R Tutorial

Gao Qinghui

Wang Yanan Institute for Studies in Economics (WISE)
Xiamen University

October 27, 2017

Outline

1. What is R
   - R introduction
   - Getting Help
2. Data Types
   - Vectors
   - Matrices
   - Lists
   - Factors
   - Missing Values
   - Data Frames
3. Subsetting
4. Control Structures and Functions
5. Other Topics

Overview

1 What is R
- R introduction
- Getting Help

2 Data Types
- Vectors
- Matrices
- Lists
- Factors
- Missing Values
- Data Frames

3 Subsetting

4 Control Structures and Functions

5 Other Topics

What is R

R is a free software environment for statistical computing and graphics.

https://www.r-project.org/

Some R Resources

1. Available from CRAN (http://cran.r-project.org)
2. Code School: Try R
3. R Core Team, An Introduction to R
4. Michonneau, F., The R Class
5. Burns, P., The R Inferno
6. R Programming: a course offered by Johns Hopkins University on Coursera
7. A list of some useful book

package: swirl

swirl help you learn R in R.

```
1 install.packages("swirl")
2 library(swirl)
3 ?swirl
4 swirl
```

Overview

1 What is R
- R introduction
- Getting Help

2 Data Types
- Vectors
- Matrices
- Lists
- Factors
- Missing Values
- Data Frames

3 Subsetting

4 Control Structures and Functions

5 Other Topics

Getting Help in R

Knowing how to get help in R is the most important part in your coding study.
There are several ways to get help in R.
(https://www.r-project.org/help.html)

- Helping Yourself
- R Help on the Internet
- Asking for Help

R Help: help(), and ?

R Help: help(), and ?

```
help(lm)
?lm
?"lm" # The quotes are optional
?help
help(rlm, package="MASS")
help(package="MASS")
example(lm) # execute examples in the current R session
```

Others

R Help on the Internet

RSiteSearch, `Rseek.org`, `Baidu.com`

- CRAN Task Views
  `https://cran.r-project.org/web/views/`
- R FAQs (Frequently Asked Questions)
  `https://cran.r-project.org/doc/FAQ/R-FAQ.html`

Asking for Help

RSiteSearch, `Rseek.org`, `Baidu.com`

- Stack Overflow
  `http://stackoverflow.com/`
- R Email Lists
  `https://www.r-project.org/mail.html`

Objects

R has five basic or "atomic" classes of objects:

- character
- numeric (real numbers)
- integer
- complex
- logical (True/False)

Attributes

R objects can have attributes

- names, dimnames
- dimensions (e.g. matrices, arrays)
- class
- length
- others

Attributes of an object can be accessed using the **attributes()** function.

Overview

1 What is R
- R introduction
- Getting Help

2 Data Types
- Vectors
- Matrices
- Lists
- Factors
- Missing Values
- Data Frames

3 Subsetting

4 Control Structures and Functions

5 Other Topics

Creating Vectors

The <- symbol is the assignment operator.
The c() function can be used to create vectors of objects.
The : operator is used to create integer sequences.
The # is for comment, the code behind # would be ignored.
The class() function can be used to check the class of an object.
The print() function can be used to print the value of an object.

```
1  x <- c(0.5, 0.6)     ## numeric
2  x <- c(TRUE, FALSE) ## logical
3  x <- c(T, F)  ## logical
4  A <- c("a", "b", "c") ## character
5  class(A) ## print the class of A
6  print(A) ## print the value of A
7  B <- 9:29 ## integer
8  class(B)
9  C <- c(A, B) # Mixing Object
10 C
11 class(C)
12 ## When different objects are mixed in a vector,
13 ## coercion occurs so that every element
14 ## in the vector is of the same class
15 x <- c(1+0i, 2+4i) ## complex
```

Explicit Coercion

Objects can be explicitly coerced from one to another using as.* functions, if available.
Nonsensical coercion results in NAs.

```
1  x <- 0:6
2  class(x)
3  as.numeric(x)
4  as.logical(x)
5  as.character(x)
6
7  # Nonsensical coercion
8  x <- c("a","b","c")
9  as.numeric(x)
10 as.logical(x)
```

Overview

1 What is R
- R introduction
- Getting Help

2 Data Types
- Vectors
- Matrices
- Lists
- Factors
- Missing Values
- Data Frames

3 Subsetting

4 Control Structures and Functions

5 Other Topics

Matrices

Matrices are vectors with a dimension attribute. The dimension attribute is itself an integer vector of length 2 (nrow, ncol)

```
m <- matrix(nrow = 2, ncol = 3)
m
dim(m)
attributes(m)
```

Matrices are constructed *column-wise*.

```
m <- matrix(1:6, nrow = 2, ncol = 3)
m
```

Matrices can also be created directly from vectors by adding a dimension attribute.

```
m <- 1:10
m
dim(m) <- c(2,5)
m
```

cbind-ing and rbind-ing

Matrices can be created by *column-binding* or *row-binding* with rbind() and cbind().

```
1 x <- 1:3
2 y <- 10:12
3 cbind(x, y)
4 rbind(x, y)
```

Overview

1 What is R
- R introduction
- Getting Help

2 Data Types
- Vectors
- Matrices
- Lists
- Factors
- Missing Values
- Data Frames

3 Subsetting

4 Control Structures and Functions

5 Other Topics

Lists

Lists are a special type of vector that can contain elements of different classes.

```
1 x <- list(1, "a", TRUE, 1 + 4i)
2 x
3 y <- list(x, list("c", 32, FALSE))
4 y
```

Overview

1 What is R
- R introduction
- Getting Help

2 Data Types
- Vectors
- Matrices
- Lists
- Factors
- Missing Values
- Data Frames

3 Subsetting

4 Control Structures and Functions

5 Other Topics

Factors (Optional)

Factors are used for represent categorical data. One can think of a factor as an integer vector where each integer has a *label*.

```
1  x <- factor(c("yes", "yes", "no", "yes", "no"))
2  x
3  table(x)
4  unclass(x)
```

Overview

1 What is R
- R introduction
- Getting Help

2 Data Types
- Vectors
- Matrices
- Lists
- Factors
- Missing Values
- Data Frames

3 Subsetting

4 Control Structures and Functions

5 Other Topics

Missing Values

Missing values are denoted by NA or NaN for undefined mathematical operations.

- is.na() is used to test objects if they are NA
- is.nan() is used to test objects if they are NaN
- NA have a class also, such as integer of character
- A NaN is also NA but the converse is not true

```
1 x <- c(1, 2, NA, 10, NaN)
2 is.na(x)
3 is.nan(x)
```

Overview

1 What is R
- R introduction
- Getting Help

2 Data Types
- Vectors
- Matrices
- Lists
- Factors
- Missing Values
- Data Frames

3 Subsetting

4 Control Structures and Functions

5 Other Topics

Data Frames

Data frames actually are matrices which can store different classes of objects in each column. But elements within each column would be with the same class. One can think of a data frame as a matrix consist of vectors of different classes.

- Data frames have a special attribute called row.names
- Data frames are usually created by calling read.table() or read.csv()
- Can be converted to a matrix by calling data.matrix()

```
1  x <- data.frame(foo = 1:4, bar = c(T, T, F, F))
2  x
3  nrow(x)
4  ncol(x)
5  colnames(x)
6  names(x) # names() can be applied to any R basic objects.
```

Subsetting

There are a number of operators can be used to extract subsets of R objects.

- [always returns an object of the same class as the original; can be used to select more than one element.
- [[is used to extract elements of a list or data frame;
- $ is used to extract elements of a list or data frame by name; semantic are similar to hat of [[

```
1  x <- c("a", "b", "c" "d")
2  x[1]
3  x[1:4]
4  x[x > "a"]
5  u <- x > "a"
6  u
7  x[u]
```

Subsetting a Matrix

Matrices can be subsetted in the usual way with *(i,j)* type indices.

```
x <- matrix(1:6, 2, 3)
x[1,2]
x[1:2, 2:3]
x[2, 1:2]
```

Indices can also be missing.

```
x <- matrix(1:6, 2, 3)
x[1,]
x[,2]
```

Subsetting Lists

The [[can be used with *computed* indices; $ can be used with literal names

```
x <- list(foo = 1:4, bar = 0.6)
x[1]
x[[1]]
x$bar
x[["bar"]]
x["bar"]
x <- list(foo = 1:4, bar = 0.6, baz="hello")
x[c(1,3)]
```

```
x <- list(foo = 1:4, bar = 0.6, baz="hello")
name <- "foo"
x[[name]] ## computed index for 'foo'
x$name ## element 'name' doesn't exist!
x$foo
```

Subsetting Nested Elements of a List

The [[can take an integer sequence.

```
1 x <- list(a = list(10, 12, 14), b = c (3.14, 2.81))
2 x[[c(1,3)]]
3 x[[1]][[3]]
4 x[[c(2,1)]]
```

Removing NA Values

A common task is to remove missing values (NAs).

```
1 x <- c(1, 2, NA, 4, NA, 5)
2 bad <- is.na(x)
3 x[!bad]
```

Other useful functions: complete.cases(), na.omit() ...

Control Structures

- if, else: testing a condition
- for: execute a loop a fixed number of times
- while: execute a loop while a condition is true
- repeat: execute a infinite loop
- break: break the execution of a loop
- next: skip an iteration of a loop
- return: exit a function

Functions

Functions are processes.

```
1 f <- function(<arguments>){
2         ## Do something interesting
3 }
```

install and library packages

- install.packages() for install packages
- library()

Reading Data

- read.table, read.csv
- readLines
- source
- load
- others: you can always get help by R resources

Thanks!