Programming and Data ANalysis with R

Thiyanga S. Talagala

Table of contents

Preface

The goal of this book is to empower you with essential skills in R programming for statistical computing and data analysis.

Each chapter of this book is designed to be hands-on and practical, with explanations, illustrative examples, and exercises to reinforce your understanding.

This book is written from Quarto.

1 Introduction to R and R Studio

1.1 Chapter Roadmap

1. What is R?
2. Why learn R?
3. R and Rstudio
4. Installing R and Rstudio
5. Familiarize with RStudio interface
6. Creating and saving an RStudio projectt

1.2 What is R?

R is a popular programming language and environment specifically designed for statistical computing, data analysis, and data visualisation. The language designers are **R**oss Ihaka and **R**obert Gentleman at the Department of Statistics, University of Auckland, New Zealand. The parent language is **S**.

1.3 Why learn R?

R is a free and open-source software package.

Here's an expansion on how R can be utilized for tasks beyond traditional statistical analysis:

1. Scientific Writing Tools: R can be used for scientific writing, particularly through the use of packages like knitr rmarkdown, and Quarto. These packages allow you to integrate R code directly into documents alongside text and figures, which is highly useful for reproducible research and automated report generation. These are useful for thesis writing, book writing or any other documentation work. This book "Programming and Data Analysis with R" is written based on Quarto.

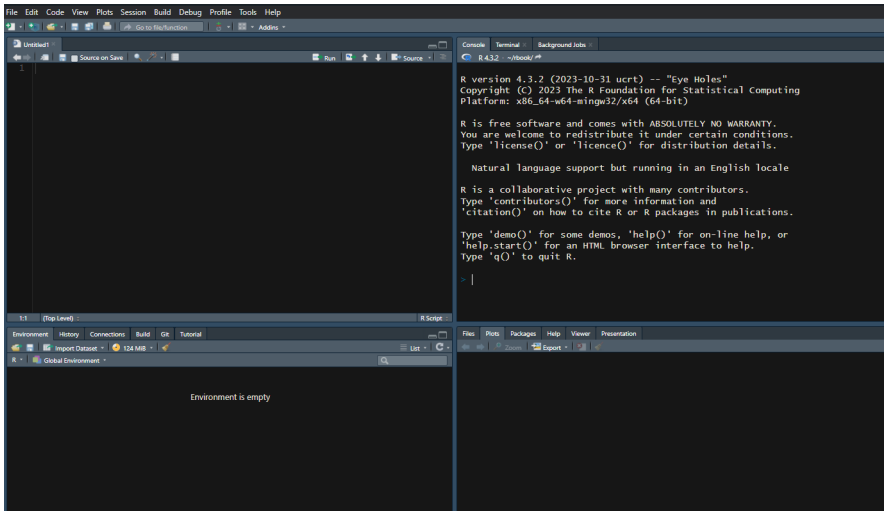
2. Website Development: R can be used developed websites, particularly through the use of packages like knitr rmarkdown, blogdown and Quarto. For example, the website https://hellor.netlify.app/ is written based on blogdown and https://thiyangt.github.io/rprogramming/ is written based on quarto.
3. Creating Presentations: R can generate dynamic and visually appealing presentations using rmarkdown, xaringan, quarto, etc. These packages enable you to embed R code, plots, and interactive elements directly into presentation slides. Here is an example presentation developed using xaringan https://thiyangt.github.io/whyR2021keynote/#1
4. Creating Posters: R can be utilized to design scientific posters using packages such as posterdown. These packages provide templates for creating professional-looking posters directly from R Markdown documents. You can include plots, tables, formatted text and graphics in your poster design.
5. Web Application Development: This capability is particularly useful for developing data-driven tools, simulations, and dashboards that can be accessed through web browsers without the need for users to install additional software.

1.4 Exercise

The goal of this exercise is to help you become familiar with the R Studio environment and create and save projects.

1. Create a new project in the RStudio IDE. Name your project as lesson1.
2. Select a suitable theme for your RStudio IDE's user interface.

   Help: Navigate to Tools > Global Options > Appearance .
3. Change the RStudio pane layout as follows:



4. Create a folder called `data` inside your lesson1 project folder.
5. Create another folder called `src` inside your lesson1 project folder.
6. Open a script file and save it as `exercise1.R` inside the src folder.
7. Type the following commands on `exercise1.R` and run it on the console. See the changes happening under the "Environment" tab and the "History" tab.

```
100 + 200
rnorm(100)
grades <- c("A+", "A-", "A", "B", "F")
random.numbers <- rnorm(100)
random.numbers*100
ls()
```

8. Close the project by saving the workspace.
9. Reopen your project by clicking the leason1.Rproj inside your lesson1 folder. Open the .RData file and the .Rhistory file and observe them.
10. Type the following commands on exercise1.R and run them on the console.

```
marks <- c(100, 70, 80, 60)
```

11. Close the project without saving the workspace.
12. Reopen the lesson1.Rproj and type `ls()` on the console, and observe the output. (`marks` is not listed, but the other objects are available. Why?)

13. Type the following command in the console to observe changes in the console, environment, history, and Viewer windows. Observe the outputs of the code and gain an understanding of the purpose of each line.

```
data("iris")
View(iris)
summary(iris)
hist(iris$Sepal.Length)
plot(x=iris$Sepal.Length, y=iris$Sepal.Width) # Method 1
plot(Sepal.Length ~ Sepal.Width, data=iris) # Method 2
plot(x=iris$Sepal.Length, y=iris$Sepal.Width, col=iris$Species)
plot(Sepal.Length ~ Sepal.Width, data=iris)
plot(Sepal.Length ~ Sepal.Width, pch=16, cex=0.6, data=iris)
plot(Sepal.Length ~ Sepal.Width, pch=16, cex=0.6, data=iris)
plot(Sepal.Length ~ Sepal.Width, col="forestgreen", pch=16, cex=0.6, data=iris)
```

14. Type the following code to obtain list of predefined colours.

```
colours()
```

15. Explore what changes the following code do on the last plot that you took.

code chunk 15.1

```
plot(Sepal.Length ~ Sepal.Width, col="forestgreen", pch=16, cex=0.6, data=iris, main = "Sc
     xlab = "Sepal Length (cm)",
     ylab = "Sepal Width (cm)")
```

code chunk 15.2

```
model <- lm(Sepal.Length ~ Sepal.Width, data=iris)
plot(Sepal.Length ~ Sepal.Width, col="forestgreen", pch=16, cex=0.6, data=iris, main = "Sc
     xlab = "Sepal Length (cm)",
     ylab = "Sepal Width (cm)")
abline(model, col="tomato1")
```

16. Type the following commands and understand what each line of code is doing. Interpret the outputs.

code chunk 16.1

```
plot(iris)
```

code chunk 16.2

```
plot(~ Petal.Length + Petal.Width + Sepal.Width, data=iris)
```

17. Type the following command and open your data folder and see the changes that had occurred.

```
write.csv(iris, file="data/iris.csv")
```

2 Summary

In summary, this book has no content whatsoever.

```
1 + 1
```

```
[1] 2
```

References