



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>)

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.

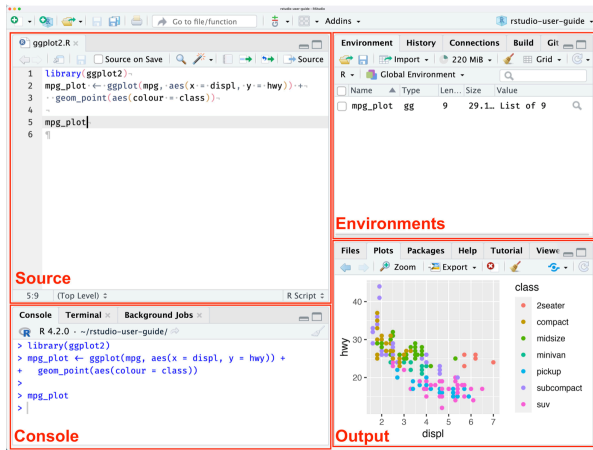


Figure: <https://docs.posit.co/ide/user/ide/guide/ui/ui-panes.html>

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.

```
# 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
<code>sum()</code>	Returns the sum
<code>mean()</code>	Returns the mean/average
<code>sd()</code>	Returns the standard deviation
<code>median()</code>	Returns the median
<code>var()</code>	Returns the variance
<code>cor()</code>	Returns the correlation between two vectors
<code>min()</code>	Returns the minimum
<code>max()</code>	Returns the maximum
<code>range()</code>	Returns the minimum and maximum
<code>summary()</code>	Returns a data summary
<code>quantile()</code>	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*

Thank you!