${\cal R}$ Learning Route and Resources

Vian Lee 💙 Iydon Liang

September 25, 2019

.CONTENTS

Ι	Overview of R	4
1	Contents	5
2	R or Python	6
3	Learning Resources3.1 Comprehensive Resources3.2 Basis	7 7 7 7
II	Preparation	8
4	Contents	9
5	Download and Install the Latest R	10
6	Download and Install the Latest RStudio	11
7	How to Use RStudio	12
8	How to Fix the Problems	13
II	I Getting Started with R	14
9	Contents	15
10	Cheat Sheets from RStudio 10.1 RStudio 10.2 Basis 10.3 Extra Packages	16 16 16

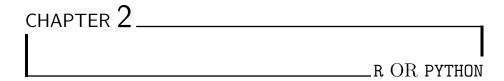
CONTENTS	3

IV	Appendix	18	
\mathbf{A}	Contents	19	
В	Getting Help B.1 Accessing the Help Files	20 20 20	
\mathbf{C}	Using Packages	21	
D	Working Directory	22	
Ε	Vectors E.1 Creating Vectors E.2 Vector Functions E.3 Selecting Vector Elements E.3.1 By Position E.3.2 By Value E.3.3 Named Vectors	23 23 24 24 24 24 24	
F	Programming F.1 For Loop F.2 While Loop F.3 If Statements F.4 Functions F.5 Reading and Writing Data	25 25 25 25 26 26	
\mathbf{G}	Matrices	27	
Н	Lists	28	
Ι	Data Frames 29		
J	Strings	30	
K	Factors 31		
${f L}$	Statistics 32		
\mathbf{M}	Distributions	33	

Part I $egin{array}{c} egin{array}{c} \egin{array}{c} \egin{array}{c} \egin{array}{c} \egin{array}{c} \egin{array}{c} \egin$

CHAPTER 1	
<u> </u>	
	CONTENTS

- R or Python
- Learning Resources
 - Comprehensive Resources
 - Basis
 - RStudio
 - Data Science



Reference from zhihu and CSDN.

- 1. R focuses on better user friendly data analysis, statistics and graphical models, while Python emphasizes productivity and code readability.
- 2. R is uncomplicated to apply complex formulas for all kinds of statistical tests and models are readily available and easily used, while Python is flexible for doing something novel like building websites.
- 3. R has a steep learning curve at start, you can easily learn advanced stuff once understand the basics. While Python pays more attention to readability and simplicity, which makes its learning curve relatively low and gradual.
- R and Python are comparable in terms of packages, the former has comprehensive archive network called CRAN, while the latter has package index called PyPi.

The closer you are to statistics, research and data science, the more you might prefer R; The closer you are to working in an engineering environment, the more you might prefer Python. Therefore, if you have enough time, you can learn both R and Python, but for different focuses. That is, use R to conduct statistical tests, graph data and inspect large data, use Python to write algorithm and deploy services. Moreover, we can do lots of interesting projects efficiently by integrating Python and R with rpython and rpy2,

After all, the set of programming languages is perfect, which means it has no isolated points.



3.1 Comprehensive Resources

1. Archived webinars, videos and learning roadmap from R Studio and its Git
Hub. $\,$

3.2 Basis

- 1. $Advanced\ R$ and its GitHub.
- 2. Cheat sheets.
- 3. Books recommended by zhihu.

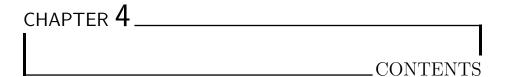
3.3 RStudio

- 1. RStudio documentation and its GitHub.
- 2. R Markdown.
- 3. RStudio Shiny.
- 4. Frequently Asked Questions.

3.4 Data Science

1. R for Data Science.

Part II Preparation



- $\bullet\,$ Download and Install the Latest R
- Download and Install the Latest RStudio
- How to Use RStudio
- How to Fix the Problems



The latest R is R-3.6.1, which released on July 05, 2019.

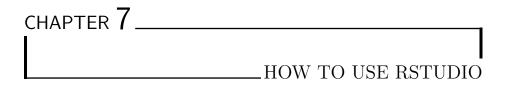
- Windows OS.
- Mac OS X
- $\bullet\,$ For Linux users, if you do not know how to install R, it is time to relearn Linux.

Install ${\tt R}$ as you normally install other softwares.

The latest RStudio is RStudio 1.2.1578, which released on September 17, 2019.

- Windows OS.
- Mac OS X
- $\bullet\,$ For Linux users, if you do not know how to install RS tudio, it is time to relearn Linux.

Install RStudio as you normally install other softwares.



Here comes an official cheat sheet and a webinar series of RStudio.

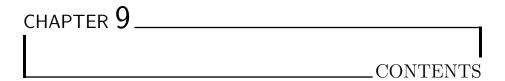


Firstly, you should know how to ask questions the smart way.

Secondly, you have several efficient ways to ask for help (the following ways are listed in descending order of priority), 1. from books or official documents; 2. from existing on-line resources, such as resources mentioned above and RStudio Support; 3. from search engines based on keywords, such as Bing; 4. from classmates or teachers; 5. from community, such as RStudio Community, Stack-Overflow and StackExchange.

Finally, after knowing how to solve the problems, you can get start with R.

Part III $\label{eq:continuous}$ Getting Started with R



- Cheat Sheets from RStudio
 - RStudio
 - Basis
 - Extra Packages
 - LaTeX and Markdown
- Quick Start



You can download all of the cheat-sheets by entering the following command in the terminal,

```
git clone https://github.com/rstudio/cheatsheets/
curl -0 https://www.rstudio.com/wp-content/uploads/2016/02/advancedR.pdf
curl -0 https://wch.github.io/latexsheet/latexsheet-a4.pdf
```

10.1 RStudio

• RStudio IDE.

10.2 Basis

- Base R.
- Advanced R.
- Dates and times.
- Strings.
- Apply functions.
- Data import.
- Animate ggplot2 plots with gganimate.
- Data visualization with ggplot2.
- Parallel computing.

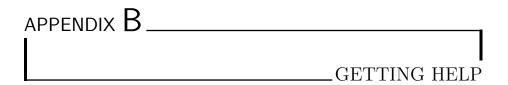
10.3 Extra Packages

- Data manipulation with data.table.
- Data transformation with dplyr.
- Factors with forcats.
- $\bullet~$ Fast, robust estimators with estimatr.
- A tabular guide to

$egin{array}{c} \mathbf{Part\ IV} \\ \mathbf{Appendix} \end{array}$

APPENDIX A ______CONTENTS

- Getting Help
 - Accessing the Help Files
 - More About An Object
- Using Packages
- Working Directory
- Vectors
 - Creating Vectors
 - Vector Functions
 - Selecting Vector Elements
 - * By Position
 - * By Value
 - $* \ {\rm Named \ Vectors}$
- Programming
 - For Loop
 - While Loop
 - If Statements
 - Functions
 - Reading and Writing Data
- Matrices
- Lists
- Data Frames
- Strings
- Factors
- Statistics
- Distributions



B.1 Accessing the Help Files

```
Get help of a particular function,

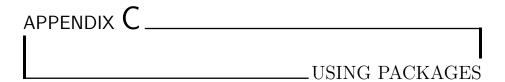
?mean

Search the help files for a word or phrase,
help.Search('weighted mean')

Find help for a package,
help(package='dplyr')
```

B.2 More About An Object

```
Get a summary of an object's structure,
str(iris)
Find the class an object belongs to,
class(iris)
```



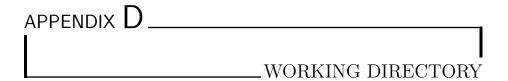
Download and install a package from CRAN,
install.packages('dplyr')

Load the package into the session, making all its functions available to use,
library(dplyr)

Use a particular function from a package,
dplyr::select

Load a built-in dataset into the environment,

data(iris)

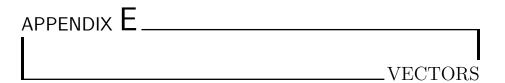


Find the current working directory (where inputs are found and outputs are sent),

getwd()

Change the current working directory,

```
# for windows
setwd('C://Users/...')
# for macosx
setwd('/Users/...')
# for linux
setwd('/home/...')
```



E.1 Creating Vectors

```
Join elements into a vector,

v <- c(2, 4, 6)

An integer sequence,

v <- 2:6

An complex sequence,

v <- seq(2, 3, by=0.5)

Repeat a vector,

v <- rep(1:2, times=3)

Repeat elements of a vector,

v <- rep(1:2, each=3)
```

E.2 Vector Functions

```
Return x sorted,
sort(x)
   Return x reversed,
rev(x)
   See counts of values,
table(x)
   See unique values,
unique(x)
```

E.3 Selecting Vector Elements

E.3.1 By Position

```
the fourth element,
```

x[4]

All but the forth,

x[-4]

Elements two to four,

x[2:4]

All elements except two to four,

x[-(2:4)]

Elements one and five,

x[c(1, 5)]

E.3.2 By Value

Elements which are equal to 10,

x[x == 10]

All elements less than zero,

x[x < 0]

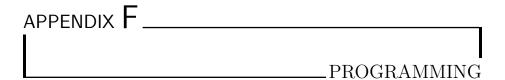
Elements in the set 1, 2, 5,

x[x %in% c(1, 2, 5)]

E.3.3 Named Vectors

Elements with name 'apple',

x['apple']



F.1 For Loop

```
# for (variable in sequence) {
# Do something
# }
for (j in 1:4) {
    j <- i + 10
    print(j)
}</pre>
```

F.2 While Loop

```
# while (condition) {
# Do something
# }
i <- 0
while (i < 5) {
    print(i)
    i <- i + 1
}</pre>
```

F.3 If Statements

```
# if (condition) {
# Do something
# } else {
# Do something different
# }
i <- 3
if (i > 3) {
    print('Yes')
```

```
} else {
    print('No')
}
```

F.4 Functions

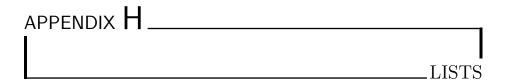
```
# function_name <- function(var) {
# Do something
# return(new_variable)
# }
square <- function(x) {
    squared <- x*x
    return(squared)
}</pre>
```

F.5 Reading and Writing Data

```
Read and write a delimited text file, - df <- read.table('file.txt') - write.table(df,
'file.txt')
   Read and write a comma separated value file, - df <- read.csv('file.csv')
- write.csv(df, 'file.csv')
   Read and write an R data file, a file type special for R, - load('file.RData')
- save(df, file='file.RData')</pre>
```

APPENDIX G _______MATRICES

```
m <- matrix(x, nrow=3, ncol=3)
    Select a row,
m[2, ]
    Select a column,
m[, 1]
    Select an element,
m[2, 3]
    Transpose,
t(m)
    Matrix multiplication,
m %*% n
    Find x in: m*x=n,
solve(m, n)</pre>
```



A list is a collection of elements which can be of different types,s

1 <- list(x=1:5, y=c('a', 'b'))

Second element of l,

1[[2]]

New list with only the first element,

1[1]

Element named x,

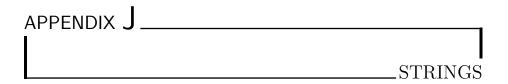
1\$x

New list with only element named y,

1['y']

APPENDIX ______DATA FRAMES

```
A special case of a list where all elements are the same length,
df <- data.frame(x=1:3, y=c('a', 'b', 'c'))</pre>
   |x|y| \ |\text{-}|\text{-}| \ |1|a| \ |2|b| \ |3|c|
   List subsetting,
df$x
df[[2]]
   See the full data frame,
View(df)
   See the first 6 rows,
head(df)
   Matrix subsetting,
df[ , 2]
df[2,]
df[2, 2]
   Number of rows,
nrow(df)
   Number of columns,
ncol(df)
   Number of columns and rows,
dim(df)
   Bind columns,
cbind
   Bind rows,
```



```
Join multiple vectors together,

paste(x, y, sep=' ')

Join elements of a vector together,

paste(x, collapse=' ')

Find regular expressoin matches in x,

grep(pattern, x)

Replace matches in x with a string,

gsub(pattern, replace, x)

Convert to uppercase,

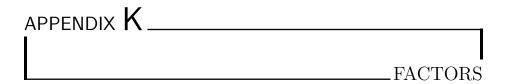
toupper(x)

Convert to lowercase,

tolower(x)

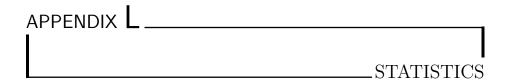
Number of characters in a string,

nchar(x)
```



Turn a vector into a factor. Can set the levels of the factor and the order, factor(x)

Turn a numeric vector into a factor by 'cutting' into sections, $\mathtt{cut}(\mathtt{x}, \mathtt{breaks=4})$



```
Linear model,

lm(y~x, data=df)

Generalized linear model,

glm(y~x, data=df)

Perform a t-test for difference between means,

t.test(x, y)

Perform a t-test for paired data,

pairwise.t.test

Test fir a difference between proportions,

prop.test

Analysis of variance,

aov
```

APPENDIX M	
	DISTRIBUTIONS