Introduction to Programming in R

From basics to Advanced

Md. Neaz Ali

M.Sc in Statistics

Department of Statistics

Islamic University, Kushtia – 7003, Bangladesh

Email: neazali77@gmail.com

Contents

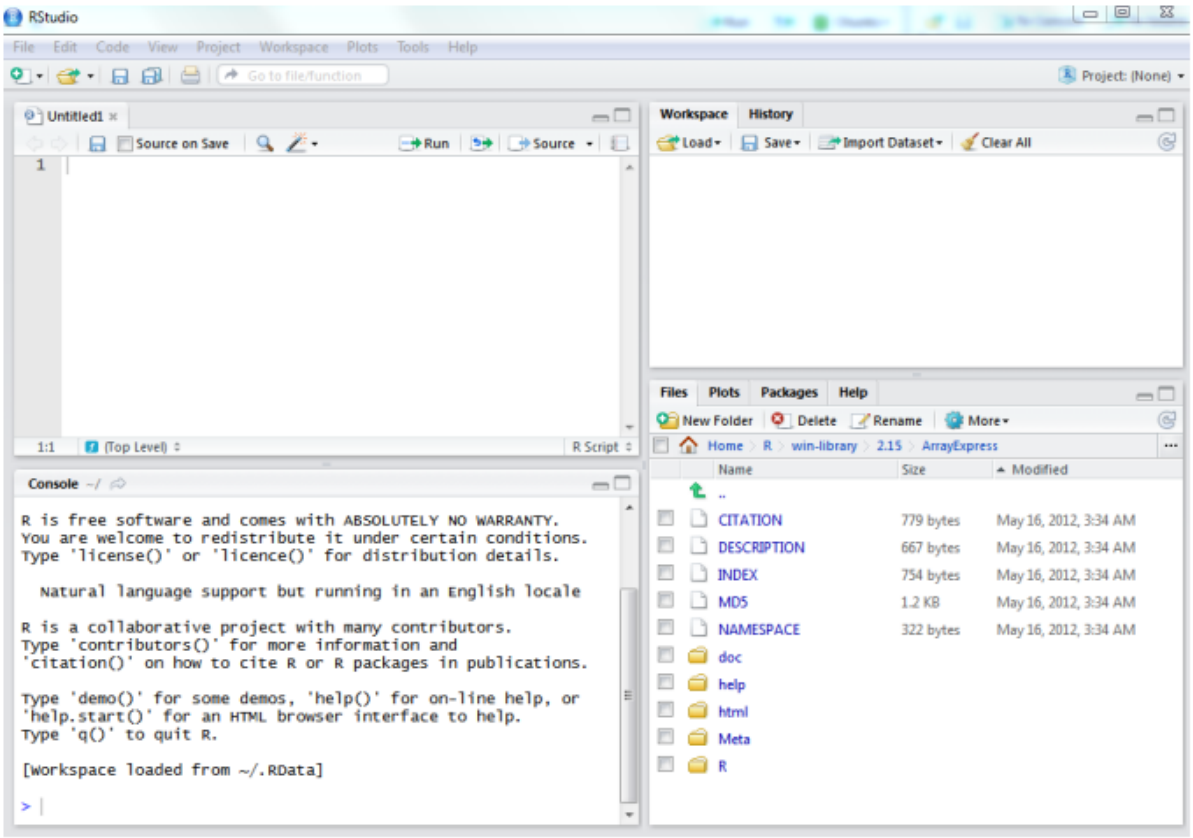
Chapter 1 Introduction to R Software

1.1 Obtaining R and RStudio

R can be downloaded from the website: [Download R-4.4.2 for Windows. The R-project for statistical computing.] and R-Studio can be downloaded from the website: [RStudio Desktop - Posit]

1.2 RStudio Interface

RStudio is a free and an open source integrated development environment for R. On startup R Studio brings up a window with 3 or 4 panels. If you only see 3 panels, click on *File → New → New R Script.*



The bottom left panel "console" is the exact same as the standard R console. RStudio just loads your local version of R. You can specify a different version of R (if you have multiple versions of R running on your machine) by clicking on *Tools → Options* and selecting R version.

1.3.1 Console

RStudio has a nice console features

- Start typing a command, for example fi, press the TAB key, it will suggest function that begin with fi
- Select fis for fishers.test

1.3.2 Editor

The top left panel is an editor which can be used to edit R scripts (.R), plain text (.txt), html web files or any other files. There are several nice features to this text editor which we will describing in the following sections. But for this time being note, that it highlights R code, and that the code is searchable.

1.3.3 Workspace, History

On the top right there is a tab menu workspace and history.

- It lists the object in the current R session. You can load, save or "Clear All" objects for a workspace.
- There is the option to **Import Dataset.**

- The history panel lists all of the command that have been types or input in the console.

1.3.4 File, Plots, Packages, Help

On the bottom right there is a tab menu Files, Plots, Packages, and Help.

- **Files** is the file browser, which allows you to create a new folder, rename a folder or delete a folder.
- The **Plots** window displays plots generated in R. Simply type the following command into the Console window

```
plot(1:10)
plot(rnorm(10), 1:10)
```

- **Packages** lists all of the packages installed in you computer. The packages with a tick marked are those loaded in your current R session. Click on a package name to view help on that package. Note that you can **install packages or check for updates.**

```
instll.packages("packagename")
# load the package
library(packagename)
```

- The **Help** menu provides an extensive R help.

```
?mean()
help(mean)
```

1.4 Starting out – setting a working directory

The first thing to do when starting an R session, is to ensure that you will be able to find your data and also that your output will be saved to a useful location on your computer hard-drive. Therefore, set a **working directory**.

There are numerous ways to set the **working directory**. To change directory:

1. In the classic R interface. Use the file menu, to change directory *File* → *Change dir.*
2. If you start R by clicking on an R icon. You may wish to change the default start location by right mouse clicking on the R icon on the desktop/start menu and changing the **Start In** property.
3. In RStudio Tools → **Set Working Directory.**
4. Use the **File** browser window to view the contents of a directory and navigate to the directory you wish to set as you home directory.
5. The commands to set the working directory

```
# What is my current directory
getwd()
# To Change the directory
setwd("File Path")
```

6. Creating a R Project: Top right corner of the window and following along with your desired name of the project and the directory to create a R project.

1.5 R as a big calculator

Type the following into an R session.

```
2 + 2
## [1] 4
2 * 2
## [1] 4
2 * 100 / 4
## [1] 52
```

1.6 A few important points on R

Elementary commands: *expressions* are evaluated, printed and value lost; *assignments* evaluate expression, passes value to a variable, but not automatically printed

```
2 * 5 ^ 2
```

```
## [1] 50
x = 2 * 5 ^ 2
print(x)
## [1] 50
```

1.7 Operators in R

Operators are the symbols directing the compiler to perform various kinds of operators between the operands e.g.

```
2 + 4
```

Here 2 and 4 are operands and (+) is the operator which is performing addition operation on the both operand 2 and 4.

R support majorly four kinds of binary operators between a set of operands. Namely they are

- Arithmetic Operators
- Logical Operators
- Relational Operators
- Assignment Operators
- Miscellaneous Operators

1.7.1 Arithmetic Operators

Arithmetic operators modulo using the specified operator between operands, which may be either scalar value, complex numbers, or vectors. The most common arithmetic operators are

- Addition (+): 5+4
- Subtraction (-): 4-2
- Multiplication (*): 7*2
- Division (/): 10/2
- Power (^): 2^5
- Modulo (%%): 24 %% 5

1.7.2 Logical Operators

Logical operators in R simulate element-wise decision operations, based on the specified operator between the operands, which are then evaluated either a TRUE or FALSE value.

- Element-wise Logical AND operator (&): TRUE & TRUE = TRUE (if both side are true then result is true).
- Element-wise Logical OR operator (|): TRUE | FALSE = TRUE (if at least one side is true then result is true).
- NOT operator (!): A unary operator that negates the status of the elements of the operand. ! TRUE = FALSE.
- Logical AND operator (&&): Returns TRUE if both the first element of the operands is True.
- Logical OR operator (||): Return TRUE if either of the first elements of the operands is True.

1.7.3 Relational Operators

The Relational Operators in R carry out comparison operations between the corresponding elements of the operands. Returns a Boolean TRUE value if the first operand statisfies the relation compared to the second.

- Less than (<): 5 < 10 (TRUE)
- Greater than (>): 5 > 10 (FALSE)
- Less than equal to (<=): 5 <= 10 (TRUE)
- Greater than equal to (>=): 5 >= 10 (FALSE)
- Not equal to (!=): 5!=10 (TRUE)
- Equal to (==): 5 == 10 (FALSE)

1.7.4 Assignment Operators

Assignment operators in R are used to assigning values to various data objects in R. the objects may be integers, vectors, functions, or dataframes etc. There are two kinds of assignment operators: Left and Right

- Left assignment (<-): vec1 <- c("ab", TRUE)
- Right assignment (->): c("ab", TRUE) -> vec2
- Or Simply equal to (=): vec1 = c("ab", TRUE)

N.B: Operations in programming flows from right to left and up to bottom manner.

1.7.5 Miscellaneous Operators

Miscellaneous operators are the mixed operators in R that simulate the printing of sequences and assignment of vectors, either left or right – handed.

- **%in% Operator**: Checks if an element belongs to a list and returns a Boolean value TRUE if value is present else FALSE e.g.,

```
val = 0.1
list1 = c(TRUE, 0.1, "apple")
print(val %in% list1)
```

- **%*% Operator**: This operator is used to multiply a matrix with its transpose. Transpose of the matrix is obtained by interchanging the rows to columns and columns to rows.

```
mat = matrix(c(1,2,3,4,5,6),nrow=2,ncol=3)
      print (mat)
      print( t(mat))
      pro = mat %*% t(mat)
      print(pro)
```

1.8 Variables in R programming

A variable is a memory allocated for the storage of specific data and the name associated with the variable is used to work around this reserved block. The name given to a variable is known as its variable name. Usually a single variable stores only the data belonging to a certain data type.

```
var1 = "hello"
print(var1)
```

Rules for Variable Naming:

- A valid variable name consists of a combination of alphabets, numbers, dot(.), and underscore(_) characters. Example: var.1_ is valid
- Apart from the dot and underscore operators, no other special character is allowed. Example: var$1 or var#1 both are invalid
- Variables can start with alphabets or dot characters. Example: .var or var is valid
- The variable should not start with numbers or underscore. Example: 2var or _var is invalid.
- If a variable starts with a dot the next thing after the dot cannot be a number. Example: .3var is invalid
- The variable name should not be a reserved keyword in R. Example: TRUE, FALSE, etc.

N.B.: Always use relative names for variables for better understanding the work of the variable.

1.9 Data Types in R Programming

R Data types are used to specify the kind of data that can be stored in a variable. For effective memory consumption and precise computation, the right data type must be selected. Each R data type has its own set of regulations and restrictions. Variables are not needed to be declare with a data type in R, data type even can be changed. Data types in R are:

- Numeric – (3, 6.7, .121)
- Integer – (2L, 42L; where 'L' declares this as an integer)
- Logical – (TRUE)

- Complex – (7+5i; where ‘I’ is imaginary number)
- Character – (“a”, “B”)
- Raw – (as.raw(55); raw creates a raw vector of the specified length)
- String – (“Hello World”)

Basic Data Types	Values	Examples
Numeric	Set of all real numbers	`"numeric_value <- 3.14"`
Integer	Set of all integers, Z	`"integer_value <- 42L"`
Logical	TRUE and FALSE	`"logical_value <- TRUE"`
Complex	Set of complex numbers	`"complex_value <- 1 + 2i"`
Character	“a”, “b”, “c”, ..., “@”, “#”, “$”, ..., “1”, “2”, ...etc	`"character_value <- "Hello Geeks"`
raw	as.raw()	`"single_raw <- as.raw(255)"`

1.10 Useful Functions in R e.g. rm(), ls()

ls(); List all the objects currently defined in your R workspace.
- ls(pattern = “my_var”)

rm(); Removes the specified objects.
- rm(“var1”)
- rm(list = ls()

1.11 R Session

For R session you may visit this site: R tutorials, R session, enter, leave, quit