Introduction to R & Data Management

Christopher Maronga

May 18, 2021

What is R

- R is both a scripted programming language and software environment for statistical computing.
- R is widely used for statistical analysis and creation of high-quality publishable graphics.
- It's open source, free to use and supported by a large community of users and developers.
- Flexible and highly scalable via user defined packages and functions.

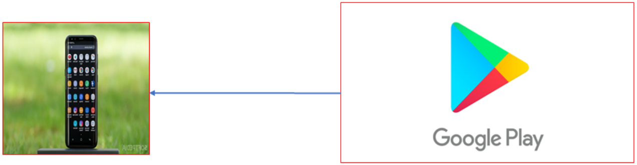
Why R

- Reproducibility - one code; multiple users and projects .
- Flexible and easy to learn.
- Scalability - You can define/customize your own functions and/or packages.
- Interoperability - same code across platforms (mac, windows, linux etc).

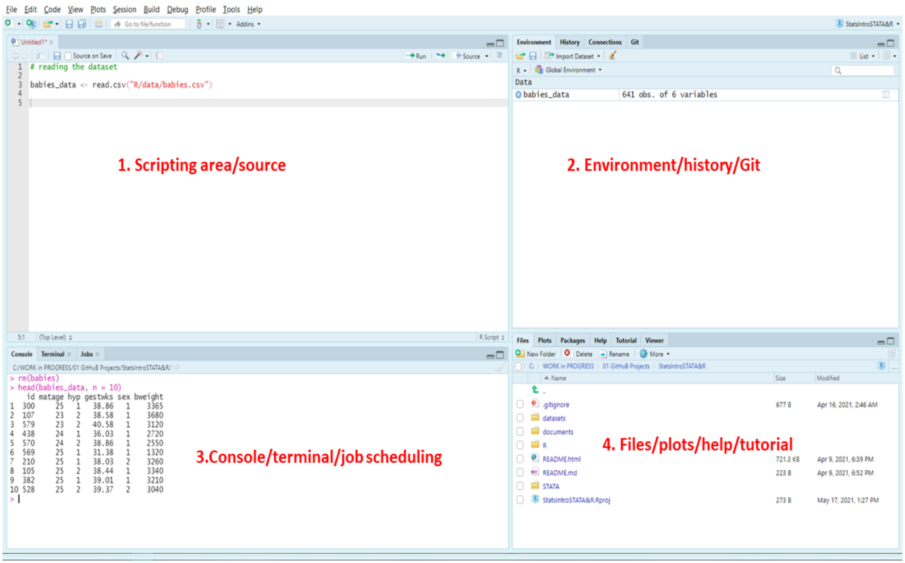
Getting started - Installing R and Rstudio

- You can install the latest version of R from the CRAN repository. CRAN stands for **C**omprehensive **R** **A**rchive **N**etwork
- The latest version or R is 4.0.5 as of time of the slide creation. This might change with time
- Rstudio is an IDE (Intergrated Development Environment for R). You can install the latest version of Rstudio here
    - **NOTE:** You do not need Rstudio to use R, BUT it makes things easier. So I will highly recommend you to install Rstudio for easy and efficient coding in R.

Introduction to R and Rstudio



Components of Rstudio


File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
StatsIntroSTATA&R
Untitled1*
Source on Save
Run
Source
1. Scripting area/source
Environment History Connections Git
Import Dataset
List
Global Environment
2. Environment/history/Git
5:1
(Top Level)
R Script
Console Terminal Jobs
3.Console/terminal/job scheduling
Files Plots Packages Help Tutorial Viewer
New Folder
Delete
Rename
More
4. Files/plots/help/tutorial

```r
# reading the dataset

babies_data <- read.csv("R/data/babies.csv")
```

Data

babies_data 641 obs. of 6 variables

C:/WORK in PROGRESS/01 GitHuB Projects/StatsIntroSTATA&R/

```
> rm(babies)
> head(babies_data, n = 10)
    id matage hyp gestwks sex bweight
1  300     25   1   38.86   1    3365
2  107     23   2   38.58   1    3680
3  579     23   2   40.58   1    3120
4  438     24   1   36.03   1    2720
5  570     24   2   38.86   1    2550
6  569     25   1   31.38   1    1320
7  210     25   1   38.03   2    3260
8  105     25   2   38.44   1    3340
9  382     25   1   39.01   1    3210
10 528     25   2   39.37   2    3040
> 
```

C: > WORK in PROGRESS > 01 GitHuB Projects > StatsIntroSTATA&R

Name	Size	Modified
..		
.gitignore	677 B	Apr 16, 2021, 2:46 AM
datasets		
documents		
R		
README.html	721.3 KB	Apr 9, 2021, 6:39 PM
README.md	223 B	Apr 9, 2021, 6:52 PM
STATA		
StatsIntroSTATA&R.Rproj	273 B	May 17, 2021, 1:27 PM

How R works (expression vs assignment)

- R syntax is easy to use. Type commands after the prompt > and hit enter to execute in console.
- Simple illustration, using R as an overgrown calculator to evaluate mathematical computation.
- Broad classification of R syntax:-
  - Expression- Commands are evaluated, printed on console and the value is lost (requires re-typing).
  - Assignments- Storing intermediate results of an expression to an object which can be re-used (<-, =, assign())
- Objects are created in memory and saved into a file called .RData.
- R commands are stored in a file names .Rhistory.
- R is case sensitive name and Name refer to different variables

Interactive R session: 01 - How R works

We switch to R and write code to demonstrate the ideas.

R data structures(1/4) - Introduction

- We assign results of expressions into objects with unique names in R.
- Objects can be of different forms. Most common types include:-
  - Vectors (building block of R language)
    - Atomic vectors - Homogeneous, contains items of ONLY one data type
    - Lists - Heterogeneous, contains objects of any type
  - Matrices
  - Data frames
  - Arrays
  - Complex numbers*
  - Raw*.
- These ecosystem of different forms of objects is referred to as data structures.

R data structures(2/4) - vectors

- Vectors are the building blocks in R.
- Atomic vectors are the simplest type of objects and are classified into:-
  - Logical
  - Integer
  - Double
  - Character
- We use the function c() meaning combine items to create vectors in R
- Generic functions ls() and rm() are used to list and remove objects in the workspace.

Interactive R session: 02 - Creating vectors in R

We switch to R and write code to demonstrate the ideas.

R data structures(3/4) - Lists

- A list is a generic vector containing objects/elements of different data types (heterogeneous).
- We construct lists using the function list(). Lists are also referred to as recursive vectors because it can contain other lists.

```r
my_list <- list(
  c(3,4,5,8),
  c("Mon","Tu","wed","Thur","Fri","Sat"),
  matrix(1:10, byrow = T, nrow = 5)
)

str(my_list)
```

Interactive R session: 03 - Getting to know lists

We switch to R and write code to demonstrate the ideas.

R data structures(4/4) - data frames

- Data frames generally refer to tabular data (rows and coloumns). A data frame is the most common way of storing data in R.
- This means that a data frame has both coloumn and row names. Basically a data frame is a list of vectors of equal length
- Data frames are created using the function data.frame() which takes named vectors as arguments.

```
my_dataframe <- data.frame(
  col_1 = c("values_to_go_to_this_coloumn1"),
  col_2 = c("values_to_go_to_this_coloumn2")
  col_n = c("values_to_go_to_this_coloumnn")
)
```

- Let's practice on data frames using other functions such as head(), tail() and str().

Interactive R session: 04 - Data frames

We switch to R and write code to demonstrate the ideas.

Exercise one

1. Create a vector whose elements are names of the first 6 months of the year and name it `monthVec`
2. Create a vector named `distances` whose elements consist of 10, 25.4, 16.8, 23, 40 and 17.1
3. Create an integer of numbers between 5 and 12. Name it `intVec`
4. Create a logical vector with TRUE, FALSE, TRUE, FALSE and FALSE as it's elements. Name it `logVec`
5. Combine 1 and 4 into a new vector named `char_log`. Notice anything?
6. Combine 2 and 3 into a new vector named `int_duble`. Notice anything?
7. Create a list containing objects 1 to 6 above and name it `ListOne`

Subsetting/Indexing elements of an object

- More often you will want to extract elements or components of a object.
- We explore how to subset `data.frames`, `lists` and `vectors`
- Combining integer vector indexing (`negative` *omits* while `positive` *retains*) with `[]` and `[[]]`
- Primarily use dollar sign (`$`) to subset dataframes and `attach/detach` to index variables directly.

Note: For **Integer vector indexing**, unlike other programming languages (`python, Java and C++`) that starts indexing from **0**, indexing of objects in R starts from **1**

Interactive R session: 05 - Subsetting objects

We switch to R and write code to demonstrate the ideas.

External R packages and getting help

- An R package is a collection of R functions and/or data with a well compiled code(s) that enable R users perform even more tasks with R(expand functionality).
- Examples of R packages include `dplyr`, `ggplot2` etc. There are currently over 14,500 contributed packages in the Comprehensive R Archive Network(known as CRAN).

Common commands

- `install.packages("package name")` # to install a package
- `library(package name)` # load an R package into the workspace for use, also require(package name)
- To get help on a particular function, use `?function_name` or `help("function_name")` and you will be presented with a documentation on arguments and use case of an R function.

Interactive R session: 06 - R packages

We switch to R and write code to demonstrate the ideas.

The concept of working directory and file paths

1. The working directory is the **specifc folder** within your computer where your live R session is hosted/pointing to. This folders by default holds all your .RData files and .Rhistory files.
   - `getwd()` to get to know where your working directory is located, get help `?getwd`
   - `setwd()` to set a new location for your working directory, get hwlp `?setwd`
2. The **file path** is the specific folder address that contains your files of interest. A file path can be a folder within the working directory or outside the working directory

NOTE: The above concepts are very key when reading data into R, therefore a clear understanding of how they work will come in handy in importing data into R suing any method defined.

Interactive R session: 07 - Working directories

We switch to R and write code to demonstrate the ideas.

Reading data into R

- Data can be read into R from various sources.
- **General way** using the command **`read.table()`**.

You can read a variety of file extensions into R using the function `read.table()`. Some examples of applicable files to read include csv, tab delimited(.txt), space demilited and any flat file so long as it's defined interms of what character separates its columns.

NOTE: Knowing the extension of the file you are reading and how its coloumns are separated is essential in using `read.table()` command.

- **More specif way** using the command **`read.csv()`**.

-used to read *comma separated files ONLY* otherwise reffered to as `.csv`. More convinient for csv files in that it pre-populates arguments for you which you would have specified in `read.table()` such as `header=T` or even `sep=","`.

Interactive R session: 08 - Getting data into R

We switch to R and write code to demonstrate the ideas.

Exercise Two

1. Read the dataset `birth_weight.csv` into R and name it `birth_weight_data`.
    - Print the count of number of rows and columns in the dataset
    - Print the first 10 observations and last 10 observations in the dataset
    - What is the mean of the variable `matage`
    - Use `summary` to print general summary of the data. How many observations are female?
2. Read the dataset `bank_churners.csv` into R and name it `churners_data`.
    - How many rows does `churners_data` contain?
    - Use `str` function to print details of the structure of churners_data.

Introduction to `dplyr` for data management in R

- We are going to focus on the **dplyr** package which is a core member of the group of packages called `tidyverse`.
- `dplyr` package makes data transformation and manipulation easy by providing simple verbs that correspond to common data manipulation tasks which helps you translate your **thoughts** into code.

The main verbs for data manipulation provided by `dplyr` include:-

- `select()` and `rename()` to select variables based on their names/rename variables.
- `mutate()` and `transmute()` to add/create new variables.
- `filter()` to select cases/rows based on their values.
- `arrange()` to reorder the cases/rows.
- `summarise()` to condense multiple values to a single value. Can be used together with `group_by()`
- `sample_n()` and `sample_frac()` to take random samples.

Interactive R session: 09 - Data management basics

We switch to R and write code to demonstrate the ideas.

Reference material

- An introduction to R hand-out
- R for beginners
- The book R for Data Science
- Introduction to R slides by Dr. Norma Coffey
- The book R in Action for data analysis and graphics
- Quick R help.
- Extensive online help twitter, Github, StackOverflow