Exploratory Data Analysis

Chapter 1

Instructor: Seokho Lee

Hankuk University of Foreign Studies

1. R Language

1.1. Introduction to R

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS.

Why do we use R?

- Freeware!!

  Most statistical softwares are so expensive.

- Flexible!!

  R is not only a statistical program but also a programming language. R is so flexible to use and develop something new as well.

- Fast updating!!

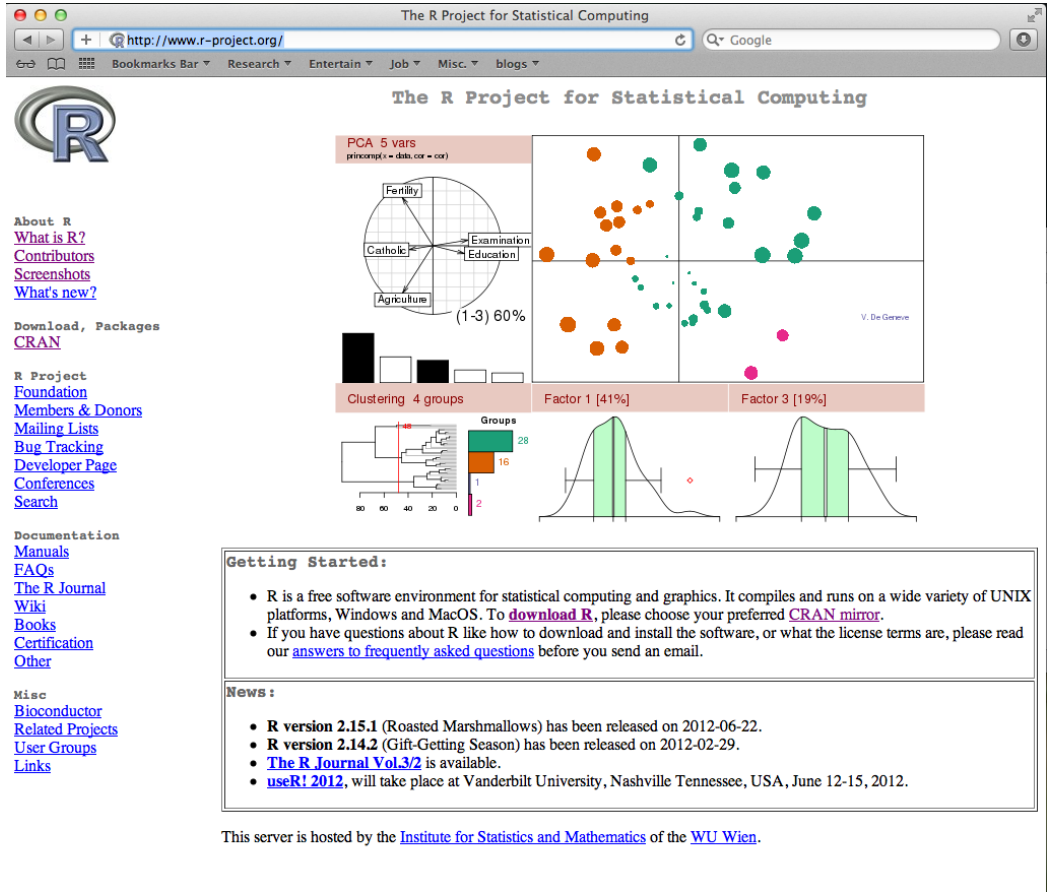
  Frontier researchers mostly use R and they develop and provide R packages. Users can use the most recent methodologies under R environment.

Limitation of R

- SPSS or Minitab are menu-based so easy to use
- R (as well as SAS) is language-based so hard to learn
- R has also a limitation on handling large-sized data, depending on your system
- Python (numpy+scipy+matplotlib) is another option.

Official webpage : `http://r-project.org`

From CRAN (you can find a CRAN tab or `http://cran.r-project.org/`), install file and its source code can be downloaded. The current version is 3.0.1 (by Sep/02/2013). Also, a lot of packages are available in CRAN.



Some useful links are here:

- Rstudio (IDE for R) : `http://www.rstudio.com`
- ggplot2 (R graphic package) : `http://ggplot2.org`
- 한국 R 사용자 모임 : `http://r-project.kr`

Example

```
> 1+1
[1] 2
>
```

- 2 is obtained as a value.
- New prompt (“>”) appears in the new line.
- [1] is an index for the value.

Example

```
> 10:30
 [1] 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
[16] 25 26 27 28 29 30
```

- The 16th value is 25.
- Indexes may vary depending on your console size.

Example

```
> 34-
+ 4
[1] 30
```

- '34-4' is submitted in two lines
- "+" is not the plus arithmetic operator, but the prompt to connect the consecutive two lines.

Example

```
> 2*pi          # pi is 3.141593
[1] 6.283185
```

- "#" is the comment operator. The following statements will be ignored by R interpreter.
- It is a very good practice to add comments in writing R programs.

1.3. R as a calculator

Try to use R for the following computation:

Example

```
> 4+5*7
[1] 39
> 4+5*7 # (1)
[1] 39
> 3+10/(2+3) # (2)
[1] 5
> (1/2+1/3)^2/(1/3^2) # (3)
[1] 6.25
> (10-4*2)^2/(-2) # (4)
[1] -2
> (3+(3*(3+4))-5)/4 #(5)
[1] 4.75
> 2^5; 2**5 # (6) both ways are possible
[1] 32
[1] 32
> 2^(1/3) # (7)
[1] 1.259921
```

(1) $4 + 5 \times 7$

(2) $3 + 10 \div (2 + 3)$

(3) $(\frac{1}{3} + \frac{1}{3})^2 \div \frac{1}{3^2}$

(4) $(10 - 4 \times 2)^2 \div (-2)$

(5) $[3 + \{3 \times (3 + 4)\} - 5] \div 4$

(6) 2^5

(7) $2^{\frac{1}{3}}$

- Note the order of priority!!

Operator	Description
{	block operator
(	parenthesis
$	extracting a component from a list or a data frame
[, [[	subscripts
^, **	power operators
-	negative operator
:	sequence operator
%*%, %/%, %%	matrix multiplication, quotient, remainder operator
*, /	multiplication, division
+, -	sum , subtraction
<, >, <=, >=, ==, !=	logical comparison operators
!	negation
&, &&, |, ||	logical operators
<-, =, ->	allocation operator

- Priority order is decreasing from the top to the bottom of the table.
- If you are not sure, then use parentheses.

Example

```
> sin(pi/4) # (1)
[1] 0.7071068
> sqrt(2) # (2)
[1] 1.414214
> sqrt(2)/2 # (1) sin(pi/4)=sqrt(2)/2
[1] 0.7071068
> exp(5) # (3)
[1] 148.4132
> log10(20) # (4)
[1] 1.30103
> log(20) # (5)
[1] 2.995732
```

(1) $\sin(45^\circ)$

(2) $\sqrt{2}$

(3) e^5

(4) $\log_{10} 20$

(5) $\log_e 20$

- Radian, not degree, should be used for arguments in triangular functions.
- `pi` is a built-in constant whose value is π.
- Logarithm with base 10 is `log10` and logarithm with base e is `log`.

Math functions

Function	Description
`abs`	absolute function
`ceiling`, `floor`, `trunc`	truncation functions
`exp`	exponential function
`gamma`	gamma function
`lgamma`	log gamma function
`log`	natural logarithm
`log10`	common logarithm
`round`, `signif`, `zapsmall`	rounding functions
`sign`	sign function
`sqrt`	square root
`cos`, `sin`, `tan`	sine, cosine, tangent functions
`acos`, `asin`, `atan`	arcsine, arccosine, arctangent functions
`cosh`, `sinh`, `tanh`	hyperbolic-sine, cosine, tangent functions

1.4. Object and Assignment

Example

```
> log(2^(1/3)+4)
[1] 1.660116
> x1 <- 2^(1/3) # value is assigned to x1
> log(x1+4) # re-use x1
[1] 1.660116
> x2 = 0.5 # value is assigned to x2
> x2+2 # re-use x2
[1] 2.5
```

- '=' and '<-' are assignment operators that assign a value on the right side to a variable in the left side.

Example

```
> 3 + 7 -> y1
> z1      <-  2   +            5
> assign('w1',2^5)
> y1; z1; w1
[1] 10
[1] 7
[1] 32
> L1 < -5              # not assignment
에러: 개체 'L1'이 없습니다
> L1 = 0                # regard '=' as a comparison operator
> L1 < -5
[1] FALSE
> L1_5                  # not assignment
에러: 개체 'L1_5'이 없습니다
```

- No white spaces are allowed between '<' and '−' in the use of '<−' or '−>'.
- White space is allowed between objects and operators except for assignment operators.
- When `assin()` function is used, object name should be in characters with quotation marks (`"w1"`) or single quotation marks (`'w1'`).
- '_' can be used in variable names. But this is not allowed in S-PLUS.

Naming convention rule:

Example

```
> 5scr <- 5                  # variable names cannot start with digits
에러:예기치 않은 심볼 입니다 in "5scr"
> _scr <- 5                  # variable names cannot start with _
에러:예기치 않은 입력입니다 in "_"
> .scr <- 5                   # variable names can start with .
> .scr
[1] 5
> 국어점수 <- 95          # variable names can be Korean
> 국어점수
[1] 95
> kor.scr <- 95
> KOR.SCR                   # R language is case-sensitive
에러:개체 'KOR.SCR'이 없습니다
> KOR.SCR <- 88
> kor.scr; KOR.SCR
[1] 95
[1] 88
```

Example

```
> ls()
[1] "kor.scr"  "KOR.SCR"  "L1"       "w1"       "x1"       "x2"
[7] "y1"       "z1"       "국어점수"
> objects()
[1] "kor.scr"  "KOR.SCR"  "L1"       "w1"       "x1"       "x2"
[7] "y1"       "z1"       "국어점수"
```

Example

```
> tmp <- 2 * pi
> tmp
[1] 6.283185
> rm(tmp)
> tmp
에러: 개체 'tmp'이 없습니다
> ls()
[1] "kor.scr"  "KOR.SCR"  "L1"       "w1"       "x1"       "x2"
[7] "y1"       "z1"       "국어점수"
> remove('L1','w1')
> ls()
[1] "kor.scr"  "KOR.SCR"  "x1"       "x2"       "y1"       "z1"
[7] "국어점수"
> rm(list=ls())        # remove all objects in the workspace
> ls()
character(0)
```