Chapter 2:
Your first R Session

Downloading and Installing R

✓ Base R software for Windows
- ➡ Download the latest R version from http://cran.r-project.org/bin/windows/base/.
  File name example: R-2.15.3-win.exe [for version 2.15.3]
- ➡ Run the .exe file and follow the steps to installation

✓ Base R software for Mac
- ➡ Download the latest R version from http://cran.r-project.org/bin/macosx/.
  File name example: R-2.15.3.pkg [for version 2.15.3]
- ➡ Open the .pkg file and follow the steps to installation

✓ R IDE for Windows
- ➡ R Studio offers a heavy duty Integrated Development Environment [IDE] for R. Link for download - http://rstudio.org/download/desktop
- ➡ Run the .exe file and follow the steps to installation

✓ R IDE for Mac
- ➡ R Studio offers an R IDE for Mac also.
- ➡ Link for download - http://rstudio.org/download/desktop
- ➡ Open the .dmg file and follow the steps to installation

URL for Linux - http://cran.r-project.org/bin/linux/

Finding your way around R

✓ Accessing Help
- ➡ Option 1: Type ?{keyword} at the command line. For ex., ?sum or ?mean
- ➡ Option 2: Type help({keyword}) at the command line. Ex., help(sum) or help(mean)
- ➡ Option 3: In the IDE menu, click Help > R Help. Then type in the keyword(s)

✓ Working Directory
- ➡ Is the directory where all input to and output from R are stored
- ➡ Function *getwd()* is used to get the current working directory.
  - Usage: Type *getwd()* at the command line
- ➡ Function *setwd()* is used to assign a desired path as working directory
  - Usage: Type *setwd({Path})* at the command line.
  - Example (Mac): *setwd*("/Users/jagannathrajagopal/Documents")
  - Example (Windows): *setwd*("C:\Program Files\R")
- ➡ In R Studio, access Tools > Set Working Directory from the menu
  - Three options available. Select "Choose Directory" to assign a desired path

✓ Navigation:
- ➡ Use arrow keys ↑ or ↓ to recall previous commands
- ➡ Depending on R GUI, [Ex. in the R Mac GUI, it is Alt + ⌘ + L] different key combinations can be used to clear the console

Exercise

Basic Commands

- ✓ Assignment
  - ➡ To assign a value or a formula to a variable, the assignment operator [= or <-] is used
    - Usage: a <- {Value} or a <- {Formula}. Example: a <- 3
    - Note: {Value} -> a or {Formula} -> a may also be used

- ✓ Continuation prompt
  - ➡ In some cases, an assignment needs to be made over a couple of lines. For example, when using two input vectors in the same formula
  - ➡ In these cases, the Enter key may be used to add a line to the command.

- ✓ Case Sensitivity
  - ➡ All function names, variable names, keywords etc in R are case-sensitive
  - ➡ Example: a <- 3 and A <- 4 will retain their case-specific values

- ✓ Assignment rules
  - ➡ Blank spaces are ignored
  - ➡ No standard names or key words should be used as object names or variables

Basic Commands

- ✓ Comments
  - ➡ Comments may be added to a function or assignment using the # key
  - ➡ Anything from the # to the end of the line is ignored
  - ➡ May be used in conjunction with a Continuation prompt
  - ➡ Example: a <- 3 #assign the value of 3 to a

- ✓ Last expression
  - ➡ The last value or expression returned by R is stored in .Last.value
  - ➡ This may be used as a variable is a function or in an assignment
  - ➡ NOTE: if the last expression was a function called, .Last.value will behave like a function
  - ➡ Example: a <- .Last.value

Exercise

Arithmetic Operators

Operator	Description	Example
+	Addition	> 2 + 3 [1] 5
-	Subtraction	> 20 - 13 [1] 7
x	Multiplication	> 2 x 3 [1] 6
/	Division	> 3 / 2 [1] 1.5
^	Exponentiation	> 2 ^ 3 [1] 8
%%	Modulo	> 87 %% 21 [1] 3

Logical Operators

Operator	Description	Example
==	Equals	> 2 == 3 [1] FALSE
!=	Not equal to	> 2 != 3 [1] TRUE
<	Less than	> 2 < 3 [1] TRUE
>	Greater than	> 2 > 3 [1] FALSE
<=	Less than or equal to	> 2 <= 3 [1] TRUE
>=	Greater than or equal to	> 2 >= 3 [1] FALSE
&	And	> 2 < 3 & 4 < 6 [1] TRUE
|	Or	> 2 > 3 | 4 < 6 [1] TRUE
!	Not	> a <- 3 [1] 3 > !a > 3 [1] TRUE

Exercise

Miscellaneous

- ✓ Finding objects
  - ➡ Function *objects*() is used to list objects in the current R environment/session
    - Usage: Type *objects*() at the command line
    - Example: objects() may result in [1] "a" "D" "x" # 3 objects a, D and x

- ✓ Removing objects
  - ➡ Function *rm*() is used to remove objects in the current R environment/session
    - Usage: Type *rm*({object}) at the command line
    - Above example: *rm*(a) will result in *object a* being removed.

- ✓ Missing values
  - ➡ If there are unknown objects in the R environment, the object may be found but the value may not be known. Code for missing value in R: NA
  - ➡ Function *is.na*() can be used to check if the value is missing
    - Usage: Type *is.na*({object}) at the command line
    - Example: If object D exists and has a value, *is.na*(D) will result in a FALSE

Miscellaneous

- ✓ Infinite values
  - ➡ Infinite values are derived from a division by or taking a log of zero. Code in R: Inf, -Inf
  - ➡ Functions is.finite() or is.infinite() may be used to check this
    - Usage: Type *is.finite*({object}) or *is.infinite*({object}) at the command line
    - Example: *is.finite*(a) where a is set to 5/0 will result in FALSE.
    - Example: is.infinite(b) where b is set to log(0) will result in TRUE.

- ✓ Indefinite values
  - ➡ An example of an indefinite value is 0/0. Code in R: NaN.
  - ➡ Functions is.nan() may be used to check this
    - Usage: Type *is.nan*({object}) at the command line
    - Example: *is.nan*(a) where a is set to 0/0 will result in TRUE.

Working with packages

- ✓ Installing packages
  - ➡ Packages are available to R through the CRAN
  - ➡ They can be installed either from the IDE. In R Studio for example, the lower right window in the Workspace has a Packages tab that allows packages to be downloaded from the CRAN
  - ➡ Updates to existing packages may also be found here

- ✓ Loading packages
  - ➡ Function *library*() is used to load a package to the current R environment/session
    - Usage: Type *library*({package}) at the command line
    - Above example: *library*("MASS") will result in MASS being loaded

- ✓ Listing loaded packages
  - ➡ Function *search*() can be used to list all loaded packages
    - Usage: Type *search*() at the command line

- ✓ Detaching packages
  - ➡ Function *detach*() is used to detach a package from the current R environment/ session
    - Usage: Type *detach*("package:{package}") at the command line
    - Above example: *detach*("package:MASS") will result in MASS being detached

Managing global preferences

- ✓ Viewing options
  - ➡ Function *options*() is used to load a package to the current R environment/session
    - Usage: Type *options*({package}) at the command line

- ✓ Changing preferences
  - ➡ Function *options*() may also be used to modify preferences
    - Usage: Type *options*({preference} =) at the command line
    - Example:

```
> options(digits = 7)
> options()
$HTTPUserAgent
[1] "R (2.15.1 x86_64-apple-darwin9.8.0 x86_64 darwin9.8.0)"

$OutDec
[1] "."

$device.ask.default
[1] FALSE

$digits
[1] 7

$dvipscmd
[1] "dvips"
```

Exercise