Teagasc

Agriculture and Food Development Authority

Introduction to R

Antonia Santos

Program

PART I

Introduction to R and base R programming

PART II

Data manipulation

PART III

Data visualisation

PART IV

Introduction to modelling in R

Bibliography

- R Manual (`https://cran.r-project.org/doc/manuals/R-intro.html`)
- R for Data Science (2e) (`https://r4ds.hadley.nz`)
- Fundamentals of Data Visualisation (`https://clauswilke.com/dataviz/`)

Part I

Introduction to R and Base R Programming

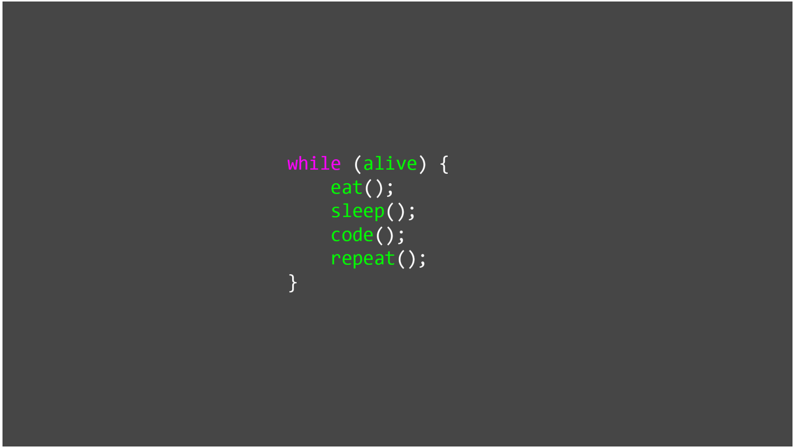
1 CODING

2 R AND RStudio

3 BASIC CONCEPTS

CODING

What is coding?



```
while (alive) {
    eat();
    sleep();
    code();
    repeat();
}
```

R AND RStudio

What is R? And RStudio?

R AND RStudio

- R
    - Created by Ross Ihaka and Robert Gentleman (University of Auckland/R Development Core Team).
    - Open source.

R AND RStudio

- R
  - Created by Ross Ihaka and Robert Gentleman (University of Auckland/R Development Core Team).
  - Open source.
- RStudio is an Integrated Development Environment (IDE).

R AND RStudio

- R
  - Created by Ross Ihaka and Robert Gentleman (University of Auckland/R Development Core Team).
  - Open source.
- RStudio is an Integrated Development Environment (IDE).
- Download:
  - R: `https://www.r-project.org`.
  - RStudio: `https://www.rstudio.com`.

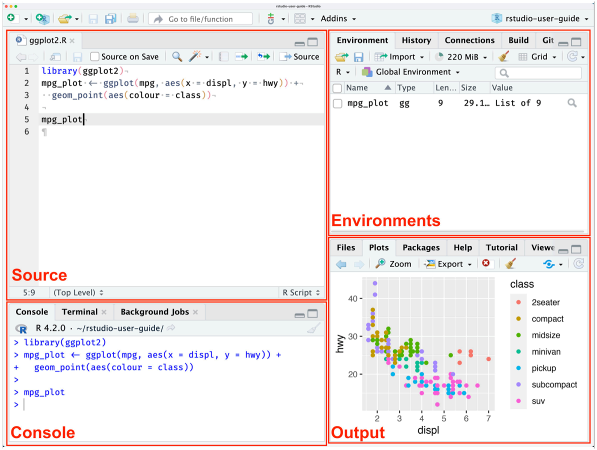
BASIC CONCEPTS

RStudio interface.

- Interface
  - Source pane
  - Console pane
  - Environment pane (Environment, History, Connections, Build, VCS, and Tutorial)
  - Output pane (Files/ Plots / Packages / Help)

BASIC CONCEPTS

RStudio interface.



```r
library(ggplot2)
mpg_plot <- ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(colour = class))

mpg_plot
```

Name	Type	Len...	Size	Value
mpg_plot	gg	9	29.1…	List of 9

```
> library(ggplot2)
> mpg_plot <- ggplot(mpg, aes(x = displ, y = hwy)) +
+   geom_point(aes(colour = class))
>
> mpg_plot
>
```

Figure: https://docs.posit.co/ide/user/ide/guide/ui/ui-panes.html

BASIC CONCEPTS

RStudio interface.

- Source pane
  - This is where you write and edit your R scripts (.R files).
  - You can write code, comments, and documentation in this area, and run selected lines by pressing Ctrl+Enter (Windows) or Cmd+Enter (macOS).
  - This area allows you to save your code, which is essential for keeping a record of your analysis.
- Console pane
  - The console is where you can execute commands interactively. It shows the output of the code you run and is useful for testing small code snippets or running analyses directly.
  - You can type commands directly into the console and see immediate feedback or results.
- Environment pane
  - Environment Tab: Displays all the objects (e.g., data frames, variables, functions) currently in your R workspace. You can inspect and manage your data and objects here.
  - History Tab: Shows a list of all the commands you have previously executed during the current R session. This is helpful for tracking your steps or re-running commands.

BASIC CONCEPTS

RStudio interface.

- Output pane
  - Files Tab: A file explorer that lets you browse files in your working directory and manage files (open, delete, move, etc.).
  - Plots Tab: Displays any plots or graphs that you generate with R (e.g., ggplot2 visualisations). You can export these plots as images or PDFs.
  - Packages Tab: Manage R packages installed in your system. You can install, load, or remove packages from here.
  - Help Tab: Provides access to R's help documentation. You can search for information about functions, packages, and datasets.
  - Viewer Tab: Used to display web-based content, such as HTML reports generated from R Markdown files.

Note 1: The RStudio layout may look slightly different based on your configuration, but the core functionality remains the same.

BASIC CONCEPTS

Files.

- Files
    - **.R** - To save codes and scripts.
    - **.RData** - To save workspace objects.
    - **.Rhistory** - To save the history of executed commands.

BASIC CONCEPTS

First steps

- Creating a script:
  - Click on “File” → “New File” → “R Script” to open a new script file.
- Writing and running code:
  - Type code in the script editor: “Hello World!”;
  - Highlight the lines you want to run, and press Ctrl+Enter (Windows) or Cmd+Enter (macOS) to execute the code in the console (or to run all the code, use the Run button at the top of the script editor).
- Saving your work:
  - You can save your R script by clicking “File” → “Save As” and giving your file a .R extension.

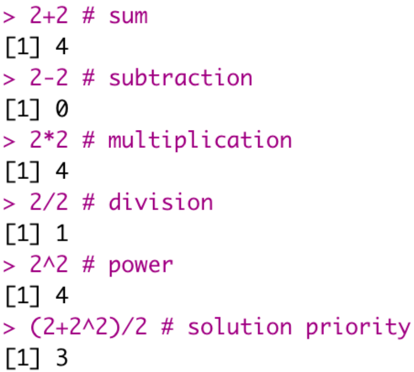
BASIC CONCEPTS

Some observations.

- Everything in R is an object.
  - Think of objects like containers that hold data or things you can work with. For example: numbers, words, list of things, group of numbers, functions...
- There are differences between uppercase characters and lowercase characters.
- Parentheses, square brackets and braces:
  - ( ): to group objects inside a function.
  - [ ]: to group functions inside other functions.
  - { }: to index objects inside other objects.
- Comments can be inserted after the # character.
- The dot (.) or underscore (_) symbols can be used, but not spaces.
- A cheatsheet providing a detailed explanation of some available functions can be found at `https://rstudio.github.io/cheatsheets/html/rstudio-ide.html`.

BASIC CONCEPTS

Calculator.



```
> 2+2 # sum
[1] 4
> 2-2 # subtraction
[1] 0
> 2*2 # multiplication
[1] 4
> 2/2 # division
[1] 1
> 2^2 # power
[1] 4
> (2+2^2)/2 # solution priority
[1] 3
```

BASIC CONCEPTS

Assigning.

To assign values to objects, just use the ← operator, which is the combination of the < operator with –. Alternatively, we can use the = operator.

```
> x <- 10 # the value 10 is saved in the object x
> y <- x + 10
```

The ← operator is preferred by many R users because it clearly distinguishes assignment from equality checking (==). The = operator can also be used for assignment, but it is more commonly used for setting function arguments. While it behaves similarly to ← when assigning variables, using ← is generally recommended for clarity and consistency in most R code.

BASIC CONCEPTS

List and remove objects.

- To list the objects in the environment use the function `ls()`;
- To remove the objects from the environment use the function `rm()`

BASIC CONCEPTS

Your turn.

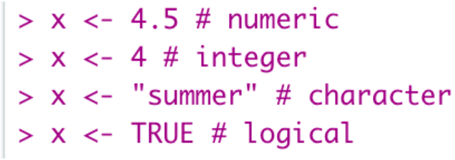
Question 1: *Find the volume of a cylindrical water tank whose base radius is 25 inches and whose height is 120 inches. Use* $\pi = 3.14$.

Remember: $volume = \pi \times radius^2 \times height.$

BASIC CONCEPTS

Types of variables.

R has different classes to accommodate different types of data.



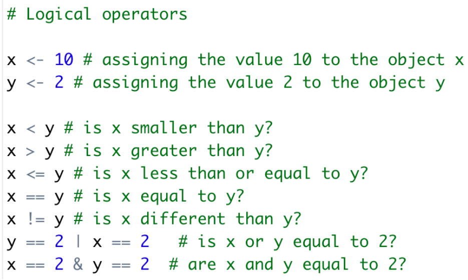
```
> x <- 4.5 # numeric
> x <- 4 # integer
> x <- "summer" # character
> x <- TRUE # logical
```

- ✓ We can check the class of any object by using the built-in `class()` function.
- ✓ We can check the structure of any object by using the built-in `str()` function.

BASIC CONCEPTS

Logical operators.

Logical operators are binary operators for performing tests between two variables (objects). These operations return the value TRUE or FALSE.



```r
# Logical operators

x <- 10 # assigning the value 10 to the object x
y <- 2 # assigning the value 2 to the object y

x < y # is x smaller than y?
x > y # is x greater than y?
x <= y # is x less than or equal to y?
x == y # is x equal to y?
x != y # is x different than y?
y == 2 | x == 2   # is x or y equal to 2?
x == 2 & y == 2  # are x and y equal to 2?
```

Basic Concepts

Your turn

Question 2: *You have three participants with scores of 85, 50, and 75. Use logical expressions (and, or, not) to answer the following:*

- *Did all participants score above 40?*
- *Did any participant score exactly 50?*
- *Is it true that none of the participants scored less than 30?*

BASIC CONCEPTS

Structures: Vectors

- Vectors are one-dimensional collections of data of the same type (e.g., all numbers or all characters).
- You can create a vector using the c( ) function.
- You can access elements of a vector using the square brackets [ ].

BASIC CONCEPTS

Some functions

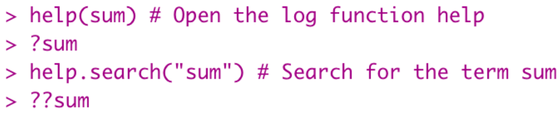
Table: Functions and descriptions.

Function	Description
sum()	Returns the sum
mean()	Returns the mean/average
sd()	Returns the standard deviation
median()	Returns the median
var()	Returns the variance
cor()	Returns the correlation between two vectors
min()	Returns the minimum
max()	Returns the maximum
range()	Returns the minimum and maximum
summary()	Returns a data summary
quantile()	Returns the quantiles of the numeric vector

BASIC CONCEPTS

Help function.

The help function (or ?) allows you to find the help file of the functions.



```
> help(sum) # Open the log function help
> ?sum
> help.search("sum") # Search for the term sum
> ??sum
```

BASIC CONCEPTS

Your turn.

Question 3: *Create a numeric vector named* **scores** *with the following values: [12, 45, 67, 89, 34, 23, 50, 8, 62]. Then:*

- *Select only the values smaller than 50.*
- *Calculate the mean and standard deviation of these values.*

BASIC CONCEPTS

Structures: Matrix

- Matrices are two-dimensional arrays that store elements of the same type (e.g., all numeric).
- You can create a matrix using the matrix( ) function.
- You can access elements of a matrix using the square brackets [ ].

BASIC CONCEPTS

Your turn.

Question 4: *Create a 3x4 numeric matrix named* **sales_data** *with the following values (filled by row):*

[15, 23, 42, 31, 8, 12, 50, 27, 20, 35, 10, 18]

- *Name the rows as "Store_A", "Store_B", and "Store_C".*
- *Name the columns as "Jan", "Feb", "Mar", and "Apr".*

Answer the following:

- *Which store had the highest sales in March ("Mar")?*
- *What is the total sales for "Store_B" across all months?*
- *Extract all sales values greater than 30.*

BASIC CONCEPTS

Structures: Data frames

- Data frames are two-dimensional tables, similar to spreadsheets or SQL tables. Each column in a data frame can hold different data types (numeric, character, etc.).
- You can create a data frame using the data.frame( ) function.

BASIC CONCEPTS

Your turn.

Question 5: *Generate a data frame named* **my_data** *with 10 rows and 3 columns:*

- *One numeric column (named ”values”) with random numbers between 1 and 100;*
- *One logical column (named ”flags”) with randomly selected TRUE/FALSE values;*
- *One character column (named ”categories”) with randomly selected categories from: ”A”, ”B”, ”C”.*

Ensure there is exactly one missing value (NA) in the numeric column. Then:

- *Display rows with missing values;*
- *Replace the missing value in the numeric column with the maximum value of that column;*
- *Verify there are no more missing values;*
- *Convert the ”categories” column to a factor;*
- *Create a frequency table showing counts for each category.*

BASIC CONCEPTS

Structures: Lists

- Lists can hold elements of different types, including numbers, strings, vectors, and even other lists.
- You can create a list using the list( ) function.

BASIC CONCEPTS

Your turn.

Question 6: *Create a list called* **my_city** *with:*

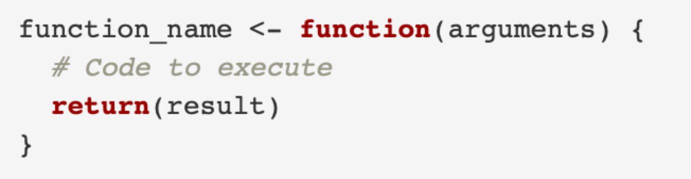
- *A character element 'name' with your city name;*
- *A numeric element 'population';*
- *A logical element 'capital' (TRUE or FALSE);*
- *A vector 'districts' with 3 district names.*

Access and print the population from your list. Then, add a new element 'area' with the city's area.

BASIC CONCEPTS

Functions

In addition to using R's built-in functions, you can write your own custom functions to perform specific tasks. Writing your own function allows you to create reusable blocks of code for operations you might need frequently.



```
function_name <- function(arguments) {
  # Code to execute
  return(result)
}
```

BASIC CONCEPTS

Your turn.

Question 7: *In agronomy, crop yield is often measured in tons per hectare. However, some researchers need the yield in kilograms per hectare for specific analyses. Create a function in R called* `convert_to_kg` *that:*

- *Takes one argument:* `yield_tons` *(the yield in tons per hectare).*
- *Converts it to kilograms per hectare (1 ton = 1,000 kilograms).*
- *Returns the converted value.*

BASIC CONCEPTS

If else

- if/else: Used for conditional execution on a single value.
  - if: Checks a condition; if it’s TRUE, runs the following code block.
  - else: Executes an alternative block if the condition is FALSE.
- ifelse: A vectorised function ideal for performing element-wise checks on a vector.
  ifelse(test_expression, value_if_true, value_if_false)

BASIC CONCEPTS

Your turn.

Question 8: *Create a function to check if a number is positive or negative.*

BASIC CONCEPTS

Loops

- For loop:
  - Iterates over each element in a sequence.
  - Use when you know the number of iterations in advance.
- While loop:
  - Repeats a block of code as long as a condition remains true.
  - Use when the number of iterations is not predetermined.

BASIC CONCEPTS

Your turn.

Question 9: *Generates a vector of 10 numbers. Uses loops to calculate the sum of all numbers and to find the largest number*

BASIC CONCEPTS

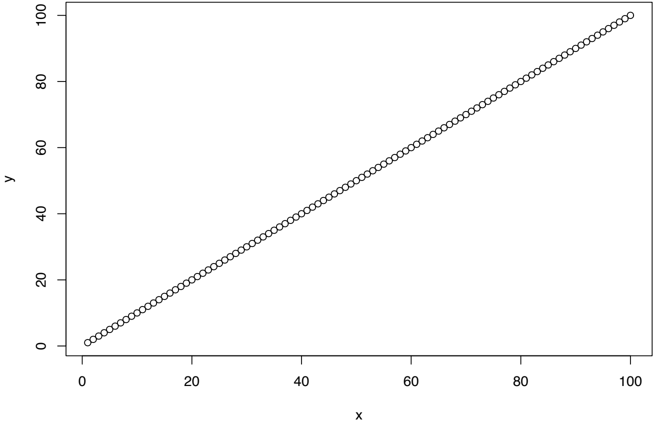
Packages.

- R base and packages.

  > A collection of functions that can be written in different programming languages that are called directly from within R. A package contains code, data and documentation.

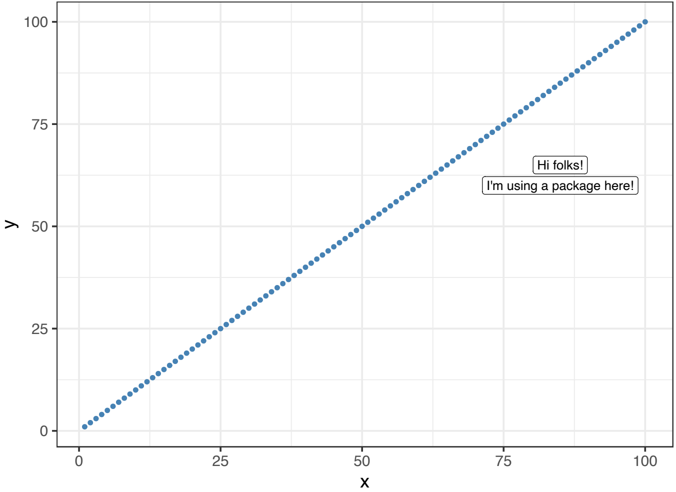
BASIC CONCEPTS

R base and packages.



BASIC CONCEPTS

R base and packages.



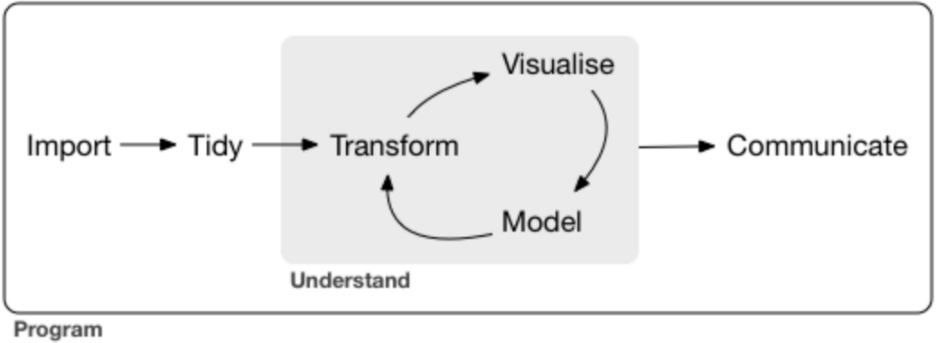
BASIC CONCEPTS

Your turn.

Question 10: *Install the following packages and look at the vignettes:*

- *dplyr*
- *tidyr*
- *lme4*
- *tidyverse*

DATA MANIPULATION



DATA MANIPULATION

Datasets

Data set 1: Seoul Bike Sharing Demand Dataset.
The dataset provides hourly counts of public bicycle rentals in the Seoul Bike Sharing System. It includes detailed weather data (temperature, humidity, wind speed, visibility, dew point, solar radiation, snowfall, and rainfall), along with rental counts and date information.

Data set 2: Sample data
This data has 6 columns and is in excel format.

WORKING WITH DIRECTORIES IN R

Checking the current directory

- The working directory is the folder where R looks for files to read or write;
- To see your current working directory: `getwd()`;
- This function returns the path to the current directory;
- To change the working directory, use: `setwd("path/to/your/directory")`
- Replace `"path/to/your/directory"` with the full path of the folder you want.

WORKING WITH DIRECTORIES IN R

Checking the current directory

- The working directory is the folder where R looks for files to read or write;
- To see your current working directory: `getwd()`;
- This function returns the path to the current directory;
- To change the working directory, use: `setwd("path/to/your/directory")`
- Replace `"path/to/your/directory"` with the full path of the folder you want.

In RStudio, you can easily set the working directory:

1. Click on Session in the menu bar.
2. Select Set Working Directory > Choose Directory....
3. Navigate to the folder and confirm.

DATA MANIPULATION

Importing data sets

Importing CSV files

- To import CSV files, use the `read.csv()` function. Example:

```
data <- read.csv("path/to/your/file.csv", header = TRUE, sep = ",")
```

- `file`: The path to the CSV file.
- `header`: Logical, TRUE if the first row contains column names.
- `sep`: Specifies the delimiter (default is ”,” for comma-separated files).

DATA MANIPULATION

Importing data sets

Importing Excel files

- To import Excel files, use the `readxl` package and the `read_excel()` function.
- First, install the package (if not already installed):

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

- Example:

```
data <- read_excel("path/to/your/file.xlsx", sheet = "Sheet1")
```

- `path`: The path to the Excel file.
- `sheet`: Specifies the sheet name or index (e.g., ”Sheet1” or 1).

DATA MANIPULATION

Importing data sets

Importing SAS files

- To import SAS files, use the **`haven`** package and the **`read_sas()`** function.
- First, install the package (if not already installed):

```
install.packages("haven")
library(haven)
```

- Example:

```
data <- read_sas("path/to/your/file.sas7bdat") )
```

- **`path`**: The path to the SAS file.

DATA MANIPULATION

Understanding missing values (NAs)

- NA stands for "Not Available" and represents missing or undefined data in R.
- Use `is.na()` to identify missing values in a dataset.
- Use `sum(is.na())` to count the total number of NAs in a dataset.
- Use `na.omit()` to remove rows with missing values.
- Use `which()` to find the exact positions of NAs in the dataset.
- Many functions allow you to ignore NAs using `na.rm = TRUE`.

BASIC CONCEPTS

Your turn.

Question 11: *Using the 'data1' dataset, check if there are any missing values in any column. Then, calculate the mean and standard deviation of the variable 'Rainfall.mm.'*

Question 12: *Using the 'data2' dataset, replace the missing values related to Frank.*

DATA MANIPULATION

Native pipe operator

- Pipes are powerful tools for simplifying and clarifying sequences of multiple operations.
- The pipe operator makes reading a sequence of code much more logical, easier, and understandable.
- The | > is R's native pipe operator, available from version 4.1 onwards.
- The | > operator takes the result on its left side and uses it as the first argument of the function on its right side.

DATA MANIPULATION

Introduction to `dplyr`

- A package for easy and efficient data manipulation.
- Provides clear and intuitive functions for working with tabular data.
- Simplifies tasks like selecting, filtering, and transforming data.
- Makes code easier to read and write.

DATA MANIPULATION

Function: `select()`

- Selects specific columns from a dataset.

```
select(data, column1, column2, ...)
```

DATA MANIPULATION

Function: `select()`

- Selects specific columns from a dataset.

```
select(data, column1, column2, ...)
```

Question 14: *Using 'dataset1', create a new object and select only the columns 'Date', 'Rented.Bike.Count', 'Hour', and 'Seasons'.*

DATA MANIPULATION

Function: `rename()`

- Renames columns in a dataset while keeping everything else unchanged.

```
rename(data, new_name = old_name)
```

DATA MANIPULATION

Function: `rename()`

- Renames columns in a dataset while keeping everything else unchanged.

```
rename(data, new_name = old_name)
```

Question 16: *Using 'data1', rename the columns as follows:*

Rented.Bike.Count = RBC
Temperature.C. = Temp
Humidity... = Humidity
Solar.Radiation..MJ.m2. = SR
Rainfall.mm. = Rainfall
Snowfall..cm. = Snowfall

DATA MANIPULATION

Function: `mutate()`

- Adds or modifies columns in the dataset.

```
mutate(data, new_column = operation)
```

DATA MANIPULATION

Function: `mutate()`

- Adds or modifies columns in the dataset.

```
mutate(data, new_column = operation)
```

Question 18: *Using the 'data1' create a new column called* ***`Humidity_new`****, where*

$$Humidity\_new = Humidity/100$$

DATA MANIPULATION

Changing variable types

- Some analyses require specific types of variables (e.g., factors for categorical data, numeric for calculations).
- Use `as.factor()` to convert a numeric variable to a factor.
- Use `as.numeric()` carefully to convert a factor to numeric.
- Use `as.character()` to convert numeric variables to text.

DATA MANIPULATION

Changing variable types

- Some analyses require specific types of variables (e.g., factors for categorical data, numeric for calculations).
- Use `as.factor()` to convert a numeric variable to a factor.
- Use `as.numeric()` carefully to convert a factor to numeric.
- Use `as.character()` to convert numeric variables to text.

Question 20: *Using 'data1', convert the columns 'Seasons' and 'Holiday' to factor.*

DATA MANIPULATION

Function: `filter()`

- Filters rows based on a condition.

```
filter(data, condition)
```

DATA MANIPULATION

Function: `filter()`

- Filters rows based on a condition.

```
filter(data, condition)
```

Question 22: *Using the 'data1' filter the rows with Rainfall above 25mm and Season 'Spring'*

DATA MANIPULATION

Function: `summarise()`

- Creates a summary of the data.

```
summarise(data, summary_name = operation(column))
```

DATA MANIPULATION

Function: `summarise()`

- Creates a summary of the data.

```
summarise(data, summary_name = operation(column))
```

Question 24: *Using the 'data1' calculate the mean and the sd of the variable Rainfall.*

DATA MANIPULATION

Function: `group_by()`

- Groups data by one or more columns.

```
group_by(data, column)
```

DATA MANIPULATION

Function: `group_by()`

- Groups data by one or more columns.

```
group_by(data, column)
```

Question 26: *Using the ’data1’, group by ’Holiday’ and calculate the average Rainfall.*

DATA MANIPULATION

Function: `arrange()`

- Sorts rows in ascending or descending order.

```
arrange(data, column)
arrange(data, desc(column))
```

Question 27: *Using data1, display the data in ascending order of Temperature.*

DATA MANIPULATION

Seoul Bike Sharing Demand Dataset

1. **Import the dataset**: Read the `sbd.csv` file into R using the appropriate function and examine the structure.
2. **Summarise the dataset**: Use the `dplyr` package to summarise the dataset, showing the mean, median, and standard deviation for `Temperature(C)` and `Rented Bike Count`.
3. **Count seasonal data**: Use `count` to determine how many records there are for each `Seasons`.
4. **Filter by time**: Filter the data to show only records where `Hour` is between 6 and 9 (inclusive).
5. **Create a new column**: Use `mutate` to create a column `Temperature_F` that converts `Temperature(C)` to Fahrenheit using the formula:

$$\text{Temperature\_F} = \text{Temperature\_C} \times \frac{9}{5} + 32$$

Seoul Bike Sharing Demand Dataset

6. **Rename a column**: Rename the column `Rented.Bike.Count` to `Bike_Rentals` using `rename`.
7. **Select specific columns**: Use `select` to create a new dataset with only the columns `Date`, `Hour`, `Seasons`, and `Bike_Rentals`.
8. **Group by and summarise**: Group the data by `Seasons` and calculate the average `Rented Bike Count` and `Temperature(C)` for each season.
9. **Arrange by temperature**: Arrange the dataset by `Temperature(C)` in ascending order and display the first 10 rows.
10. **Subset by weather conditions**: Filter and display rows where `Humidity(%)` is greater than 80 and `Solar Radiation (MJ/m2)` is equal to 0.

PRINCIPLES OF DATA VISUALISATION

- What is data visualisation?
- What are the benefits?
- A practical example: A graph tells a story.

PRINCIPLES OF DATA VISUALISATION

Florence Nightingale's Rose Diagram

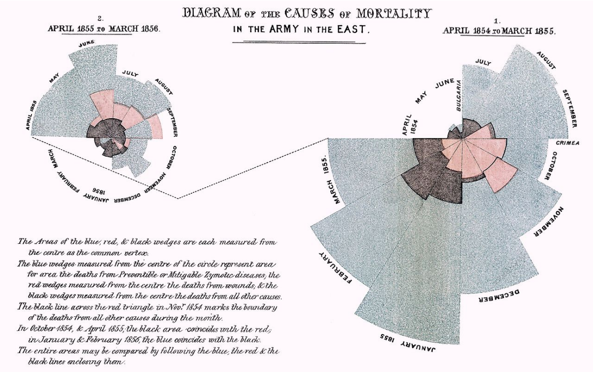


DIAGRAM OF THE CAUSES OF MORTALITY

IN THE ARMY IN THE EAST.

The Areas of the blue, red, & black wedges are each measured from the centre as the common vertex.

The blue wedges measured from the centre of the circle represent area for area the deaths from Preventible or Mitigable Zymotic diseases; the red wedges measured from the centre the deaths from wounds; & the black wedges measured from the centre the deaths from all other causes.

The black line across the red triangle in Novr. 1854 marks the boundary of the deaths from all other causes during the month.

In October 1854, & April 1855, the black area coincides with the red; in January & February 1856, the blue coincides with the black.

The entire areas may be compared by following the blue, the red & the black lines enclosing them.

PRINCIPLES OF DATA VISUALISATION

Florence Nightingale's Rose Diagram

Why does this chart tell a story?

- Context: It highlights a real problem (preventable deaths) within a specific context (the Crimean War).
- Clarity: The visualisation is simple and easy to understand, even for those without a background in statistics.
- Impact: The plot led to tangible changes (improvements in sanitary conditions).

PRINCIPLES OF DATA VISUALISATION

Fundamental Principles

- Clarity: Visualisations should be clear and easy to interpret.
- Accuracy: Represent the data correctly.
- Efficiency: Maximise information with minimal elements.
- Aesthetics: Make the chart visually appealing.
- Relevance: Only visualise data that is relevant to the message or story you are trying to tell.
- Choosing the right graph: Make sure you select an appropriate graph type for the data and insight you want to pass on.

PRINCIPLES OF DATA VISUALISATION

What to avoid?

- Confusing or cluttered charts.
- Inappropriate choice of chart type.
- Incorrect use of colours and scales.

PRINCIPLES OF DATA VISUALISATION

Misleading and bad visualisations

- Lets say we want to report on whether the number of crimes has increased over two selected years.

PRINCIPLES OF DATA VISUALISATION

Misleading and bad visualisations

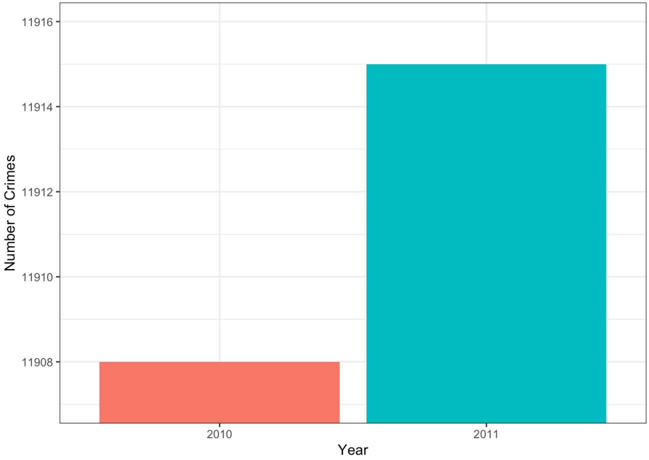


Figure: Example 1.

PRINCIPLES OF DATA VISUALISATION

Misleading and bad visualisations

- From the previous plot, it seems that the crime rate has jumped significantly from 2010 to 2011. However, can you notice anything suspicious about the plot?

PRINCIPLES OF DATA VISUALISATION

Misleading and bad visualisations

- From the previous plot, it seems that the crime rate has jumped significantly from 2010 to 2011. However, can you notice anything suspicious about the plot?
- Now, if we set the scale to start at zero we get:

PRINCIPLES OF DATA VISUALISATION

Misleading and bad visualisations

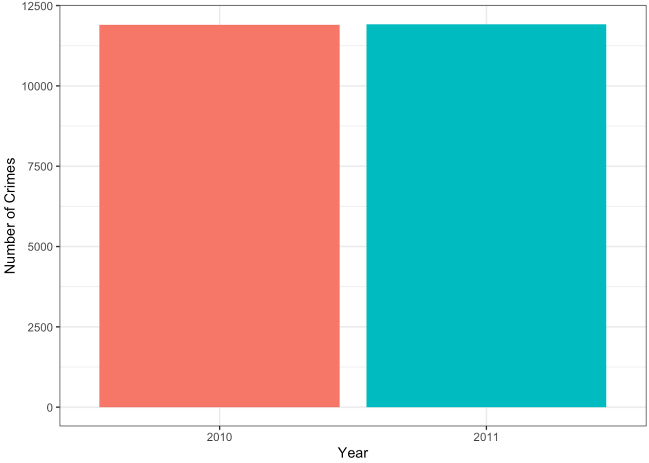


Figure: Example 1.

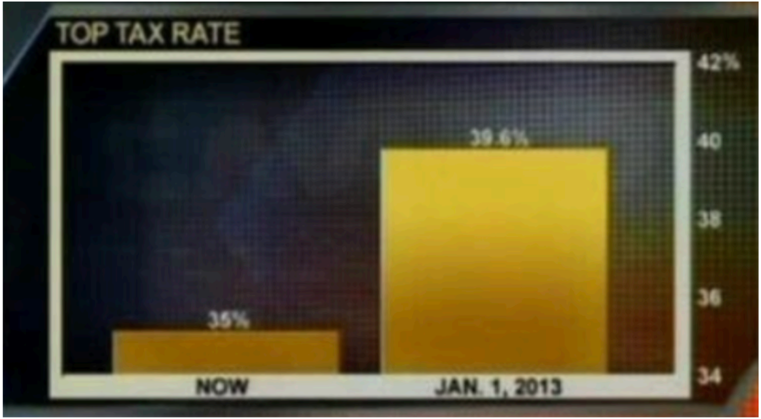
PRINCIPLES OF DATA VISUALISATION

Misleading and bad visualisations

- Now we can see that in reality, the crime rate has only increased marginally. This is a common tactic used in politics and the news when reporting. For example:

PRINCIPLES OF DATA VISUALISATION

Misleading and bad visualisations



Tax rate as reported on Fox news. Left bar is 35%. Right bar is 39.6%

Figure: Example 2.

PRINCIPLES OF DATA VISUALISATION

Misleading and bad visualisations

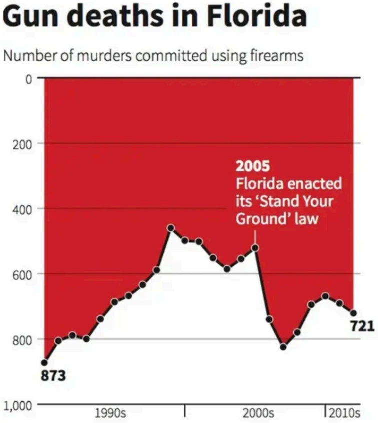


Figure: Example 3.

PRINCIPLES OF DATA VISUALISATION

Not All Bad

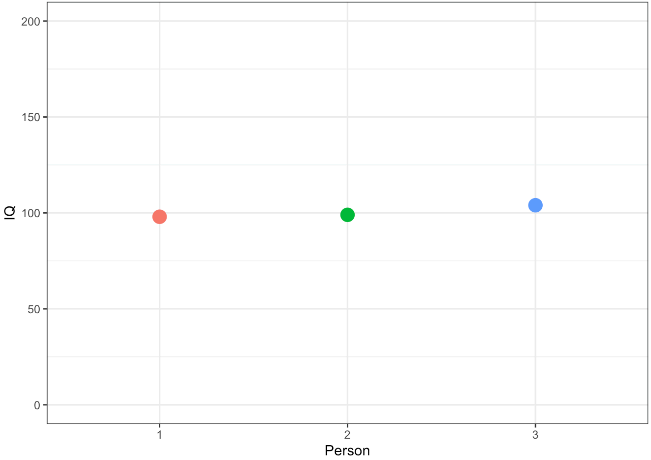


Figure: Example 4.

PRINCIPLES OF DATA VISUALISATION

Not All Bad

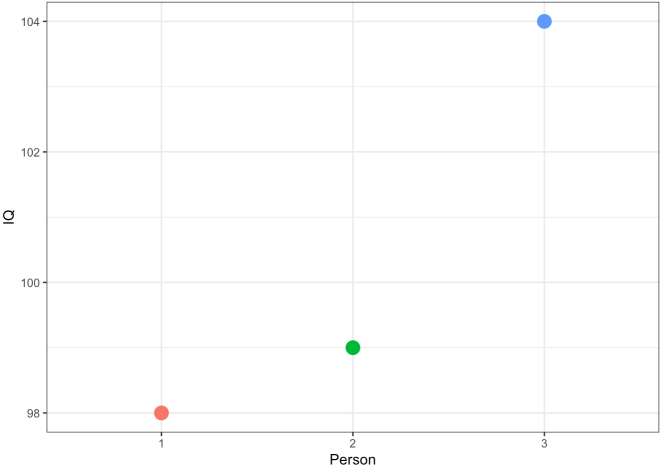


Figure: Example 4.

PRINCIPLES OF DATA VISUALISATION

Common visualisation types.

- Scatter plots: For visualising the relationship between two continuous variables.
- Bar plots: For comparing categorical data.
- Histograms: For displaying the distribution of a numeric variable.
- Line plots: For trends over time or continuous data.

DATA VISUALISATION

R base vs ggplot2

DATASETS

Diamonds Dataset

- Contains prices and attributes of over 50,000 diamonds.
- Variables:
  - carat (numeric): Weight of the diamond.
  - cut (factor): Quality of the cut (Fair, Good, Very Good, Premium, Ideal).
  - color (factor): Diamond color (D to J).
  - clarity (factor): Clarity measurement (I1, SI2, SI1, VS2, VS1, VVS2, VVS1, IF).
  - depth (numeric): Total depth percentage.
  - table (numeric): Width of the top relative to the widest point.
  - price (numeric): Price in USD.
  - x, y, z (numeric): Dimensions of the diamond.
- How to load it in R:

```r
data("diamonds")
```

DATASETS

Airquality Dataset

- Daily air quality measurements in New York (May to September 1973).
- Variables:
  - Ozone (numeric): Ozone concentration (ppb).
  - Solar.R (numeric): Solar radiation (Langley).
  - Wind (numeric): Wind speed (mph).
  - Temp (numeric): Temperature (F).
  - Month (integer): Month (1 = January, 12 = December).
  - Day (integer): Day of the month.
- How to load it in R:

```
data("airquality")
```

DATASETS

Iris Dataset

- Measurements of 150 iris flowers from three species.
- Variables:
  - Sepal.Length (numeric): Sepal length (cm).
  - Sepal.Width (numeric): Sepal width (cm).
  - Petal.Length (numeric): Petal length (cm).
  - Petal.Width (numeric): Petal width (cm).
  - Species (factor): Species of iris (setosa, versicolor, virginica).
- How to load it in R:

```
data("iris")
```

DATASETS

mtcars Dataset

- Data on 32 cars from the 1974 Motor Trend magazine.
- Variables:
  - mpg (numeric): Miles per gallon (fuel efficiency).
  - cyl (numeric): Number of cylinders.
  - disp (numeric): Displacement (cubic inches).
  - hp (numeric): Horsepower.
  - drat (numeric): Rear axle ratio.
  - wt (numeric): Weight (1000 lbs).
  - qsec (numeric): Quarter-mile time (seconds).
  - vs (numeric): Engine type (0 = V-shaped, 1 = straight).
  - am (numeric): Transmission type (0 = automatic, 1 = manual).
  - gear (numeric): Number of forward gears.
  - carb (numeric): Number of carburetors.
- How to load it in R:

```
data("mtcars")
```

PLOTTING IN BASE R

Histograms

- A histogram helps visualise the distribution of a continuous variable.
- Let’s create a histogram for the price of diamonds.

```
hist(diamonds$price,
main = "Histogram of Diamond Prices",
xlab = "Price",
col = "orange",
border = "black")
```

PLOTTING IN BASE R

Bar plots

- A bar plot visualises the frequency of categories in a factor variable.
- Let’s create a bar plot for the cut variable, which represents the quality of the diamond’s cut.

```
barplot(table(diamonds$cut),
main = "Bar Plot of Diamond Cut",
xlab = "Cut",
ylab = "Frequency",
col = "lightblue")
```

PLOTTING IN BASE R

Scatter Plot

- A scatter plot shows the relationship between two continuous variables.
- Let’s create a scatter plot between carat (diamond size) and price.

```
plot(diamonds$carat, diamonds$price,
main = "Scatter Plot of Carat vs Price",
xlab = "Carat",
col = "blue",
pch = 19)
```

PLOTTING IN BASE R

Line Plots

- A line plot is ideal to use when you want to show trends over time, compare multiple series, and display relationships between variables.
- Let's plot the average price of diamonds by carat size.

```
x = plot(avg_price$carat, y = avg_price$price,
type = "l",
main = "Line Plot of Average Price by Carat",
xlab = "Carat",
ylab = "Average Price",
col = "blue",
lwd = 2)
```

PLOTTING IN BASE R

Box Plots

- A box plot is ideal to use when you want to summarise the distribution of data, identify outliers, and understand data variability.
- Let’s create a boxplot.

```
boxplot(price   cut, data = diamonds,
main = ”Boxplot of Diamond Prices by Cut”,
xlab = ”Cut”,
ylab = ”Price (USD)”,
col = ”lightblue”,
border = ”black”)
```

PLOTTING IN BASE R

Your turn.

Question 28:

- *Create a histogram to visualize the distribution of temperature (Temp) in the airquality dataset.*
- *Customize the plot with appropriate colors, titles, and labels.*
- *Comment on what the histogram reveals about the distribution of temperatures.*

PLOTTING IN BASE R

Your turn.

Question 29:

- *Use the dplyr package to calculate the average ozone concentration (Ozone) by month (Month) in the airquality dataset.*
- *Create a bar plot to visualize the average ozone concentration for each month.*
- *Customize the plot with appropriate colors, titles, and labels.*
- *Comment on which months have the highest and lowest average ozone concentrations.*

PLOTTING IN BASE R

Your turn.

Question 30:

- *Create a scatter plot to explore the relationship between wind speed (Wind) and ozone concentration (Ozone) in the airquality dataset.*
- *Customize the plot with appropriate colors, titles, and labels.*
- *Comment on the observed relationship.*

PLOTTING IN BASE R

Your turn.

Question 31:

- *Create a scatter plot to explore the relationship between wind speed (Wind) and ozone concentration (Ozone) in the airquality dataset.*
- *Customize the plot with appropriate colors, titles, and labels.*
- *Comment on the observed relationship.*

PLOTTING IN BASE R

Your turn.

Question 32:

- *Use the airquality dataset to create a boxplot that compares the distribution of Ozone levels (Ozone) across different months (Month).*
- *Customize the boxplot to include:*
  - *A title: "Distribution of Ozone Levels by Month"*
  - *Axis labels: "Month" (x-axis) and "Ozone*
  - *Concentration (ppb)" (y-axis)*
  - *Different colors for each month's boxplot.*
- *Interpret the boxplot:*
  - *Which month has the highest median Ozone level?*
  - *Are there any outliers in the data? If so, in which months do they occur?*

PLOTTING IN GGPLOT2

Philosophy of ggplot2

- Grammar of graphics:
  - ggplot2 is based on the Grammar of Graphics, a systematic way to describe and build visualisations.
  - It breaks down plots into layers and components, making it highly flexible and consistent.
- Layered approach:
  - Plots are built step-by-step by adding layers (e.g., data, aesthetics, geometries, scales, themes).
  - Each layer can be modified independently, allowing for complex and customisable visualisations.

PLOTTING IN GGPLOT2

Basic Components of ggplot2

- Data: The dataset you want to visualise; Passed as the first argument to ggplot().
- Aesthetics (aes): Maps variables in the data to visual properties (e.g., x-axis, y-axis, color, size, shape).
- Geometries (geom_*): Defines the type of plot (e.g., scatter plot, bar plot, line plot).
- Scales: Control how variables are mapped to aesthetics (e.g., color scales, axis scales).
- Facets: Splits the data into subsets and creates multiple plots (small multiples).
- Themes: Controls the non-data elements of the plot (e.g., background, fonts, grid lines).
- Labels and Annotations: Adds titles, axis labels, and annotations to the plot.

PLOTTING IN GGPLOT2

Basic Components of ggplot2

- Consistency.
- Flexibility.
- Automatic Legends.
- Publication-Quality Plots.
- Faceting.
- Active Community.

PLOTTING IN GGPLOT2

Scatter Plot

- Let’s create a scatter plot of carat vs price, similar to the base R example, but using ggplot2.

```
ggplot(data = diamonds, aes(x = carat, y = price)) +
geom_point(color = ”blue”) +
labs(title = ”Scatter Plot of Carat vs Price”, x = ”Carat”, y = ”Price”)
```

PLOTTING IN GGPLOT2

Your turn.

Question 33:

- *Using the iris dataset, create a scatter plot to analyse the relationship between Sepal.Length and Sepal.Width. Colour the points by Species.*

PLOTTING IN GGPLOT2

Bar Plot

- Let’s recreate the bar plot of the cut variable using ggplot2.

```
ggplot(data = diamonds, aes(x = cut)) +
geom_bar(fill = ”lightblue”) +
labs(title = ”Bar Plot of Diamond Cut”, x = ”Cut”, y = ”Frequency”)
```

PLOTTING IN GGPLOT2

Your turn.

Question 34: *Using the iris dataset, create a bar plot to display the average Petal.Length for each species. Colour the bars by Species.*

PLOTTING IN GGPLOT2

Histogram

- Let’s create a histogram of price using ggplot2.

```
ggplot(data = diamonds, aes(x = price)) +
geom_histogram(binwidth = 1000, fill = ”orange”, color = “black”) +
labs(title = ”Histogram of Diamond Prices”, x = ”Price”, y = ”Count”)
```

PLOTTING IN GGPLOT2

Your turn.

Question 35: *Using the iris dataset, create a histogram to visualise the distribution of Sepal.Length. Use different colours to represent each Species in the dataset.*

PLOTTING IN GGPLOT2

Line plots

- Let’s create a line plot using the data airquality

```
ggplot(data = avg_price, aes(x = carat, y = price)) + geom_line() + labs(title = ”Average
Price by Carat (Faceted by Cut)”, x = ”Carat”, y = ”Average Price (USD)”)
```

PLOTTING IN GGPLOT2

Your turn.

Question 36: *Using the airquality dataset, create a line plot to visualise the trend of Ozone levels over the days of the month. Separate the lines by Month so that each month's trend is clearly visible.*

PLOTTING IN GGPLOT2

Box plots

- Let’s create a line plot of price using ggplot2.

```
ggplot(airquality, aes(x = Month, y = Ozone, color = factor(Month))) +
geom_boxplot() +
labs( x = ”Month”, y = ”Ozone Levels (ppb)”, color = ”Month” )
```

PLOTTING IN GGPLOT2

Your turn.

Question 37: *Using the iris dataset, create a box plot to compare the distribution of Petal.Width across the three Species. Ensure the plot includes:*

- *Different colours for each species.*
- *Proper axis labels and a descriptive title.*

PLOTTING IN GGPLOT2

Customising Plots

- Adding titles, axis labels, and captions;
- Adjusting themes;
- Modifying scales;
- Faceting for subplots;
- Adding annotations;

PLOTTING IN GGPLOT2

Your turn

Question 38:

- *Use the dplyr package to summarise the data:*
  - *Calculate the mean Ozone levels and mean Wind speed for each Month.*
  - *Include the number of observations (n) in each month.*
- *Create a scatter plot using ggplot2 to visualise the relationship between Wind and Ozone:*
  - *Plot Wind on the x-axis and Ozone on the y-axis.*
  - *Use different colours for each Month to distinguish them.*
  - *Add a regression line.*
- *Use facet_wrap to create individual scatter plots for each month to better visualise monthly trends.*

Part IV

Statistical models

1 HYPOTHESIS TESTING

2 T-TESTS

3 ANOVA

4 LINEAR MODELS

HYPOTHESIS TESTING

Tonight, you're going to a party. The weather forecast says there's an 80% chance of rain. Do you take an umbrella?

HYPOTHESIS TESTING

A statistical method used to make decisions or inferences about a population based on sample data.

HYPOTHESIS TESTING

A statistical method used to make decisions or inferences about a population based on sample data.

H_0 : Null hypothesis
H_1 : Alternative hypothesis

HYPOTHESIS TESTING

A person comes into court charged with a crime. A jury must decide whether the person is innocent or guilty. What is the null hypothesis?

HYPOTHESIS TESTING

A person comes into court charged with a crime. A jury must decide whether the person is innocent or guilty. What is the null hypothesis?

H_0 : The person is innocent.
H_1 : The person is guilty.

HYPOTHESIS TESTING

- A company claims that their new energy drink increases productivity. A researcher wants to test whether the drink has a significant effect on productivity.
- A scientist is testing a new medication to reduce blood pressure. They want to determine if the medication is effective in lowering blood pressure compared to a placebo.
- A school wants to know if there is a difference in average test scores between students who attend tutoring sessions and those who do not.
- A factory claims that their light bulbs last an average of 1000 hours. A quality control team tests whether the average lifespan of the bulbs is different from 1000 hours.

HYPOTHESIS TESTING

Type of errors

Since the decision to accept or reject H_0 is based solely on information from a sample of the population, it is possible to commit one of the following errors:

- Rejecting H_0 when H_0 is true (Type I error);
- Failing to reject H_0 when H_0 is false (Type II error).

Decision	**Reality**	
	H_0 true	H_0 false
Do not reject H_0	Correct decision	Type II error (β)
Reject H_0	Type I error (α)	Correct decision

HYPOTHESIS TESTING

Type of errors

$$H_0 \text{ : It will rain tonight.}$$

$$H_A \text{ : It will not rain tonight.}$$

- **Type I Error:** You reject H_0, believe it won’t rain, go without an umbrella, and get wet.
- **Type II Error:** You do not reject H_0, believe it will rain, take an umbrella, and spend the whole night carrying it without needing to use it.

Decision	**Reality**	
	H_0 : It rains	H_0 : It doesn’t rain
Takes umbrella	Correct decision	Type II error (β)
Doesn’t take umbrella	Type I error (α)	Correct decision

HYPOTHESIS TESTING

Basic concepts

- **Hypotheses**: These establish the beliefs (statements) to be tested. They are defined based on the knowledge of the problem and can be either simple or composite.
- **Level of significance (α)**: Associated with the decision rule. It is the probability of committing a Type I Error:

$$\alpha = P(\text{reject } H_0 \mid H_0 \text{ is true}).$$

- **Test statistic**: A statistic that depends on the parameter of interest, but has a known distribution independent of this parameter.
- **Decision rule**: A rule that, based on the data obtained and the significance level α, establishes when H_0 will be rejected.
- **Descriptive level (p-value)**: The probability of obtaining more extreme statistics for the rejection of H_0 than the one provided by the sample.

HYPOTHESIS TESTING

Steps in hypothesis testing

1. State the hypotheses: define H_0 and H_A.
2. Choose a significance level (α): typically 0.05.
3. Select the appropriate test: t-test, ANOVA, etc.
4. Determine the p-value: Compare to α.
5. Make a decision: reject or fail to reject H_0.
6. Draw conclusions: interpret the results in context.

HYPOTHESIS TESTING

Test for the mean of a population

- **Null Hypothesis (H_0)**: The population mean is equal to a specific value, μ_0.

$$H_0 : \mu = \mu_0$$

- **Alternative Hypothesis (H_A)**:
  - The population mean is different from ($\neq$) μ_0:

$$H_A : \mu \neq \mu_0$$

  - The population mean is greater than ($>$) μ_0:

$$H_A : \mu > \mu_0$$

  - The population mean is less than ($<$) μ_0:

$$H_A : \mu < \mu_0$$

HYPOTHESIS TESTING

Test for the mean of a population

In R, we use the function `t.test`:

```
t.test(x, y = NULL,
alternative = c("two.sided", "less", "greater"),
mu = 0, paired = FALSE, var.equal = FALSE,
conf.level = 0.95, ...)
```

HYPOTHESIS TESTING

Your turn.

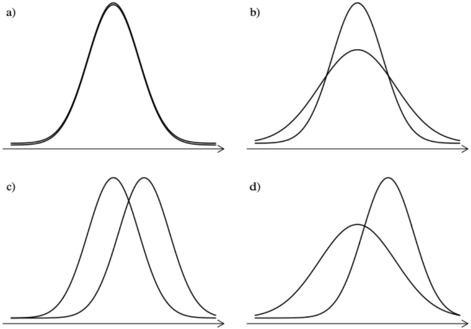
Question 39:

- *airquality data: Is the mean ozone level (Ozone) in the airquality dataset significantly different from 50 ppb*
- *mtcars data: Test if the mean miles per gallon (mpg) of cars with 6 cylinders is less than 20.*
- *Iris data: Test if the mean sepal length of the species setosa is different from 5.0 cm.*

HYPOTHESIS TESTING

Test for comparing means

Given two populations, the objective in this case is to test comparative statements about the parameters of the two populations.



HYPOTHESIS TESTING

Test for comparing means

A t-test for two populations (two-sample t-test) is used to determine whether the means of two independent groups are significantly different from each other.

- **Independent t-test**: Compares the means of two unrelated groups (e.g., Group A vs. Group B, men vs. women).
- **Paired t-test**: Compares the means of two related groups (e.g., before vs. after treatment)

HYPOTHESIS TESTING

Test for comparing means

- Null hypothesis (H_0)

$\mu_1 = \mu_2$ "There is **no difference** between the group means"

- Alternative hypothesis (H_1)

$\mu_1 \neq \mu_2$ "Means are **different** (either greater or less)"

or $\mu_1 > \mu_2$ "(Group 1 **greater** than Group 2)"

or $\mu_1 < \mu_2$ "(Group 1 **less** than Group 2)"

HYPOTHESIS TESTING

Your turn.

Question 40:

- *iris data: Test if the mean sepal length of the species setosa is different from that of versicolor.*
- *mtcars: Test if the mean miles per gallon (mpg) of cars with 4 cylinders is different from that of cars with 6 cylinders.*
- *airquality data: Is there a significant difference in mean temperature (Temp) between May and September?*

Analysis of Variance (ANOVA)

- **ANOVA** stands for **Analysis of Variance**.
- Test the equality of three or more population means.
- The sample data are divided into groups according to a characteristic.
- **Factor (or treatment)**: is a characteristic that allows distinguishing different populations from one another. Each factor contains two or more groups.

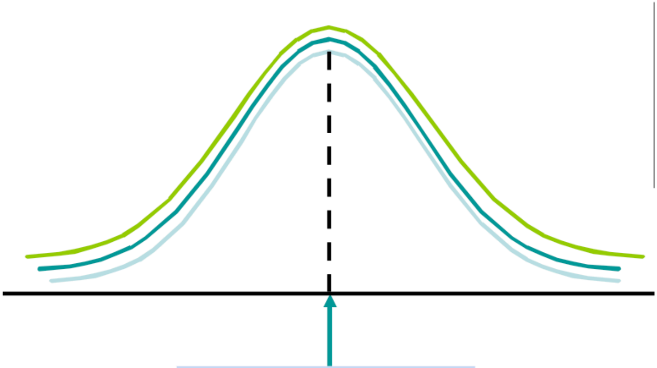
Analysis of Variance (ANOVA)

- We can compare more than two groups at once.
- The null hypothesis (H_0): All group means are equal.
- The alternative hypothesis (H_A): At least one group mean is different.

$$H_0 : \mu_1 = \mu_2 = ... = \mu_t$$

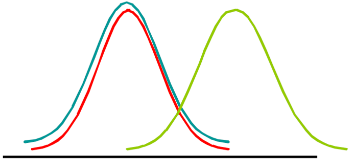
$$H_A : \mu_i \neq \mu_j, \text{ for at least one pair } (i, j)$$

Analysis of Variance (ANOVA)



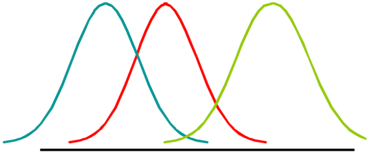
$$\mu_1 = \mu_2 = \mu_3$$

Analysis of Variance (ANOVA)



$$\mu_1 = \mu_2 \neq \mu_3$$

or



$$\mu_1 \neq \mu_2 \neq \mu_3$$

Analysis of Variance (ANOVA)

Assumptions

- Populations are normally distributed.
- Populations have the same variance.
    - To test the homogeneity of variances in an ANOVA model, you can use Bartlett's test or Levene's test.
    - If the hypothesis of equal variances is rejected, another version of the ANOVA can be used: the Welch ANOVA.
    - Welch's ANOVA is typically used for one-way ANOVA when the assumption of equal variances across groups is violated.
- Samples are random and mutually independent.

HYPOTHESIS TESTING

one-way ANOVA

In R, we use the function `oneway.test` for 1-way ANOVA:

```
oneway.test(formula, data, subset, na.action, var.equal = FALSE)
```

or the function `aov`:

```
aov(formula, data = NULL, projections = FALSE, qr = TRUE, contrasts = NULL,
...)
```

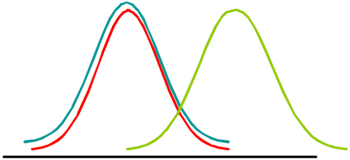
Analysis of Variance (ANOVA)

Your turn.

Question 41: *We want to analyse whether the sepal length (Sepal.Length) differs significantly among the three species.*

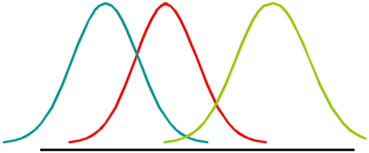
- *Create a box plot using ggplot2 to visualise the distribution of Sepal.Length for each species.*
- *Perform a one-way ANOVA to test if there is a significant difference in Sepal.Length among the three species.*
- *Write down the null and alternative hypotheses before running the test.*
- *Check whether the residuals of the ANOVA model follow a normal distribution using a histogram and a Q-Q plot. Perform a Shapiro-Wilk test as well.*
- *Use Levene's test (from the car package) or Bartlett's test to check if the variance is equal across groups.*
- *Interpret the ANOVA results.*
- *If the assumptions are violated, suggest an alternative analysis.*

Analysis of Variance (ANOVA)



$\mu_1 = \mu_2 \neq \mu_3$

or



$\mu_1 \neq \mu_2 \neq \mu_3$

Analysis of Variance (ANOVA)

Two-way ANOVA

- Two categorical independent variables on a continuous dependent variable.
- It is also used to check the interaction effects between the two factors.
- Factors: independent variables.
- Levels: categories within each factor.
- Interaction effect: ehen the effect of one factor depends on the level of another factor.

Simple linear regression

It mmodels the relationship between two variables:

- One independent (predictor) variable (X).
- One dependent (response/outcome) variable (Y).

The simple linear regression model is:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

The estimated regression line is:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$$

The residual is:

$$e_i = Y_i - \hat{Y}_i$$

Simple linear regression

Assumptions

- Linearity: relationship between X and Y is linear.
- Independence: residuals are uncorrelated (no autocorrelation).
- Homoscedasticity: residuals have constant variance.
- Normality: residuals are normally distributed.

Linear models

Your turn.

Question 42: *Use a linear regression to asnwer the following questions:*

- *Does car weight (wt) affect fuel efficiency (mpg)? (mtcars)*
- *Does petal length (Petal.Length) predict sepal length (Sepal.Length)? (iris)*

Final task

Your turn.

Question 43:

- *mtcars*
- *iris*
- *trees*
- *npk*
- *cars*

Thank you!