Introduction to R

P. Adames

11/14/2020

Contents

1 Target audience

This tutorial is aimed at undergraduate and graduate students in the areas of science and engineering.

The goal is to stengthen their computational skills with the most popular statistical computing language and the ecosystem of open source and free packages and tools.

Students are usually expected to learn how to use these specialized tools outside of their normal curriculum, this it becomes important to expose them to them as early as possible in their engineering or scientific careers.

2 Introduction to R

The introduction to R document in the official repository of packages for the R language has 105 pages and 14 sections (https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf). It covers data structures, reading and writing (I/O) to files, probability distributions, statistical models, conditional statements and other forms of controlling the execution of a program, user functions, graphics, extension via packages, and commands to interact with the host operating system.

Undoubably too much for a 45 minute session. Therefore the focus will be on two aspects:

- language specific syntax
- basic I/O and interoperability with the operating system

These two aspects can give the student a quick sense of usability for the R langauage. This approrach leaves an essential strengtgh of R for later, its nature as a statistical computing and graphics language.

3 Language Syntax

This section introduces the basic structural components used to write an R script. Because this is an introductory tutorial, the semantics and the tools will not be emphasized.

At its core, R represents everything as an object. Even functions and operators are special objects. Programing languages model information using data structures. Let's have a look at the objects available in R to model information.

3.1 Data structures

R uses the data structures illustrated in the table below taken from (Wickham 2015, 13). Vectors, matrices, and arrays can store values of the same data type, whether they are are numeric, integer, logical, or character. In contrast, lists and data frames can store mixed data types. For 3d and higher dimensionality the native data structure is an object of class *array*.

	Homogeneous	Heterogeneous
1d	Atomic vector	List
2d	Matrix	Data frame
nd	Array	

Because every data in R is an object, all of these structures have attributes like name, dimension, and a `class`. They can also receive and carry user-defined attributes.

As a consequence of this design, scalars are implemented in R as vectors of length 1. This section contains exercises to illustrate and practice how to create, access, and do simple operations with the basic data structures. Valid identifiers and variable assignment is explained at this point.

3.2 Data types

- *numeric:* double precision floating point real numbers.
- *integer:* model integer numbers.
- *logical:* can be TRUE, T, 1 or FALSE, F, 0.
- *character:* models aa text character in an encoding.

This section has exercises that give examples and questions students on the meaning of operating on mixed data types and the concept of type coercion.

3.3 Built in functions

In this section two basic ideas are presented: mathematical and logical operations are functions. The magic of the generic dispatch mechanism for functions like *print*.

In addition to this the concept of functions as first-class objects is emphasized: functions can be values, thus they can be used as arguments to functions and returned from other functions for assignment to a variable.

The examples illustrate how to search and interpret the R documentation to use functions and their arguments.

3.4 User-defined functions

The main concept presented is how to write a function and what parts does the function object have. A brief introduction to the concepts and concequences of lexical scope, free variable resolution, and environments are

presented via examples.

An important example is that functions can maintain state as a consequence of the use of environments and lexycal scope rules (Ihaka and Gentleman 1996). This is radically different from many other main stream languages and it is a powerful tool for simulation.

3.4.1 Iteration over data structures

The basic apply functions are introduced in this section: apply, lapply, sapply and vapply. The emphasis is made on the safety of vapply because of its explicit declaration of the expected type of the input, thus limiting the effects of type coercion in the case of mixed input types during the apply operation.

3.4.2 I/O

Basic file reading functions, scan, read.table. Students get to practice reading numerical and text data from small files provided to experiment with given command line instructions.

Writing data from data structures to text files.

3.4.3 Interoperatibility with the host operating system

The students will get familiar with the R command line to find the contents of the local directory. The concept of present working directory is also introduced. Other basic commands are introduced to obtain information from the operating system as well as for file and folder manipulation.

3.5 Execution control

Introduction of the basic key words for execution control while writing R scripts.

4 Methodology

This part of the tutorial series in R can be delivered in one or two sessions of 45 minutes each, depending on the convenience and commitment from the students and their instructors. The intended format is brief instructor oral explanations followed by guided hands on student practice on a computer.

The computational environment will be provided as a docker container with RStudio server and all the packages and files required. After Docker is downloaded and installed, the image for this tutorial can be downloaded, built and run on. At the end of the process the students can log in to the server from a browser while the server is running locally on their machines from the container.

Each section on this course has code examples and data for students to run, complete, and modify according to the instructor's interpretation of the students level of competency and engagement. These exercises are preared with the R package learnr.

The students can use the exercises to explore further after the tutorial.

References

Ihaka, Ross, and Robert Gentleman. 1996. "R: A Language for Data Analysis and Graphics." *Journal of Computational and Graphical Statistics* 5 (3). [American Statistical Association, Taylor & Francis, Ltd., Institute of Mathematical Statistics, Interface Foundation of America]: 299–314. http://www.jstor.org/stable/1390807.

Wickham, Hadley. 2015. *Advanced R*. Chapman & Hall/Crc the R Series (Crc Press).