Exercise 5

Introduction to R

Estimated time

05:00 Hours

Overview

It is assumed that you have basic programming skills and, you are familiar with high level programming languages (i.e. `C++`, `Java`). The exercise leverages on this familiarity in order to introduce you to `R` programming language.

In this exercise, you write simple `R` programs.

Objectives

After completing this exercise, you should be:

- Install `CRAN` precompiled library distribution on a PC.
- Install `RStudio` IDE on a PC.
- Write and execute `R` scripts.

Prerequisites

Basic understanding of high-level programming OR completed Exercise 1: *Introduction to programming* or Exercise 2: *Object-oriented programming*.

Requirements

This exercise requires a PC that has Internet access and, has a Web browser installed (i.e. `Chrome`).

Exercise instructions

In this exercise, you complete the following tasks:

- Write and compile code excerpts as answers to exercise questions.
- Add a comment with the following information: **Name**, **Student Number**, and **Date**.
- Instructions for uploading your R scripts will be provided by the instructor.

Part 1: R Overview

R is a system for statistical computation and graphics. It consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files. R was initially created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, and is currently developed by the R Development Core Team.

The core of R is an interpreted computer language which allows branching and looping as well as modular programming using functions. The R distribution contains functionality for a large number of statistical procedures. Among these are:

- linear and generalized linear models,
- nonlinear regression models,
- time series analysis,
- classical parametric and non-parametric tests,
- clustering and smoothing.

There is also a large set of functions which provide a flexible graphical environment for creating various kinds of data presentations/visualizations/graphs. Additional modules are available via *"add-on packages from CRAN"*.

R software has many uses, some popular ones include:

- computations on arrays, lists, vectors and matrices using a suite of operators;
- data analysis and generating graphs using a coherent and integrated collection of tools;
- writing statistical functions and machine learning models.

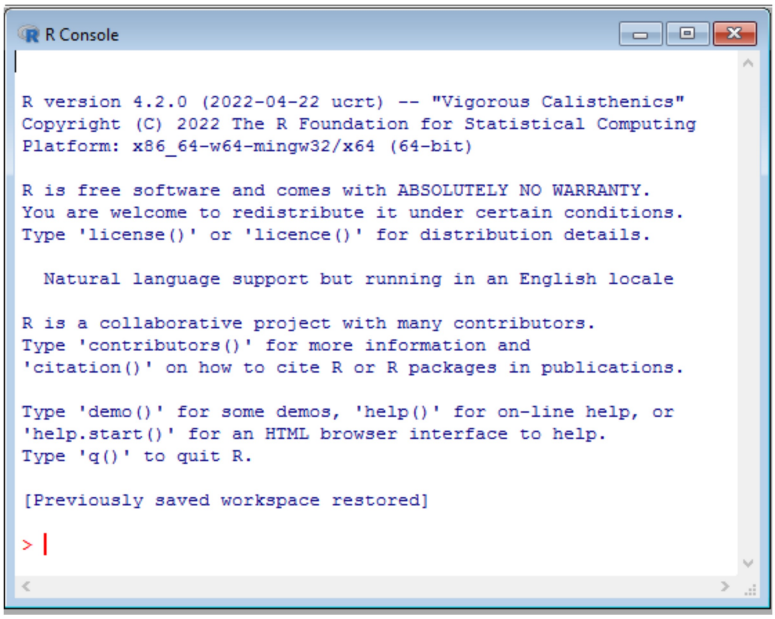
R has a well-developed, simple and effective programming language that is a great resource for data analysis, data visualization, data science and machine learning. It is easy to draw graphs in R, like pie charts, histograms, box plot, scatter plot, etc. R is open-source and it works on different platforms (Windows, Mac, Linux) and it provides many free libraries.

Part 2: Installing CRAN

Before you begin writing R scripts, you need R software on your computer. The *"Comprehensive R Archive Network"* (CRAN) is R's central software repository and it is supported by the R Foundation. It contains an archive (distributed on a collection of sites) consisting of the R distribution(s), the contributed extensions, documentation for R, and binaries.

CRAN includes source packages and pre-compiled binaries for Linux, Windows and MacOS. When you install CRAN, you install a pre-compiled R binary distribution (must be $> v3.3$ to work with RStudio) which comes with a console for using R interactively.

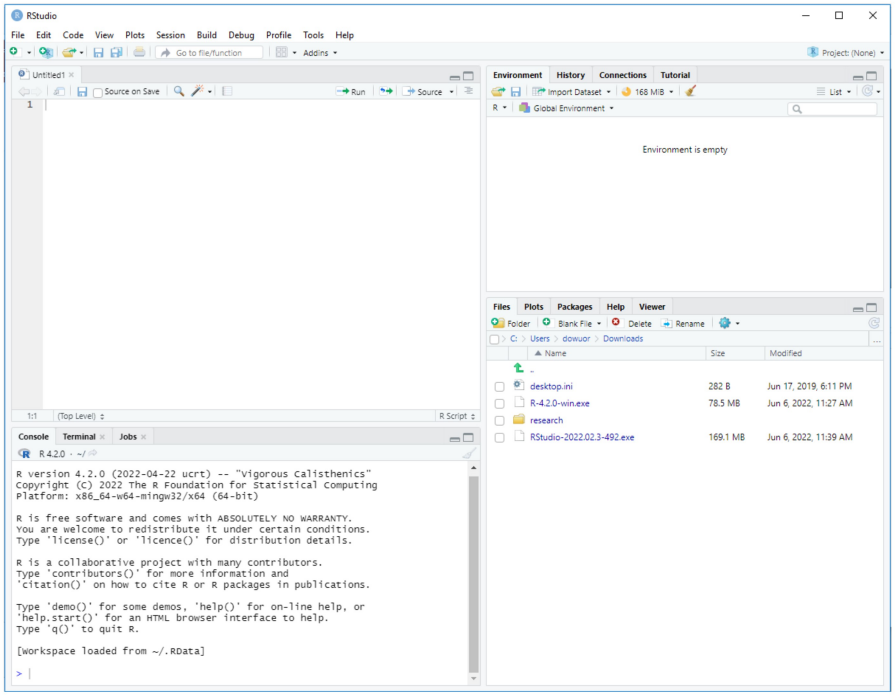
- Installing CRAN via Download, follow this link https://cran.r-project.org/.



Part 3: Installing RStudio IDE

Again before getting started, you may need an IDE or a text editor that has been tailored to make editing, interpreting and executing R scripts. This course recommends RStudio IDE for these tasks since it comes with a set of integrated tools designed to help you be more productive with R and Python. It includes a console, syntax-highlighting editor that supports direct code execution, and a variety of robust tools for plotting, viewing history, debugging and managing your work-space.

- Installing RStudio via Download, follow this link https://www.rstudio.com/products/rstudio/download/.



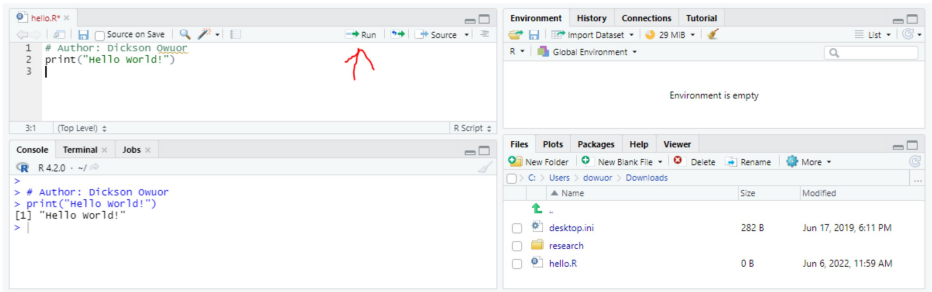
Part 4: Writing your first R script

In this exercise, you write and execute your first R script. Follow the steps that follow:

1. Launch RStudio, create a 'New R Script' and you may save it as *'hi.R'*.
2. Write the code shown below.

```
# Author: Dickson Owuor
print("Hello World!")
```

3. To interpret and execute your code, click on the 'run' button shown by the red arrow in the figure below. The results should be displayed in the Console window.



4. The same code can be written and executed interactively (line by line) using the `R Console` application.

Part 5: R Language syntax

R language has a simple syntax and grammar. Below are the guidelines to adhere to when writing Python code:

- Designed for readability, and has some similarities to the English language with influence from mathematics.
- Case sensitive (i.e., *'Print'* is different from *'print'*)
- When executing an R script in RStudio, execution occurs from the line which is highlighted by the cursor.
- The `help()` function and ? help operator in R provide access to the documentation pages for R functions, data sets, and other objects. (i.e., `help(solve)` or `?solve`).

Part 5.1: Comments

symbol is used to display comments in R. It is used for single-line and multi-line comments.

```
# Author: Dickson Owuor
# Created: 06-06-2022
```

Part 5.2: Variables

Variables are containers for storing data values. There is no precise command for declaring variables. We use the <- sign to assign a value to a variable.

```
# Author: Dickson Owuor
# Created: 06-06-2022

name <- "Jackline"
yob <- 1992
age = 2022 - yob
print(name)
print(paste("Age: ", age))
```

Note: If you try to combine a string (text) and a number, R will return an error.