.30908783  0.32613058  0.32710869
hours       -0.009504302  1.000000000  0.07383930  0.11388938  0.09100889
IQ           0.309087827  0.073839301  1.00000000  0.41351552  0.51569701
KWW          0.326130577  0.113889381  0.41351552  1.00000000  0.38813424
educ         0.327108690  0.091008888  0.51569701  0.38813424  1.00000000
exper        0.002189702 -0.062126227 -0.22491253  0.01745245 -0.45557312
tenure       0.128266391 -0.055528006  0.04215883  0.14139800 -0.03616655
age          0.156701761  0.024811636 -0.04374091  0.39305297 -0.01225396
married      0.136582670  0.032563350 -0.01466753  0.08994782 -0.05856602
south       -0.159387287 -0.029519177 -0.20978466 -0.09439242 -0.09703298
urban        0.198406472  0.016573046  0.03893553  0.09819025  0.07215091
sibs        -0.159203728 -0.049602555 -0.28477277 -0.28497534 -0.23928810
brthord     -0.145485385 -0.043129582 -0.17943947 -0.15358472 -0.20499246
meduc        0.214831839  0.076619806  0.33180383  0.24079168  0.36423913
feduc        0.237586922  0.063172297  0.34390758  0.23488927  0.42692545
hourly_wage  0.931240501 -0.317645466  0.26502635  0.26059936  0.27167136
ln_wage      0.953141156 -0.047219079  0.31478770  0.30627128  0.31211665
                   exper      tenure          age      married       south
wage         0.002189702  0.12826639  0.156701761  0.136582670 -0.15938729
hours       -0.062126227 -0.05552801  0.024811636  0.032563350 -0.02951918
IQ          -0.224912532  0.04215883 -0.043740911 -0.014667528 -0.20978466
KWW          0.017452446  0.14139800  0.393052967  0.089947816 -0.09439242
educ        -0.455573115 -0.03616655 -0.012253956 -0.058566019 -0.09703298
exper        1.000000000  0.24365440  0.495329763  0.106349115  0.02125724
tenure       0.243654402  1.00000000  0.270601647  0.072605374 -0.06169141
age          0.495329763  0.27060165  1.000000000  0.106980249 -0.02947768
married      0.106349115  0.07260537  0.106980249  1.000000000  0.02275672
south        0.021257241 -0.06169141 -0.029477681  0.022756718  1.00000000
urban       -0.047385845 -0.03848582 -0.006749288 -0.040248179 -0.10989797
```

```
sibs         0.064310470 -0.03916116 -0.040719238 -0.004327422  0.06631979
brthord      0.088300019 -0.02847775  0.005435916 -0.014737189  0.09370679
meduc       -0.186317286 -0.01496769 -0.029319099 -0.022763437 -0.15787359
feduc       -0.256792630 -0.05924123 -0.071303285 -0.020324390 -0.17236334
hourly_wage  0.017757793  0.13541822  0.126683019  0.115115701 -0.14716118
ln_wage      0.020601158  0.18585262  0.161822314  0.149975894 -0.19481092
                   urban         sibs      brthord       meduc       feduc
wage         0.198406472 -0.159203728 -0.145485385  0.21483184  0.23758692
hours        0.016573046 -0.049602555 -0.043129582  0.07661981  0.06317230
IQ           0.038935525 -0.284772765 -0.179439471  0.33180383  0.34390758
KWW          0.098190247 -0.284975345 -0.153584717  0.24079168  0.23488927
educ         0.072150908 -0.239288104 -0.204992462  0.36423913  0.42692545
exper       -0.047385845  0.064310470  0.088300019 -0.18631729 -0.25679263
tenure      -0.038485824 -0.039161158 -0.028477749 -0.01496769 -0.05924123
age         -0.006749288 -0.040719238  0.005435916 -0.02931910 -0.07130328
married     -0.040248179 -0.004327422 -0.014737189 -0.02276344 -0.02032439
south       -0.109897970  0.066319792  0.093706790 -0.15787359 -0.17236334
urban        1.000000000 -0.031468824  0.002419787  0.03402366  0.11223944
sibs        -0.031468824  1.000000000  0.593913799 -0.28715120 -0.23202649
brthord      0.002419787  0.593913799  1.000000000 -0.27593376 -0.23037060
meduc        0.034023660 -0.287151198 -0.275933760  1.00000000  0.57649476
feduc        0.112239438 -0.232026494 -0.230370600  0.57649476  1.00000000
hourly_wage  0.189240304 -0.131364072 -0.120293460  0.18348733  0.20469678
ln_wage      0.203797585 -0.152809172 -0.141852712  0.21357476  0.22338514
             hourly_wage     ln_wage
wage         0.93124050  0.95314116
hours       -0.31764547 -0.04721908
IQ           0.26502635  0.31478770
KWW          0.26059936  0.30627128
educ         0.27167136  0.31211665
exper        0.01775779  0.02060116
tenure       0.13541822  0.18585262
age          0.12668302  0.16182231
married      0.11511570  0.14997589
south       -0.14716118 -0.19481092
urban        0.18924030  0.20379758
sibs        -0.13136407 -0.15280917
brthord     -0.12029346 -0.14185271
meduc        0.18348733  0.21357476
feduc        0.20469678  0.22338514
hourly_wage  1.00000000  0.89974921
ln_wage      0.89974921  1.00000000
```

The above table is informative but the reported numbers have far too many decimals. It is distracting our focus. Below, we round these in two decimal points, which is enough to have an idea about the strength of the correlation between our variables

```
# Calculate pairwise correlations and store them under name cor_matrix
cor_matrix <- cor(wage2, use = "pairwise.complete.obs")

# Round the correlation values to 2 decimal places and save them under the name rounded_cor_m
rounded_cor_matrix <- round(cor_matrix, 2)

# Display the rounded correlation matrix
print(rounded_cor_matrix)
```

```
             wage hours    IQ   KWW  educ exper tenure   age married south
wage         1.00 -0.01  0.31  0.33  0.33  0.00   0.13  0.16    0.14 -0.16
hours       -0.01  1.00  0.07  0.11  0.09 -0.06  -0.06  0.02    0.03 -0.03
IQ           0.31  0.07  1.00  0.41  0.52 -0.22   0.04 -0.04   -0.01 -0.21
KWW          0.33  0.11  0.41  1.00  0.39  0.02   0.14  0.39    0.09 -0.09
educ         0.33  0.09  0.52  0.39  1.00 -0.46  -0.04 -0.01   -0.06 -0.10
exper        0.00 -0.06 -0.22  0.02 -0.46  1.00   0.24  0.50    0.11  0.02
tenure       0.13 -0.06  0.04  0.14 -0.04  0.24   1.00  0.27    0.07 -0.06
age          0.16  0.02 -0.04  0.39 -0.01  0.50   0.27  1.00    0.11 -0.03
married      0.14  0.03 -0.01  0.09 -0.06  0.11   0.07  0.11    1.00  0.02
south       -0.16 -0.03 -0.21 -0.09 -0.10  0.02  -0.06 -0.03    0.02  1.00
urban        0.20  0.02  0.04  0.10  0.07 -0.05  -0.04 -0.01   -0.04 -0.11
sibs        -0.16 -0.05 -0.28 -0.28 -0.24  0.06  -0.04 -0.04    0.00  0.07
brthord     -0.15 -0.04 -0.18 -0.15 -0.20  0.09  -0.03  0.01   -0.01  0.09
meduc        0.21  0.08  0.33  0.24  0.36 -0.19  -0.01 -0.03   -0.02 -0.16
feduc        0.24  0.06  0.34  0.23  0.43 -0.26  -0.06 -0.07   -0.02 -0.17
hourly_wage  0.93 -0.32  0.27  0.26  0.27  0.02   0.14  0.13    0.12 -0.15
ln_wage      0.95 -0.05  0.31  0.31  0.31  0.02   0.19  0.16    0.15 -0.19
             urban  sibs brthord meduc feduc hourly_wage ln_wage
wage          0.20 -0.16   -0.15  0.21  0.24        0.93    0.95
hours         0.02 -0.05   -0.04  0.08  0.06       -0.32   -0.05
IQ            0.04 -0.28   -0.18  0.33  0.34        0.27    0.31
KWW           0.10 -0.28   -0.15  0.24  0.23        0.26    0.31
educ          0.07 -0.24   -0.20  0.36  0.43        0.27    0.31
exper        -0.05  0.06    0.09 -0.19 -0.26        0.02    0.02
tenure       -0.04 -0.04   -0.03 -0.01 -0.06        0.14    0.19
age          -0.01 -0.04    0.01 -0.03 -0.07        0.13    0.16
married      -0.04  0.00   -0.01 -0.02 -0.02        0.12    0.15
south        -0.11  0.07    0.09 -0.16 -0.17       -0.15   -0.19
```

```
urban        1.00 -0.03    0.00  0.03  0.11        0.19    0.20
sibs        -0.03  1.00    0.59 -0.29 -0.23       -0.13   -0.15
brthord      0.00  0.59    1.00 -0.28 -0.23       -0.12   -0.14
meduc        0.03 -0.29   -0.28  1.00  0.58        0.18    0.21
feduc        0.11 -0.23   -0.23  0.58  1.00        0.20    0.22
hourly_wage  0.19 -0.13   -0.12  0.18  0.20        1.00    0.90
ln_wage      0.20 -0.15   -0.14  0.21  0.22        0.90    1.00
```

5.2.6 Task 6

5.2.6.1 Task

Examine the relationship between education and wage using a scatter plot.

5.2.6.2 Guidance

We use the `ggplot2` package to draw plots. First install the package and call the library.

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

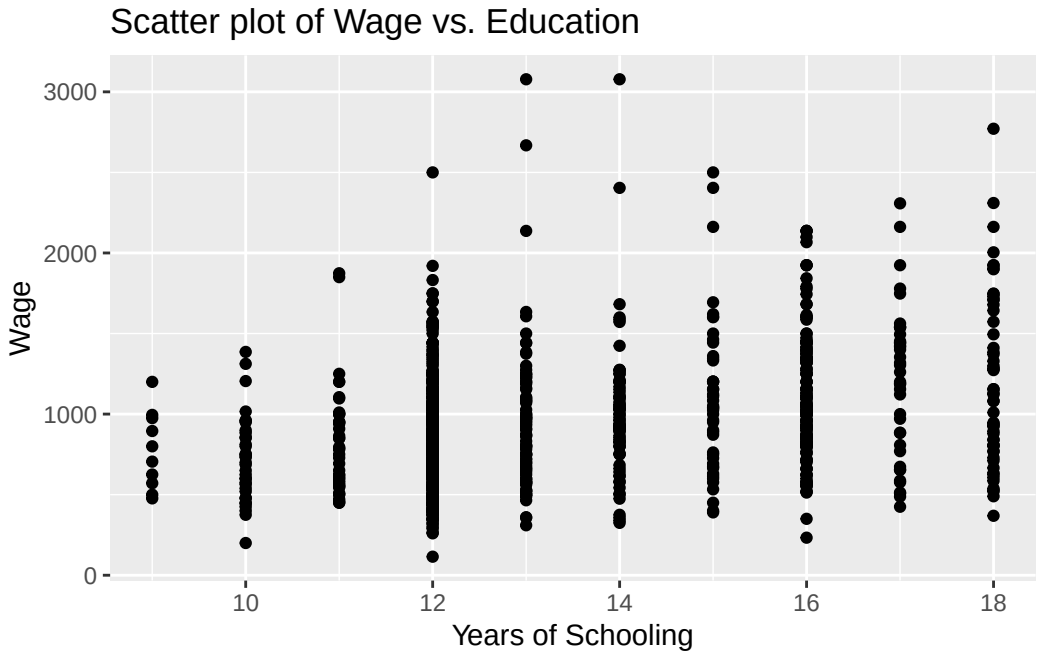
```
Attaching package: 'ggplot2'
```

```
The following object is masked from 'package:expss':
```

```
    vars
```

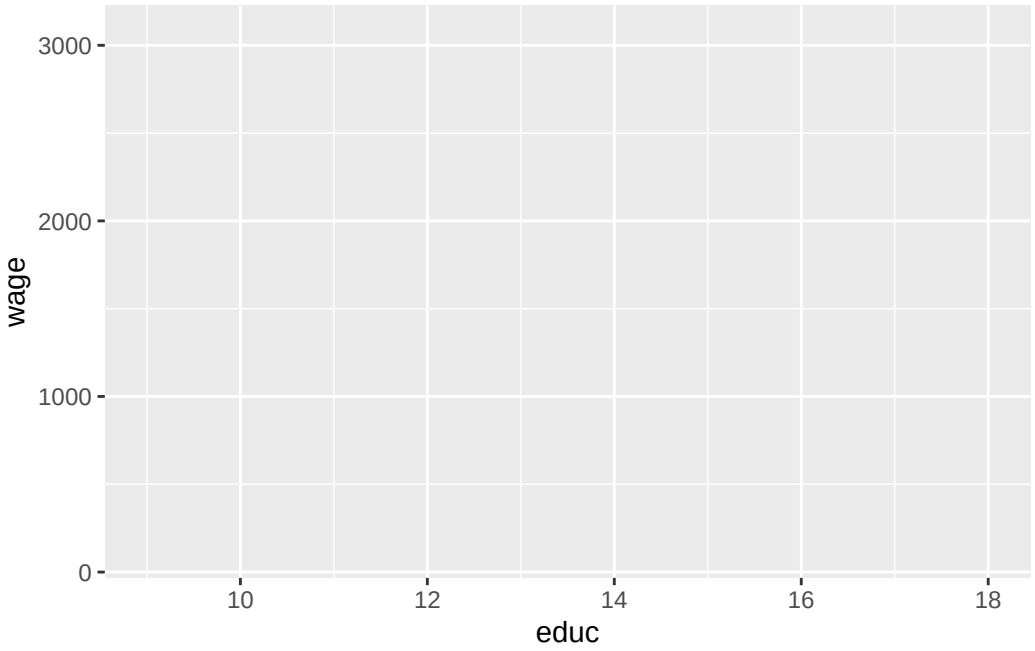
Education is expected to have a positive impact on wage. In our scatter plot, `educ` will be on the horizontal-axis while `wage` will be on the vertical-axis.

```
# Scatter plot
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  labs(title = "Scatter plot of Wage vs. Education", x = "Years of Schooling", y = "Wage")
```



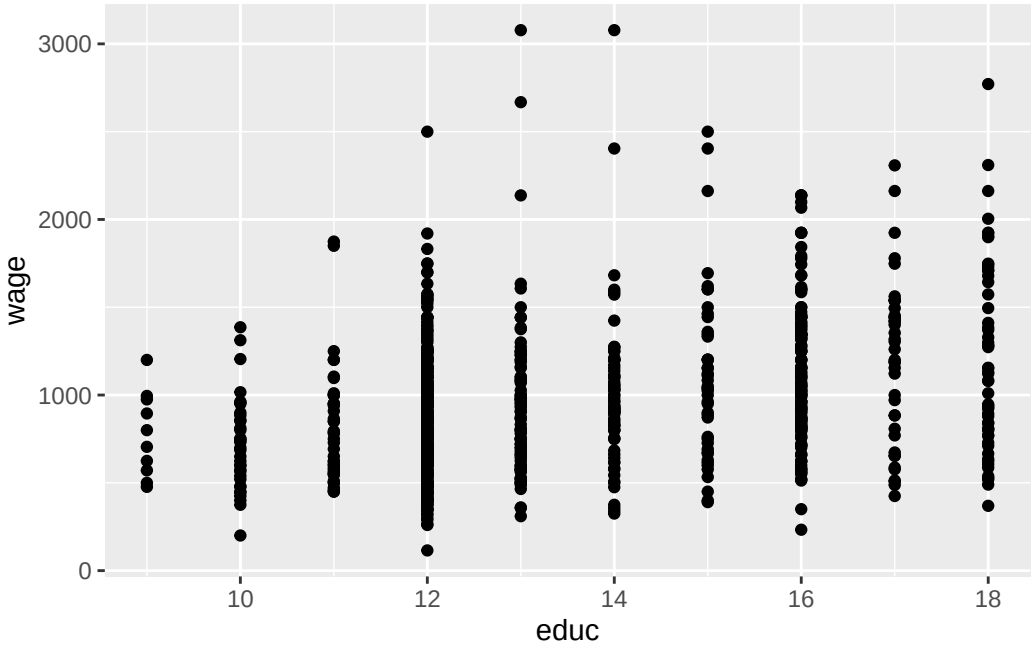
You see above the full set of lines to create this plot. But let us do this step by step to have a better understanding. First, we bring the `educ` and `wage` variables from the `wage2` data and position these on our plot.

```
ggplot(wage2, aes(x = educ, y = wage))
```



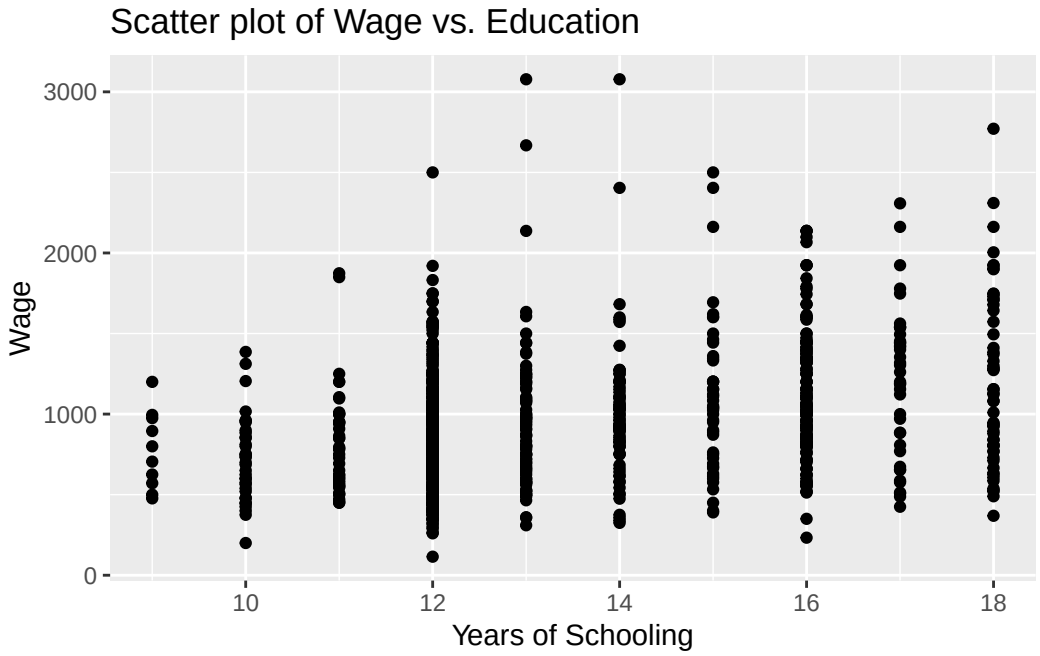
We then add (using the + sign), the observations in our data, represented by dots.

```
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point()
```



It is always good practice to give a title for your plot. Notice also that the horizontal and vertical axes above are labelled by the variable names. We may also replace these with proper definitions of the variables. This is to make it easier for the readers to understand your plots:

```
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  labs(title = "Scatter plot of Wage vs. Education", x = "Years of Schooling", y = "Wage")
```



5.2.7 Task 7

5.2.7.1 Task

Tabulate the `urban` variable to see the distribution of observations in rural and urban areas

5.2.7.2 Guidance

We use the `table()` function for that purpose.

```
table(wage2$urban)
```

```
  0   1
264 671
```

5.2.8 Task 8

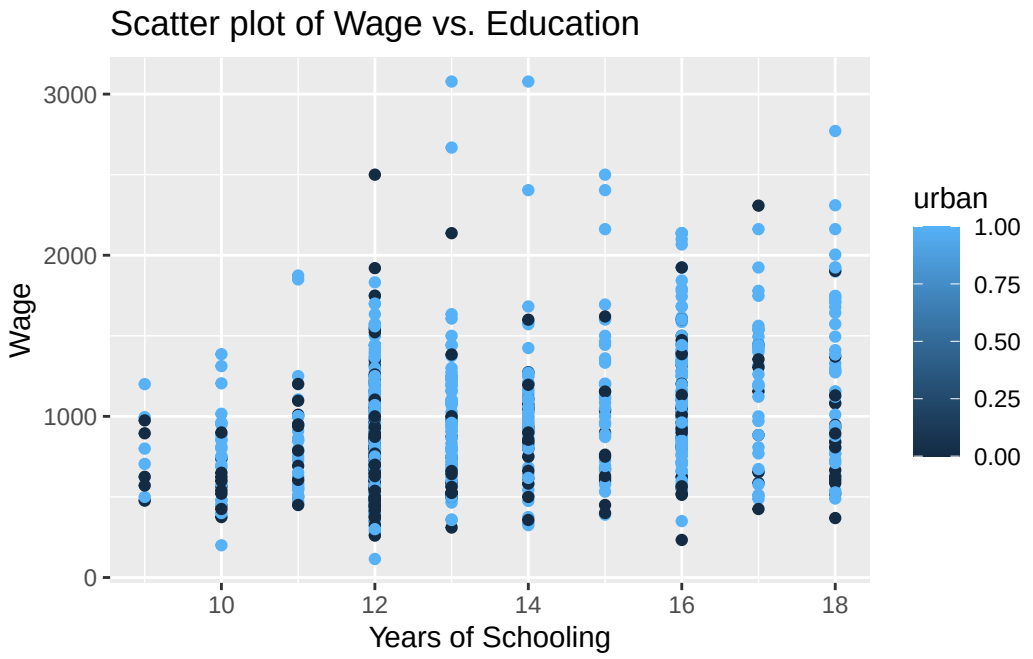
5.2.8.1 Task

Let's say we are interested to plot the education-wage relationship differentiating between people in rural and urban areas. Replicate the scatter plot above, but this time, using different colors for rural and urban.

5.2.8.2 Guidance

Notice how we add the `color = urban` option below. We do the same for the label too.

```
# Scatter plot - colored by urban
ggplot(wage2, aes(x = educ, y = wage, color = urban)) +
  geom_point() +
  labs(title = "Scatter plot of Wage vs. Education", x = "Years of Schooling", y = "Wage", co
```



The labelling of the above plot looks as if we have a range of values for the urban variable, changing from zero to one. The urban variable, in fact, is a dummy, taking two values only: zero for rural and one for urban residence. If you look into this variable entry in more detail, you will see that it is stored as `num`. We can change this using the `factor()` function. Instead of overriding the urban variable, let's create a new variable `urban_residence` to see a comparison.

```
wage2$urban_residence <- factor(wage2$urban, levels = c(0,1), labels = c("rural", "urban"))
```

Below, we view the two variables using R's `dplyr` package.

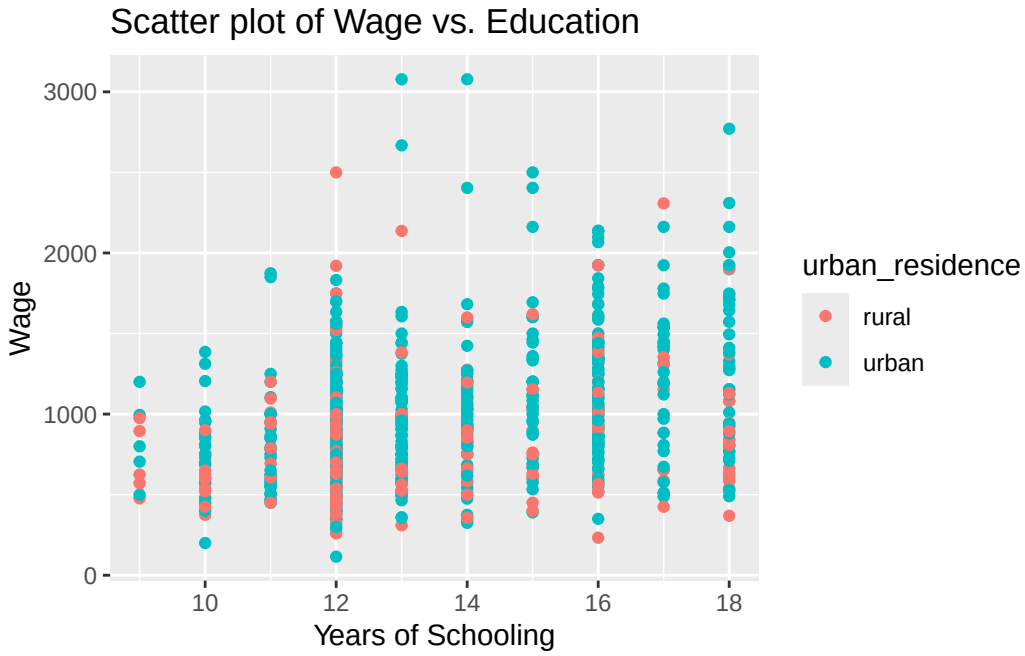
```
# install.packages("dplyr")
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:expss':

    compute, contains, na_if, recode, vars, where

The following objects are masked from 'package:maditr':

    between, coalesce, first, last

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
#View(select(wage2, urban, urban_residence))
```

Let's re-run our scatter plot code again (but replacing `urban` with `urban_residence`:

```
# Scatter plot - colored by urban
ggplot(wage2, aes(x = educ, y = wage, color = urban_residence)) +
  geom_point() +
  labs(title = "Scatter plot of Wage vs. Education", x = "Years of Schooling", y = "Wage", co
```



5.2.9 Task 9

5.2.9.1 Task

Estimate a regression model where wage is regressed on education. Interpret the results.

5.2.9.2 Guidance

We use the `lm()` function to estimate linear regression models. You may read `~` in `wage ~ educ` below as "approximately modelled as" James et al. (2023). We may also say "wage is regressed on education".

```
# Linear regression
model_1 <- lm(wage ~ educ, data = wage2)
summary(model_1)
```

```

Call:
lm(formula = wage ~ educ, data = wage2)

Residuals:
    Min      1Q  Median      3Q     Max
```

```
-877.38 -268.63  -38.38  207.05 2148.26

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  146.952     77.715   1.891   0.0589 .
educ          60.214      5.695  10.573   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 382.3 on 933 degrees of freedom
Multiple R-squared:  0.107, Adjusted R-squared:  0.106
F-statistic: 111.8 on 1 and 933 DF,  p-value: < 2.2e-16
```

In the above regression output, we see that education has a statistically significant impact on wages. Each year of schooling increases wage by around £60, on average. The F test tells us that the regression model has an explanatory power, even though the R-squared value is low.

5.2.10 Task 10

5.2.10.1 Task

Using the regression model above, predict what the wage would be for given values of education (how much do we expect the wage would be for given years of schooling).

5.2.10.2 Guidance

Below, we recall `model_1` to calculate predicted values; save the predictions under name `wage_hat` under `wage2` data.

```
# Save predicted values under name wage_hat
wage2$wage_hat <- predict(model_1)
```

5.2.11 Task 11

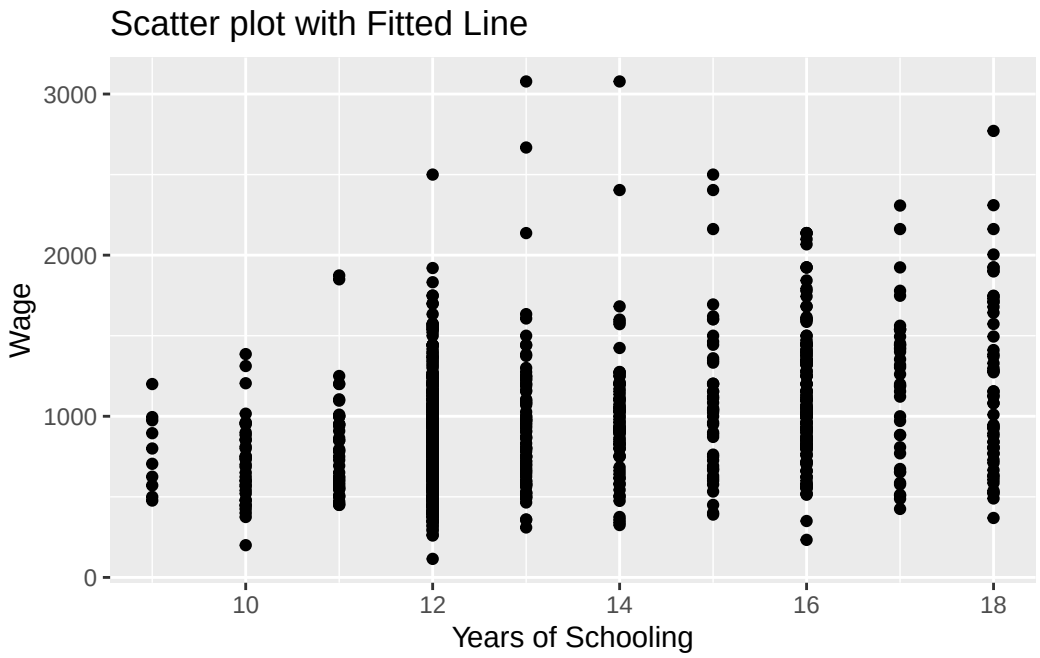
5.2.11.1 Task

Add the estimated regression line to the wage-education scatter plot.

5.2.11.2 Guidance

We will be adding the regression line to the scatter plot we produced above. We use `geom_smooth` for this purpose. Let's first remember what we did before:

```
# Scatter plot of education and wage
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  labs(title = "Scatter plot with Fitted Line", x = "Years of Schooling", y = "Wage")
```



Now, let's add the regression line:

```
# Scatter plot with fitted line
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Scatter plot with Fitted Line", x = "Years of Schooling", y = "Wage")
```

```
`geom_smooth()` using formula = 'y ~ x'
```

the `geom_smooth(method = "lm")` asks R to add a line estimating a "linear model" (i.e. a regression) of wage on educ.

Note that we could save this plot as an object by assigning it a name on the left hand side of the command. We will do that below and name the plot as `scatter_wage_educ`.

Can you guess what the plot would look if we changed `se = FALSE` to `se = TRUE` above? We can also try that below:

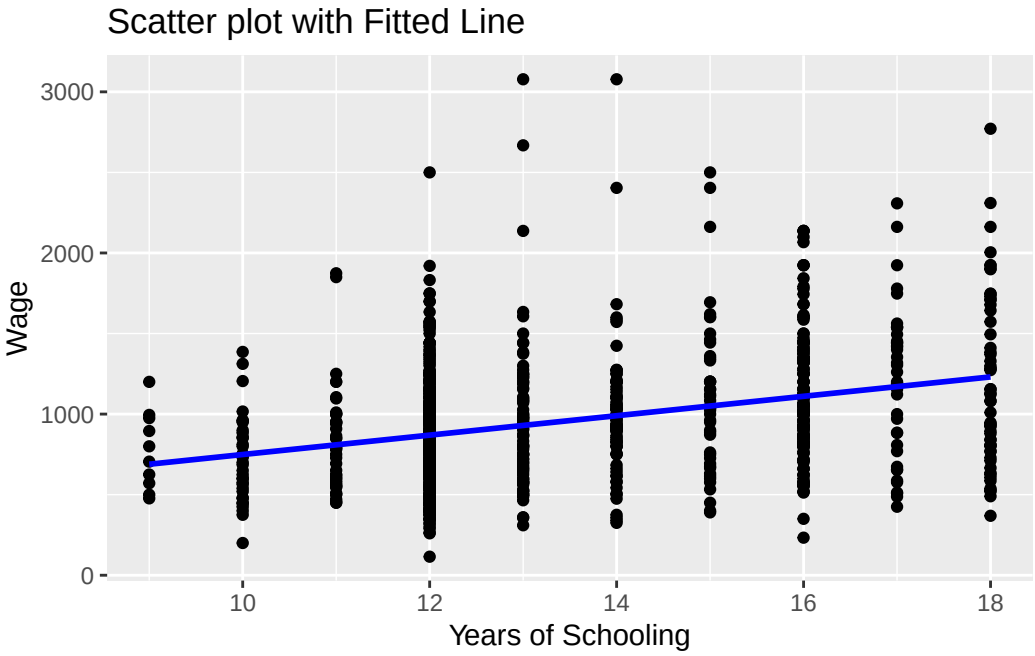
```
# Scatter plot with fitted line
scatter_wage_educ <- ggplot(wage2, aes(x = educ, y = wage)) +
                        geom_point() +
                        geom_smooth(method = "lm", se = TRUE) +
                        labs(title = "Scatter plot with Fitted Line", x = "Years of Schoolin
print(scatter_wage_educ)
```

```
`geom_smooth()` using formula = 'y ~ x'
```

We could also add this sample regression line by using the `wage_hat` variable. `wage_hat` shows the predicted value of wage given observed values of education.

```
# Scatter plot with fitted line
# we add the wage_hat variable
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  geom_line(aes(y = wage_hat), color = "blue", size = 1) +
  labs(title = "Scatter plot with Fitted Line", x = "Years of Schooling", y = "Wage")
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```



Note that we used `geom_line()` this time to add a line plot of an already existing variable in the data set.

- `ggplot(wage2, aes(x = educ, y = wage))` creates a canvas, a plot area with educ at the horizontal and wage at the vertical axis
- **`geom_point()`** adds a scatterplot of `wage` against `educ`.
- **`geom_line(aes(y = wage_hat))`** adds the line for the predicted `wage_hat` values. The `aes(y = wage_hat)` ensures the line graph uses `wage_hat` on the y-axis while sharing the x-axis (`educ`).
- **`color`** and **`size`** are optional for styling the line. Try experimenting with these and observe the changes.

5.2.12 Task 12

5.2.12.1 Task

Estimate a multiple regression model by adding experience and urban residence into the above regression. Save it under name `model_2`

5.2.12.2 Guidance

We will add `exper` and `urban` variables into the regression model using `+` sign.

```r
# Linear regression
model_2 <- lm(wage ~ educ + exper + urban, data = wage2)
summary(model_2)
```

```
Call:
lm(formula = wage ~ educ + exper + urban, data = wage2)

Residuals:
    Min      1Q  Median      3Q     Max
-799.67 -234.04  -34.26  197.89 2119.62

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -362.821    106.419  -3.409 0.000679 ***
educ          74.119      6.193  11.968  < 2e-16 ***
exper         17.940      3.105   5.777 1.03e-08 ***
urban        160.306     26.920   5.955 3.69e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 369.5 on 931 degrees of freedom
Multiple R-squared:  0.1676,    Adjusted R-squared:  0.1649
F-statistic: 62.47 on 3 and 931 DF,  p-value: < 2.2e-16
```

How does `model_2` compare to `model_1`?

5.2.13 Task 13

5.2.13.1 Task

Save your data to keep the newly created `hourly_wage` and `ln_wage` variables.

5.2.13.2 Guidance

```r
# Save data in R format
save(wage2, file = "./assets/data/wage2.Rdata")
```

5.2.14 A Gentle Introduction to dplyr library

The `dplyr` library comes with R's `tidyverse` package. The `ggplot2` library we used above to produce plots is also a part of the `tidyverse` package.

I will replicate below a few of the tasks that we performed above using the `dplyr` library

5.2.14.1 Viewing data

We have seen before to use `View` to see the contents of data in a spreadsheet format:

```
head(wage2)
```

```
# A tibble: 6 x 19
   wage hours    IQ   KWW educ      exper tenure   age married south urban  sibs
  <dbl> <dbl> <dbl> <dbl> <labelle> <lab>  <dbl> <dbl>   <dbl> <dbl> <dbl> <dbl>
1   769    40    93    35 12        11         2    31       1     0     1     1
2   808    50   119    41 18        11        16    37       1     0     1     1
3   825    40   108    46 14        11         9    33       1     0     1     1
4   650    40    96    32 12        13         7    32       1     0     1     4
5   562    40    74    27 11        14         5    34       1     0     1    10
6  1400    40   116    43 16        14         2    35       1     0     1     1
# i 7 more variables: brthord <dbl>, meduc <dbl>, feduc <dbl>,
#   hourly_wage <dbl>, ln_wage <dbl>, urban_residence <fct>, wage_hat <dbl>
```

```
#View(wage2)
```

We may use `dplyr` to select variables for viewing. Using `select` allows us to "keep or drop columns using their names and types".

```
#View(select(wage2, wage, educ, exper, urban, urban_residence))
```

5.2.14.2 Generating new variables

We used the following lines to create `hourly_wage` and `ln_wage` variables:

```
# Generate new variables
wage2$hourly_wage <- wage2$wage / wage2$hours
wage2$ln_wage <- log(wage2$wage)
```

`dplyr` 's `mutate` us used to "create, modify, and delete columns". Let us create a new data frame, `wage2_new` to see what it does:

```
wage2_new <- wage2 %>%
  mutate (
    hourly_wage_n = wage / hours,
    ln_wage_n = log(wage)
  )
```

In the above lines, we create a new data frame based on `wage2` . Note the `%>%` above. This is a part of the command and is called the `pipe operator`. It helps us to simply the code and do the operations one step after another. We first call `wage2` and create the new variables, `hourly_wage_n` and `ln_wage_n` .

Note how we avoided the use of `wage2$` every time we referred to a variable in wage2 data.

Another example we used to create a new variable was when we predicted values of wage for given levels of education after estimating model_1.

Below is the code we used:

```
wage2$wage_hat <- predict(model_1)
```

We can do this as follows using `dplyr`

```
wage2 <- wage2 %>%
  mutate(
    wage_hat_n = predict(model_1)
  )
```

5.2.14.3 Tabulating Variables

We used the code below to tabulate values of `urban` variable

```
table(wage2$urban)
```

```
  0   1
264 671
```

we may use `count` in `dplyr` for this purpose

```
wage2 %>%
  count(urban)
```

```
# A tibble: 2 x 2
  urban     n
  <dbl> <int>
1     0   264
2     1   671
```

Remember that we could save this as a new object:

```
urban_table <- wage2 %>%
  count(urban)
print(urban_table)
```

```
# A tibble: 2 x 2
  urban     n
  <dbl> <int>
1     0   264
2     1   671
```

Which output do you prefer?

5.3 Further Exercises

Download the data set called EAWE21.Rdata from the module page on Aula and save it. This is a subset of the Educational Attainment and Wage Equations data set used in Dougherty (2016) available from https://global.oup.com/uk/orc/busecon/economics/dougherty5e/student/datasets/eawe/. For this exercise we are interested in two variables:

- EXP : Total out-of-school work experience (years) as of the 2002 interview
- EARNINGS : Current hourly earnings in $ reported at the 2002 interview

5.3.1 Tasks

1. Calculate summary statistics (mean, median, minimum, maximum) for the variables EXP and EARNINGS
2. Draw scatter plot of EARNINGS on EXP.
3. Calculate the covariance and correlation between earnings and exp and comment on the values
4. Regress EARNINGS on EXP and comment on
    1. the sign and size of the regression coefficients
    2. the goodness of fits of the estimated model.
5. Add a regression line to the scatter plot.

Part III

Seminar 3 (4 February 2025)

6 Multiple Regression and Diagnostic Checks

6.1 Example: wage data

We will use the wage2 data set, which is already saved in `Rdata` format.

6.1.1 Task 1

Open `wage2.Rdata` (if it is not already open). You may so this through the menu or the command line using the `load()` function:

```
load("~/Desktop/R-workshops/assets/data/wage2.Rdata")
```

6.1.2 Task 2

Estimate a multiple regression model by regressing `wage` on `IQ`, `educ`, `exper`, `urban` and save it under name `model_3` . Display the estimation results.

6.1.2.1 Guidance

```
# Linear regression
model_3 <- lm(wage ~ IQ + educ + exper + urban, data = wage2)
summary(model_3)
```

```
Call:
lm(formula = wage ~ IQ + educ + exper + urban, data = wage2)

Residuals:
    Min      1Q  Median      3Q     Max
-797.64 -229.84  -38.35  185.10 2082.22
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -628.8654   115.5135  -5.444 6.66e-08 ***
IQ             5.0564     0.9234   5.476 5.60e-08 ***
educ          56.0554     6.9340   8.084 1.94e-15 ***
exper         17.7194     3.0583   5.794 9.41e-09 ***
urban        159.9813    26.5107   6.035 2.30e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 363.9 on 930 degrees of freedom
Multiple R-squared:  0.1936,    Adjusted R-squared:  0.1901
F-statistic: 55.81 on 4 and 930 DF,  p-value: < 2.2e-16
```

6.1.3 Task 3

6.1.3.1 Task

Test for the normality of the residuals

6.1.3.2 Guidance

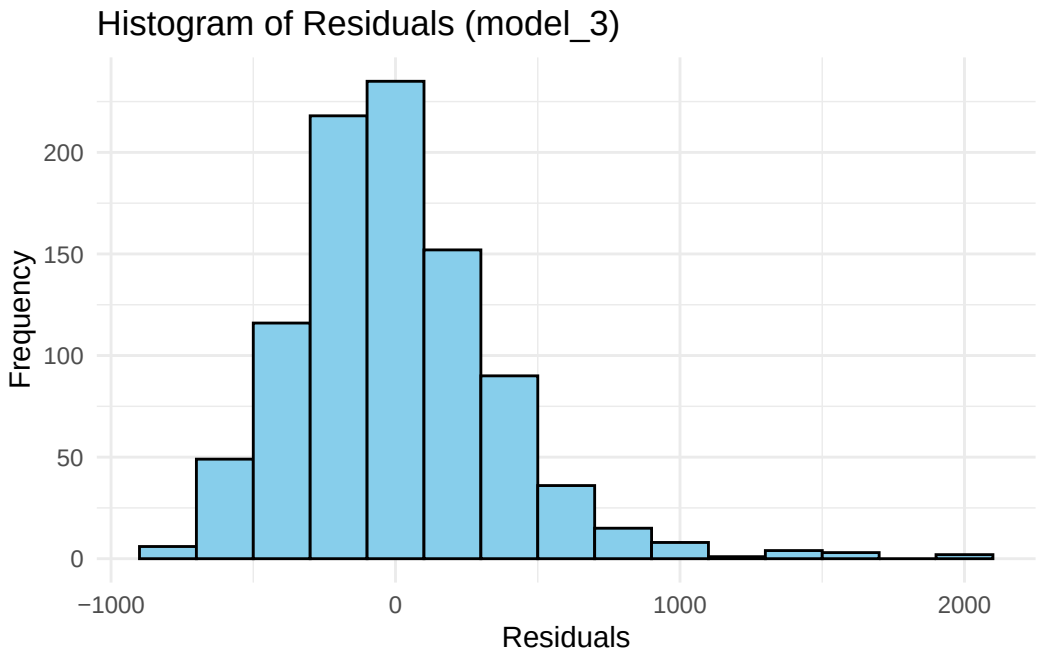
We will be using the Jarque-Bera test for this purpose.

We first save the residuals from `model_3` .

```r
wage2$resid_m3 <- residuals(model_3)
```

Plot the residuals to see the distribution. Please note that is not a part of the test but visualisation helps us to understand the data better.

```r
library(ggplot2)
ggplot(wage2, aes(x = resid_m3)) +
  geom_histogram(binwidth = 200, fill = "skyblue", color = "black") +
  labs(title = "Histogram of Residuals (model_3)", x = "Residuals", y = "Frequency") +
  theme_minimal()
```

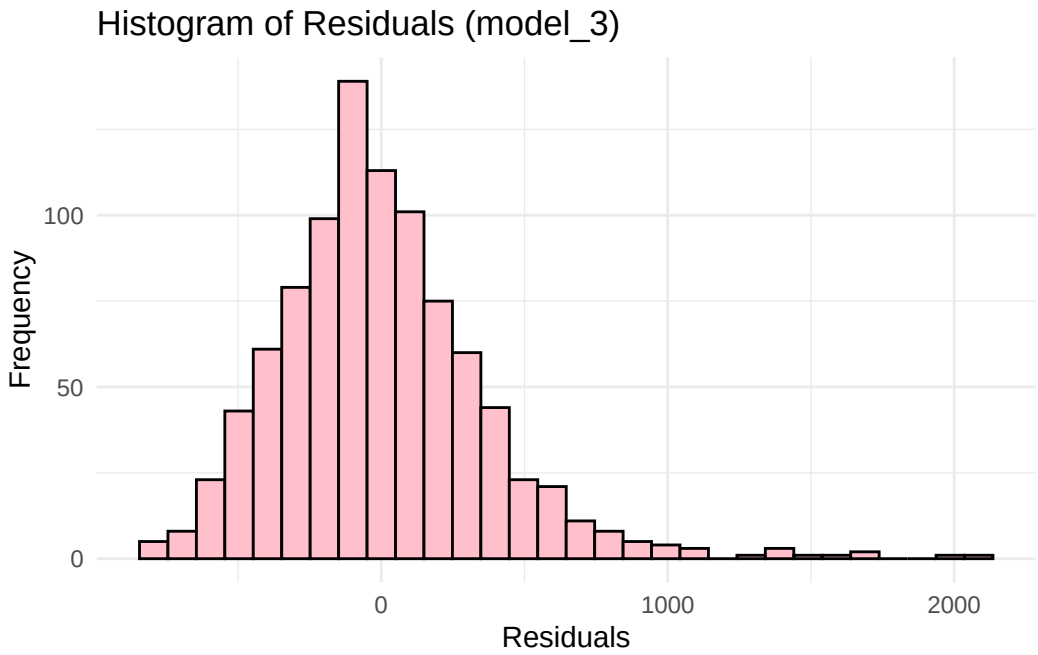


- **aes(x = resid)** specifies the residuals as the variable for the x-axis.
- **geom_histogram()** is used to create the histogram:
  - **binwidth = 200** controls the width of the bins. You can adjust this depending on how detailed you want the histogram to be.
  - **fill** sets the color inside the bars, and **color** adds a border around them for better visibility.
- **labs()** adds labels for the title and axes.
- **theme_minimal()** gives a clean, simple look to the plot - try the plot with and without this.

You may also let **ggplot** choose the number of bins automatically:

```
ggplot(wage2, aes(x = resid_m3)) +
  geom_histogram(fill = "pink", color = "black") +
  labs(title = "Histogram of Residuals (model_3)", x = "Residuals", y = "Frequency") +
  theme_minimal()
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We may use `jarque.bera.test` for the normality test. It is in the `tseries` package.

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

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

```
jarque.bera.test(wage2$resid_m3)
```

```

    Jarque Bera Test

data:  wage2$resid_m3
X-squared = 699.59, df = 2, p-value < 2.2e-16
```

The p-value of the test is almost zero. We reject the null hypothesis of normal distribution. The residuals from model_3 are **not** normally distributed.

6.1.4 Task 4

6.1.4.1 Task

Test for the functional form.

6.1.4.2 Guidance

We may use this to check whether there are any omitted variables or non-linearity in the model. The test is Ramsey RESET.

```
library(lmtest)
```

```
Loading required package: zoo
```

```

Attaching package: 'zoo'
```

```
The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
resettest(model_3)
```

```

    RESET test

data:  model_3
RESET = 3.8665, df1 = 2, df2 = 928, p-value = 0.02127
```

The default `resettest` includes second and third powers of the fitted values in the test regression. You may change this using the `power` option. Below we include from second to the fourth power of fitted values.

```
resettest(model_3, power = 2:4)
```

```
    RESET test

data:  model_3
RESET = 2.8504, df1 = 3, df2 = 927, p-value = 0.03646
```

The decision depends on the chosen significance level. We reject the null hypothesis of correct functional form if we choose a 5% significance level.

6.1.5 Task 5

6.1.5.1 Task

Test for heteroscedasticity.

6.1.5.2 Guidance

We apply Breusch-Pagan heteroscedasticity test.

```
bptest(model_3)
```

```
    studentized Breusch-Pagan test

data:  model_3
BP = 16.355, df = 4, p-value = 0.002578
```

The p-value is smaller than 0.05. Hence, we reject the null of no heteroscedasticity at 5% significance level. There is heteroscedasticity.

6.1.6 Task 6

6.1.6.1 Task

Test for autocorrelation in the model

6.1.6.2 Guidance

This is a trick question! Autocorrelation problem is related to time series data whereas we have cross-section data here. Autocorrelation problem is irrelevant here.

6.1.7 Task 7

6.1.7.1 Task

Replicate the above using logarithmic wages. Has there been a change in model diagnostics? Which form do you prefer to use for inference?

6.1.7.2 Guidance

You may use the script file to copy-paste all the code and make the minor changes (i.e. replacement of `wage` with `ln_wage`).

Part IV

Seminar 4 (11 February 2025)

7 Introduction to Time Series Analysis

7.1 Example: GAP Sales data

We start by loading the required libraries.

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(ggplot2)
library(dplyr)
library(lmtest)
```

```
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
library(tseries)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

The GAP_Sales data that we will be using in this session is obtained from (Wilson and Keating 2007). It shows the sales figures of GAP.

7.1.1 Task 1

Start a new project in R and name it as GAP-sales-analysis. Import GAP_Sales.csv data into this project. GAP_Sales is a quarterly time series data covering time period 1985:Q1 to 2004:Q4. In this example, we would like to estimate a regression model explaining sales of GAP.

7.1.1.1 Guidance

Use the menu import GAP_Sales.csv file into R. You need to choose `From Text` (base) because `csv` is a text format. The GAP_Sales data we have is comma separated, but you may encounter a different form of separation, for example, tab or semi-column. In the opening window, give a name for your data frame under the `Name` field and remember to check the `Heading` as `Yes` because we have variable names in the first row of the csv file. Also, note the `strings as factors` option, which asks R to import text-based content (variables) as categorial (`factor` is the terminology R uses).

You could alternatively run the code below

```
df <- read.csv("~/Desktop/R-workshops/assets/data/GAP_Sales.csv", stringsAsFactors=TRUE)
#View(df)
```

For ease of typing, I called this data as `df`. In the code below, `df` will refer to the GAP_Sales data we imported.

7.1.2 Task 2

Browse the data and see the contents of the variables.

7.1.2.1 Guidance

We have done this above, using

```
#View(df)
```

You may also use `head()` function to see the first 6 rows of data

```
head(df)
```

```
  Year quarter   Yqrt  Sales Time T.squared Q2 Q3 Q4 D911      ICS
1 1985      q1 1985q1 105715    1          1  0  0  0    0 94.46667
2 1985      q2 1985q2 120136    2          4  1  0  0    0 94.30000
3 1985      q3 1985q3 181669    3          9  0  1  0    0 92.83333
4 1985      q4 1985q4 239813    4         16  0  0  1    0 91.06667
5 1986      q1 1986q1 159980    5         25  0  0  0    0 95.53333
6 1986      q2 1986q2 164760    6         36  1  0  0    0 96.76667
```

7.1.3 Task 3

Provide a time series plot of the `Sales` variable.

7.1.3.1 Guidance

GAP_Sales data is a quarterly data. However, R would not recognise this until we tell it that is a quarterly time series. R has a built-in time series class, `ts` for basic data manipulation. Some other popular packages (more advanced than the `ts` in base R) include `tseries` and `zoo`.

As (Kleiber and Zeileis 2008) explains, `ts` is aimed at regular series observed in annual, quarterly, and monthly intervals. Time series objects can be created by supplying the data along with the arguments `start`, `end`, and `frequency`. The data can be:

- a numeric vector (a single variable), or
- a matrix (including a set of variables).

It includes time-series specific methods such as `lag()` (for the lagged values of the variables) and `diff()` (for time differencing the variable).

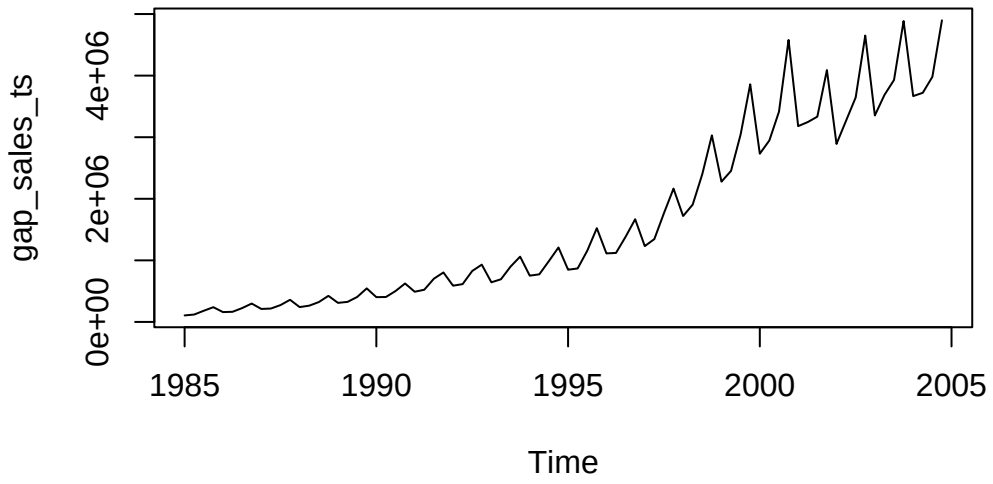
`Sales` is the variable we are interested in our data. So, let us start by introducing a time dimension to that series. In the code below, we create a single numeric vector, `gap_sales_ts`

by defining the start date and the frequency of the `Sales` variable. Our variable starts from the first quarter of 1985 with a frequency of 4 (it is a quarterly data, repeating every 3 months).

```
gap_sales_ts <- ts(df$Sales, start = c(1985, 1), frequency =4)
```

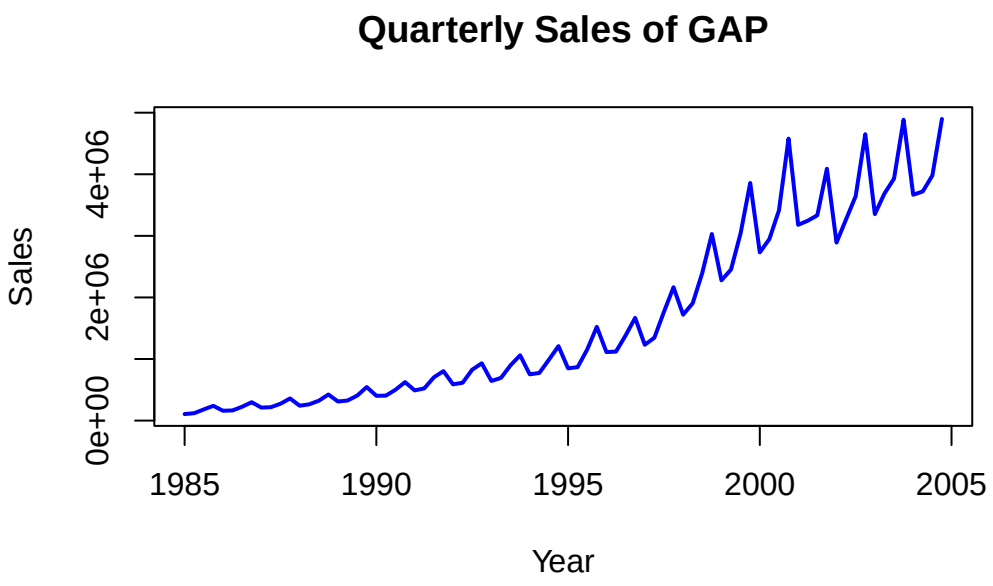
R's basic `plot` function will give us the following:

```
plot(gap_sales_ts)
```



You may add labels and color with some additional options:

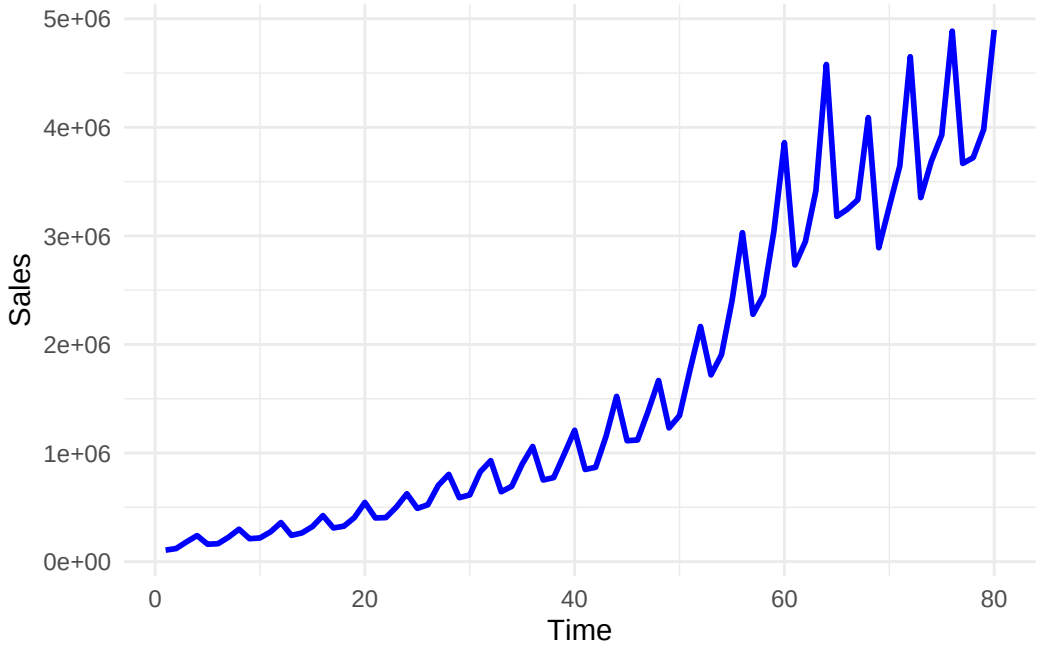
```
plot(gap_sales_ts, col = "blue", lwd = 2, xlab = "Year", ylab = "Sales", main = "Quarterly Sa
```



You may also use ggplot to plot the Sales data:

```
ggplot(df, aes(x = Time, y = Sales)) +
  geom_line(color = "blue", size = 1) +
  theme_minimal()
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```



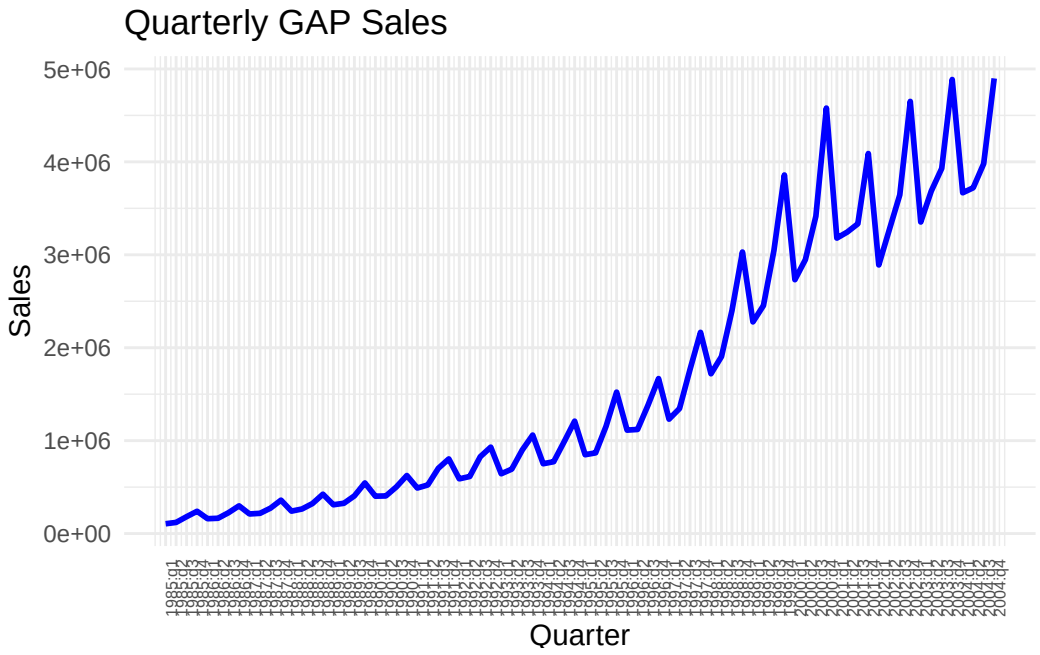
In the above plot, although we can see the pattern of the Sales variable quite clearly, the Time variable labels fail to show us the respective quarter values. We may change these labels by using the following lines of code.

We first define labels to correspond to each data point

```
# First, create a new column for formatted quarter labels
df$Quarter_label <- paste0(df$Year, ":", df$quarter)
# You can achieve the same as above using the code below:
# (note that you do not need this once you create Quarter_Label above )
df$Quarter_label_v2 <- with(df, paste(Year, quarter, sep = ":"))
```

Check the values of `Quarter_label` in the `df`. You will see that it goes on like 1985:q1, 1985:q2, and so on. We may now use these labels instead of the values of the `Time` variable.

```r
ggplot(df, aes(x = Time, y = Sales)) +
  geom_line(color = "blue", size = 1) +
  scale_x_continuous(
    breaks = df$Time,  # Position the breaks at each quarter, i.e. at each value of Time
    labels = df$Quarter_label  # Label each point using Quarter_label variable created above
  ) + # provide a title and axes labels below
  labs(title = "Quarterly GAP Sales", x = "Quarter", y = "Sales") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, size=6))  # Rotate labels for better readabili
```



Looking at this plot, what can you say about the sales figures over time? What kind of time-series characteristics it reveals?

We can see above that the gap sales have some repeating fluctuations around a positive trend. The trend is not linear. The regression that we estimate should capture this non-linear trend as well as the seasonal fluctuations.

The following tasks will take us to the best fit possible with the available data.

7.1.4 Task 4

Fit a linear trend line to the `Sales` variable.

a. Provide an interpretation of the slope coefficient.

b. Check how well this model fits the data by plotting the predictions of the model and the observed values against time.

c. Plot the residuals of this model and explain whether or not you see a pattern.

7.1.4.1 Guidance

The `Time` variable will be used to fit a linear trend to `Sales`. The `Time` variables takes values from 1 to 80, increasing by 1 in each data point (quarter).

```
# Fit a linear trend line to Sales data
model_1 <- lm(Sales ~ Time, data = df)
summary(model_1)
```

```
Call:
lm(formula = Sales ~ Time, data = df)

Residuals:
    Min      1Q  Median      3Q     Max
-889709 -390551  -60886  325202 1600763

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -680044     121435   -5.60 3.08e-07 ***
Time           57162       2605   21.95  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 538000 on 78 degrees of freedom
Multiple R-squared:  0.8606,    Adjusted R-squared:  0.8588
F-statistic: 481.6 on 1 and 78 DF,  p-value: < 2.2e-16
```

Time variable takes values starting from 1 and increasing by 1 in each quarter. Including this variable will allow us to fit a trend to sales.

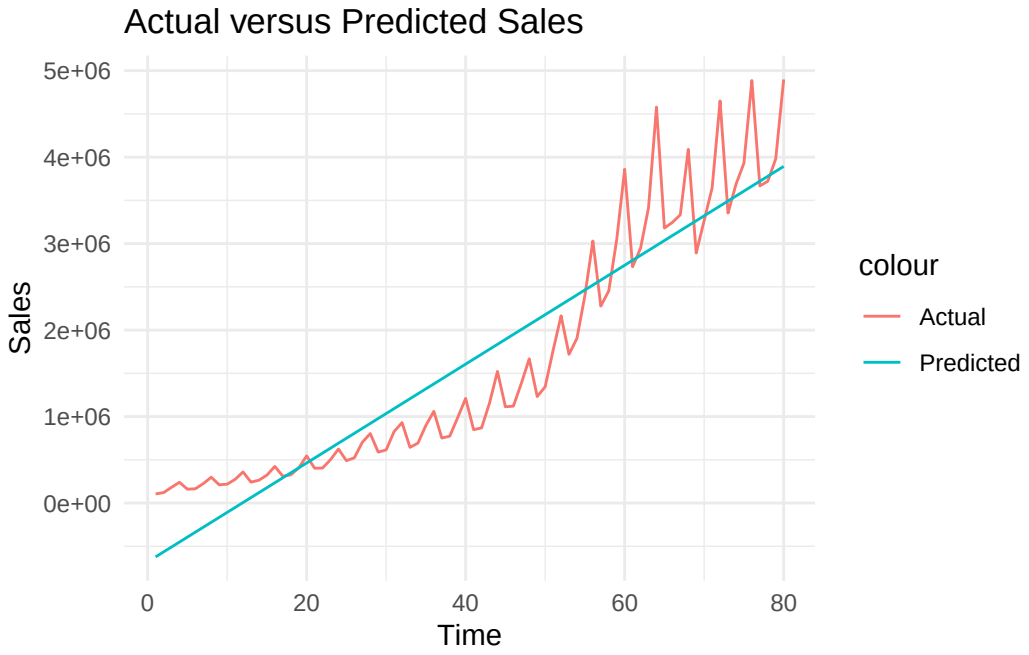
Interpretation of the coefficient of `Time` variable: In each quarter, the GAP sales increases by $57,162 thousand, on average (note that in the data, `Sales` is measured in thousand dollars)

Obtain predictions using `predict()` function.

```r
# Obtain predictions
df$sales_hat_m1 <- predict(model_1)
```

We can use R's base time series plot but we will need to convert the predictions into a time series. Alternatively, ggplot is easier to use.

```r
# Plot actual versus predicted Sales
ggplot(df, aes(x = Time)) +
  geom_line(aes(y = Sales, color = "Actual")) +
  geom_line(aes(y= sales_hat_m1, color = "Predicted")) +
  theme_minimal() +
  labs(title = "Actual versus Predicted Sales", x = "Time", y = "Sales")
```

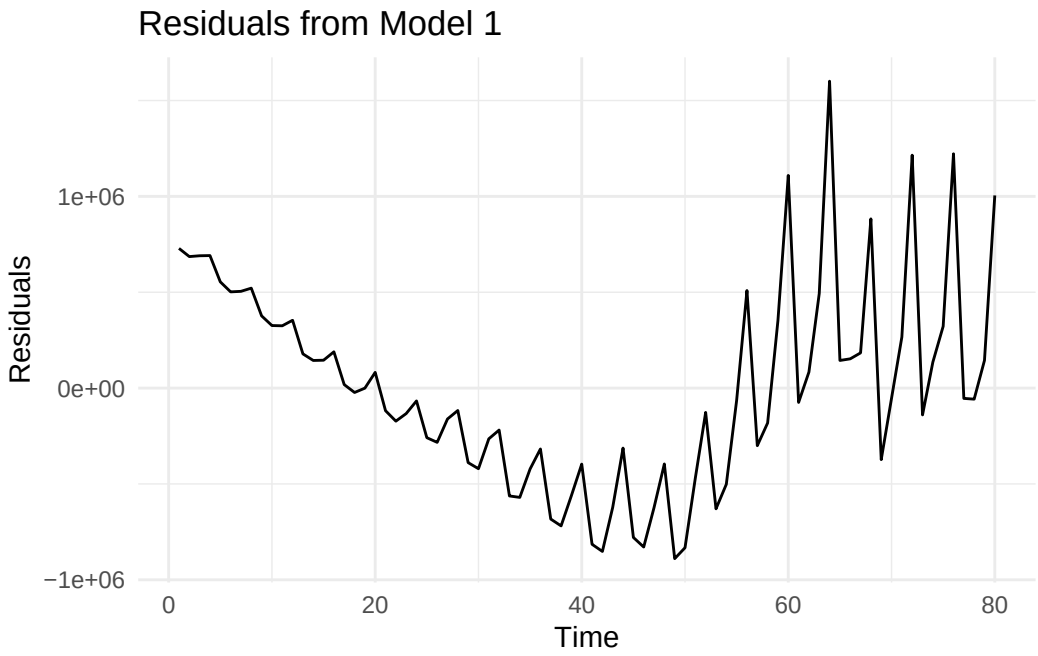


We can see above that although we could estimate the trend roughly, it is not a perfect fit. `Sales` has a positive trend, it is not linear. We will be using natural logarithm of sales below.

Below, we save and plot the residuals from model_1

```r
# Save residuals from model_1
df$residuals_m1 <- residuals(model_1)

# Residual plot
ggplot(df, aes(x = Time, y = residuals_m1)) +
```

```
  geom_line() +
  theme_minimal() +
  labs(title = "Residuals from Model 1", x = "Time", y = "Residuals")
```



When we look at the plot of residuals, we can see that the deviations from the linear trend (i.e. the non-linearities and the fluctuations) are reflected in residuals.

7.1.5 Task 5

Replicate the same analysis using logarithm of `Sales`

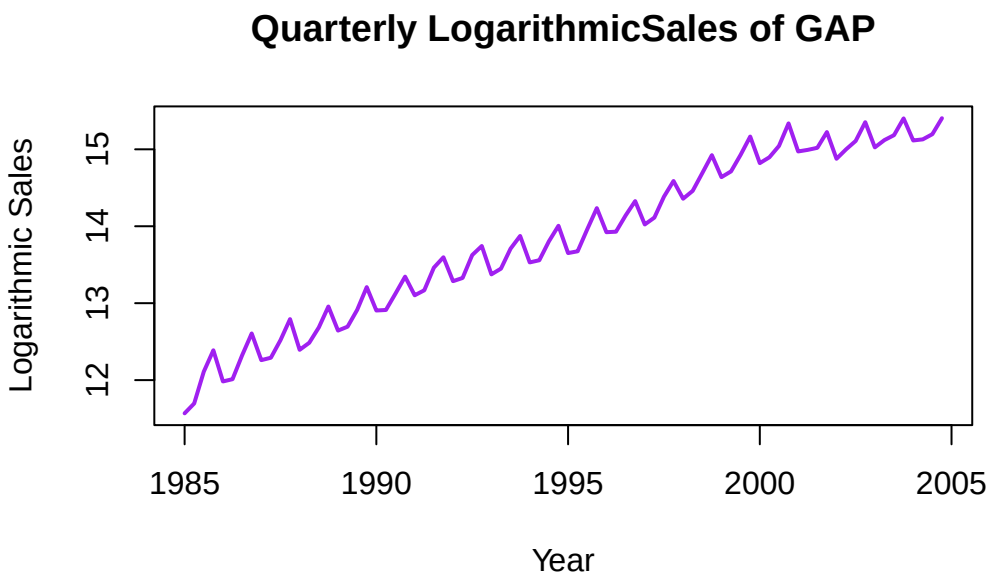
7.1.5.1 Guidance

We start by taking the logarithm of `Sales` variable.

```
# Logarithmic Sales data
df$ln_sales <- log(df$Sales)
```

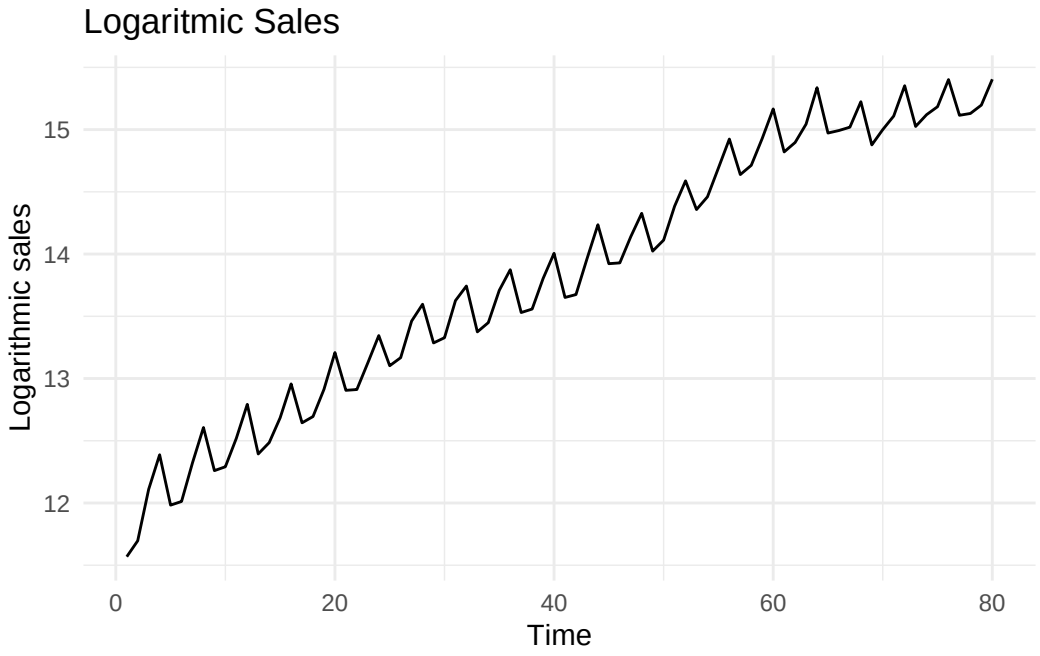
Plot logarithmic sales. Let's first do this base R's time series plot. We start by converting our `ln_sales` into quarterly time series, and then use the `plot()` function. `lwd` option below sets the line width of the plot. Change the color and the `lwd` values and see what you get.

```
# Plot of logarithmic sales (first approach - convert to time series)
ln_sales_ts <- ts(df$ln_sales, start = c(1985, 1), frequency =4) # covenrt the ln_sales into
plot(ln_sales_ts, col = "purple", lwd = 2, xlab = "Year", ylab = "Logarithmic Sales", main =
```



We may also use `ggplot` for the same purpose

```
# Plot of logarithmic sales (second approach - use ggplot)
ggplot(df, aes(x = Time, y = ln_sales)) +
  geom_line() +
  theme_minimal() +
  labs(title = "Logaritmic Sales", x = "Time", y = "Logarithmic sales")
```



Fit a trend line to logarithmic sales

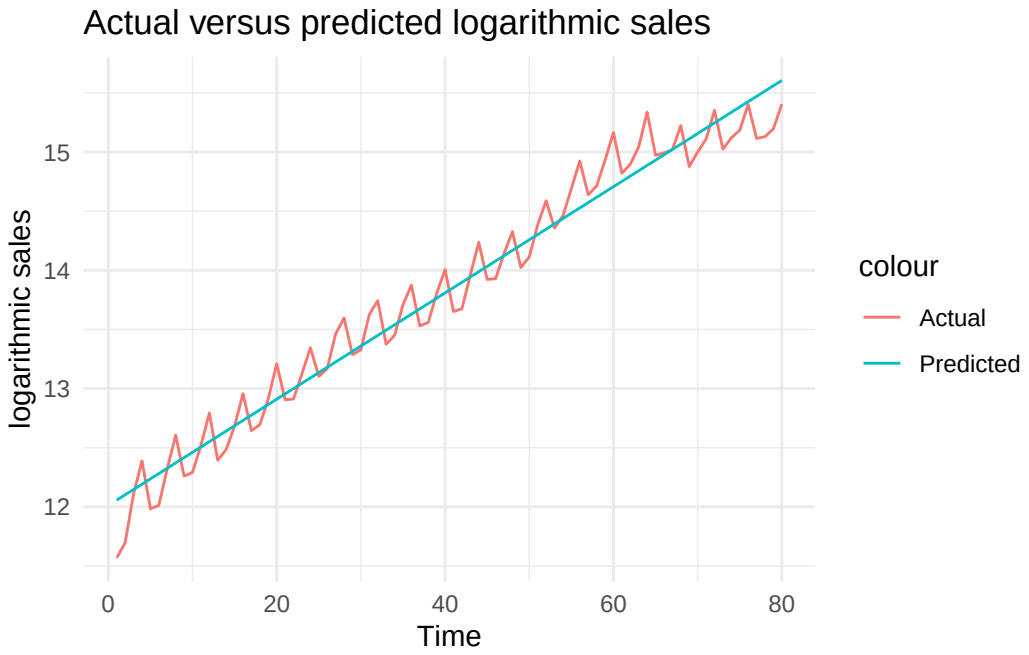
```
# Fit a trend line to logarithmic sales
model_2 <- lm(ln_sales ~ Time, data = df)
summary(model_2)
```

```
Call:
lm(formula = ln_sales ~ Time, data = df)

Residuals:
    Min      1Q  Median      3Q     Max
-0.4883 -0.1559 -0.0026  0.1684  0.4589

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 12.011834   0.046964  255.76   <2e-16 ***
Time         0.044919   0.001007   44.59   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2081 on 78 degrees of freedom
Multiple R-squared:  0.9623,    Adjusted R-squared:  0.9618
F-statistic:  1988 on 1 and 78 DF,  p-value: < 2.2e-16
```

Both the intercept and slope coefficients are statistically significant (with very low p-values). Slope coefficient this time shows that in each quarter, sales increase by around 4.5%, on average.

Let's now plot the predictions from this model with the actual `ln_sales` figures

```
# Obtain predictions from the logarithmic model
df$ln_sales_hat_m2 <- predict(model_2)

# Plot actual versus predicted log sales
ggplot(df, aes(x = Time)) +
  geom_line(aes(y = ln_sales, color = "Actual")) +
  geom_line(aes(y = ln_sales_hat_m2, color = "Predicted")) +
  theme_minimal() +
  labs(title = "Actual versus predicted logarithmic sales", x = "Time", y = "logarithmic sale
```



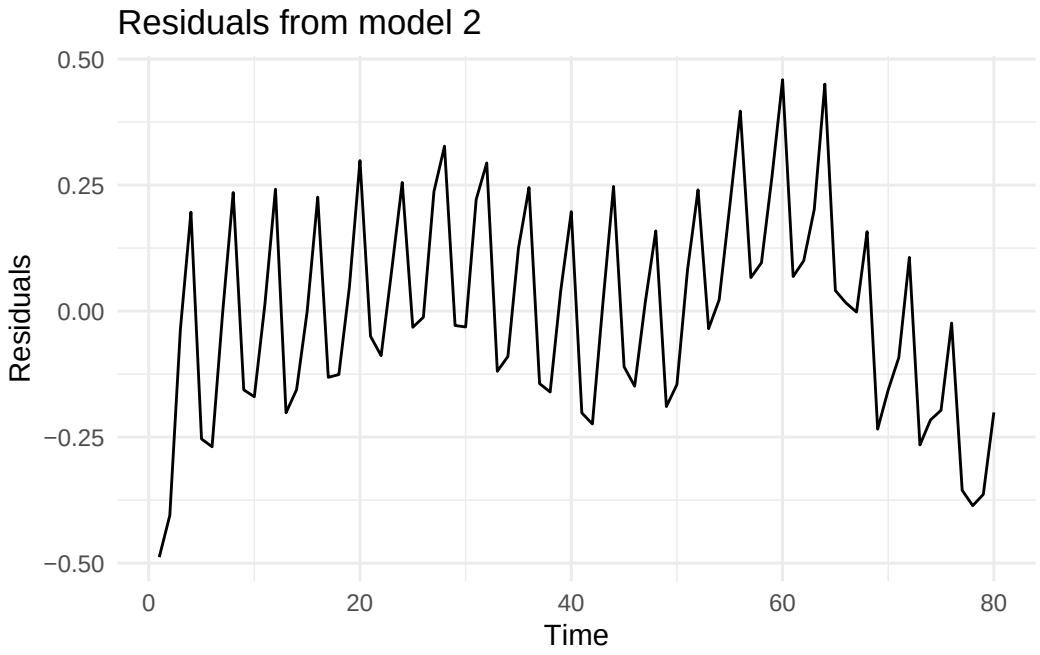
What do you think about this fit?

We can see above that using a logarithmic transformation helped to obtain a better fit for sales.

Note that although it is very tempting to use R^2 to compare the goodness-of-fit of these two models (with and without the logarithmic transformation), we cannot do that as R^2 cannot be used to compare models with different dependent variables.

Let's check what the residuals from the above estimation look like

```r
# Residuals from model_2
df$residuals_m2 <- residuals(model_2)

# Plot residuals from model_2
ggplot(df, aes(x = Time, y= residuals_m2))+
  geom_line() +
  theme_minimal() +
  labs(title = "Residuals from model 2", x = "Time", y = "Residuals")
```



We can see that the residuals repeatedly fluctuate in certain intervals. This is due to the seasonality in the sales data. We will be using quarter dummies to control for the seasonality. In comparison to the previous regression specification, residuals do not reveal a trend (because by fitting a trend on the logarithmic data, we have controlled for the non-linear trend.

7.1.6 Task 6

Do you have any suggestions to improve the fit of this model?

7.1.6.1 Guidance

Check the residual plot above. Do you see a specific pattern? What can we do to capture the fluctuations that you see?

7.1.7 Task 7

Add quarter dummies to the model you estimated above.

a. Interpret the coefficients in this model.

b. Check how well this model fits the data by plotting the predictions of the model and the observed values against time.

c. Plot the residuals of this model and explain whether or not you see a pattern.

d. Does the inclusion of the quarter dummies improve the fit of the model? Test for the joint significance of the quarter dummies.

e. If you were to choose one the models that you have estimated using the GAP sales data, which one would you choose? Why?

7.1.7.1 Guidance

The quarter dummies that we need for this model are already in the data: `Q2`, `Q3`, `Q4`. If these were not in the data, we could create them using the `ifelse()` function. This is provided below

```
df$qrt_1 <- ifelse(df$quarter == "q1", 1, 0)
df$qrt_2 <- ifelse(df$quarter == "q2", 1, 0)
df$qrt_3 <- ifelse(df$quarter == "q3", 1, 0)
df$qrt_4 <- ifelse(df$quarter == "q4", 1, 0)
```

We could also use the `dplyr` package (note that I assign different names to these variables to be able to differentiate alternative ways of creating the dummies. You may choose a name of your own):

```
df <- df %>%
  mutate(
    quarter1 = ifelse(quarter == "q1", 1, 0),
    quarter2 = ifelse(quarter == "q2", 1, 0),
    quarter3 = ifelse(quarter == "q3", 1, 0),
    quarter4 = ifelse(quarter == "q4", 1, 0)
  )
```

Check the values of these newly created dummies (`quarter1`, `quarter2`, `quarter3`, and `quarter4`) in the data.

We can now estimate the model including these quarter dummies together with a linear trend

```r
# Estimate the model using trend and quarter dummies
model_3 <- lm(ln_sales ~ Time + quarter2 + quarter3 + quarter4, data = df)
summary(model_3)
```

```
Call:
lm(formula = ln_sales ~ Time + quarter2 + quarter3 + quarter4,
    data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.41512 -0.06014 -0.00347  0.09422  0.23885

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 11.882418   0.042709 278.219  < 2e-16 ***
Time         0.044620   0.000707  63.110  < 2e-16 ***
quarter2     0.013792   0.046130   0.299 0.765783
quarter3     0.184808   0.046146   4.005 0.000145 ***
quarter4     0.367414   0.046173   7.957 1.44e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1459 on 75 degrees of freedom
Multiple R-squared:  0.9822,    Adjusted R-squared:  0.9812
F-statistic:  1032 on 4 and 75 DF,  p-value: < 2.2e-16
```

There are 4 quarters in a year. Quarter 1 is excluded from the model. This is the reference category. Our interpretation of the other quarter dummies will be in reference to the excluded category. Looking at the p-values, Q3 and Q4 are statistically significant while Q2 is statistically insignificant. Statistical insignificance implies that sales in quarter 2 are not different than sales in quarter 1. In other words, average sales in the first 2 quarters are around the same level.

A rough interpretation of the Q3 and Q4 coefficients would be as follows (**note that this approach could be reliable only when the effects (i.e. the coefficients) are very small**)

- Holding everything else constant, sales in quarter 3 (i.e. in months July-August- September) are around 18.5% higher than sales in quarter 1 (i.e. in comparision to sales in the first 3 months of the year.

- Holding everything else constant, sales in quarter 4 (i.e. in months October- November- December) are around 36.7% higher than sales in quarter 1 (i.e. in comparison to sales in the first 3 months of the year.

For a more precise interpretation of dummy variable coefficients in a logarithmic dependent variable model, we need to transform the estimated coefficients first:

$$[exp(0.1848) - 1] \times 100 = 20.30$$

$$[exp(0.3674) - 1] \times 100 = 44.40$$

- Holding everything else constant, sales in quarter 3 (i.e. in months July-August- September) are around 20.30% higher than sales in quarter 1 (i.e. in comparison to sales in the first 3 months of the year
- Holding everything else constant, sales in quarter 4 (i.e. in months October- November- December) are around 44.40% higher than sales in quarter 1 (i.e. in comparison to sales in the first 3 months of the year.

Please note that this transformation is applied only when the dependent variable is in logarithmic form and if we are commenting on the effect of a dummy variable.

Are these quarterly dummies contributing to the explanatory power of the model? In other words, are they jointly statistically significant? We can check this using an F-test for restrictions. This could be done using the `anova` function in R.

Below are the steps we follow to test for the restrictions:

1. Estimate the full (**unrestricted**) model. We have done that above. It is saved under `model_3`.
2. Estimate the **restricted** model where quarter dummy coefficients take value zero. This is in fact, our `model_2` above.
3. Perform an F-test to compare the restricted and unrestricted models using `anova()` :

   `anova(restricted_model, unrestricted_model)`

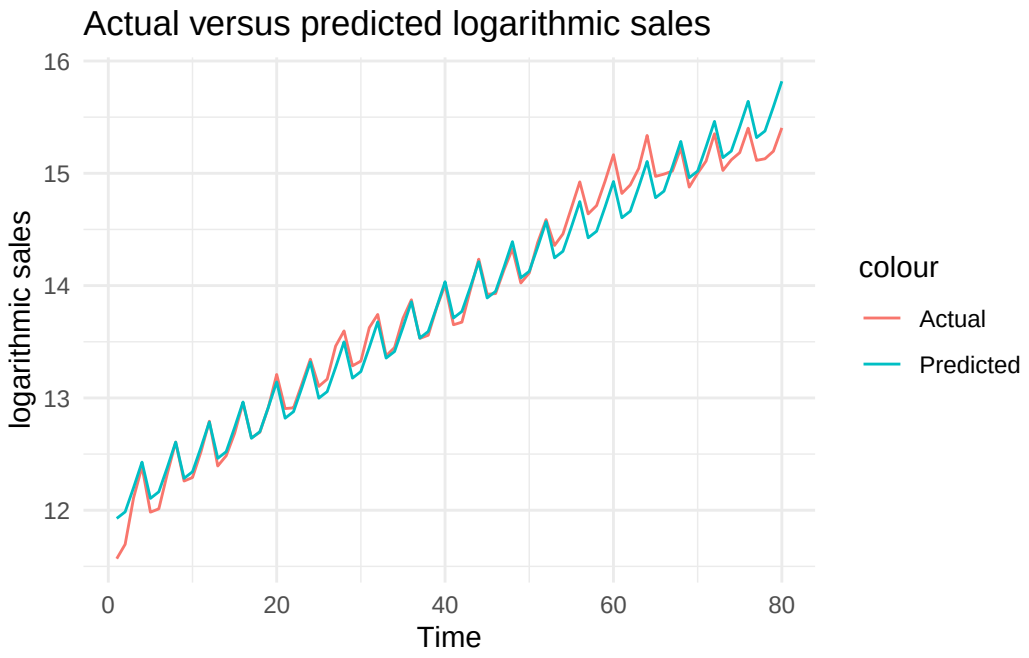
```
# Perform an F-test
anova(model_2, model_3)
```

```
Analysis of Variance Table

Model 1: ln_sales ~ Time
Model 2: ln_sales ~ Time + quarter2 + quarter3 + quarter4
  Res.Df    RSS Df Sum of Sq      F   Pr(>F)
1     78 3.3767
2     75 1.5956  3    1.7811 27.906 3.17e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis in the above test is that the coefficients of quarter dummies are jointly equal to zero versus the alternative that at least one is different than zero. We have a very small p-value. Hence we reject the null hypothesis and conclude that the quarter dummies are jointly statistically significant.

Let's plot the actual values against predictions to see the improvement by the inclusion of the quarter

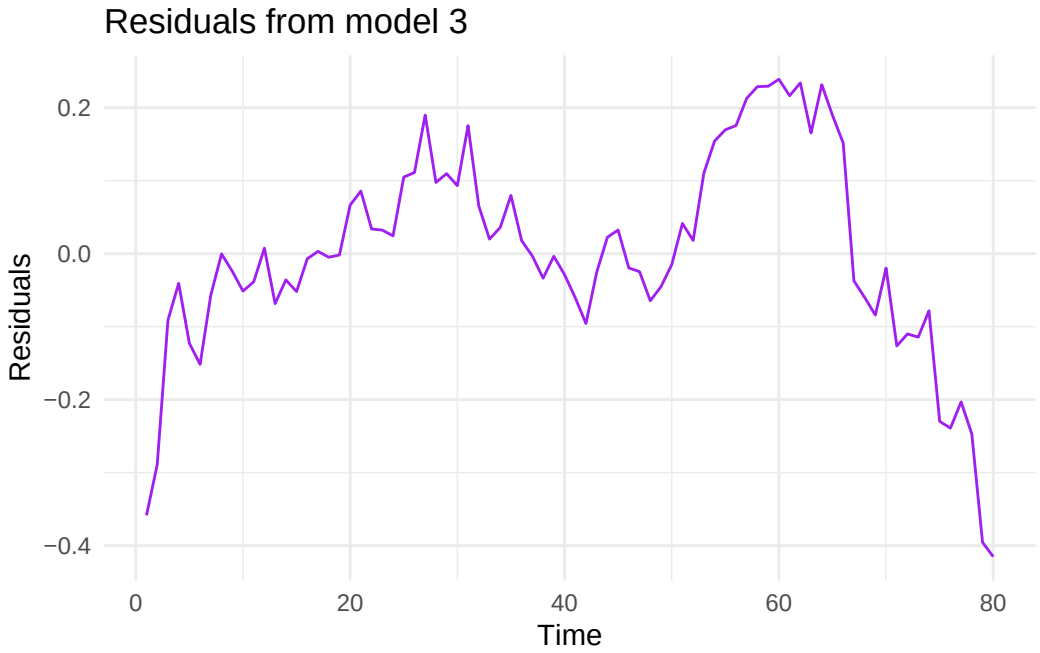
```r
# Obtain predictions from model_3
df$ln_sales_hat_m3 <- predict(model_3)

# Plot actual versus predicted log sales
ggplot(df, aes(x = Time)) +
  geom_line(aes(y = ln_sales, color = "Actual")) +
  geom_line(aes(y = ln_sales_hat_m3, color = "Predicted")) +
  theme_minimal() +
  labs(title = "Actual versus predicted logarithmic sales", x = "Time", y = "logarithmic sale
```



We can see from the plot that the regression model now have a better fit to the actual data. All fluctuations are captured by quarter dummies.

And finally, let's have a look at the residuals

```
# Residuals from model_2
df$residuals_m3 <- residuals(model_3)

# Plot residuals from model_2
ggplot(df, aes(x = Time, y= residuals_m3))+
  geom_line(color = "purple") +
  theme_minimal() +
  labs(title = "Residuals from model 3", x = "Time", y = "Residuals")
```



Among the 3 models estimated, we would choose the last one because it has a better fit than the others. In addition to the checks we have done above, you could also compare the residual sum of squares to see which one fits better. But please note that if you follow that approach, you will need to make these values comparable. For example, in this case, take the anti-log of residuals from the regressions that use logarithmic sale.

7.1.8 Task 8

Conduct the conventional misspecification tests on the last model estimated.

7.1.8.1 Guidance

We may start with the **normality of the residuals**. For this test, we will be using the `jarque.bera.test()` from the `tseries` package.

```
# Normality of residuals
jarque.bera.test(df$residuals_m3)
```

```
    Jarque Bera Test

data:  df$residuals_m3
X-squared = 6.4227, df = 2, p-value = 0.0403
```

The null hypothesis of normal distribution is rejected at 5% significance level.

For the tests that follow, we will using the `lmtest` package.

Autocorrelation Test

We use the `bgtest()` function below. It performs the Breusch-Godfrey Test. We first test for the first order autocorrelation and then, because we have quarterly data, the existence of autocorrelation up to order 4.

```
# Autocorrelation
bgtest(model_3)
```

```
    Breusch-Godfrey test for serial correlation of order up to 1

data:  model_3
LM test = 60.429, df = 1, p-value = 7.628e-15
```

```
bgtest(model_3, order = 4)
```

```
    Breusch-Godfrey test for serial correlation of order up to 4

data:  model_3
LM test = 62.487, df = 4, p-value = 8.703e-13
```

There is autocorrelation problem in our model.

Heteroscedasticity

We will use `bptest()` function for heteroscedasticity. It performs the Breusch-Pagan Test.

```
# Heteroscedasticity
bptest(model_3)
```

```
    studentized Breusch-Pagan test

data:  model_3
BP = 9.4108, df = 4, p-value = 0.05161
```

The null of no heteroscedasticity cannot be rejected at 5% significance level.

Functional Form

We will use `resettest()` for Ramsey's RESET.

```
# Ramsey RESET
resettest(model_3)
```

```
    RESET test

data:  model_3
RESET = 36.442, df1 = 2, df2 = 73, p-value = 1.06e-11
```

The null of correct functional form is rejected.

The estimated model suffers from autocorrelation, heteroscedasticity, functional misspecification, and a structural break at the third quarter of 2001. Also, the residuals are non-normally distributed.

Please see your textbook for explanations of possible implications of each of these misspecification.

7.1.9 Task 9

Using the last model, forecast the sales value for each quarter of 2005.

7.1.9.1 Guidance

There are more advanced ways of producing forecasts in R. But we need at this stage is explained below.

Define a `forecast_2005` function using the coefficients of `model_3`. Let us see what `model_3` coefficients are

```
summary(model_3)
```

```
Call:
lm(formula = ln_sales ~ Time + quarter2 + quarter3 + quarter4,
    data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.41512 -0.06014 -0.00347  0.09422  0.23885

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 11.882418   0.042709 278.219  < 2e-16 ***
Time         0.044620   0.000707  63.110  < 2e-16 ***
quarter2     0.013792   0.046130   0.299 0.765783
quarter3     0.184808   0.046146   4.005 0.000145 ***
quarter4     0.367414   0.046173   7.957 1.44e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1459 on 75 degrees of freedom
Multiple R-squared:  0.9822,    Adjusted R-squared:  0.9812
F-statistic:  1032 on 4 and 75 DF,  p-value: < 2.2e-16
```

Define a function to forecast future values:

```r
forecast_2005 <- function(Time, quarter2, quarter3, quarter4) {
  exp(11.882418 + 0.044620 * Time + 0.013792 * quarter2 + 0.184808 * quarter3 + 0.367414 * qu
}
```

Use the above function for forecasts.

```r
# Forecasts from 2005
# quarter 1
y2005_q1 <- forecast_2005(81,0,0,0)
print(y2005_q1)
```

```
[1] 5371609
```

```r
# quarter 2
y2005_q2 <- forecast_2005(82,1,0,0)
print(y2005_q2)
```

```
[1] 5694720
```

```
# quarter 3
y2005_q3 <- forecast_2005(83,0,1,0)
print(y2005_q3)
```

```
[1] 7065158
```

```
# quarter 4
y2005_q4 <- forecast_2005(84,0,0,1)
print(y2005_q4)
```

```
[1] 8867576
```

Part V

Seminar 5 (18 February 2025)

8 Unit Root (Non-stationary Time Series): Augmented Dickerm(y-Fuller Test

8.1 Example: Pepper Price

The Pepper Price example provided in this section is taken from (Kleiber and Zeileis 2008). We start by loading the required libraries.

```
library(AER) # Applied Econometrics with R, Kleiber and Zeileis, 2008
```

```
Loading required package: car
```

```
Loading required package: carData
```

```
Loading required package: lmtest
```

```
Loading required package: zoo
```

```

Attaching package: 'zoo'
```

```
The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
Loading required package: sandwich
```

```
Loading required package: survival
```

```
library(tseries) #Required for the adf test
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

```
library(vars)
```

```
Loading required package: MASS
```

```
Loading required package: strucchange
```

```
Loading required package: urca
```

```
library(urca)
```

Load Pepper Price time series data and check the first 6 rows of observations.

```
data("PepperPrice")
head(PepperPrice)
```

```
        black   white
[1,]  884.050 1419.78
[2,]  919.329 1503.55
[3,]  930.350 1536.62
[4,] 1102.310 1629.22
[5,] 1150.810 1737.24
[6,] 1093.490 1629.22
```

There are two series here: black pepper and white pepper. Let's understand the time series components better:

```
# tsp stands for "Time Series Properties"
tsp(PepperPrice)
```

```
[1] 1973.75 1996.25   12.00
```

`tsp` above stands for time series properties. It seems like we have monthly data (frequency of 12), starting in year 1973 and ending in year 1996.

We can use the `window()` function to inspect the values of the variables by setting start and end dates. In the example below, we start from the start of the sample and display values up until the 6th data point of 1974.

```
window(PepperPrice, end = c(1974, 6))
```

```
            black   white
Oct 1973  884.050 1419.78
Nov 1973  919.329 1503.55
Dec 1973  930.350 1536.62
Jan 1974 1102.310 1629.22
Feb 1974 1150.810 1737.24
Mar 1974 1093.490 1629.22
Apr 1974 1117.740 1620.40
May 1974 1168.450 1671.11
Jun 1974 1117.740 1578.51
```

8.1.1 Task 1

Change the names of the variables `black` and `white` to `black_pepper` and `white_pepper`.

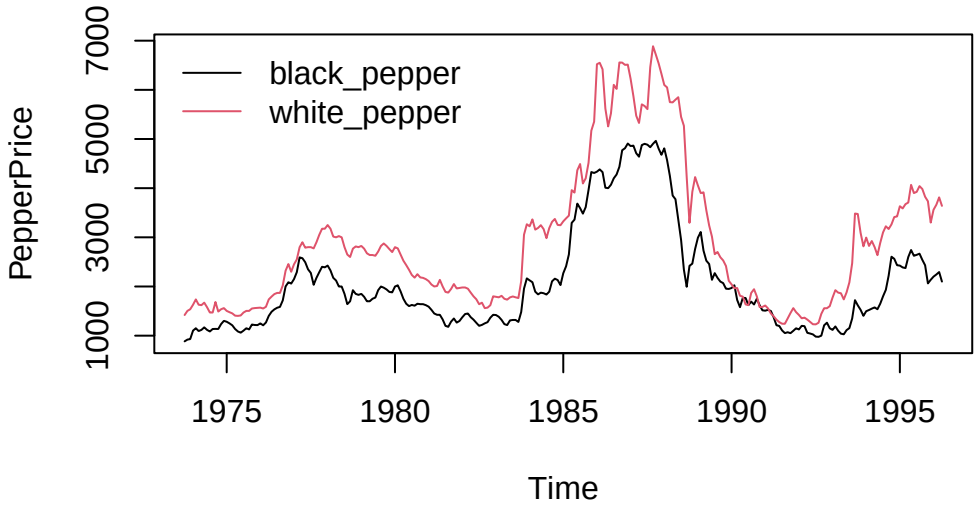
```
colnames(PepperPrice) <- c("black_pepper", "white_pepper")
```

8.1.2 Task 2

Provide a time-series plot of the `white_pepper` and `black_pepper` prices.

Let us start by plotting the data:

```
plot(PepperPrice, plot.type = "single", col = 1:2)
legend("topleft", c("black_pepper", "white_pepper"),
       bty = "n", col = 1:2, lty = rep(1,2))
```



- `plot(PepperPrice, ...)` is the base R plot function for time series objects.
- `plot.type = "single"` ensures that multiple time series within `PepperPrice` are plotted on the same graph (rather than separate subplots).
- `col = 1:2` assigns different colors to the time series (we use R defaults above)

The second line after `plot` is about legend.

- `legend("topleft", ...)` places the legend in the top-left corner of the plot.
- `c("black", "white")` are the legend labels for the two time series.
- `bty = "n"` removes the legend box (makes it look cleaner).
- `col = 1:2` matches the line colors (black and red).
- `lty = rep(1,2)` sets line type to solid (`lty = 1`) for both series.

8.1.3 Task 3

Find the order of integration of `white_pepper` and `black_pepper` prices.

8.1.3.1 Guidance

Apply Dickey Fuller test without trend in test regression

```
adf.test(PepperPrice[, "white_pepper"])
```

```
    Augmented Dickey-Fuller Test

data:  PepperPrice[, "white_pepper"]
```

```
Dickey-Fuller = -1.6001, Lag order = 6, p-value = 0.7444
alternative hypothesis: stationary
```

```
adf.test(PepperPrice[, "black_pepper"])
```

```
    Augmented Dickey-Fuller Test

data:  PepperPrice[, "black_pepper"]
Dickey-Fuller = -1.6434, Lag order = 6, p-value = 0.7262
alternative hypothesis: stationary
```

We cannot reject the null hypothesis of unit root.

Let us apply Augmented Dickey-Fuller (ADF) Test with 12 lags (because of the monthly frequency of the data, it likely to observe Autocorrelation up to 12 lags). Here, we have selected the lag length with some intuition. We will use another R command to choose the optimum lag length with the help of AIC (Akaike Information Criterion).

```
adf.test(PepperPrice[, "white_pepper"], k = 12)
```

```
    Augmented Dickey-Fuller Test

data:  PepperPrice[, "white_pepper"]
Dickey-Fuller = -2.5763, Lag order = 12, p-value = 0.3332
alternative hypothesis: stationary
```

```
adf.test(PepperPrice[, "black_pepper"], k = 12)
```

```
    Augmented Dickey-Fuller Test

data:  PepperPrice[, "black_pepper"]
Dickey-Fuller = -2.3677, Lag order = 12, p-value = 0.4211
alternative hypothesis: stationary
```

Both series are still with a unit root. They are non-stationary.

Since the plots did not show a clear deterministic trend, we can proceed by differencing. We re-apply the Augmented Dickey-Fuller Test on the first differenced series.

```
adf.test(diff(PepperPrice[, "white_pepper"]))
```

```
Warning in adf.test(diff(PepperPrice[, "white_pepper"])): p-value smaller than
printed p-value
```

```

    Augmented Dickey-Fuller Test

data:  diff(PepperPrice[, "white_pepper"])
Dickey-Fuller = -5.8575, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary
```

```
adf.test(diff(PepperPrice[, "black_pepper"]))
```

```
Warning in adf.test(diff(PepperPrice[, "black_pepper"])): p-value smaller than
printed p-value
```

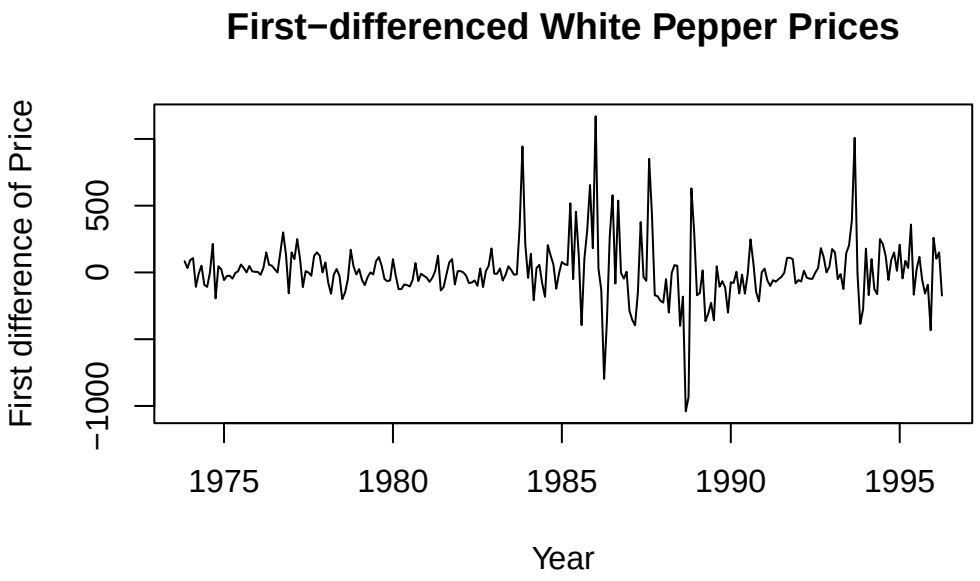
```

    Augmented Dickey-Fuller Test

data:  diff(PepperPrice[, "black_pepper"])
Dickey-Fuller = -4.973, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary
```

The differenced series are statonary for both indicators.

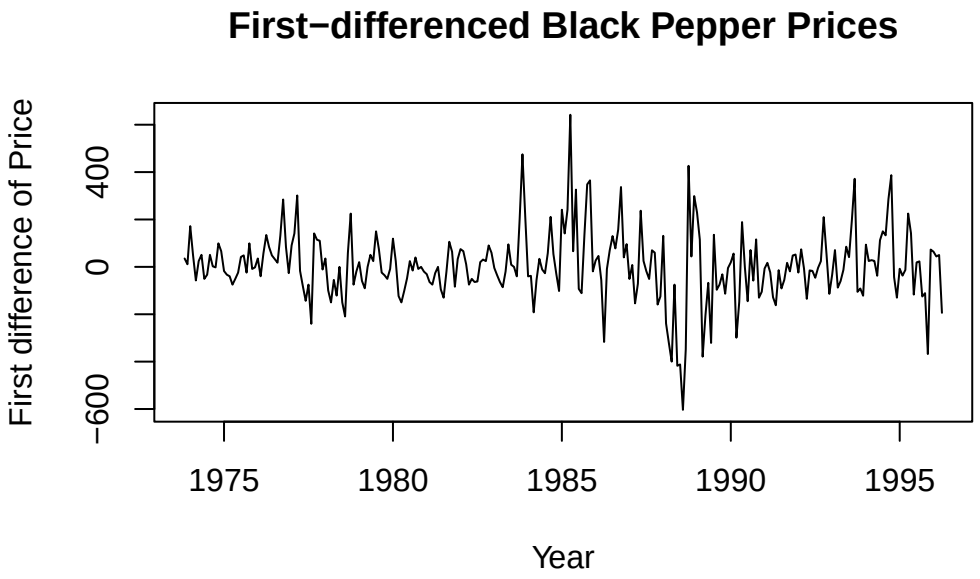
8.1.4 Task 4

Plot the first differenced series

```
# Plot the first differenced white pepper price
plot(diff(PepperPrice[, "white_pepper"]),
     xlab = "Year", ylab = "First difference of Price", main = "First-differenced White Pepp
```




```
# Plot the first differenced black pepper price
plot(diff(PepperPrice[, "black_pepper"]),
     xlab = "Year", ylab = "First difference of Price", main = "First-differenced Black Pepp
```



We can also see from the above plots that the first differenced series are stationary.

The white and black pepper prices are integrated of order 1, i.e. they are $I(1)$.

8.1.5 Task 5

Re-apply the ADF test by choosing the optimum lag length using AIC (Akaike Information Criterion). Set a maximum of 12 lags.

8.1.5.1 Guidance

We use the `vars` R package to find the optimum lag length for our ADF test.

```
lag_selection <- VARselect(PepperPrice[, "white_pepper"], lag.max = 12, type = "const")
# print the results
head(lag_selection)
```

```
$selection
AIC(n)  HQ(n)  SC(n) FPE(n)
    12      2      2     12

$criteria
                1           2           3           4           5           6
AIC(n)    10.88957    10.80854    10.81078    10.81678    10.82324    10.82629
HQ(n)     10.90061    10.82510    10.83286    10.84439    10.85637    10.86494
SC(n)     10.91704    10.84973    10.86571    10.88544    10.90564    10.92242
FPE(n) 53614.21642 49441.05649 49552.12848 49850.44270 50174.03385 50327.43278
                7           8           9          10          11          12
AIC(n)    10.81944    10.82684    10.81711    10.79352    10.79293    10.78408
HQ(n)     10.86362    10.87654    10.87233    10.85425    10.85919    10.85586
SC(n)     10.92931    10.95044    10.95444    10.94458    10.95773    10.96260
FPE(n) 49984.31397 50355.88525 49868.81434 48706.52773 48678.78476 48250.47433
```

```
# Extract suggested lag from AIC criterion
optimum_lag <- lag_selection$selection["AIC(n)"]
print(optimum_lag)
```

```
AIC(n)
    12
```

```
# Run ADF test with the selected lag
adf.test(PepperPrice[, "white_pepper"], k = optimum_lag)
```

```
    Augmented Dickey-Fuller Test

data:  PepperPrice[, "white_pepper"]
Dickey-Fuller = -2.5763, Lag order = 12, p-value = 0.3332
alternative hypothesis: stationary
```

You may replicate the above for the black pepper price and also for the first-differenced series.

Also, you may replicate the above analysis using logarithmic series.

8.1.6 Task 6

Replicate the ADF test including trend in the test regression.

Please note that this test is only for exercise purposes (to show how it could be done in R).The plot of our white and pepper price series do not suggest existence of a deterministic trend. Hence, the results obtained below will not be used in the analysis further.

8.1.6.1 Guidance

We use R's `urca` library.

```r
# library(urca)
# Run the ADF test with trend
adf_trend <- ur.df(PepperPrice[, "white_pepper"], type = "trend", lags = 1)

# Print test results
summary(adf_trend)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression trend


Call:
lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-952.60  -92.97  -18.41   65.97 1155.13

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 50.491863  34.431212   1.466   0.1437
z.lag.1     -0.017740   0.009639  -1.841   0.0668 .
tt           0.043479   0.177088   0.246   0.8062
z.diff.lag   0.290817   0.058798   4.946 1.35e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 218.1 on 265 degrees of freedom
Multiple R-squared:  0.09145,   Adjusted R-squared:  0.08117
F-statistic: 8.892 on 3 and 265 DF,  p-value: 1.235e-05


Value of test-statistic is: -1.8406 1.2023 1.7202

Critical values for test statistics:
      1pct  5pct 10pct
tau3 -3.98 -3.42 -3.13
phi2  6.15  4.71  4.05
phi3  8.34  6.30  5.36
```

```
# type can take on: none; drift; trend
```

In the next section, we will check for a cointegrating relationship between these variables.

9 Cointegration: Engle-Granger Test

```
library(AER)
```

```
Loading required package: car
```

```
Loading required package: carData
```

```
Loading required package: lmtest
```

```
Loading required package: zoo
```

```

Attaching package: 'zoo'
```

```
The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
Loading required package: sandwich
```

```
Loading required package: survival
```

```
library(tseries)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

```
library(vars) # to select optimal lag length
```

```
Loading required package: MASS
```

```
Loading required package: strucchange
```

```
Loading required package: urca
```

```
library(urca)
```

9.1 Example: Pepper Price

We continue from the previous section, where we concluded that the white and black pepper prices are $I(1)$.

In this section, we test whether they have a cointegrating relationship.

<div style="color:green>

There are two conditions for cointegration:

1. The time-series should be integrated of the same order
2. There is a stationary linear relationship between the variables.

In the previous section, we established that the series are integrated of the same order. The next task finds out whether they have a stationary linear relationship.

```
data("PepperPrice")
colnames(PepperPrice) <- c("black_pepper", "white_pepper")
```

9.1.1 Task 1

Regress `white_pepper` on `black_pepper` and obtain the residuals.

9.1.1.1 Guidance

```
# regression of white_pepper on black_pepper
m_1 <- lm(white_pepper ~ black_pepper, as.data.frame(PepperPrice))
```

```
# Obtain residuals
resid_m1 <- residuals(m_1)
```

We need to perform a unit root test on the residuals. But the test regression should not include a constant term. The `adf.test()` function we used above does not have an option to remove the constant from test regression. Hence, we revert to `urca` package.

```
library(urca)
adf_no_constant <- ur.df(resid_m1, type = "none", lags = 0)
summary(adf_no_constant)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression none


Call:
lm(formula = z.diff ~ z.lag.1 - 1)

Residuals:
     Min       1Q   Median       3Q      Max
-1316.48   -79.12    -4.80    74.85  1087.12

Coefficients:
        Estimate Std. Error t value Pr(>|t|)
z.lag.1 -0.11587    0.02916  -3.973 9.11e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 208.1 on 269 degrees of freedom
Multiple R-squared:  0.05543,   Adjusted R-squared:  0.05192
F-statistic: 15.79 on 1 and 269 DF,  p-value: 9.111e-05
```

```
Value of test-statistic is: -3.9733

Critical values for test statistics:
      1pct  5pct 10pct
tau1 -2.58 -1.95 -1.62
```

The null of unit root is rejected. Hence the residuals of model `m_1` are stationary. This implies a cointegrating relationship between `white_pepper` and `black_pepper` prices. Hence, we can use these variables and estimate a regression without the need for differencing.

9.1.2 Task 2

Estimate the Error Correction Mechanism (ECM) between the two types of pepper prices and interpret your results.

9.1.2.1 Guidance

The ECM will be in the form of a first differenced regression of the two variables while also including the lagged error term from the regression we estimated above.

First, let's convert the residuals we obtained above to a `ts` (time-series) object while matching the dates to the `PepperPrice` object

```
# Convert residuals we obtained above to a ts object (match ts object PepperPrice)
resid_m1 <- ts(resid_m1, start = start(PepperPrice),
               frequency = frequency(PepperPrice))
```

We then create the difference and lag variables that we need for an Error Correction Mechanism:

$$\Delta white\_pepper_t = \alpha_0 + \alpha_1 \Delta black\_pepper_t + \alpha_2 resid\_m1_{t-1} + \epsilon_t$$

```
d_white <- diff(PepperPrice[, "white_pepper"])  # First difference of white_pepper
d_black <- diff(PepperPrice[, "black_pepper"])  # First difference of black_pepper
lag_resid <- stats::lag(resid_m1, -1)  # Lagged residuals (one period back)
```

Each of the above are separate `ts` objects. we will bind them under one. I will save this as `PepperPrice_2`.

```r
PepperPrice_2 <- cbind(d_white, d_black, lag_resid)
colnames(PepperPrice_2) <- c("d_white", "d_black", "lag_resid") # assign column names
```

We can now run our ECM:

```r
ecm <- lm(d_white ~ d_black + lag_resid, data = as.data.frame(PepperPrice_2))
summary(ecm)
```

```
Call:
lm(formula = d_white ~ d_black + lag_resid, data = as.data.frame(PepperPrice_2))

Residuals:
     Min       1Q   Median       3Q      Max
-1059.66   -62.79   -15.21    61.56  1092.60

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.60426   11.65811   0.395    0.693
d_black      0.72575    0.07993   9.080  < 2e-16 ***
lag_resid   -0.12866    0.02689  -4.784 2.84e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 191.5 on 267 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.2949,    Adjusted R-squared:  0.2896
F-statistic: 55.83 on 2 and 267 DF,  p-value: < 2.2e-16
```

In the above output, we are interested with the `lag_resid` term. The coefficient of this term needs to be negative for a valid error correction mechanism. We see that the coefficient is statistically significant and is negative. `-0.12866` shows that in the case of a deviation from the long-run equilibrium between the white pepper and black pepper prices, 12.9% of this deviation is adjusted in the next period. Here, we have monthly data, so 12.9% of the deviatiton from the long-run equilibrium is adjusted in the next month. The system appears to come back to its equilibrium state in less than a year.

Part VI

Seminar 6 (25 February 2025)

10 Data Visualisation Using `ggplot2`

10.1 Example 1: Scatter plot with wage data

The exercises below are a selection from the "introduction to regression analysis" section. Load necessary libraries

```
library(readxl)
library(ggplot2)
```

10.1.1 Task 1

10.1.1.1 Task

Import `wage.xls` data. Alternatively, you may download and work on the `wage-analysis` project file.

10.1.1.2 Guidance

Use `read_excel()` and `head()` functions.

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

# Import Excel data
wage2 <- read_excel("./assets/data/wage2.xls", sheet = "wage2")
```

10.1.2 Task 2

10.1.2.1 Task

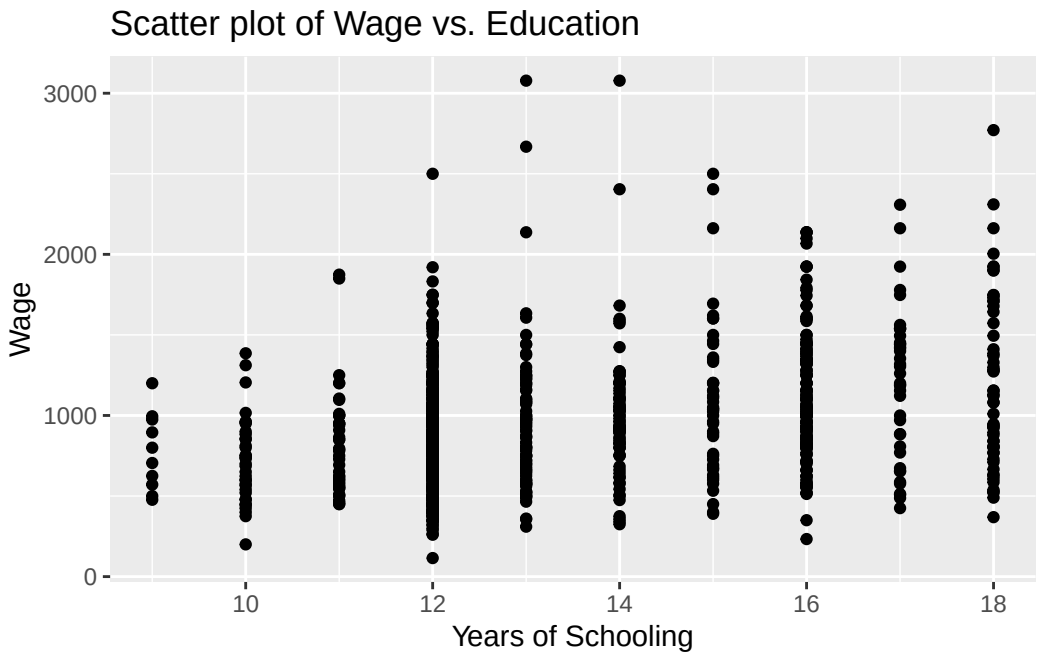
Examine the relationship between education and wage using a scatter plot.

10.1.2.2 Guidance

We use the `ggplot2` package to draw plots.

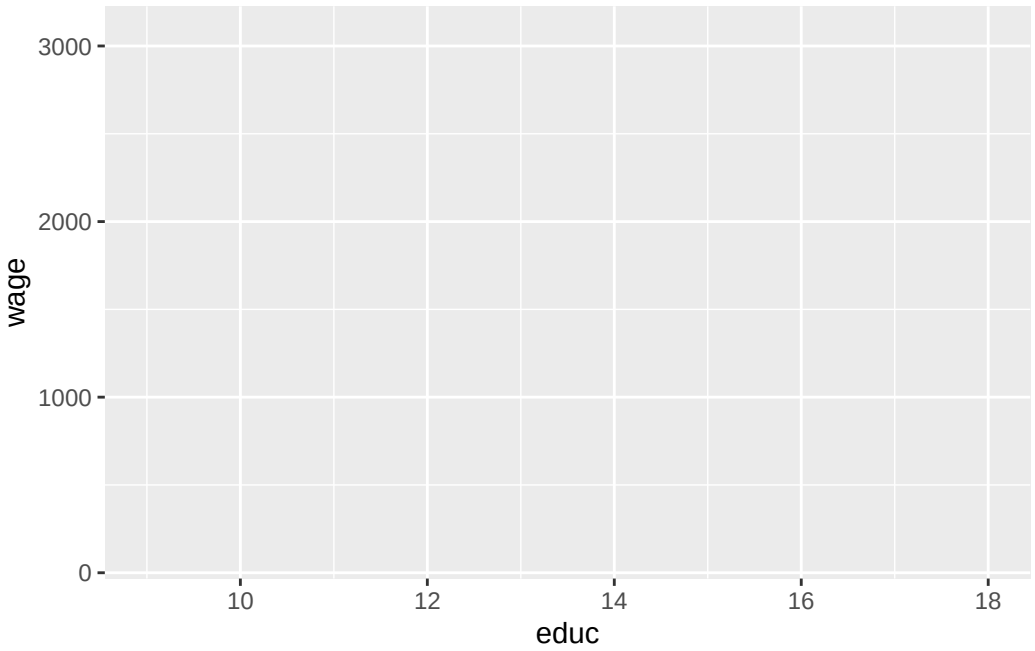
Education is expected to have a positive impact on wage. In our scatter plot, `educ` will be on the horizontal-axis while `wage` will be on the vertical-axis.

```
# Scatter plot
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  labs(title = "Scatter plot of Wage vs. Education", x = "Years of Schooling", y = "Wage")
```



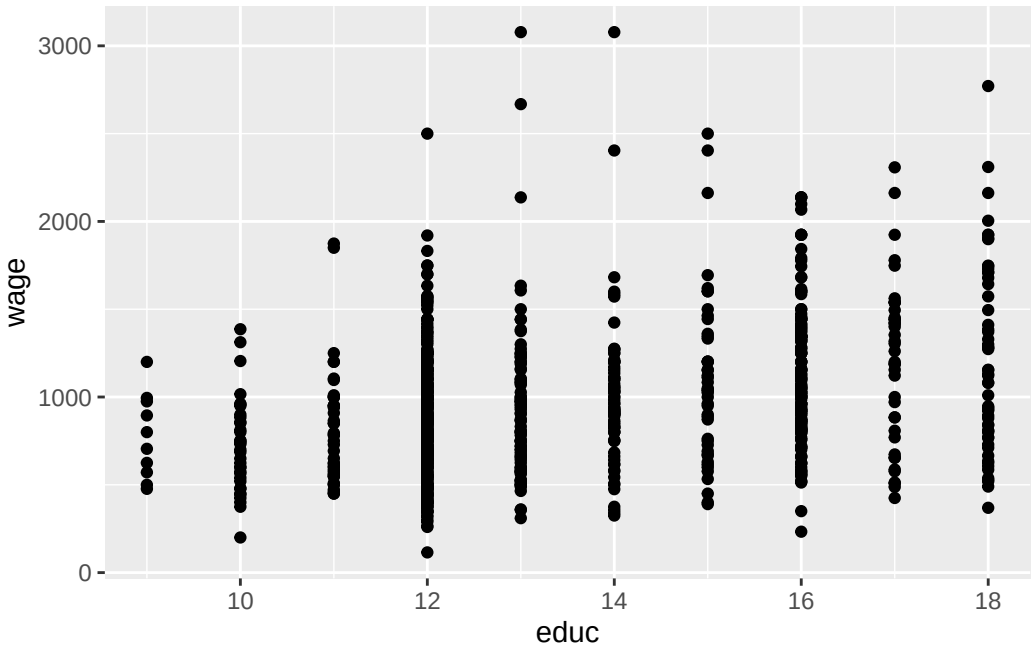
You see above the full set of lines to create this plot. But let us do this step by step to have a better understanding. First, we bring the `educ` and `wage` variables from the `wage2` data and position these on our plot.

```
ggplot(wage2, aes(x = educ, y = wage))
```



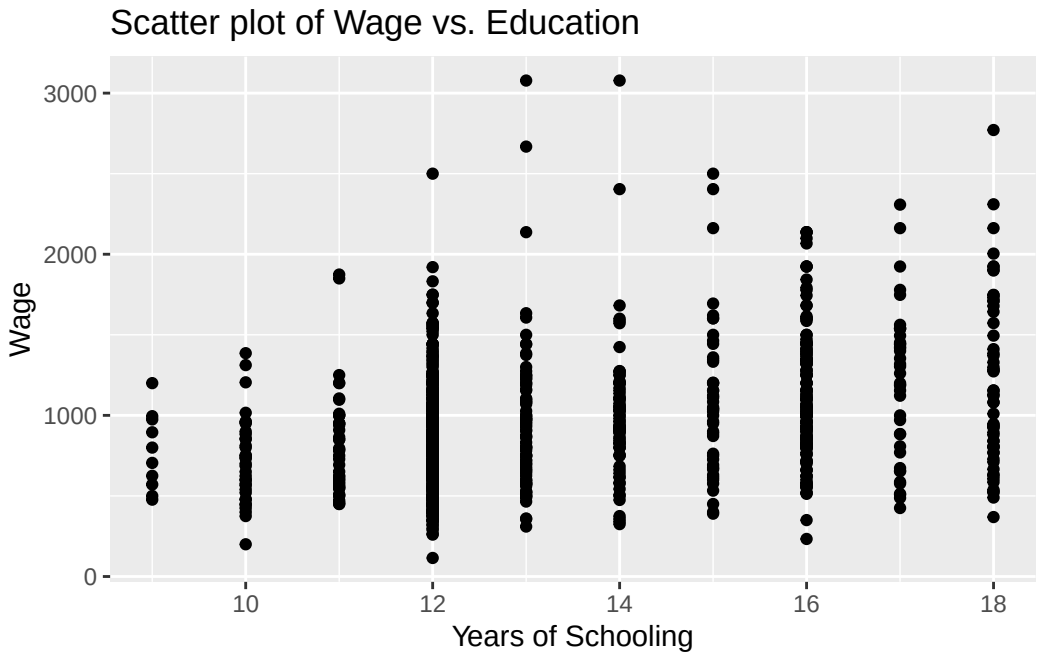
We then add (using the + sign), the observations in our data, represented by dots.

```
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point()
```



It is always good practice to give a title for your plot. Notice also that the horizontal and vertical axes above are labelled by the variable names. We may also replace these with proper definitions of the variables. This is to make it easier for the readers to understand your plots:

```
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  labs(title = "Scatter plot of Wage vs. Education", x = "Years of Schooling", y = "Wage")
```



10.1.3 Task 3

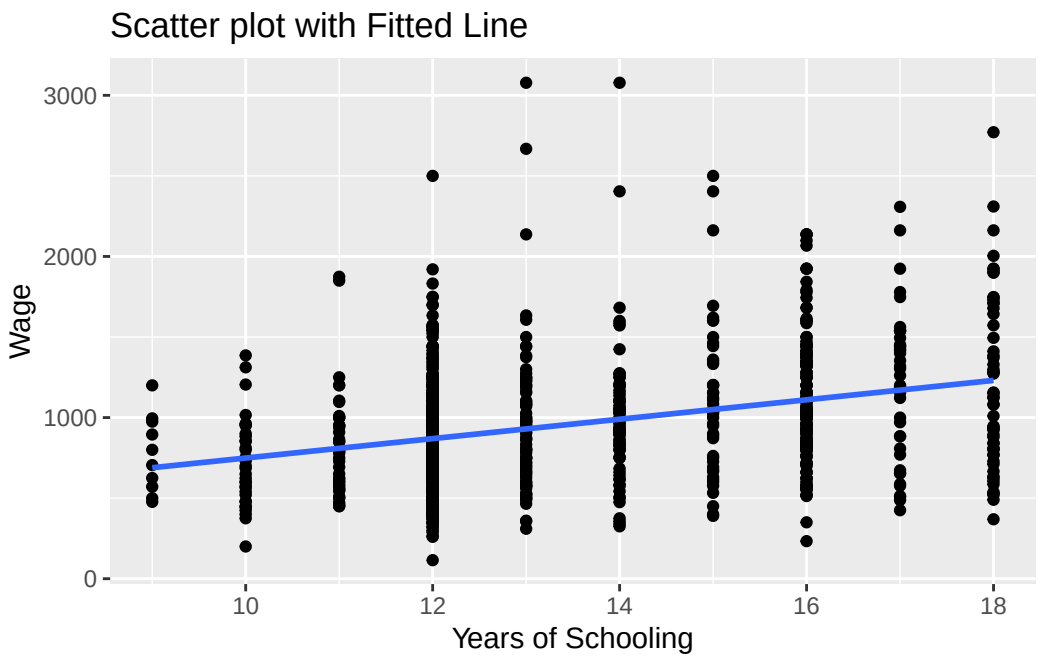
Fit a regression line to the scatterplot you created

10.1.3.1 Guidance

We will be adding the regression line to the scatter plot we produced above.

```
# Scatter plot with fitted line
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Scatter plot with Fitted Line", x = "Years of Schooling", y = "Wage")
```

```
`geom_smooth()` using formula = 'y ~ x'
```



the `geom_smooth(method = "lm")` asks R to add a line estimating a "linear model" (i.e. a regression) of wage on educ.

Note that we could save this plot as an object by assigning it a name on the left hand side of the command. We will do that below and name the plot as `scatter_wage_educ`.

Can you guess what the plot would look if we changed `se = FALSE` to `se = TRUE` above? We can also try that below:

```
# Scatter plot with fitted line
scatter_wage_educ <- ggplot(wage2, aes(x = educ, y = wage)) +
                        geom_point() +
                        geom_smooth(method = "lm", se = TRUE) +
                        labs(title = "Scatter plot with Fitted Line", x = "Years of Schooling
print(scatter_wage_educ)
```

```
`geom_smooth()` using formula = 'y ~ x'
```

We could also add this sample regression line by saving predictions after estimation of a wage regression and using these predictons.

We use the `lm()` function to estimate linear regression models.

```
# Linear regression
model_1 <- lm(wage ~ educ, data = wage2)
summary(model_1)
```

```
Call:
lm(formula = wage ~ educ, data = wage2)

Residuals:
    Min      1Q  Median      3Q     Max
-877.38 -268.63  -38.38  207.05 2148.26

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  146.952     77.715   1.891   0.0589 .
educ          60.214      5.695  10.573   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 382.3 on 933 degrees of freedom
Multiple R-squared:  0.107, Adjusted R-squared:  0.106
F-statistic: 111.8 on 1 and 933 DF,  p-value: < 2.2e-16
```

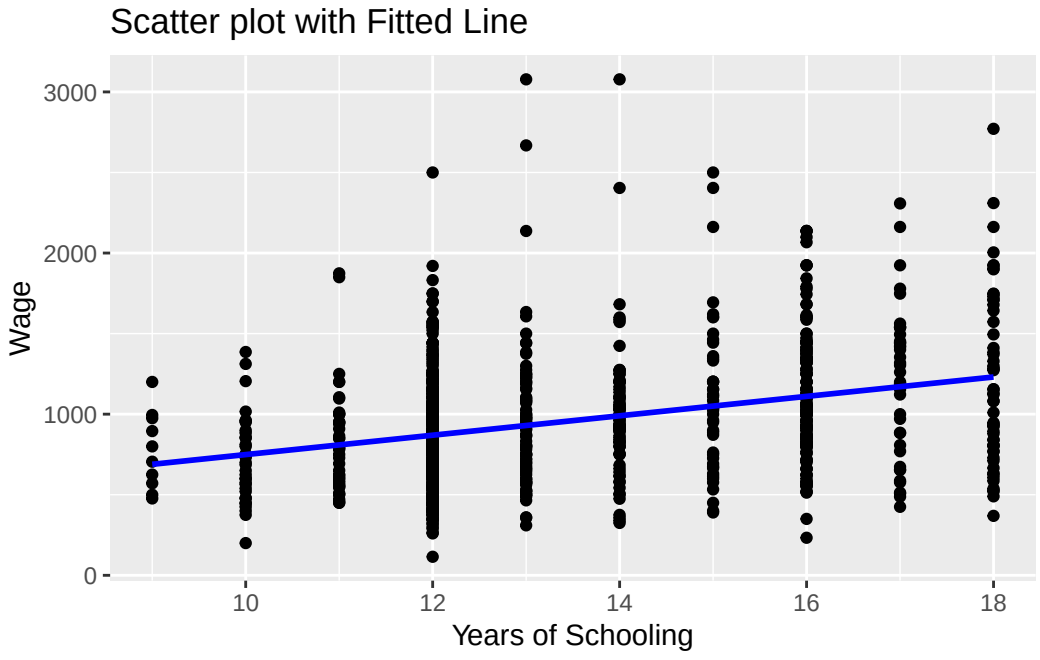
Using the regression model above, we can predict what the wage would be for given values of education (how much do we expect the wage would be for given years of schooling).

```
# Save predicted values under name wage_hat
wage2$wage_hat <- predict(model_1)
```

We can now add the estimated regression line to the wage-education scatter plot.

```
# Scatter plot with fitted line
# we add the wage_hat variable
ggplot(wage2, aes(x = educ, y = wage)) +
  geom_point() +
  geom_line(aes(y = wage_hat), color = "blue", size = 1) +
  labs(title = "Scatter plot with Fitted Line", x = "Years of Schooling", y = "Wage")
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```



Note that we used `geom_line()` this time to add a line plot of an already existing variable in the data set.

- `ggplot(wage2, aes(x = educ, y = wage))` creates a canvas, a plot area with educ at the horizontal and wage at the vertical axis
- **`geom_point()`** adds a scatterplot of `wage` against `educ`.
- **`geom_line(aes(y = wage_hat))`** adds the line for the predicted `wage_hat` values. The `aes(y = wage_hat)` ensures the line graph uses `wage_hat` on the y-axis while sharing the x-axis (`educ`).
- **`color`** and **`size`** are optional for styling the line. Try experimenting with these and observe the changes.

10.2 Example 2: Histogram of error term using wage2 data

The following example is taken from "multiple regression and diagnostic checks" section.

10.2.1 Task 4

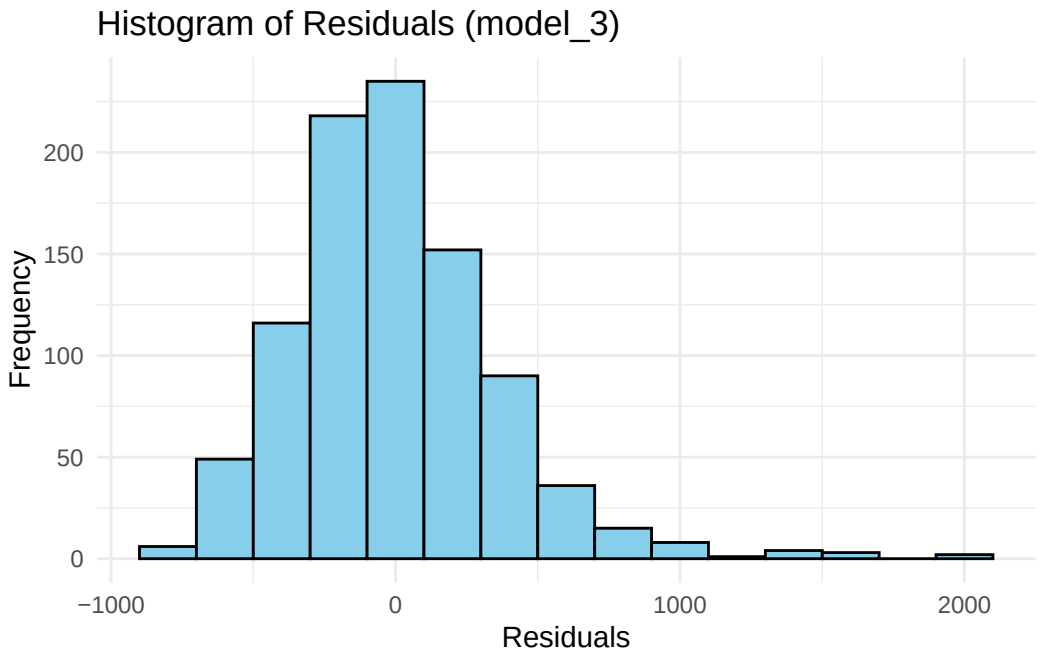
Estiamte a multiple regression of wage using `IQ`, `educ`, `exper`, `urban`. Provide a histogram of the error terms.

10.2.1.1 Guidance

```
model_3 <- lm(wage ~ IQ + educ + exper + urban, data = wage2)
wage2$resid_m3 <- residuals(model_3)
```

Plot the residuals to see the distribution.

```
ggplot(wage2, aes(x = resid_m3)) +
  geom_histogram(binwidth = 200, fill = "skyblue", color = "black") +
  labs(title = "Histogram of Residuals (model_3)", x = "Residuals", y = "Frequency") +
  theme_minimal()
```

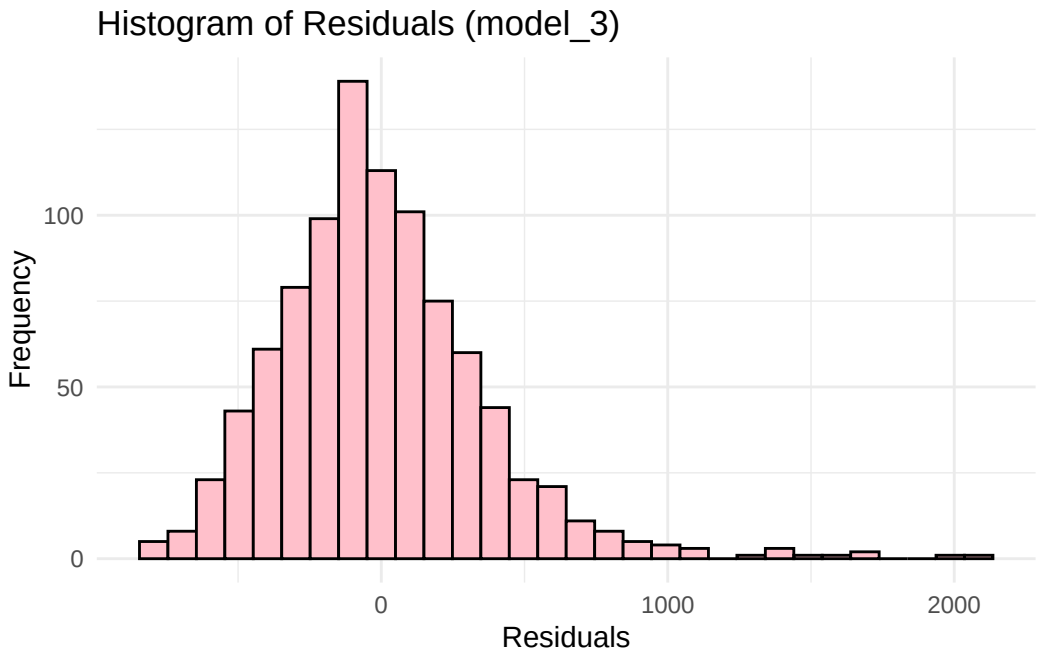


- **aes(x = resid)** specifies the residuals as the variable for the x-axis.
- **geom_histogram()** is used to create the histogram:
  - **binwidth = 200** controls the width of the bins. You can adjust this depending on how detailed you want the histogram to be.
  - **fill** sets the color inside the bars, and **color** adds a border around them for better visibility.
- **labs()** adds labels for the title and axes.
- **theme_minimal()** gives a clean, simple look to the plot - try the plot with and without this.

You may also let **ggplot** choose the number of bins automatically:

```
ggplot(wage2, aes(x = resid_m3)) +
  geom_histogram(fill = "pink", color = "black") +
  labs(title = "Histogram of Residuals (model_3)", x = "Residuals", y = "Frequency") +
  theme_minimal()
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



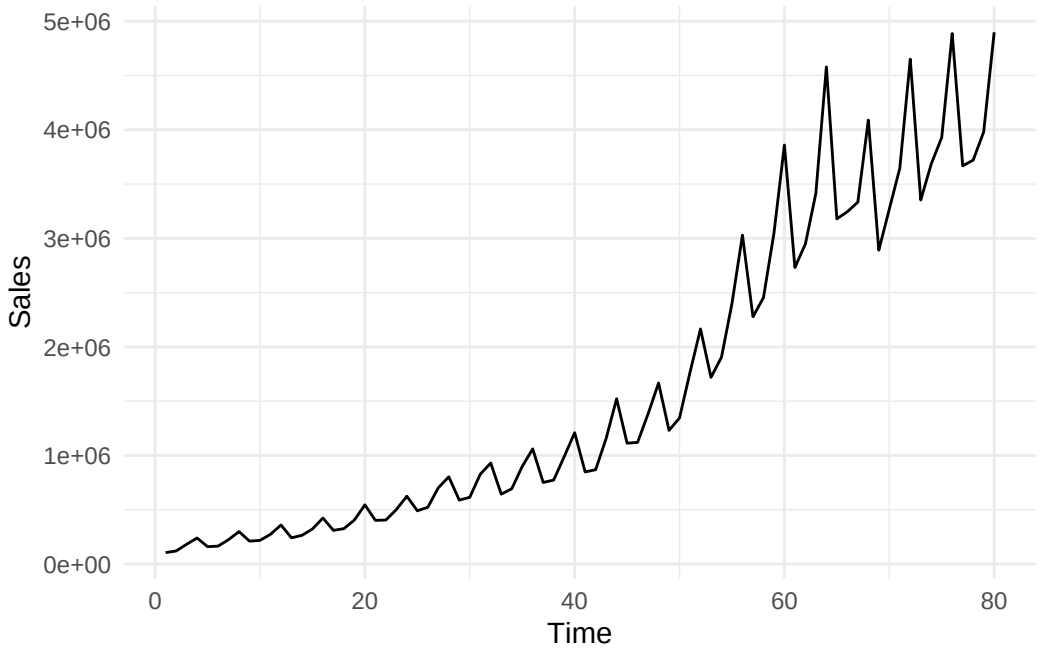
10.3 Example 3: Line plots using Gap_sales data

The following examples are from the "introduction to time series analysis" section.

Start by reading the data

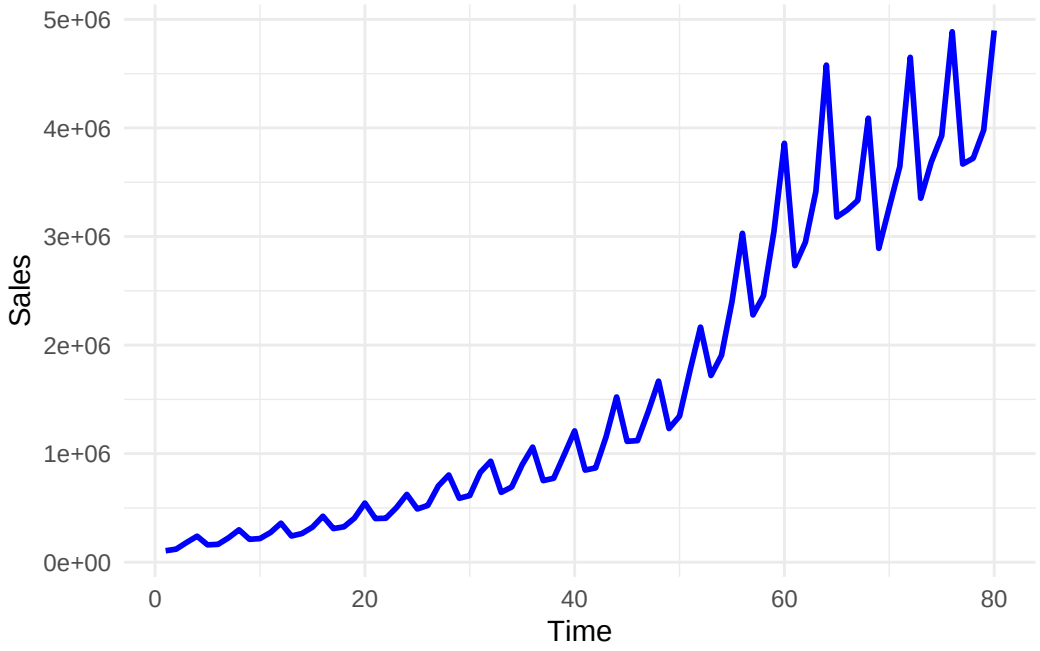
```
df <- read.csv("~/Desktop/R-workshops/assets/data/GAP_Sales.csv", stringsAsFactors=TRUE)
```

```
ggplot(df, aes(x = Time, y = Sales)) +
  geom_line() +
  theme_minimal()
```



You may add some color and change the size of the line

```
ggplot(df, aes(x = Time, y = Sales)) +
  geom_line(color = "blue", size = 1) +
  theme_minimal()
```



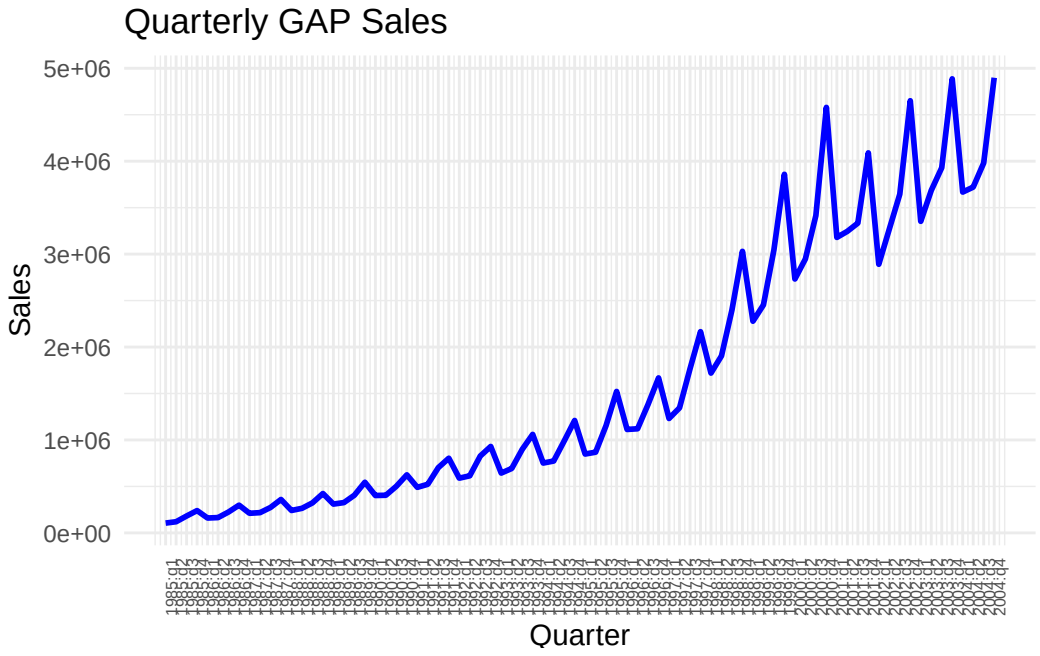
In the above plot, although we can see the pattern of the Sales variable quite clearly, the Time variable labels fail to show us the respective quarter values. We may change these labels by using the following lines of code.

We first define labels to correspond to each data point

```
# First, create a new column for formatted quarter labels
df$Quarter_label <- paste0(df$Year, ":", df$quarter)
```

Check the values of `Quarter_label` in the `df`. You will see that it goes on like 1985:q1, 1985:q2, and so on. We may now use these labels instead of the values of the `Time` variable.

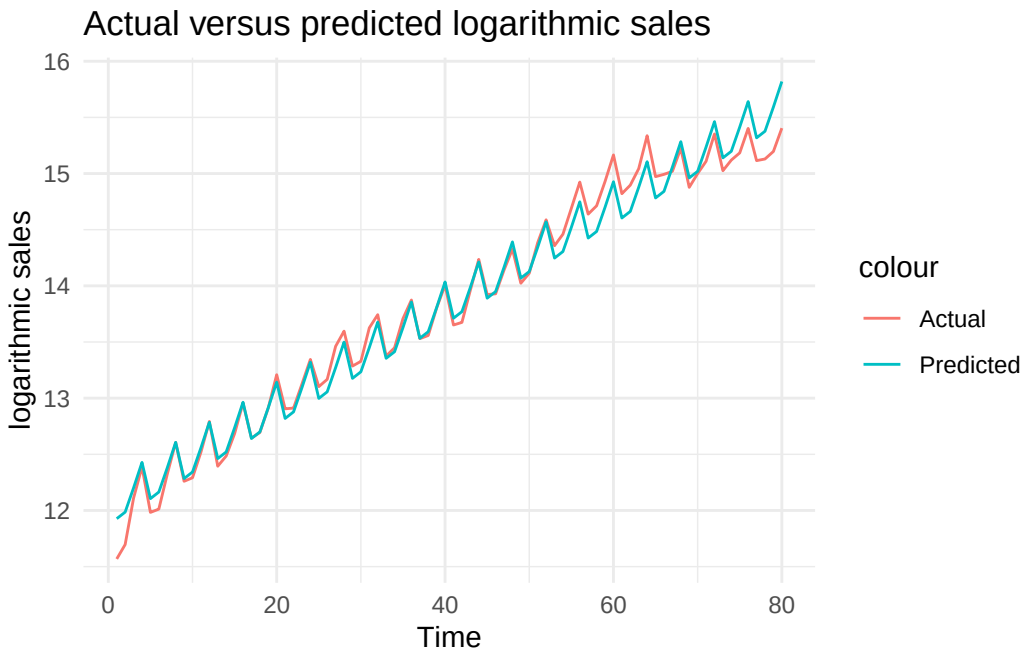
```
ggplot(df, aes(x = Time, y = Sales)) +
  geom_line(color = "blue", size = 1) +
  scale_x_continuous(
    breaks = df$Time,  # Position the breaks at each quarter, i.e. at each value of Time
    labels = df$Quarter_label  # Label each point using Quarter_label variable created above
  ) + # provide a title and axes labels below
  labs(title = "Quarterly GAP Sales", x = "Quarter", y = "Sales") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, size=6))  # Rotate labels for better readabili
```



```r
# Estimate the model using trend and quarter dummies
df$ln_sales = log(df$Sales)
model_3 <- lm(ln_sales ~ Time + Q2 + Q3 + Q4, data = df)
# Obtain predictions from model_3
df$ln_sales_hat_m3 <- predict(model_3)
```

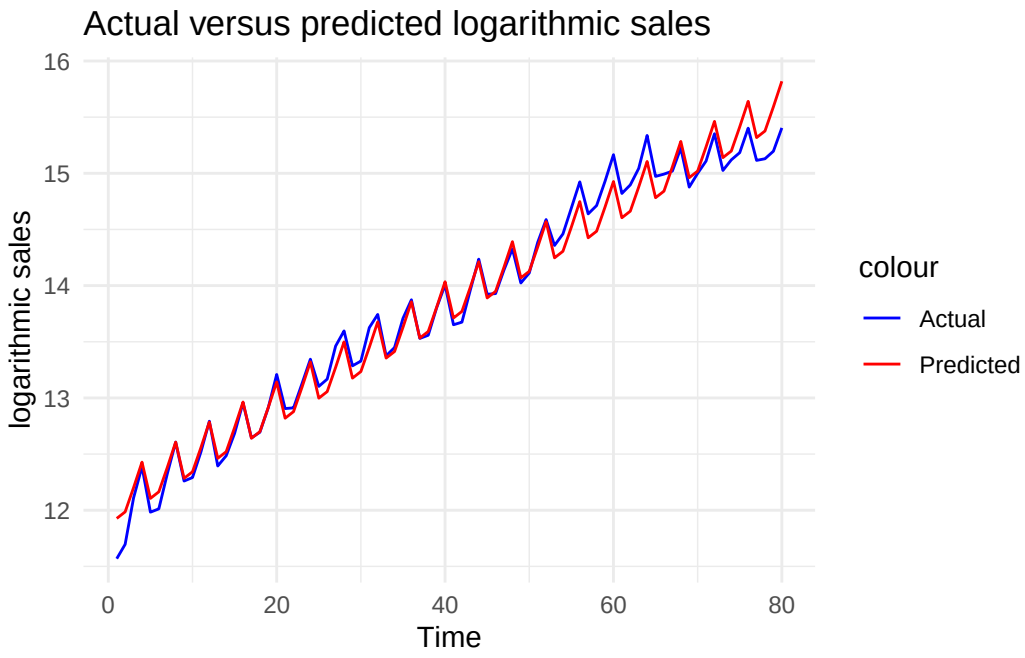
Plot the actual values against predictions to see the improvement by the inclusion of the quarter

```r
# Plot actual versus predicted log sales
ggplot(df, aes(x = Time)) +
  geom_line(aes(y = ln_sales, color = "Actual")) +
  geom_line(aes(y = ln_sales_hat_m3, color = "Predicted")) +
  theme_minimal() +
  labs(title = "Actual versus predicted logarithmic sales", x = "Time", y = "logarithmic sale
```



Change the color of the lines

```r
ggplot(df, aes(x = Time)) +
  geom_line(aes(y = ln_sales, color = "Actual")) +
  geom_line(aes(y = ln_sales_hat_m3, color = "Predicted")) +
  scale_color_manual(values = c("Actual" = "blue", "Predicted" = "red")) +
  theme_minimal() +
  labs(title = "Actual versus predicted logarithmic sales", x = "Time", y = "logarithmic sale
```


Actual versus predicted logarithmic sales
logarithmic sales
16
15
14
13
12
0
20
40
60
80
Time
colour
Actual
Predicted

11 Introduction to Panel Data Analysis

11.1 Example: European Countries Gasoline Consumption Data

Data is obtained from Abay Mulatu 316ECN Applied Econometrics lecture material, Coventry University.

We start by loading the required libraries.

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
library(Hmisc) # add labels to variables
```

```
Attaching package: 'Hmisc'

The following objects are masked from 'package:dplyr':

    src, summarize

The following objects are masked from 'package:base':

    format.pval, units
```

```
library(ggplot2)
library(dplyr)
```

We will start by looking a a sub-group of two countries: Italy and Denmark and then move on to do estimations using the complete data on 18 countries.

11.1.1 Task 1

Import the data into R.

```
df <- read.csv("~/Desktop/R-workshops/assets/data/gasoline-demand-IT-DK.csv", stringsAsFacto
#View(df)
```

For ease of typing, I called this data as `df`. In the code below, `df` will refer to the `gasoline-demand-IT-DK` data we imported.

11.1.2 Task 2

Label variables.

```
# library(Hmisc)

label(df$L_gas_cons_pcar) <- "Logarithm of gasoline consumption per car"
label(df$L_income_pc) <- "Logarithm of real income per capita"
label(df$L_gas_price) <- "Logarithm of real gasoline price per gallon"
label(df$L_cars_pc) <- "Logarithm of number of cars per capita"
```

11.1.3 Task 3

Produce a scatter plot of gasoline consumption by car versus income per capita separately for Italy and Denmark.

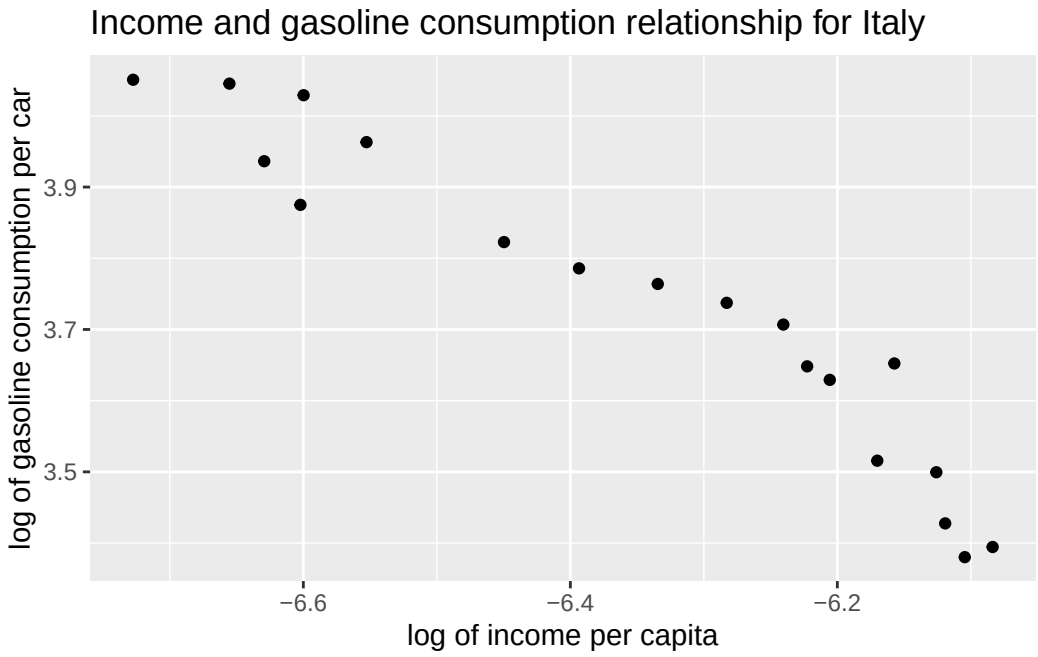
11.1.3.1 Guidance

Let's first create data frames for Italy and Denmark.

```r
df_Italy <- df %>%
  filter(country == "ITALY")

df_Denmark <- df %>%
  filter(country == "DENMARK")
```

Create scatterplots using `dplyr`

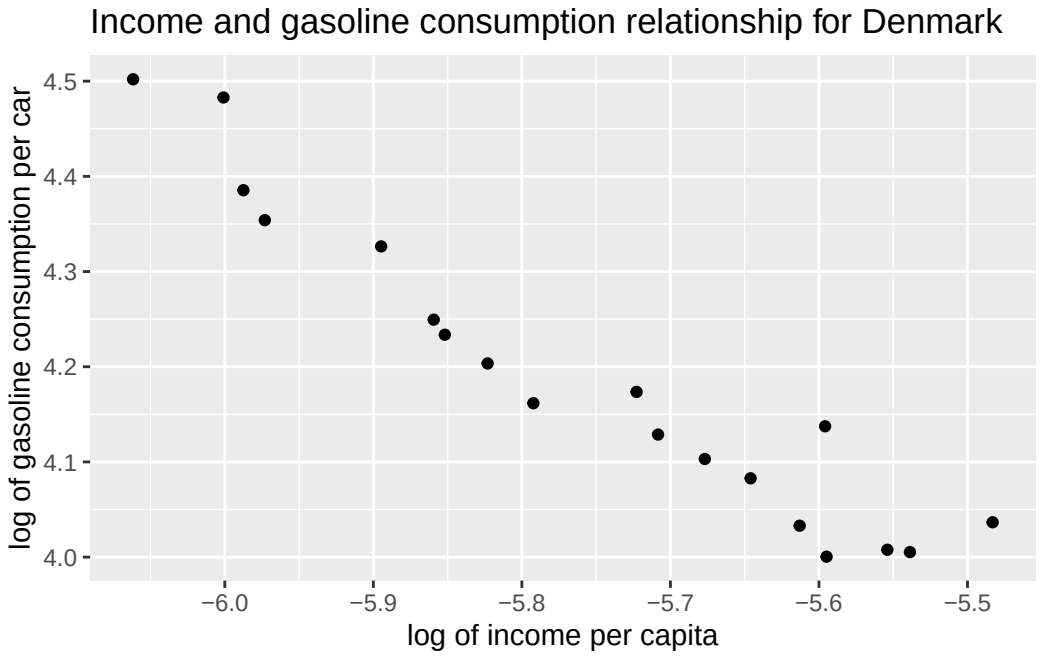
```r
ggplot(df_Italy, aes(x = L_income_pc, y = L_gas_cons_pcar)) +
  geom_point() +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```




```r
# Save the plot
# ggsave("./plots/panel-data-analysis/italy.png")
```

We can do the same for Denmark

```r
ggplot(df_Denmark, aes(x = L_income_pc, y = L_gas_cons_pcar)) +
  geom_point() +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```




```
# Save the plot
# ggsave("./plots/panel-data-analysis/denmark.png")
```

Both countries depict a negative relationship between income per capita and gasoline consumption by car. As the income per capita of Italy or Denmark increases, the gasoline consumption per car declines. What could be the reason of this pattern? Please provide a reasonable explanation.

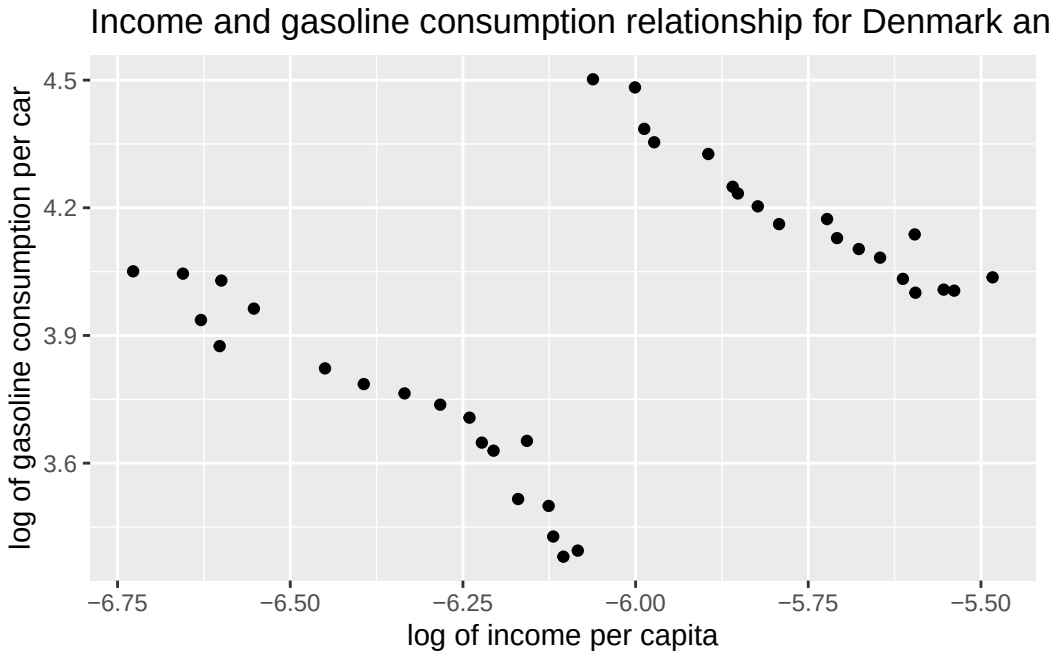
11.1.4 Task 4

Now, rather than focussing on each country separately, let's work with a panel data of these two countries.

Provide a scatter plot of the two indicators in both countries. Assign different colors to each country data points.

Let's first draw the scatter plot

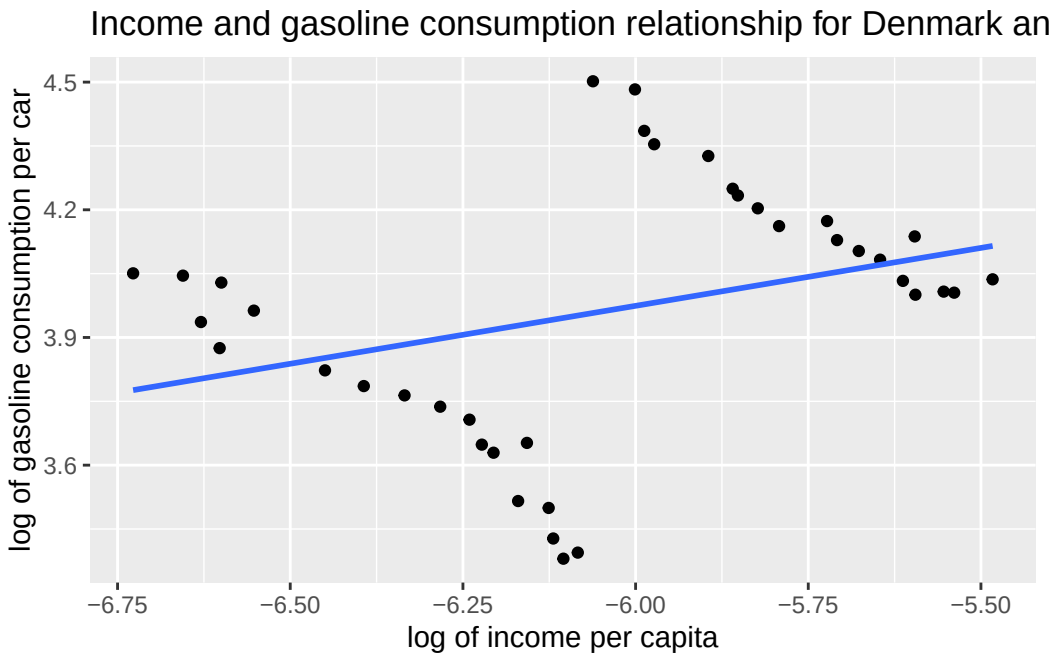
```
ggplot(df, aes(x = L_income_pc, y = L_gas_cons_pcar)) +
  geom_point() +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```



Add a regression line to the data points above:

```
ggplot(df, aes(x = L_income_pc, y = L_gas_cons_pcar)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```

```
`geom_smooth()` using formula = 'y ~ x'
```




```
# ggsave("./plots/panel-data-analysis/pooled-fit.png")
```

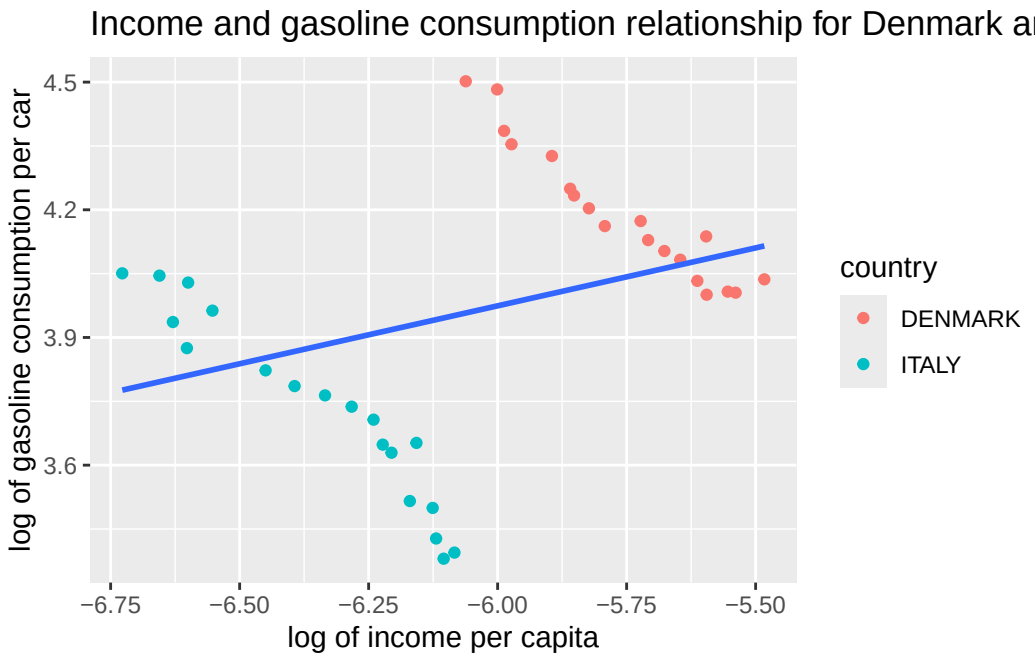
The above plot fit an OLS regression line to the data points in our scatter plot. But it seems like something is wrong. Separate scatter plots for Denmark and Italy revealed a negative relationship between log income per capita and log gasoline consumption per car whereas the above regression line suggests a positive relationship. Which one is correct?

Let us dive into this deeper. First, let us see which data points belong to which country and then move on from there.

In the scatter plot, add separate colors for each country

```
ggplot(df, aes(x = L_income_pc, y = L_gas_cons_pcar)) +
  geom_point(aes(color = country)) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```

```
`geom_smooth()` using formula = 'y ~ x'
```

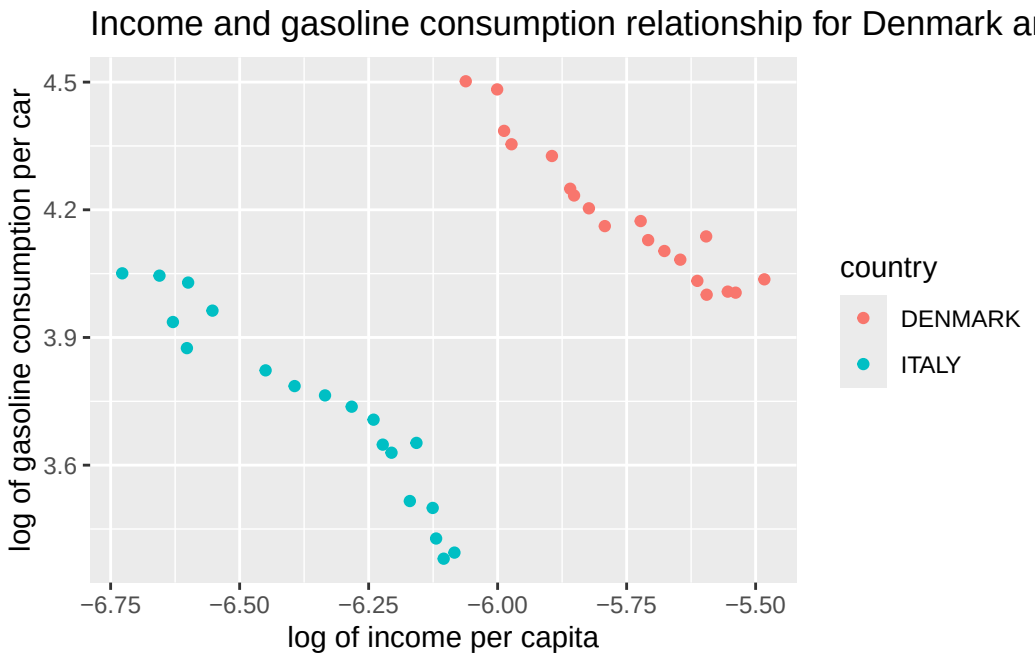



```
ggsave("./plots/panel-data-analysis/pooled-color-fit.png")
```

```
Saving 5.5 x 3.5 in image
`geom_smooth()` using formula = 'y ~ x'
```

We can see a clear separation of Denmark and Italy. If we were to use separate country samples, rather than a pooled sample of the two, we would fit the regression lines below. Let's start with the scatter and then add the regression lines:

```
ggplot(df, aes(x = L_income_pc, y = L_gas_cons_pcar, color = country)) +
  geom_point() +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```

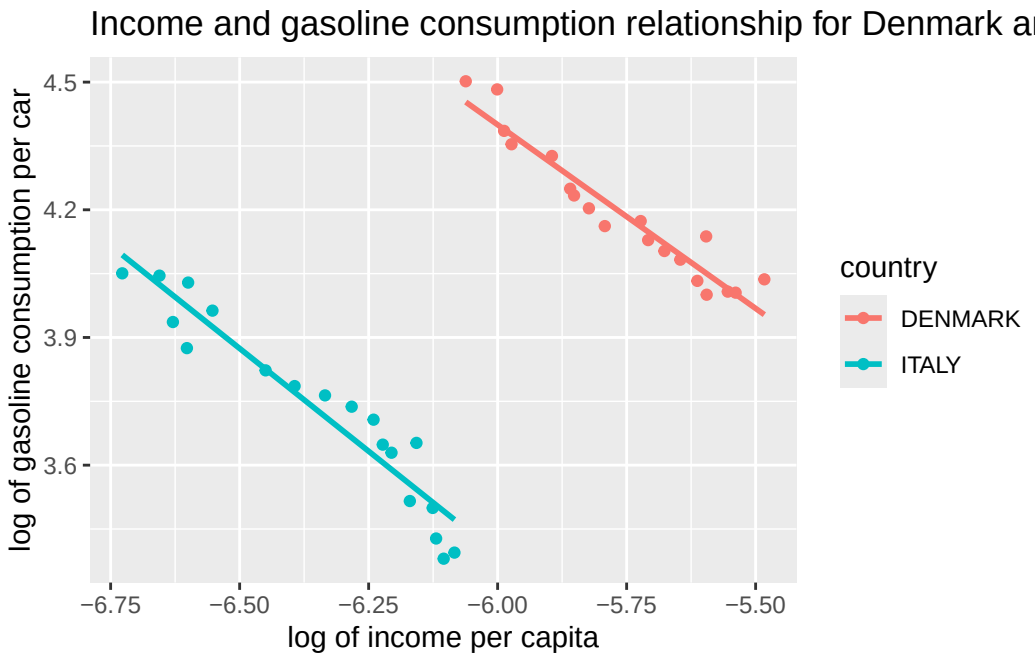



```
# ggsave("./plots/panel-data-analysis/italy-denmark.png")
```

Add regression lines to each plot

```
ggplot(df, aes(x = L_income_pc, y = L_gas_cons_pcar, color = country)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```

```
`geom_smooth()` using formula = 'y ~ x'
```




```
# ggsave("./plots/panel-data-analysis/italy-denmark-fits.png")
```

So, which pattern is correct?!

11.1.5 Task 5

Estimate a pooled regression of gasoline consumption on income per capita.

We will be using our `lm()` function as we have done before:

```
pooled_ols <- lm(L_gas_cons_pcar ~ L_income_pc, data = df)
summary(pooled_ols)
```

```
Call:
lm(formula = L_gas_cons_pcar ~ L_income_pc, data = df)

Residuals:
Logarithm of gasoline consumption per car
     Min       1Q   Median       3Q      Max
-0.56567 -0.14980  0.02645  0.20870  0.54445
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.6079     0.8004   7.006 3.22e-08 ***
L_income_pc   0.2723     0.1320   2.063   0.0464 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2878 on 36 degrees of freedom
Multiple R-squared:  0.1057,    Adjusted R-squared:  0.08085
F-statistic: 4.254 on 1 and 36 DF,  p-value: 0.04642
```

According to the results above, 1% increase in income per capita increases the gasoline consumption per car by 0.27%, on average.

11.1.6 Task 6

Create dummy variables representing each country in the sample and run the same regression above, this time including a country dummy (in the context of panel data anlysis, we will refer to this as the "country fixed effect").

Create country dummies using `ifelse()` function.

```
df$italy <- ifelse(df$country == "ITALY", 1, 0)
df$denmark <- ifelse(df$country == "DENMARK", 1, 0)
```

Re-run the above regression with `denmark` dummy. Can you explain why we do not include both `italy` and `denmark` but choose to include one of these countries only?

```
lsdv <- lm(L_gas_cons_pcar ~ L_income_pc + denmark, data = df)
summary(lsdv)
```

```
Call:
lm(formula = L_gas_cons_pcar ~ L_income_pc + denmark, data = df)

Residuals:
Logarithm of gasoline consumption per car
      Min        1Q    Median        3Q       Max
-0.121945 -0.038747 -0.001717  0.035970  0.101286

Coefficients:
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.14743    0.31063  -6.913 4.95e-08 ***
L_income_pc -0.92547    0.04887 -18.937  < 2e-16 ***
denmark      1.00963    0.03457  29.205  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.05795 on 35 degrees of freedom
Multiple R-squared:  0.9647,    Adjusted R-squared:  0.9627
F-statistic: 478.9 on 2 and 35 DF,  p-value: < 2.2e-16
```

We see that in Denmark, on average, gasoline consumption per car is around 174% higher than in Italy, holding income per capita levels constant. Can you figure out how did we get that number?

Looking at the coefficient of `L_income_pc`, we can say that a 1% increase in income per capita, on average, decreases the gasoline consumption per car by around 0.93%. This is a very different figure than what we obtained using pooled OLS.

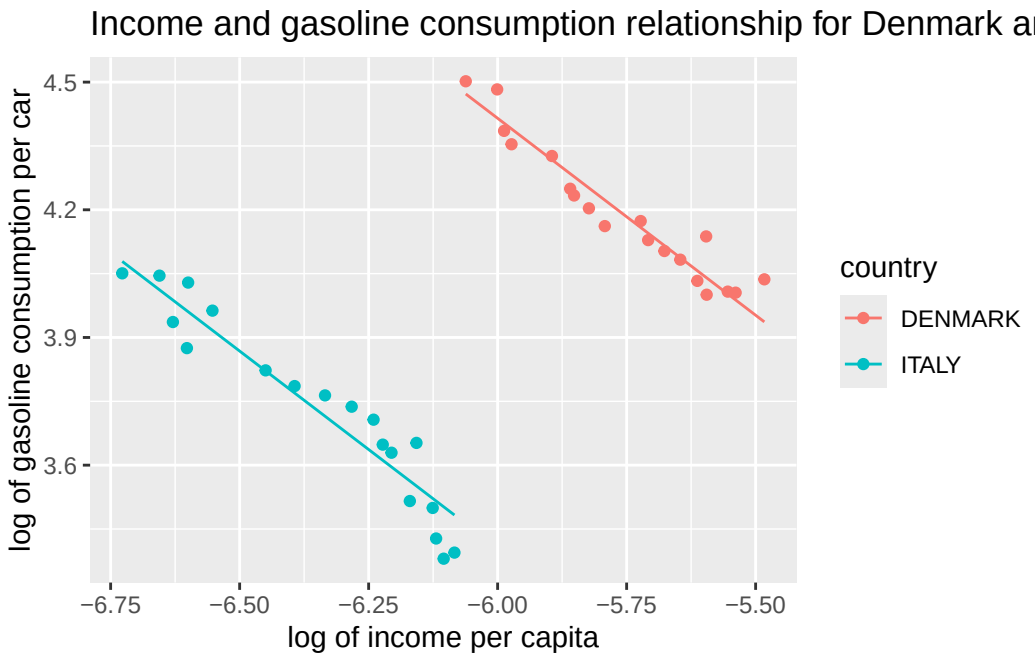
The model we estimated above using country dummies is using the Least Squares Dummy Variable approach.

11.1.7 Task 7

Plot predictions from the model estimated above.

```
df$lsdv_hat <- predict(lsdv)
```

```
ggplot(df, aes(x = L_income_pc, y = L_gas_cons_pcar, color = country)) +
  geom_point() +
  geom_line(aes(y = lsdv_hat)) +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```



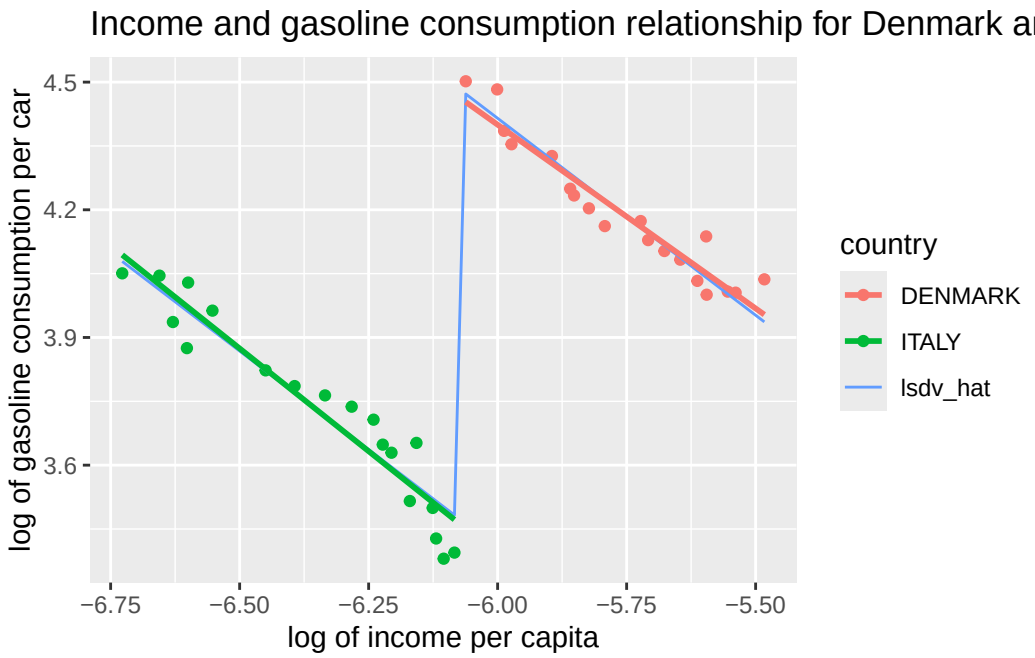

```
# ggsave("./plots/panel-data-analysis/italy-denmark-lsdv.png")
```

The LSDV approach assumes a command slope for Italy and Denmark, but captures the constant distance between them through a country dummy. Note that the slope coefficient will give us the average effect for all countries in sample and will be different than the coefficients we would obtain if we were to estimate separate regressions for each country.

How does this compare to individual regressions (i.e. if we were to estimate each separately?). In the case of this example, we get very similar slope coefficients. But we will see on a larger sample that this is not always the case.

```
ggplot(df, aes(x = L_income_pc, y = L_gas_cons_pcar, color = country)) +
  geom_point() +
  geom_line(aes(y = lsdv_hat, color = "lsdv_hat")) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "log of income per capita", y = "log of gasoline consumption per car", title = "In
```

```
`geom_smooth()` using formula = 'y ~ x'
```




```
# ggsave("./plots/panel-data-analysis/italy-denmark-lsdv.png")
```

11.1.8 Task 8

For `L_gas_cons_pcar` and `L_income_pc` variables, create deviations from within-group averages.

11.1.8.1 Guidance

We can use the `dplyr` package. We first create within group averages

```
# library(dplyr)
df <- df %>%
  group_by(country) %>%
  mutate(m_L_gas_cons_pcar = mean(L_gas_cons_pcar, na.rm = TRUE),
         m_L_income_pc = mean(L_income_pc, na.rm = TRUE)) %>%
  ungroup()
```

View the data to see what we have done above

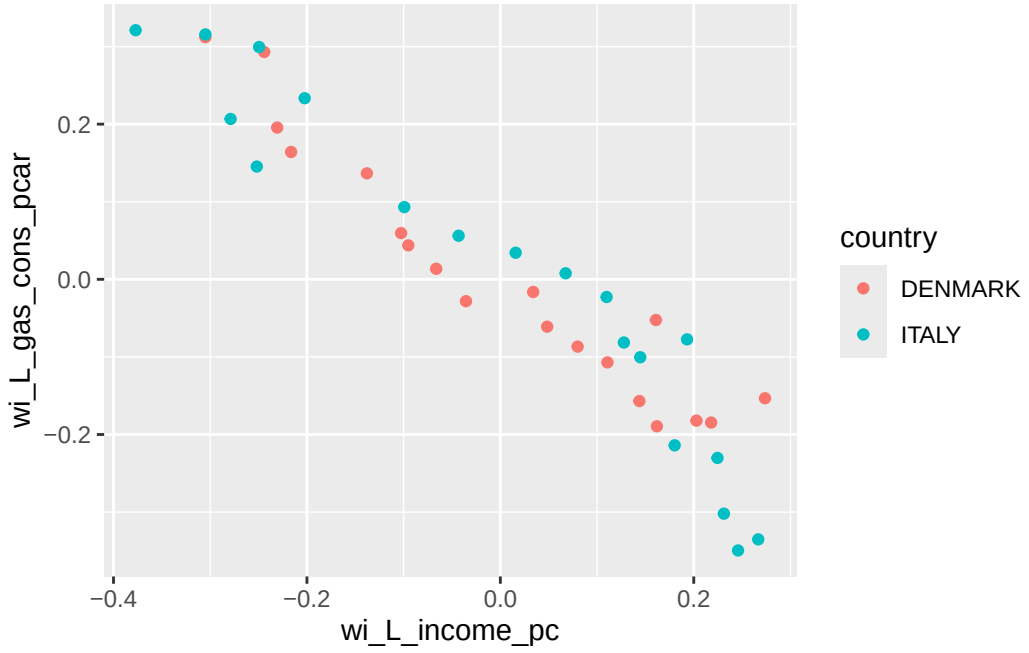
```
#View(df[, c("country", "year", "L_gas_cons_pcar", "m_L_gas_cons_pcar")])
```

Create deviations from the mean:

```
df <- df %>%
  mutate(wi_L_gas_cons_pcar = L_gas_cons_pcar - m_L_gas_cons_pcar,
         wi_L_income_pc = L_income_pc - m_L_income_pc)
```

Plot deviations from mean

```
ggplot(df, aes(x = wi_L_income_pc, y = wi_L_gas_cons_pcar, color = country)) +
       geom_point()
```




```
# ggsave("./plots/panel-data-analysis/deviations-from-mean.png")
```

11.1.9 Task 9

Run an OLS regression on the deviations from group averages without a constant.

11.1.9.1 Guidance

This is called the *within-groups estimator*. The slope coefficient that we obtain for our explanatory variable will be the same as the one obtained from LSDV approach, with a small difference in standard error estimates.

```
wg <- lm(wi_L_gas_cons_pcar ~ 0 + wi_L_income_pc, data = df)
summary(wg)
```

```
Call:
lm(formula = wi_L_gas_cons_pcar ~ 0 + wi_L_income_pc, data = df)

Residuals:
Logarithm of gasoline consumption per car
      Min        1Q    Median        3Q       Max
-0.121945 -0.038747 -0.001717  0.035970  0.101286

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
wi_L_income_pc -0.92547    0.04753  -19.47   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.05636 on 37 degrees of freedom
Multiple R-squared:  0.9111,    Adjusted R-squared:  0.9087
F-statistic: 379.1 on 1 and 37 DF,  p-value: < 2.2e-16
```

Part VII

Seminar 7 and 8 (4 and 11 March 2025)

12 Panel Data Models

European Countries Gasoline Consumption data is obtained from Abay Mulatu 316ECN Applied Econometrics lecture material, Coventry University.

We will be using the all country sample of the gasoline demand data.

We start by loading the required libraries.

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(Hmisc) # add labels to variables
```

```
Attaching package: 'Hmisc'

The following objects are masked from 'package:dplyr':

    src, summarize

The following objects are masked from 'package:base':

    format.pval, units
```

```
library(ggplot2)
library(dplyr) # for data manipulation
library(plm) # to estimate linear panel data models
```

```
Attaching package: 'plm'

The following objects are masked from 'package:dplyr':

    between, lag, lead
```

```
library(fastDummies) # create dummies based on categorical (factor) variable
```

12.1 Least Squares Dummy Variables (LSDV) Approach

We estimated this model in the previous section using two-country sample of gasoline consumption data. We will replicate it with full sample of countries.

12.1.1 Task 1

Import the `gasoline-demand-all-countries.csv` data, label variables and create country dummies.

12.1.1.1 Guidance

This is a replication of what we have done in the previous section.

```
df <- read.csv("~/Desktop/R-workshops/assets/data/gasoline-demand-all-countries.csv", strings
#View(df)
# label variables
label(df$L_gas_cons_pcar) <- "Logarithm of gasoline consumption per car"
label(df$L_income_pc) <- "Logarithm of real income per capita"
label(df$L_gas_price) <- "Logarithm of real gasoline price per gallon"
label(df$L_cars_pc) <- "Logarithm of number of cars per capita"
# create country dummies
# below is what we have done before
# df$italy <- ifelse(df$country == "ITALY", 1, 0)
# df$denmark <- ifelse(df$country == "DENMARK", 1, 0)
```

Because the data now has 18 countries, I will use another approach to create the country dummies:

```
df <- dummy_cols(df, select_columns = "country", remove_first_dummy = FALSE, remove_selected
```

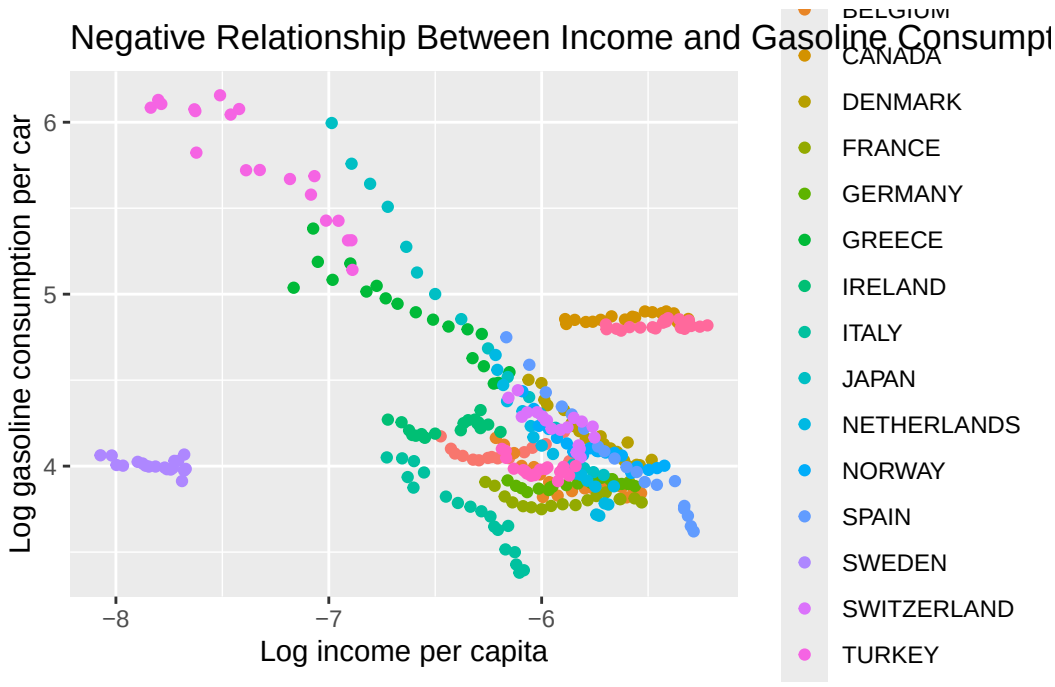
`View(df)` and you will see that country dummies are created with names `country_AUSTRIA`, `country_BELGIUM`, etc. You may remove the `country_` in front of all these dummy country names and convert letters to lower case by running the following line:

```
df <- df %>%
  rename_with(~ tolower(gsub("country_", "", .)), starts_with("country_"))
```

12.1.2 Task 2

Provide a scatter plot of `L_gas_cons_pcar` and `L_income_pc`. Differentiate each country data point by adding color.

```
ggplot(df, aes(x = L_income_pc, y = L_gas_cons_pcar, color = country)) +
  geom_point() +
  labs(x = "Log income per capita", y = "Log gasoline consumption per car",
       title = "Negative Relationship Between Income and Gasoline Consumption")
```



```
#ggsave("plots/panel-data-analysis/scatter-all-countries.png")
```

12.1.3 Task 3

Estimate a Least Squares Dummy Variable Model that regresses logarithm of gasoline consumption per car (`L_gas_cons_pcar`) on log of income per capita (`L_income_pc`), log of gasoline price per gallon (`L_gas_price`) and number of cars per capita (`L_cars_pc`). Comment on the results.

12.1.3.1 Guidance

There are multiple ways of approaching data analysis. We can use the country dummies that we created above or alternatively, we can ask R to add dummies based on our `country` variable. I will show the latter below. Try replicating this using dummies we created in the previous task.

Least Squares Dummy Variable Estimation:

```
# Least Squares Dummy Variable
lsdv <- lm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc + factor(country), data =
summary(lsdv)
```

```
Call:
lm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc +
    factor(country), data = df)

Residuals:
Logarithm of gasoline consumption per car
     Min       1Q   Median       3Q      Max
-0.37877 -0.03976  0.00465  0.04541  0.36286

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                2.28586    0.22832  10.011  < 2e-16 ***
L_income_pc                0.66225    0.07339   9.024  < 2e-16 ***
L_gas_price               -0.32170    0.04410  -7.295 2.35e-12 ***
L_cars_pc                 -0.64048    0.02968 -21.580  < 2e-16 ***
factor(country)BELGIUM    -0.12030    0.03415  -3.523 0.000489 ***
factor(country)CANADA      0.75598    0.04075  18.554  < 2e-16 ***
```

```
factor(country)DENMARK      0.10360    0.03660   2.830 0.004944 **
factor(country)FRANCE      -0.08108    0.03356  -2.416 0.016256 *
factor(country)GERMANY     -0.13599    0.03188  -4.266 2.63e-05 ***
factor(country)GREECE       0.05125    0.04153   1.234 0.218049
factor(country)IRELAND      0.30647    0.03529   8.683  < 2e-16 ***
factor(country)ITALY       -0.05331    0.03711  -1.436 0.151868
factor(country)JAPAN        0.09007    0.03861   2.333 0.020262 *
factor(country)NETHERLANDS -0.05106    0.03358  -1.521 0.129280
factor(country)NORWAY      -0.06916    0.04041  -1.711 0.087967 .
factor(country)SPAIN       -0.60408    0.09122  -6.622 1.49e-10 ***
factor(country)SWEDEN       0.74049    0.18008   4.112 4.99e-05 ***
factor(country)SWITZERLAND  0.11665    0.03471   3.360 0.000872 ***
factor(country)TURKEY       0.22413    0.04764   4.704 3.79e-06 ***
factor(country)UK           0.05959    0.03019   1.974 0.049237 *
factor(country)USA          0.76940    0.04458  17.260  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.09233 on 321 degrees of freedom
Multiple R-squared:  0.9734,    Adjusted R-squared:  0.9717
F-statistic: 586.6 on 20 and 321 DF,  p-value: < 2.2e-16
```

12.1.4 Task 4

Test if the the country dummies in the LSDV model above are jointly statistically significant.

12.1.4.1 Guidance

We need to perform an F test for restrictions. The LSDV model estimated above is the unrestricted model. We may re-estimate a restricted version of it by removing the country dummies. This model is our pooled OLS.

```r
pooled_ols <- lm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc, data = df)
summary(pooled_ols)
```

```
Call:
lm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df)

Residuals:
```

```
Logarithm of gasoline consumption per car
     Min       1Q   Median       3Q      Max
-0.38412 -0.15307 -0.04981  0.16529  0.59684

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.39133    0.11693   20.45   <2e-16 ***
L_income_pc  0.88996    0.03581   24.86   <2e-16 ***
L_gas_price -0.89180    0.03031  -29.42   <2e-16 ***
L_cars_pc   -0.76337    0.01861  -41.02   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.21 on 338 degrees of freedom
Multiple R-squared:  0.8549,    Adjusted R-squared:  0.8536
F-statistic:   664 on 3 and 338 DF,  p-value: < 2.2e-16
```

$H_0 : \beta_{belgium} = \beta_{canada} = \cdots = \beta_{usa} = 0$

H_1 : at least one is different than zero

Now, compare the restricted model with the unrestricted one:

```r
# anova_r <- anova(restricted_model, full_model)
anova_r <- anova(pooled_ols, lsdv)

print(anova_r)
```

```
Analysis of Variance Table

Model 1: L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
Model 2: L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc + factor(country)
  Res.Df     RSS Df Sum of Sq      F    Pr(>F)
1    338 14.9044
2    321  2.7365 17    12.168 83.961 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We reject H_0 . The country dummies are jointly statistically significant.

12.1.5 Task 5

Replicate the above test, this time using R's fixed effects estimation.

`plm` is the function we will use to estimate a panel linear model, with the `index` showing our panel data cross-section and time identifiers and the `model` telling R which panel linear model to estimate:

```
# Panel Data Fixed Effects Model
fixed_1 <- plm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
               data = df, index = c("country", "year"), model = "within")
summary(fixed_1)
```

```
Oneway (individual) effect Within Model

Call:
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df, model = "within", index = c("country", "year"))

Balanced Panel: n = 18, T = 19, N = 342

Residuals:
     Min.   1st Qu.    Median   3rd Qu.      Max.
-0.378774 -0.039758  0.004650  0.045412  0.362856

Coefficients:
             Estimate Std. Error  t-value  Pr(>|t|)
L_income_pc  0.662250   0.073386   9.0242 < 2.2e-16 ***
L_gas_price -0.321702   0.044099  -7.2950 2.355e-12 ***
L_cars_pc   -0.640483   0.029679 -21.5804 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    17.061
Residual Sum of Squares: 2.7365
R-Squared:      0.8396
Adj. R-Squared: 0.82961
F-statistic: 560.093 on 3 and 321 DF, p-value: < 2.22e-16
```

Note that the slope coefficients we obtain above are the same as those we obtained in our LSDV model.

The test between Fixed Effects and Pooled OLS models

```
pFtest(fixed_1, pooled_ols)
```

```
    F test for individual effects

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
F = 83.961, df1 = 17, df2 = 321, p-value < 2.2e-16
alternative hypothesis: significant effects
```

Again, note that the F-statistic reported above (83.961) is the same as what we obtained through `anova()` comparison.

12.1.6 Task 6

Estimate the Within-groups model manually.

12.1.6.1 Guidance

For this, we will need to regress the deviations of observations from group means using a linear model without a constant.

Let's first create the group averages using the `dplyr` package

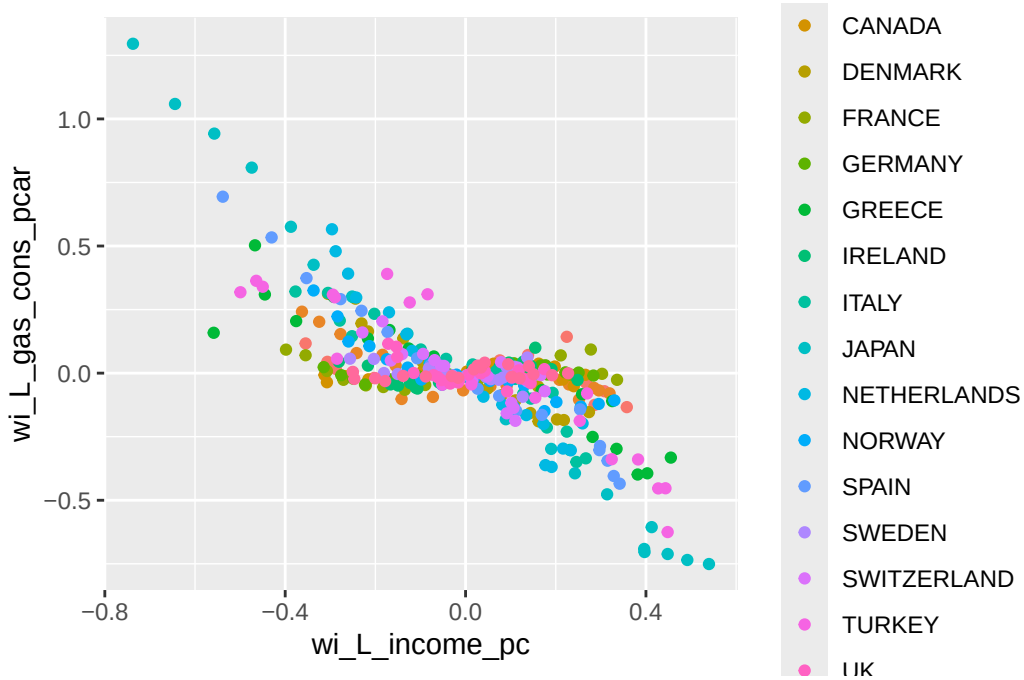
```
df <- df %>%
  group_by(country) %>%
  mutate(m_L_gas_cons_pcar = mean(L_gas_cons_pcar, na.rm = TRUE),
         m_L_income_pc = mean(L_income_pc, na.rm = TRUE),
         m_L_gas_price = mean(L_gas_price, na.rm = TRUE),
         m_L_cars_pc = mean(L_cars_pc, na.rm = TRUE)) %>%
  ungroup()
```

We then find the deviations of each observation from the group average.

```
df <- df %>%
  mutate(wi_L_gas_cons_pcar = L_gas_cons_pcar - m_L_gas_cons_pcar,
         wi_L_income_pc = L_income_pc - m_L_income_pc,
         wi_L_gas_price = L_gas_price - m_L_gas_price,
        wi_L_cars_pc = L_cars_pc - m_L_cars_pc)
```

In the final step, we use these deviations in a regression without a constant term. But let us see the scatter plot of gasoline consumption per car and income per capita after this data transformation.

```
ggplot(df, aes(x = wi_L_income_pc, y = wi_L_gas_cons_pcar, color = country)) +
  geom_point()
```



Regression (within-groups estimation):

```
wg <- lm(wi_L_gas_cons_pcar ~ 0 + wi_L_income_pc + wi_L_gas_price + wi_L_cars_pc,
         data = df)
summary(wg)
```

```
Call:
lm(formula = wi_L_gas_cons_pcar ~ 0 + wi_L_income_pc + wi_L_gas_price +
    wi_L_cars_pc, data = df)

Residuals:
Logarithm of gasoline consumption per car
     Min       1Q   Median       3Q      Max
-0.37877 -0.03976  0.00465  0.04541  0.36286
```

```
Coefficients:
               Estimate Std. Error t value Pr(>|t|)
wi_L_income_pc  0.66225    0.07141   9.274  < 2e-16 ***
wi_L_gas_price -0.32170    0.04291  -7.497 5.77e-13 ***
wi_L_cars_pc   -0.64048    0.02888 -22.177  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.08985 on 339 degrees of freedom
Multiple R-squared:  0.8396,    Adjusted R-squared:  0.8382
F-statistic: 591.5 on 3 and 339 DF,  p-value: < 2.2e-16
```

12.1.7 Task 7

Estimate a random effects model and compare this with Pooled OLS.

To estimate a panel random effect model, we use the same panel data linear model `plm()` function as above, but will change the model to `random`.

```
# Panel Data Random Effects Model
random_1 <- plm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
                data = df, index = c("country", "year"), model = "random")
summary(random_1)
```

```
Oneway (individual) effect Random Effect Model
   (Swamy-Arora's transformation)

Call:
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df, model = "random", index = c("country", "year"))

Balanced Panel: n = 18, T = 19, N = 342

Effects:
                   var  std.dev share
idiosyncratic 0.008525 0.092330 0.182
individual    0.038238 0.195545 0.818
theta: 0.8923

Residuals:
      Min.    1st Qu.     Median    3rd Qu.       Max.
```

```
-0.3977058 -0.0520350  0.0050877  0.0582288  0.3763726

Coefficients:
             Estimate Std. Error  z-value  Pr(>|z|)
(Intercept)  1.996698   0.184326  10.8324 < 2.2e-16 ***
L_income_pc  0.554986   0.059128   9.3861 < 2.2e-16 ***
L_gas_price -0.420389   0.039978 -10.5155 < 2.2e-16 ***
L_cars_pc   -0.606840   0.025515 -23.7836 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    18.054
Residual Sum of Squares: 3.0817
R-Squared:      0.82931
Adj. R-Squared: 0.8278
Chisq: 1642.2 on 3 DF, p-value: < 2.22e-16
```

The comparison between the above the Pooled OLS is done by Breusch-Pagan LM Test.

```
plmtest(random_1, type = "bp")
```

```
    Lagrange Multiplier Test - (Breusch-Pagan)

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
chisq = 1465.6, df = 1, p-value < 2.2e-16
alternative hypothesis: significant effects
```

12.1.8 Task 8

Apply the Hausman Test to compare Random Effects and Fixed Effects Models.

```
phtest(fixed_1, random_1)
```

```
    Hausman Test

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
chisq = 302.8, df = 3, p-value < 2.2e-16
alternative hypothesis: one model is inconsistent
```

The null of Random Effects is rejected. We choose the Fixed Effects.

12.1.9 Task 9

Estimate a two-way fixed effects model and test for the joint significance of the time effects.

```
# Panel Data Fixed Effects Model
fixed_2 <- plm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc + factor(year),
               data = df, index = c("country", "year"), model = "within")
summary(fixed_2)
```

```
Oneway (individual) effect Within Model

Call:
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc +
    factor(year), data = df, model = "within", index = c("country",
    "year"))

Balanced Panel: n = 18, T = 19, N = 342

Residuals:
       Min.     1st Qu.      Median     3rd Qu.        Max.
-0.41920085 -0.03886111  0.00018502  0.04199566  0.23067839

Coefficients:
                  Estimate Std. Error  t-value  Pr(>|t|)
L_income_pc       0.051369   0.091386   0.5621 0.5744611
L_gas_price      -0.192850   0.042860  -4.4995 9.718e-06 ***
L_cars_pc        -0.593448   0.027669 -21.4479 < 2.2e-16 ***
factor(year)1961  0.040970   0.027248   1.5036 0.1337236
factor(year)1962  0.044249   0.027595   1.6035 0.1098635
factor(year)1963  0.064744   0.028277   2.2897 0.0227268 *
factor(year)1964  0.105995   0.029297   3.6179 0.0003479 ***
factor(year)1965  0.124134   0.030049   4.1310 4.677e-05 ***
factor(year)1966  0.167830   0.031046   5.4058 1.310e-07 ***
factor(year)1967  0.198832   0.032048   6.2042 1.801e-09 ***
factor(year)1968  0.230077   0.033201   6.9299 2.537e-11 ***
factor(year)1969  0.242999   0.035304   6.8831 3.374e-11 ***
factor(year)1970  0.275080   0.037182   7.3982 1.365e-12 ***
factor(year)1971  0.304198   0.038516   7.8980 5.262e-14 ***
factor(year)1972  0.332136   0.040532   8.1944 7.160e-15 ***
factor(year)1973  0.369707   0.043209   8.5562 5.913e-16 ***
factor(year)1974  0.327938   0.042232   7.7652 1.266e-13 ***
factor(year)1975  0.362392   0.041877   8.6538 2.984e-16 ***
```

```
factor(year)1976  0.370891   0.043626   8.5016 8.651e-16 ***
factor(year)1977  0.385702   0.044559   8.6559 2.939e-16 ***
factor(year)1978  0.400956   0.046409   8.6397 3.295e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    17.061
Residual Sum of Squares: 1.997
R-Squared:      0.88295
Adj. R-Squared: 0.86827
F-statistic: 108.839 on 21 and 303 DF, p-value: < 2.22e-16
```

```r
# fixed_1 : One-way fixed effects model (cross-section effects included)
# fixed_2 : Two-way fixed effects model (cross-section and time-effects included)
# Joint statistical significance of the time effects
pFtest(fixed_2, fixed_1)
```

```
    F test for individual effects

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc + factor(year)
F = 6.2338, df1 = 18, df2 = 303, p-value = 5.36e-13
alternative hypothesis: significant effects
```

Time effects are jointly statistically significant. We choose the two-way FE model over one-way FE model.

Note that we can estimate the above two-way model by adding an `effect = "twoways"` option in R, without the need for an explicit inclusion of time dummies.

```r
# Panel Data Fixed Effects Model
fixed_3 <- plm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
               data = df, index = c("country", "year"), model = "within", effect = "twoways")
summary(fixed_3)
```

```
Twoways effects Within Model

Call:
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df, effect = "twoways", model = "within", index = c("country",
        "year"))
```

```
Balanced Panel: n = 18, T = 19, N = 342

Residuals:
       Min.     1st Qu.      Median     3rd Qu.        Max.
-0.41920085 -0.03886111  0.00018502  0.04199566  0.23067839

Coefficients:
             Estimate Std. Error  t-value  Pr(>|t|)
L_income_pc  0.051369   0.091386   0.5621    0.5745
L_gas_price -0.192850   0.042860  -4.4995 9.718e-06 ***
L_cars_pc   -0.593448   0.027669 -21.4479 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    10.644
Residual Sum of Squares: 1.997
R-Squared:      0.81239
Adj. R-Squared: 0.78886
F-statistic: 437.338 on 3 and 303 DF, p-value: < 2.22e-16
```

12.1.10 Task 10

Estimate a two-way random effects model.

```r
# Panel Data Fixed Effects Model
random_2 <- plm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
                data = df, index = c("country", "year"),
                model = "random", effect = "twoways")
summary(random_2)
```

```
Twoways effects Random Effect Model
   (Swamy-Arora's transformation)

Call:
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df, effect = "twoways", model = "random", index = c("country",
        "year"))

Balanced Panel: n = 18, T = 19, N = 342
```

```
Effects:
                   var  std.dev share
idiosyncratic 0.006591 0.081183 0.147
individual    0.038340 0.195805 0.853
time          0.000000 0.000000 0.000
theta: 0.9053 (id) 0 (time) 0 (total)

Residuals:
      Min.    1st Qu.     Median    3rd Qu.       Max.
-0.3956900 -0.0499700  0.0075613  0.0543043  0.3748214

Coefficients:
             Estimate Std. Error  z-value  Pr(>|z|)
(Intercept)  2.040793   0.191508  10.6564 < 2.2e-16 ***
L_income_pc  0.564562   0.060854   9.2773 < 2.2e-16 ***
L_gas_price -0.404936   0.040369 -10.0309 < 2.2e-16 ***
L_cars_pc   -0.609360   0.025970 -23.4641 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    17.829
Residual Sum of Squares: 3.0145
R-Squared:      0.83092
Adj. R-Squared: 0.82942
Chisq: 1661.05 on 3 DF, p-value: < 2.22e-16
```

12.1.11 Task 11

Perform a Hausman Test comparing two-way fixed and random effect models.

```
phtest(fixed_2, random_2)
```

```
    Hausman Test

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc + factor(year)
chisq = 134.07, df = 3, p-value < 2.2e-16
alternative hypothesis: one model is inconsistent
```

We reject the null hypothesis of random effects and choose the fixed effects model.

To sum up, we started by estimating a Pooled OLS model, followed by a fixed effects specification. We have seen alternative ways of estimating the Fixed Effects Model, i.e. the Least Squares Dummy Variable Apporoach, the Within Groups Estimator and R's built in function `plm` with the argument `model = "within"`.

We then moved on to Random Effects Model using the `plm` function with the argument `model = "random"`.

The F-test comparison between the Pooled OLS and Fixed Effects Model indicated the existence of unobserved heterogeneity (time invariant country-specific effects).

The Breusch-Pagan Test was applied to compare the Pooled and Random Effects models. This test also rejected the Pooled OLS and indicated that the variation of the country-specific effects is different than zero.

A comparison between the Fixed Effects and Random Effects model through the Hausman Test rejected the null of Random Effects specification in favour of a Fixed Effects Model.

We then moved on to check existence of time-specific effects that are common to all cross-sectional units. This was done by an F-test, which this time, compared one-way and two way fixed effects specification and tested for the joint significance of the time effects.

The null of time effects being jointly equal to zero is rejected, suggesting estimation of two-way models.

A final comparison was between the Two-way Fixed Effects and Two-way Random effects models. The Hausman Test again rejected the Random Effects specification, which led us to decide on the Two-way Fixed Effects model as our final specification.

13 Panel Data Misspecification Tests

Please note that the information provided in this section (panel data misspecification tests) are only for those of you who are interested to explore further. You are not responsible from these tests in your 6036ECN exam.

In this section, we will continue with the examples from the previous section and test for Autocorrelation and Heteroscedasticity.

European Countries Gasoline Consumption data is obtained from Abay Mulatu 316ECN Applied Econometrics lecture material, Coventry University.

We will be using the all country sample of the gasoline demand data.

We start by loading the required libraries.

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
library(Hmisc) # add labels to variables
```

```
Attaching package: 'Hmisc'

The following objects are masked from 'package:dplyr':

    src, summarize
```

```
The following objects are masked from 'package:base':

    format.pval, units
```

```
#library(ggplot2)
#library(dplyr) # for data manipulation
library(plm) # to estimate linear panel data models
```

```
Attaching package: 'plm'

The following objects are masked from 'package:dplyr':

    between, lag, lead
```

```
#library(fastDummies) # create dummies based on categorical (factor) variable
```

13.1 Testing for Autocorrelation

13.1.1 Task 1

Using the `gasoline-demand-all-countries.csv` data, estimate a one-way fixed effects gasoline demand model.

13.1.1.1 Guidance

This is a replication of what we have done in the previous section.

```
df <- read.csv("~/Desktop/R-workshops/assets/data/gasoline-demand-all-countries.csv", strings
#View(df)
# label variables
label(df$L_gas_cons_pcar) <- "Logarithm of gasoline consumption per car"
label(df$L_income_pc) <- "Logarithm of real income per capita"
label(df$L_gas_price) <- "Logarithm of real gasoline price per gallon"
label(df$L_cars_pc) <- "Logarithm of number of cars per capita"
```

We estimate a fixed effects model below:

```r
# Panel Data Fixed Effects Model
fixed_1 <- plm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
               data = df, index = c("country", "year"),
               model = "within")
summary(fixed_1)
```

```
Oneway (individual) effect Within Model

Call:
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df, model = "within", index = c("country", "year"))

Balanced Panel: n = 18, T = 19, N = 342

Residuals:
     Min.   1st Qu.    Median   3rd Qu.      Max.
-0.378774 -0.039758  0.004650  0.045412  0.362856

Coefficients:
             Estimate Std. Error  t-value  Pr(>|t|)
L_income_pc  0.662250   0.073386   9.0242 < 2.2e-16 ***
L_gas_price -0.321702   0.044099  -7.2950 2.355e-12 ***
L_cars_pc   -0.640483   0.029679 -21.5804 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    17.061
Residual Sum of Squares: 2.7365
R-Squared:      0.8396
Adj. R-Squared: 0.82961
F-statistic: 560.093 on 3 and 321 DF, p-value: < 2.22e-16
```

13.1.2 Task 2

Test for the existence of Autocorrelation.

13.1.2.1 Guidance

Wooldridge Test for AR(1) Errors in FE Panel Models.

- Could be used for short and long panels.

- Could be used for fixed effects models only.
- Tests for Autocorrelation of order 1.

```
pwartest(fixed_1)
```

```
    Wooldridge's test for serial correlation in FE panels

data:  fixed_1
F = 212, df1 = 1, df2 = 322, p-value < 2.2e-16
alternative hypothesis: serial correlation
```

According to the test results, there is autocorrelation of order 1.

Breusch-Godfrey Test for Panel Models

- Suitable for long panels.
- Allows to choose the order to test for.
- It could be used for both the fixed effects and random effects models.

```
pbgtest(fixed_1)
```

```
    Breusch-Godfrey/Wooldridge test for serial correlation in panel models

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
chisq = 185.08, df = 19, p-value < 2.2e-16
alternative hypothesis: serial correlation in idiosyncratic errors
```

There is autocorrelation of order 1.

Let's say we want to test for Autocorrelation of order 3:

```
pbgtest(fixed_1, order = 3)
```

```
    Breusch-Godfrey/Wooldridge test for serial correlation in panel models

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
chisq = 172.76, df = 3, p-value < 2.2e-16
alternative hypothesis: serial correlation in idiosyncratic errors
```

There is autocorrelation up to order 3.

13.1.3 Task 3

Estimate a random effects model

13.1.3.1 Guidance

We have this in the previous section. Here is a random effects version of the above fixed effects model.

```
# Panel Data Fixed Effects Model
random_1 <- plm(L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
                data = df, index = c("country", "year"),
                model = "random")
summary(random_1)
```

```
Oneway (individual) effect Random Effect Model
   (Swamy-Arora's transformation)

Call:
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df, model = "random", index = c("country", "year"))

Balanced Panel: n = 18, T = 19, N = 342

Effects:
                   var  std.dev share
idiosyncratic 0.008525 0.092330 0.182
individual    0.038238 0.195545 0.818
theta: 0.8923

Residuals:
      Min.    1st Qu.     Median    3rd Qu.       Max.
-0.3977058 -0.0520350  0.0050877  0.0582288  0.3763726

Coefficients:
             Estimate Std. Error  z-value  Pr(>|z|)
(Intercept)  1.996698   0.184326  10.8324 < 2.2e-16 ***
L_income_pc  0.554986   0.059128   9.3861 < 2.2e-16 ***
L_gas_price -0.420389   0.039978 -10.5155 < 2.2e-16 ***
L_cars_pc   -0.606840   0.025515 -23.7836 < 2.2e-16 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    18.054
Residual Sum of Squares: 3.0817
R-Squared:      0.82931
Adj. R-Squared: 0.8278
Chisq: 1642.2 on 3 DF, p-value: < 2.22e-16
```

13.1.4 Task 4

Test for autocorrelation in the above estimated random effects model.

13.1.4.1 Guidance

We can start by applying the Breusch-Godfrey test for panel models.

```
pbgtest(random_1)
```

```
    Breusch-Godfrey/Wooldridge test for serial correlation in panel models

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
chisq = 193.42, df = 19, p-value < 2.2e-16
alternative hypothesis: serial correlation in idiosyncratic errors
```

There is autocorrelation of order 1.

Baltagi and Li Serial Dependence Test for Random Effect Models

- The test can be used with random effects models only.

```
pbltest(random_1)
```

```
    Baltagi and Li two-sided LM test

data:  formula(x$formula)
chisq = 225.16, df = 1, p-value < 2.2e-16
alternative hypothesis: AR(1)/MA(1) errors in RE panel model
```

This test confirms the results of the previous one, in that, there is autocorrelation of order 1.

13.2 Heteroscedasticity

13.2.1 Task 5

Test for the existence of heteroscedasticity in each of the models estimated above.

13.2.1.1 Guidance

Lagrange FF Multiplier Tests for Panel Data

- We will apply `plmtest()` with `type = "bp"`.

```
plmtest(fixed_1, type = "bp")
```

```
    Lagrange Multiplier Test - (Breusch-Pagan)

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
chisq = 1465.6, df = 1, p-value < 2.2e-16
alternative hypothesis: significant effects
```

```
plmtest(random_1, type = "bp")
```

```
    Lagrange Multiplier Test - (Breusch-Pagan)

data:  L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc
chisq = 1465.6, df = 1, p-value < 2.2e-16
alternative hypothesis: significant effects
```

There is heteroscedasticity.

13.2.2 Task 6

Report the fixed and random effects estimation results with autocorrelation and heteroscedasticity robust standard errors.

13.2.2.1 Guidance

Existence of autocorrelation and heteroscedasticity affects the standard error estimates. Although the coefficients will remain unbiased, the standard errors will be biased and hence all tests we perform on the model will be misleading. We may either try to resolve these issues though changing our model or estimation strategy or computer heteroskedasticity or autocorrelation adjusted standard errors.

Calculation of robust standard errors is common in Applied Econometrics. Since the models presented here are panel data estimation, we will be using `plm` package's `vcovHC` function. Below is the information from R help about the arguments that this function can take:

```
vcovHC(
  x,
  method = c("arellano", "white1", "white2"),
  type = c("HC0", "sss", "HC1", "HC2", "HC3", "HC4"),
  cluster = c("group", "time"),
  ...
)
```

- All `methods` assume no intragroup (serial) correlation between errors and allow for heteroskedasticity across groups (time periods).
- `white1` allows for general heteroskedasticity but no serial (cross–sectional) correlation;
- `white2` is `white1` restricted to a common variance inside every group (time period)
- `arellano` allows a fully general structure w.r.t. heteroskedasticity and serial (cross–sectional) correlation.

Hence, we will be using **`arellano`** here.

The `type` argument in `vcovHC()` allows you to specify different weighting schemes for the robust standard errors. The most common options include:

- **`"HC0"`**: The classic White (1980) estimator. It does not account for small sample bias.
- **`"HC1"`**: A finite-sample correction used in Eicker-Huber-White heteroskedasticity-consistent covariance matrix estimators.
- **`"HC2"`**: Corrects for leverage by dividing residuals by $(1 - h_{ii})$, where h_{ii} is the leverage of observation i.
- **`"HC3"`**: A further adjustment for leverage, dividing residuals by $(1 - h_{ii})^2$, making it more robust in small samples.
- **`"HC4"`**: A more extreme correction for leverage, giving higher influence to high-leverage points.

R's default:

```
summary(fixed_1, vcov = vcovHC)
```

```
Oneway (individual) effect Within Model

Note: Coefficient variance-covariance matrix supplied: vcovHC

Call:
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df, model = "within", index = c("country", "year"))

Balanced Panel: n = 18, T = 19, N = 342

Residuals:
     Min.   1st Qu.    Median   3rd Qu.      Max.
-0.378774 -0.039758  0.004650  0.045412  0.362856

Coefficients:
             Estimate Std. Error t-value  Pr(>|t|)
L_income_pc  0.662250   0.153279  4.3205 2.078e-05 ***
L_gas_price -0.321702   0.122275 -2.6310  0.008925 **
L_cars_pc   -0.640483   0.096654 -6.6266 1.451e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    17.061
Residual Sum of Squares: 2.7365
R-Squared:      0.8396
Adj. R-Squared: 0.82961
F-statistic: 17.005 on 3 and 17 DF, p-value: 2.3028e-05
```

Use of `method = "arellano"` with `type = "HC1"`:

```
summary(fixed_1, function(x) vcovHC(x, method = "arellano", type = "HC1"))
```

```
Oneway (individual) effect Within Model

Note: Coefficient variance-covariance matrix supplied: function(x) vcovHC(x, method = "arella

Call:
```

```
plm(formula = L_gas_cons_pcar ~ L_income_pc + L_gas_price + L_cars_pc,
    data = df, model = "within", index = c("country", "year"))

Balanced Panel: n = 18, T = 19, N = 342

Residuals:
     Min.   1st Qu.    Median   3rd Qu.      Max.
-0.378774 -0.039758  0.004650  0.045412  0.362856

Coefficients:
            Estimate Std. Error t-value  Pr(>|t|)
L_income_pc  0.66225    0.15396  4.3016 2.254e-05 ***
L_gas_price -0.32170    0.12281 -2.6194  0.009227 **
L_cars_pc   -0.64048    0.09708 -6.5975 1.726e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    17.061
Residual Sum of Squares: 2.7365
R-Squared:      0.8396
Adj. R-Squared: 0.82961
F-statistic: 16.8559 on 3 and 17 DF, p-value: 2.4334e-05
```

Part VIII

Seminar 9 (18 March 2025)

14 Endogeneity and Instrumental Variable Estimation

In this section we will be using `mroz` data, obtained from Wooldridge's Econometric Analysis of Cross Section and Panel Data book's official site for downloadable materials. This is a "PSID data on the wages of 428 working, married women".

We start by installing the required packages and loading the required libraries. Note that you do not need to install an already installed package if you are using your private computer. But you will have to do this every time you require the package if you are on a university gadget.

```
# install required packages
#install.packages("tidyverse")
#install.packages("haven")
#install.packages(ivreg)
#install.packages("sandwich")
#install.packages("lmtest")

library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
library(haven) # to import data, which is in Stata format
library(ivreg) # IV estimation
library(sandwich) # for robust se calculations
library(lmtest) # for coeftest
```

```
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
library(stargazer) # create formatted tables
```

```
Please cite as:

 Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.
 R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

```
#library(Hmisc) # add labels to variables
#library(ggplot2)
#library(dplyr) # for data manipulation
#library(plm) # to estimate linear panel data models
#library(fastDummies) # create dummies based on categorical (factor) variable
```

Import the `mroz_v2.dta` data. `mroz_v2` data is provided to you in Stata format. Stata is an other statistical package, widely used for data analysis and econometric modelling. You will see that even when Stata is not installed in your system, you will be able to import it into R using the `haven` package.

```
mroz <- read_dta("assets/data/mroz_v2.dta")
#View(mroz)
```

See the the fist few rows:

```
head(mroz)
```

```
# A tibble: 6 x 19
  hours kidslt6 kidsge6   age  educ  wage repwage hushrs husage huseduc huswage
  <dbl>   <dbl>   <dbl> <dbl> <dbl> <dbl>   <dbl>  <dbl>  <dbl>   <dbl>   <dbl>
1  1610       1       0    32    12  3.35    2.65   2708     34      12    4.03
2  1656       0       2    30    12  1.39    2.65   2310     30       9    8.44
3  1980       1       3    35    12  4.55    4.04   3072     40      12    3.58
```

```
4   456       0       3    34    12  1.10    3.25   1920     53      10    3.54
5  1568       1       2    31    14  4.59    3.60   2000     32      12   10
6  2032       0       0    54    12  4.74    4.70   1040     57      11    6.71
# i 8 more variables: faminc <dbl>, motheduc <dbl>, fatheduc <dbl>, unem <dbl>,
#   city <dbl>, exper <dbl>, lwage <dbl>, expersq <dbl>
```

Instrumental variable approach is widely referred to as Two-stage Least Squares. These two will be used interchangeably. Please also note the abbreviations IV and 2SLS, respectively, for the former and the latter.

14.1 Case I: 2SLS with one endogenous, one exogenous variable

14.1.1 Task 1

Estimate a regression of logarithmic wage `lwage` using education (`educ`) and experience (`exper`) as independent variables. Include experience in quadratic form.

```
ols <- lm(lwage ~ educ + exper + expersq, data = mroz)
summary(ols)
```

```
Call:
lm(formula = lwage ~ educ + exper + expersq, data = mroz)

Residuals:
     Min       1Q   Median       3Q      Max
-3.08404 -0.30627  0.04952  0.37498  2.37115

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.5220407  0.1986321  -2.628  0.00890 **
educ         0.1074896  0.0141465   7.598 1.94e-13 ***
exper        0.0415665  0.0131752   3.155  0.00172 **
expersq     -0.0008112  0.0003932  -2.063  0.03974 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6664 on 424 degrees of freedom
Multiple R-squared:  0.1568,    Adjusted R-squared:  0.1509
F-statistic: 26.29 on 3 and 424 DF,  p-value: 1.302e-15
```

Because it is highly likely that we will observe heteroscedasticity, let us summarise the results with heteroscedasticity-robust standard errors. For this, we will use `coeftest`, which comes with the `lmtest` package.

```
coeftest(ols, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

               Estimate  Std. Error t value Pr(>|t|)
(Intercept) -0.52204068  0.20165046 -2.5888 0.009961 **
educ         0.10748965  0.01321897  8.1315 4.72e-15 ***
exper        0.04156651  0.01527304  2.7216 0.006765 **
expersq     -0.00081119  0.00042007 -1.9311 0.054139 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Have a look at the coefficient estimates, standard errors, t-statistics and the p-values reported with and without the heteroscedasticity-robust standard errors. How do they compare?

Calculation of robust standard errors corrects for the bias in standard error estimates due to heteroscedasticity or autocorrelation (heteroscedasticity in the case of this example). Therefore, the coefficient estimates remain the same while standard errors are adjusted for the bias. Because of this change in the standard errors, the t-statistics and the p-values (which use standard error in calculations) also change.

14.1.2 Task 2

Explain why there may be an endogeneity issue in the model we estimated above.

Ability is an important determinant of wages, which is not included in the given regression. It is also expected to be highly correlated with education. If that's the case, omission of ability from the model will lead to a correlation between education and the error term. This causes an issue of endogeneity.

14.1.3 Task 3

Identify the potential instruments that you may use in the data you are given and explain your choice.

The three potential instruments provided in data are: mother's education, father's education and husband's education. Each of these three variables are likely to be highly correlated with individual's education but not with their wage.

14.1.4 Task 4

Estimate the above equation by two-stage least squares (i.e. instrumental variable estimation) manually.

14.1.4.1 Guidance

Step 1

Let's say we want to use husband's education `huseduc` as an instrument for the woman's education.

Regress the endogenous variable on the instrument and all other exogenous variables of the model.

```
stage1_1 <- lm(educ ~ huseduc + exper + expersq, data = mroz)
summary(stage1_1)
```

```
Call:
lm(formula = educ ~ huseduc + exper + expersq, data = mroz)

Residuals:
    Min      1Q  Median      3Q     Max
-6.5093 -0.9785 -0.2310  1.2857  6.3994

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.694524   0.454849  14.718   <2e-16 ***
huseduc      0.448194   0.029495  15.195   <2e-16 ***
exper        0.042220   0.036338   1.162    0.246
expersq     -0.001017   0.001085  -0.937    0.349
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.841 on 424 degrees of freedom
Multiple R-squared:  0.3558,    Adjusted R-squared:  0.3512
F-statistic: 78.05 on 3 and 424 DF,  p-value: < 2.2e-16
```

Obtain predictions of education using the above model.

```
mroz$educ_hat <- predict(stage1_1)
```

Step 2

Estimate the wage regression, replacing the endogenous `educ` variable, with its exogenous version `educ_hat`

```
stage2_1 <- lm(lwage ~ educ_hat + exper + expersq, data = mroz)
summary(stage2_1)
```

```
Call:
lm(formula = lwage ~ educ_hat + exper + expersq, data = mroz)

Residuals:
     Min       1Q   Median       3Q      Max
-3.12116 -0.34011  0.05149  0.38784  2.36570

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.2980989  0.3248571  -0.918 0.359334
educ_hat     0.0893851  0.0250210   3.572 0.000394 ***
exper        0.0425893  0.0138836   3.068 0.002296 **
expersq     -0.0008457  0.0004148  -2.039 0.042080 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6999 on 424 degrees of freedom
Multiple R-squared:   0.07, Adjusted R-squared:  0.06342
F-statistic: 10.64 on 3 and 424 DF,  p-value: 9.339e-07
```

Let us report these results using the heteroscedasticity-robust standard errors:

```
coeftest(stage2_1, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

               Estimate  Std. Error t value  Pr(>|t|)
(Intercept) -0.29809893  0.33474934 -0.8905 0.3736950
educ_hat     0.08938509  0.02448649  3.6504 0.0002945 ***
```

```
exper        0.04258927  0.01591354  2.6763 0.0077327 **
expersq     -0.00084567  0.00044034 -1.9205 0.0554634 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

14.1.5 Task 5

Estimate the above equation by 2SLS (IV estimation) using R's built-in command `ivreg`. Compare the coefficients and standard errors with what you obtained manually.

```
iv_1 <- ivreg(lwage ~ educ + exper + expersq | huseduc + exper + expersq, data = mroz)
summary(iv_1)
```

```
Call:
ivreg(formula = lwage ~ educ + exper + expersq | huseduc + exper +
    expersq, data = mroz)

Residuals:
     Min       1Q   Median       3Q      Max
-3.07677 -0.32148  0.03525  0.37605  2.36256

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.2980989  0.3099189  -0.962 0.336668
educ         0.0893851  0.0238705   3.745 0.000206 ***
exper        0.0425893  0.0132451   3.215 0.001402 **
expersq     -0.0008457  0.0003957  -2.137 0.033155 *

Diagnostic tests:
                 df1 df2 statistic p-value
Weak instruments   1 424   230.900  <2e-16 ***
Wu-Hausman         1 423     0.892   0.346
Sargan             0  NA        NA      NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6677 on 424 degrees of freedom
Multiple R-Squared: 0.1536, Adjusted R-squared: 0.1476
Wald test: 11.69 on 3 and 424 DF,  p-value: 2.26e-07
```

The coefficients reported by the `ivreg` are the same as our manual two-stage estimation. The standard errors are slightly different. This is because in the manual calculations, during the estimation of the second stage, we use predictions from the first stage (`educ_hat`). This introduces additional uncertainty into the model. Although the statistical packages (such as R) follow a similar approach, they correct for the bias in standard error calculations before reporting these numbers.

Note that this is a different adjustment than calculation of heteroscedasticity-robust standard errors. So let's integrate that too

```
coeftest(iv_1, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

               Estimate  Std. Error t value  Pr(>|t|)
(Intercept) -0.29809893  0.31885872 -0.9349 0.3503753
educ         0.08938509  0.02306960  3.8746 0.0001237 ***
exper        0.04258927  0.01525285  2.7922 0.0054716 **
expersq     -0.00084567  0.00041999 -2.0135 0.0446901 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

14.1.6 Task 6

Compare the coefficient on education in OLS and 2SLS approaches.

14.1.6.1 Guidance

We can compare the three models summarising their results in one table using `stargazer` package. Please note that the table below does not provide heteroscedasticity corrected standard errors. We can replace these conventional standard errors reported using a matrix of robust ones but this requires a few more steps that is beyond this module. You may change these manually for the moment.

```
stargazer(ols, stage2_1, iv_1, type = "text")
```

```
===============================================================
                                        Dependent variable:
```

	lwage		
	OLS		instrumental variable
	(1)	(2)	(3)
educ	0.107***		0.089***
	(0.014)		(0.024)
educ_hat		0.089***	
		(0.025)	
exper	0.042***	0.043***	0.043***
	(0.013)	(0.014)	(0.013)
expersq	-0.001**	-0.001**	-0.001**
	(0.0004)	(0.0004)	(0.0004)
Constant	-0.522***	-0.298	-0.298
	(0.199)	(0.325)	(0.310)
Observations	428	428	428
R2	0.157	0.070	0.154
Adjusted R2	0.151	0.063	0.148
Residual Std. Error (df = 424)	0.666	0.700	0.668
F Statistic (df = 3; 424)	26.286***	10.638***	
Note:		*p<0.1; **p<0.05; ***p<0.01	

As expected, the coefficient on education is higher in OLS estimation than the coefficient obtained through 2SLS. This is because education captures not only the genuine impact of years of schooling but also ability. Each of these indicators are expected to have a positive impact on wages, and they are positively correlated with each other. Hence, omission of ability from the wage regression creates a positive bias on the education's coefficient.

14.2 Case II: 2SLS with one endogenous and multiple exogenous variables

14.2.1 Task 7

Replicate the 2SLS estimation manually (using OLS), this time with 3 instruments for education: husband's education (`huseduc`), mother's education (`motheduc`) and father's education (`fatheduc`).

14.2.1.1 Guidance

Step1

First, estimate the first stage regression: regress the endogenous variable on the instrument and all other exogenous variables of the model.

```
stage1_2 <- lm(educ ~ huseduc + motheduc + fatheduc + exper + expersq,
               data = mroz)
summary(stage1_2)
```

```
Call:
lm(formula = educ ~ huseduc + motheduc + fatheduc + exper + expersq,
    data = mroz)

Residuals:
    Min      1Q  Median      3Q     Max
-6.6882 -1.1519  0.0097  1.0640  5.7302

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.5383110  0.4597824  12.046  < 2e-16 ***
huseduc      0.3752548  0.0296347  12.663  < 2e-16 ***
motheduc     0.1141532  0.0307835   3.708 0.000237 ***
fatheduc     0.1060801  0.0295153   3.594 0.000364 ***
exper        0.0374977  0.0343102   1.093 0.275059
expersq     -0.0006002  0.0010261  -0.585 0.558899
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.738 on 422 degrees of freedom
```

```
Multiple R-squared:  0.4286,    Adjusted R-squared:  0.4218
F-statistic:  63.3 on 5 and 422 DF,  p-value: < 2.2e-16
```

Obtain predictions of education using the above model.

```
mroz$educ_hat_2 <- predict(stage1_2)
```

Step 2

Estimate the wage regression, replacing the endogenous `educ` variable, with its exogenous version `educ_hat_2`

```
stage2_2 <- lm(lwage ~ educ_hat_2 + exper + expersq, data = mroz)
summary(stage2_2)
```

```
Call:
lm(formula = lwage ~ educ_hat_2 + exper + expersq, data = mroz)

Residuals:
    Min      1Q  Median      3Q     Max
-3.1407 -0.3382  0.0594  0.3798  2.3860

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.1868574  0.2985449  -0.626 0.531722
educ_hat_2   0.0803918  0.0227772   3.529 0.000462 ***
exper        0.0430973  0.0138760   3.106 0.002024 **
expersq     -0.0008628  0.0004144  -2.082 0.037957 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7001 on 424 degrees of freedom
Multiple R-squared:  0.06935,   Adjusted R-squared:  0.06277
F-statistic: 10.53 on 3 and 424 DF,  p-value: 1.078e-06
```

Let us report these results using the heteroscedasticity-robust standard errors:

```
coeftest(stage2_2, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

               Estimate  Std. Error t value  Pr(>|t|)
(Intercept) -0.18685736  0.31671471 -0.5900 0.5555141
educ_hat_2   0.08039177  0.02303514  3.4900 0.0005337 ***
exper        0.04309732  0.01606065  2.6834 0.0075728 **
expersq     -0.00086280  0.00044548 -1.9368 0.0534340 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

14.2.2 Task 8

Estimate the above equation by 2SLS (IV estimation) using R's built-in command `ivreg`. Compare the coefficients and standard errors with what you obtained manually.

```
iv_2 <- ivreg(lwage ~ educ + exper + expersq | huseduc + motheduc + fatheduc +
                exper + expersq, data = mroz)
summary(iv_2)
```

```
Call:
ivreg(formula = lwage ~ educ + exper + expersq | huseduc + motheduc +
    fatheduc + exper + expersq, data = mroz)

Residuals:
     Min       1Q   Median       3Q      Max
-3.08378 -0.32135  0.03538  0.36934  2.35829

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.1868574  0.2853959  -0.655 0.512996
educ         0.0803918  0.0217740   3.692 0.000251 ***
exper        0.0430973  0.0132649   3.249 0.001250 **
expersq     -0.0008628  0.0003962  -2.178 0.029976 *

Diagnostic tests:
                 df1 df2 statistic p-value
Weak instruments   3 422   104.294  <2e-16 ***
Wu-Hausman         1 423     2.732  0.0991 .
Sargan             2  NA     1.115  0.5726
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6693 on 424 degrees of freedom
Multiple R-Squared: 0.1495, Adjusted R-squared: 0.1435
Wald test: 11.52 on 3 and 424 DF,  p-value: 2.817e-07
```

Let us also integrate heteroscedasticity-robust standard errors.

```
coeftest(iv_2, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

               Estimate  Std. Error t value  Pr(>|t|)
(Intercept) -0.18685736  0.30126251 -0.6202 0.5354280
educ         0.08039177  0.02170330  3.7041 0.0002402 ***
exper        0.04309732  0.01530642  2.8156 0.0050951 **
expersq     -0.00086280  0.00042166 -2.0462 0.0413549 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

14.2.3 Task 9

Test for the relevance of the chosen instruments.

14.2.3.1 Guidance

The chosen instruments should sufficiently explain the variation in the endogenous variable (i.e. the education level). The F-statistic obtained in the first stage is 63.3, which is greater than the widely accepted threshold of 10. Hence, we conclude that all the instrument set sufficiently explain the variations in education (at least one of the them has an impact that is different than zero). The instruments in this example are are relevant.

14.2.4 Task 10

Test for the overidentifying restrictions in the above estimation. Explain what this test does.

14.2.4.1 Guidance

Instruments used in IV estimation should not normally belong to the main model of interest (i.e. in the case of this example, should not be a determinant of individual's wage) and they should satisfy the relevance and exogeneity assumptions. We confirmed above that the instrument set is relevant. We can check for the exogeneity assumption by applying Sarjan's J test of overidentifying restrictions.

First, we save the residuals from the iv estimation.

```
mroz$resid_iv_2 <- residuals(iv_2)
```

We then regress these saved residuals on all avilable exogenous variables (the instruments + the exogenous variables of the wage model)

```
overid_test <- lm(resid_iv_2 ~ huseduc + motheduc + fatheduc + exper + expersq
                  , data = mroz)
summary(overid_test)
```

```

Call:
lm(formula = resid_iv_2 ~ huseduc + motheduc + fatheduc + exper +
    expersq, data = mroz)

Residuals:
     Min       1Q   Median       3Q      Max
-3.07503 -0.32777  0.04156  0.37759  2.33621

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.606e-03  1.773e-01   0.049    0.961
huseduc      6.781e-03  1.143e-02   0.593    0.553
motheduc    -1.039e-02  1.187e-02  -0.875    0.382
fatheduc     6.734e-04  1.138e-02   0.059    0.953
exper        5.603e-05  1.323e-02   0.004    0.997
expersq     -8.882e-06  3.956e-04  -0.022    0.982

Residual standard error: 0.67 on 422 degrees of freedom
Multiple R-squared:  0.002605,  Adjusted R-squared:  -0.009212
F-statistic: 0.2205 on 5 and 422 DF,  p-value: 0.9537
```

We then calculate the chi-squared test-statistic by multiplying the number of observations in sample (n) with the R-squared from above regression. Note how we call these two in the R code provided below.

```
# Calculate chi-squared test statistic
sum_stat <- nrow(mroz) * summary(overid_test)$r.squared

print(sum_stat)
```

```
[1] 1.115043
```

We can either compare this with a chi-squared table value with 3-1=2 degrees of freedom, or ask R to calculate the corresponding p-value. I find the latter easier:

```
# p-value for the calculated test-statistic with 2 degrees of freedom
pchisq(sum_stat, df = 2, lower.tail = FALSE)
```

```
[1] 0.5726264
```

The p-value is 0.57. There is not enough evidence to reject the null hypothesis that "the instrument set is exogenous". Hence, overidentifying restrictions are valid.

The results of this task and the previous one confirm that we have a valid set of instruments.

14.2.5 Task 11

Test for the existence of endogeneity in the wage regression.

14.2.6 Guidance

We will be applying the Durbin-Wu-Hausman test for endogeneity. We will need the saved residuals from the first stage of the 2SLS estimation. Let's save this under the name `resid_2` (to differentiate from the single IV case we ran at the beginning)

```
mroz$resid_2 <- residuals(stage1_2)
```

We then estimate the original model of interest by additionally including these saved residuals.

```r
dwh_test <- lm(lwage ~ educ + exper + expersq + resid_2,
               data = mroz)
# Report the results with robest standard errors
coeftest(dwh_test, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

               Estimate  Std. Error t value  Pr(>|t|)
(Intercept) -0.18685736  0.29772776 -0.6276 0.5305971
educ         0.08039177  0.02136857  3.7621 0.0001922 ***
exper        0.04309732  0.01510368  2.8534 0.0045373 **
expersq     -0.00086280  0.00041557 -2.0762 0.0384826 *
resid_2      0.04718901  0.02630679  1.7938 0.0735598 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

14.3 Further Reading

See https://cran.r-project.org/web/packages/ivreg/vignettes/ivreg.html for more information and another empirical example.

Part IX

Seminar 10 (25 March 2025)

15 IV Estimation: The Role of Institutions in Economic Growth

In their study titled *The colonial origins of comparative development*, Acemoglu, Johnson and Robinson (2001) explore the role of institutions on economic performance, measured as income per capita.

Acemoglu, D., Johnson, S., Robinson, J.A. (2001) The colonial origins of comparative development: an empirical investigation, *The American Economic Review*, 91(5): 1369-1401. Available from https://economics.mit.edu/sites/default/files/publications/colonial-origins-of-comparative-development.pdf

Data Source: https://economics.mit.edu/people/faculty/daron-acemoglu/data-archive

Some of the variables in their data are listed below:

Variable	Definition
`shortnam`	3 letter country name
`logpgp95`	log PPP GDP pc in 1995, World Bank
`avexpr`	average protection against expropriation risk
`f_brit`	British Colony (Flopsexpsn)
`f_french`	French Colony (Flopsexpans)
`logem4`	log settler mortality

Before continuing with the analysis below, it is recommended that you read through the highlighted text in the paper (provided on module Aula page). Try to find answers to the following: - Why institutions or institutional quality may be considered as *endogenous* in a growth or national income model. - What is the authors' strategy to break this endogeneity?

On the first page, the authors explain that "[c]ountries with better *institutions*, more secure property rights and less distortionary policies will invest more in physical and human capital, and will use these factors more efficiently to achieve a greater level of income." While better institutions have a positive impact on national income, more developed countries are more likely to have better institutions and more established property rights. This simultaneity between national income and institutional quality creates an endogeneity problem. Because any shock affecting the national income (through the error term) will in return influence institutions, creating a correlation between the error term and institutions.

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(ivreg) # IV estimation
library(sandwich) # for robust se calculations
library(lmtest) # for coeftest
```

```
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
library(stargazer) # create formatted tables
```

```
Please cite as:

 Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.
 R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

```
library(lmtest) # use coeftest function to display resuls
```

Load the `acemoglu_2001` data. The data is provided to you in `RData` format.

```
load("./assets/data/acemoglu_2001.RData")
```

Let's assign a shorter name for our data

```
df <- acemoglu_2001
```

15.0.1 Task 1

Estimate the following regression using OLS and comment on the estimation results.

$$logpgp95 = \beta_1 + \beta_2 avexpr + \beta_3 f\_brit + \beta_4 f\_french + u$$

15.0.1.1 Guidance

```
ols <- lm(logpgp95 ~ avexpr + f_brit + f_french, data = df)
coeftest(ols, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

             Estimate Std. Error t value  Pr(>|t|)
(Intercept)  4.838521   0.359791 13.4481 < 2.2e-16 ***
avexpr       0.527621   0.051401 10.2648 7.873e-15 ***
f_brit      -0.306681   0.218114 -1.4061   0.16486
f_french    -0.377069   0.204934 -1.8400   0.07072 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The British and French colony dummies are statistically insignificant at 5% level. The origin of the colonising country does not appear to have a statistically significant impact on the GDP per capita of the colony.

The institution variable is statistically significant with a positive sign, implying that as expected, better institutions have an increasing impact on the national income.

15.0.2 Task 2

Re-estimate the above equation, this time by using settler mortality as an instrument for institutional quality.

15.0.2.1 Guidance

Instrumental variable estimation is also referred to as two-stage least squares because of the two-stage estimation that it requires.

Stage 1: Regress the endogenous variable on all exogenous variables (instruments and the independent variables in the model other than the endogenous variable) and calculate predictions for the endogenous variable.

```
# Regress endogenous variable on IV and other exogenous variables
step1 <- lm(avexpr ~ logem4 + f_brit + f_french, data = df)

# Obtain predictions for the enodgenous variable
df$avexpr_hat <- predict(step1)
```

Stage 2: In stage two, we use the predicted values of the endogenous variable from the first stage. We estimate the main model by replacing the endogenous variable with its predictions.

```
step2 <- lm(logpgp95 ~ avexpr_hat + f_brit + f_french, data = df)
coeftest(step2, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

            Estimate Std. Error t value  Pr(>|t|)
(Intercept)  1.37240    0.94952  1.4454   0.15356
avexpr_hat   1.07785    0.15791  6.8257 4.957e-09 ***
f_brit      -0.77770    0.29962 -2.5957   0.01185 *
f_french    -0.11697    0.21860 -0.5351   0.59456
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The coefficient of the institution variable is lower in the OLS estimation than in the 2SLS estimation. It implies that because of endogeneity, OLS underestimates the impact of institutions (i.e. negative bias).

The comparison of standard errors on the other hand, reveals the inefficiency of the 2SLS estimation. This is because we are instrumenting the institution variable. The higher the correlation between the endogenous variable and the instrument, the lower will be the difference in standard errors of the OLS and 2SLS; the lower the correlation between the endogenous variable and the instrument, the higher the 2SLS standard errors will be. The latter case implies weak instrumentation.

15.0.3 Task 3

Is settler mortality a good (valid) instrument? Please test and discuss.

15.0.3.1 Guidance

Let's print the results of first stage regression (`step`) from above.

```
coeftest(step1, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

             Estimate Std. Error t value  Pr(>|t|)
(Intercept)  8.746647   0.741542 11.7952 < 2.2e-16 ***
logem4      -0.534399   0.156066 -3.4242  0.001117 **
f_brit       0.629348   0.371462  1.6942  0.095404 .
f_french     0.047405   0.402817  0.1177  0.906712
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can see that the instrument log settler mortality (*logem4*) has a statistically significant impact on average protection against expropriation risk (*avexpr*) (the endogenous variable), implying that it has an explanatory power in predicting the values of average protection against expropriation risk. This confirms its **relevance** as is an instrument. We can also check the correlation coefficient between the two.

```
cor(df$avexpr, df$logem4)
```

```
[1] -0.5197417
```

The pairwise correlation coefficient between the two variables is moderate (-0.5197). Log settler mortality is a relevant instrument, though better alternatives could also be sought.

15.0.4 Task 4

Test whether there is endogeneity problem in estimation of the above equation.

15.0.4.1 Guidance

We apply the Durbin-Wu-Hausman Test. The test consists of two-stages. The first stage is the same as the first stage regression of the IV estimation (i.e. Two-Stage Least Squares). In the first stage we regress the instrument on the exogenous variables (the instrument and the other exogenous variables in the main model) and save the residuals from this model. We then estimate the main model, this time by additionally including the residuals from the first stage. Statistical significance of this residual term will imply endogeneity.

The null hypothesis of this test is that *there is no endogeneity* and the alternative hypothesis is *there is endogeneity*.

Step 1 Save the residuals from the first stage of 2SLS.

```
df$resid_step1 <- residuals(step1)
```

Step 2 Estimate the main model of interest with adding these residuals as one of the independent variables.

```
dwh <- lm(logpgp95 ~ avexpr + f_brit + f_french + resid_step1,
          data = df)
coeftest(dwh, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

             Estimate Std. Error t value  Pr(>|t|)
(Intercept)  1.37240    0.76424  1.7958  0.077651 .
avexpr       1.07785    0.12441  8.6639 4.166e-12 ***
f_brit      -0.77770    0.23425 -3.3199  0.001548 **
f_french    -0.11697    0.17953 -0.6515  0.517228
resid_step1 -0.68466    0.15548 -4.4037 4.545e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The residual term is highly statistically significant. We reject the null hypothesis of no endogeneity. This implies that there is endogeneity. Since we have a valid instrument, we choose IV regression over OLS.

15.0.5 Task 5

Use R's `ivreg` function to obtain the 2SLS estimation results.

15.0.5.1 Guidance

```
iv <- ivreg(logpgp95 ~ avexpr + f_brit + f_french |
           logem4 + f_brit + f_french,
          data = df)
coeftest(iv, vcov = vcovHC, type = "HC1")
```

```
t test of coefficients:

            Estimate Std. Error t value  Pr(>|t|)
(Intercept)  1.37240    1.60014  0.8577   0.39448
avexpr       1.07785    0.24746  4.3556 5.264e-05 ***
f_brit      -0.77770    0.37703 -2.0627   0.04348 *
f_french    -0.11697    0.34731 -0.3368   0.73744
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The manually calculated 2SLS regression results have inflated standard errors because of the two-stage approach of using the predictions of the endogenous variable from first stage. R's `ivreg()` function corrects for that bias in standard error estimates. Hence, although we have estimated the first and second stage regressions by OLS (the `lm()` function), to conduct the necessary checks and tests, use `ivreg()` when it is time to report the results!

15.0.6 Task 6

Compare the estimates for institutional quality in OLS and 2SLS regressions.

15.0.6.1 Guidance

The below table presents OLS, the second stage of 2SLS and the results of `ivreg()`.

```
stargazer(ols, step2, iv, type = "text")
```

```
===============================================================
                                Dependent variable:
                            ---------------------------------
                                       logpgp95
```

	OLS		instrumental variable
	(1)	(2)	(3)
avexpr	0.528***		1.078***
	(0.065)		(0.218)
avexpr_hat		1.078***	
		(0.161)	
f_brit	-0.307	-0.778***	-0.778**
	(0.211)	(0.262)	(0.354)
f_french	-0.377	-0.117	-0.117
	(0.232)	(0.262)	(0.355)
Constant	4.839***	1.372	1.372
	(0.436)	(1.027)	(1.388)
Observations	64	64	64
R2	0.565	0.479	0.048
Adjusted R2	0.543	0.453	0.001
Residual Std. Error (df = 60)	0.705	0.772	1.043
F Statistic (df = 3; 60)	25.950***	18.387***	
Note:			*p<0.1; **p<0.05; ***p<0.01

The coefficient of the institution variable is lower in the OLS estimation than in the 2SLS estimation. It implies that, because of endogeneity, OLS underestimates the impact of institutions (i.e. negative bias).

The comparison of standard errors on the other hand, reveals the inefficiency of the 2SLS estimation. This is because we are instrumenting the institution variable. The higher the correlation between the endogenous variable and the instrument, the lower will be the difference in standard errors of the OLS and 2SLS; the lower the correlation between the endogenous variable and the instrument, the higher the 2SLS standard errors will be. The latter case implies weak instrumentation.

Part X

References

References

James, Gareth, Daniela Witten, Trevor Hastie, and Rob Tibshirani. 2023. *An Introduction to Statistical Learning*. 2nd edition. Springer. https://www.statlearning.com.

Kleiber, C., and A. Zeileis. 2008. *Applied Econometrics with r*. Springer.

Riegler, Robert. 2022. "R Workbook - Guidance for Worksheets." Aston University.

Wickham, Hadley, Mine Cetinkaya-Rundel, and Garrett Grolemund. n.d. *R for Data Science*. 2nd edition. O'Reilly. https://r4ds.hadley.nz/preface-2e.

Wilson, J. H., and B. Keating. 2007. *Business Forecasting*. 5th edition. McGraw-Hill.