Introduction to R - Part I

Matt Arthur

GradQuant
University of California Riverside

29 Sep 2022

Outline

Introduction

S

R is an open source software used for statistical computing and graphics.

- Easy to download, with no license restrictions.
- Provides support for powerful computing tasks, such as machine learning.
- Intuitive and flexible graphics support.
- Works well with other programs; i.e, SQL and Excel

Outline

Installing R

Go to `https://www.r-project.org`

- Click the "Download" link on the left.
- Select a mirror from which to download.
- Follow the download instructions for your operating system.

Installing R Studio

R Studio is a graphical user interface that makes R much easier to use.

To download R Studio, go to `https://www.rstudio.com`.

- R must be installed for R Studio to run.
- Go to Products > R Studio
- Follow the instructions for downloading the free version of R Studio Desktop.

Outline

Using R as a Calculator

One of the simplest ways to use R is as a calculator:

```
2 + 2
```

```
## [1] 4
```

```
2^3
```

```
## [1] 8
```

```
2 + (3^2 - sin(46))
```

```
## [1] 10.09821
```

```
exp(1.3)
```

$e^{1.3}$

```
## [1] 3.669297
```

R uses standard order of operations and has many predefined functions which make it easy to use in this way.

Testing for Equality

We can also use R to conduct logical tests:

```
2 == 3
```

```
## [1] FALSE
```

```
2 < 3
```

```
## [1] TRUE
```

As in other languages, we use "==" instead of "=" to test for equality.

Outline

Data Structures in R

R can store and use objects that we create in **variables**

```
x <- 2 * 3
print(x)
```

```
## [1] 6
```

R can also store sequences of values, called **vectors**

```
y <- c(2, 4, 6, 8)
print(y)
```

```
## [1] 2 4 6 8
```

```
z <- 2:4
print(z)
```

```
## [1] 2 3 4
```

Note: The use of <- as an assignment operator in R.

We can access all or part of vectors defined in R by using square brackets: []

```
y[1]
```

```
## [1] 2
```

```
y[2:4]
```

→ y[c(2,3,4)] y[2,3,4]

```
## [1] 4 6 8
```

Vector Operations

Most mathematical operations that can be applied to scalars can also be applied to vectors in R:

- Operators to vectors are applied elementwise:

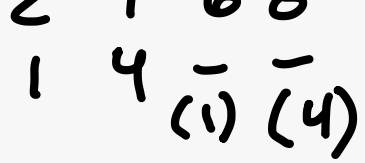
```
x <- c(1, 2, 3, 4)
y <- c(2, 4, 6, 8)
print(x + y)
```

```
## [1]  3  6  9 12
```

- When a shorter vector is combined with a longer vector, elements in the shorter vector are **recycled**:

```
z <- c(1, 4)
print(y + z)
```

```
## [1]  3  8  7 12
```



Vector Operations

- Single numbers can be thought of as vectors of length-1

```
y <- c(2, 4, 6, 8)
y * 2
```

```
## [1]  4  8 12 16
```

Matrices

A matrix is essentially a 2D array in R. Any vector can be put into matrix form using the `matrix()` function:

- Matrices can be ordered in different directions:

```
m1 <- matrix(c(1:9), nrow = 3, ncol = 3)
print(m1)
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
```

```
m2 <- matrix(c(1:9), nrow = 3, ncol = 3, byrow = T)
print(m2)
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

Matrices

m1[1,4]

- We can access matrix components using again using square brackets:

```
 m1[1,3] # element in row 1, column 3
## [1] 7
 m2[,3] # all elements in column 3
## [1] 3 6 9
 m1[1,] # all elements in row 1
## [1] 1 4 7
```

Note: Vectors and matrices are indexed starting at 1.

Matrix Operations

- Elementwise multiplication:

```
 m1 * m2
##      [,1] [,2] [,3]
## [1,]    1    8   21
## [2,]    8   25   48
## [3,]   21   48   81
```

- Matrix Multiplication

```
 m1 %*% m2
##      [,1] [,2] [,3]
## [1,]   66   78   90
## [2,]   78   93  108
## [3,]   90  108  126
```

Matrix Operations

- Transpose

```
 t(m1)
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

- Diagonal Matrices

```
 diag(c(1, 2, 3))
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    2    0
## [3,]    0    0    3
```

Matrix Operations

- Inverse of a square Matrix:

```
 A <- matrix(c(1, 3, 5, 7), nrow = 2)
 solve(A)
```

```
##         [,1]   [,2]
## [1,] -0.875  0.625
## [2,]  0.375 -0.125
```

- Row and Column Means

```
 rowMeans(A)
```

```
## [1] 3 5
```

```
 colMeans(A)
```

```
## [1] 2 6
```

Outline

The Working Directory

A working directory is the default location where R looks for files.

- To view your current working directory, use the `getwd()` function:

```
getwd()
```

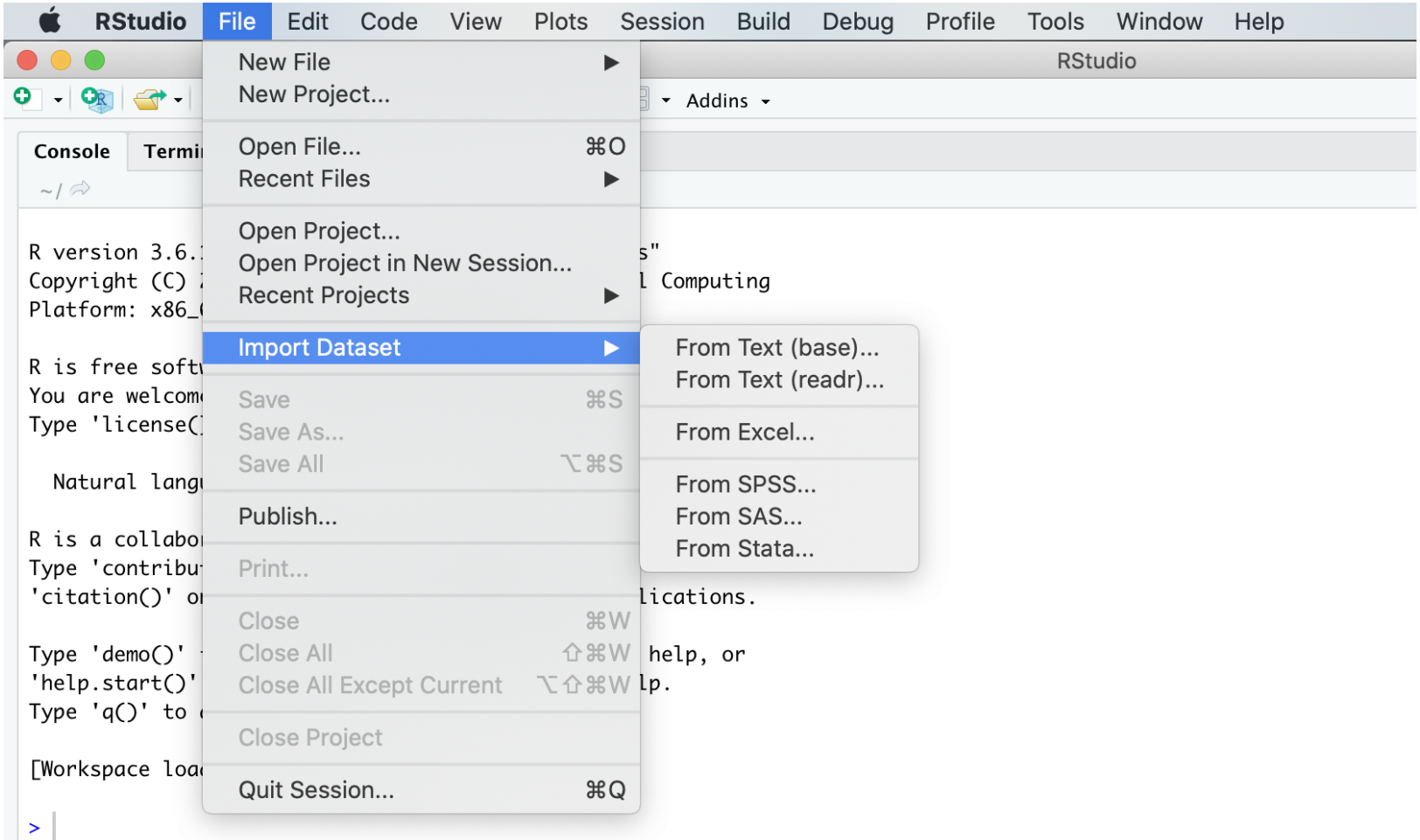
- To change your working directory, use `setwd()`:

```
setwd("desired_path")
```

("~/Downloads")

Importing Data: Using the File Tab

R Studio has a File tab that can be used to import data files:



Importing Data: Using R Directly

Alternatively, you can read data by using commands directly from R:

- From a Comma-Delimited File:

```
new_data <- read.table('your_file_path', sep = ',', header = T)
```

or,

Flat/delimited

```
new_data <- read.csv('your_file_path', header = T)
```

- From Excel:

```
library(readxl)
new_data <- read_excel('your_file_path')
```

- From SPSS:

```
library(foreign)
new_data <- read.spss('your_file_path', to.data.frame = T)
```

Package Management

Many data-import utilities reside in auxiliary libraries which are not loaded in base R. To install these libraries, use the `install.packages()` function:

```
install.packages("readxl")
```

R may need to be restarted prior to loading the packages you installed.

Outline

Concrete Example

We will do several hands-on exercises with the "Concrete Compressive Strength" dataset from the UC Irvine Machine Learning Data Repository[1]. You may access the data using the following link:

- `https://archive.ics.uci.edu/ml/datasets/Concrete+Compressive+Strength`

Concrete Example

To download the data:

- Navigate to the link on the previous slide
- Click on the "Data Folder" link at the top of the page
- Download `Concrete_Data.xls`

Concrete Example

To load the data into R: First determine the directory to which you saved the Concrete Data.

- Usually will be in your "Downloads" folder
- "~/Downloads" on Unix
- Something like "C:/Users/your-user-name/Downloads" on Windows

```
library(readxl)
concrete <- read_excel("~/Downloads/Concrete_Data.xls")
View(concrete)
```

Exercises for Next Time

Next week, we'll continue working with the Concrete Dataset in R:

- Basic viewing/summarizing of the data
- Running a linear regression in R to predict concrete compressive strength
- Managing tabular data

Related Workshops

If you are interested in learning more about R, please consider the following workshop, which is offered later this quarter:

- Introduction to R: Part II: Thursday, 10/6/2022, 1:00PM - 1:50PM
- Linear Models in R: Monday 10/3 & 10/10: 2:00PM - 2:50PM
- Data Visualization in R: Tuesday 10/11 & 10/18: 1:00PM - 1:50PM
- Remember to register!

Services in GradQuant

- GradQuant offers individual consultations via Skype. You are always welcome to make an appontment with us.
- Weekly Drop-in hours are held Wednesdays 10AM–12PM. Graduate students and postdocs can meet in GradQuant consultants without an appointment on a first-come, first-served basis.
- Weekly Hacky Hours are held on Mondays 1PM–3PM. Hacky Hours are open to the whole campus, serving undergraduate and graduate students, faculty, and staff. No appointment is needed.
- For detailed information about how to make an appointment, visit our website: `https://gradquant.ucr.com`

Credits

This presentation was adapted from a previous workshop delivered by Lead Consultant Ruihan Lu in 2020.

The Concrete Data were owned and donated to the UCI Machine Learning Repository by I-Cheng Yeh:


I-Cheng Yeh.
Modeling of strength of high performance concrete using artificial neural networks.
Cement and Conrete Research, 28(12):1797–1808, 1998.

Questions?