R Basics

Michael Malick

Updated: 15 May 2013

What is R?

An integrated suite of software facilities that provides an environment for data analysis, computation, and graphical displays

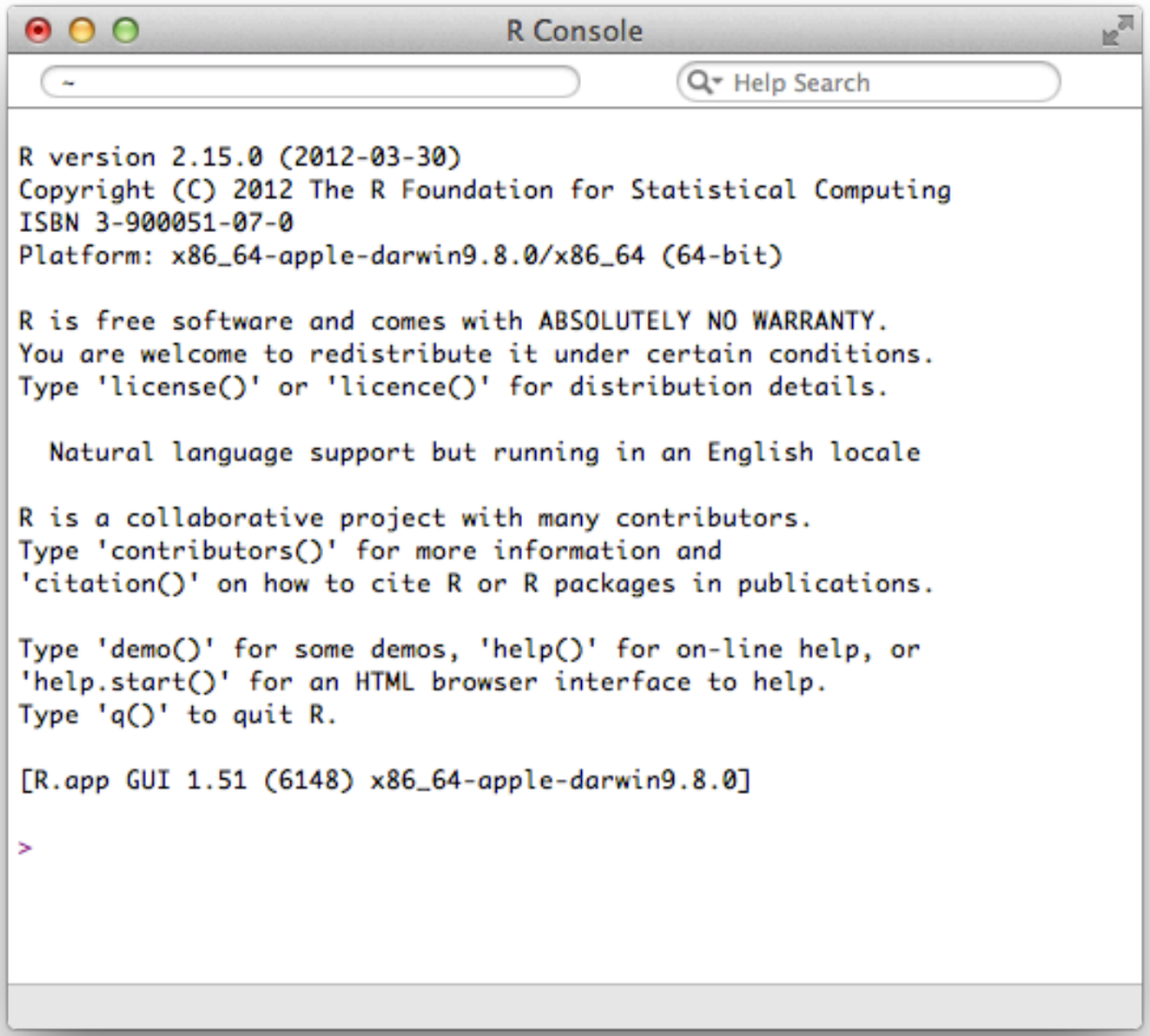
- Large collection of tools (i.e., functions) for data analysis
- A language for expressing statistical models
- Univariate and multivariate graphical facilities for data analysis
- An extendable programming language

(Venables and Ripley 1999)

R Overview

- First released in 1997
- Open-source software
- Dialect of the S programming language
- High level programming language
- Functional programming language
- Object oriented programming language

R Console


R Console
~
Help Search
R version 2.15.0 (2012-03-30)
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-apple-darwin9.8.0/x86_64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[R.app GUI 1.51 (6148) x86_64-apple-darwin9.8.0]
>

Basic Syntax

- R prompt: `>`
- `# This is a comment`
- `D` is different than `d`
- Assignment operator: `<-`

```
# Assign 10 to the letter x
x <- 10
```

Working Directory

- Each R project should be in a separate folder
- The project folder usually is the working directory
- Default search location for reading data and saving output

```
getwd()

setwd("~/Desktop")
```

Setting Working Directory

- Mac (section 2.3.3)
- Windows (sections 2.2 and 2.14)
- R Studio

Workspace

- Current working environment
- Contains any user-defined objects
- Workspace saved in .RData files

```
save.image()
save.image("project.RData")

load("project.RData")

ls() # view objects in the workspace
rm(x)# remove object x from workspace
```

History

- History is a record of executed commands
- Up arrow key accesses last executed command

Getting Help

R has extensive documentation for every function

```
help(plot) # view help for function plot
?plot      # same as help(plot)

# search help system for a pattern
help.search("plot")
??plot # same as help.search()

# Get a vector of search results
apropos("plot")
```

Packages

- R packages are a collection of functions, compiled code, help files, and datasets
- Packages are often written and distributed to solve a specific problem (e.g., bootstrapping)
- Two types of packages
  1. Packages distributed with base R (no installation required)
  2. User-defined or contributed (requires installation)
- Need to load all packages to use them

Packages Cont'd

```
install.packages("plyr")

library(plyr)

library() # view installed packages
```

R Scripts

- An R script is a collection of R commands saved in a text file (.R extension)
- Code should be saved in R scripts
- Built in script editor
- Many external script editors (e.g., Rstudio, Tinn-R, Emacs, Vim, ...)
- Send commands from script to R console
  - Mac: CMD + Enter
  - Windows: CTRL + R

R Workflow

Script

script.R

<functions> | Help search

```
# Assign the number 10 to the variable mm
mm <- 10
mm

# Assign a vector of length 3 to the variable nn
nn <- c(10, 11, 12)
nn

# Assign a sequence of numbers from 1 to 10 to the variable oo
oo <- 1:10
oo

# The assignment operator also works going to the right
1:10 -> oo
oo

# Assign a sequence of numbers from 1 to 10 by 0.5 to pp
pp <- seq(1, 10, 0.5)
pp

# Replicate 99 ten times
qq <- rep(99, 10)
qq

# Vectors can also be composed of other data types. The above vectors
# are all of class 'numeric'. We can also create 'character' vectors
# and 'logical' vectors.
class(pp)

# 'character' vector
rr <- c("A", "B", "C")
rr
class(rr)

# 'logical' vector
ss <- c(TRUE, TRUE, FALSE, TRUE, FALSE)
ss
class(ss)


# Each of the variables we created above are considered objects in R.
# We can view all the objects we created during the R session.
ls()

# If we don't need any of objects anymore we can remove them
rm(mm, nn)
```

R Console

R Console

~ | Help Search

```
R version 3.0.0 (2013-04-03) -- "Masked Marvel"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin10.8.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.60 (6476) x86_64-apple-darwin10.8.0]

[History restored from /Users/malickmj/.Rapp.history]

>
```

Intro to R Script

- script_intro_r.R

You Try...

1. Create a new folder on your desktop and name it ‘IntroR’
2. Set your working directory in R to the ‘IntroR’ folder
3. Create a new script and save it with the filename ‘intro.R’ to the IntroR folder
4. Using the script, assign the number 13 to the variable x `(x <- 13)` and send it to the R console

R as a Calculator

```
2 + 2   # Add

2 - 2   # Subtract

2 * 2   # Multiply

2 / 2   # Divide

2^2     # Exponents
```

Equivalence, Greater/Less Than

```
2 == 3   # Test for equivalence

2 != 3   # Test for non-equivalence
```

```
2 > 3    # Greater than
2 < 3    # Less than
2 >= 3   # Greater than or equal to
2 <= 3   # Less than or equal to
```

What is a Function?

A function takes some *inputs* and delivers a specific *output*

- Internal functions (mean, plot, ...)
- Functions in packages
- User defined functions

```
function(argument1, argument2, ...)
```

Basic Math Functions

```
x <- 10

log(x)    # log base e
log10(x)  # log base 10
exp(x)    # e raised to x
```

```
abs(-x)   # absolute value
sqrt(x)   # square root
```

Trig Functions

```
x <- seq(from = 0.1, to = 1, by = 0.1)

sin(x)    # sine
cos(x)    # cosine
tan(x)    # tangent
asin(x)   # arc-sine
acos(x)   # arc-cosine
atan(x)   # arc-tangent

pi
```

Basic Stats Functions

```
x <- 1:100

mean(x)          # average
median(x)        # median
var(x)           # variance
sd(x)            # standard deviation
quantile(x)      # main quantiles
max(x)           # maximum value
min(x)           # minimum value
summary(x)       # summary statitics
```

You Try...

1. What is the second argument to the `sd()` function?
   - Hint: look at the help file
2. What is the square root of 4.5?
3. What is the log of 5?

Warning Messages

Warning messages tell you something unusual happened while running the code

- Warnings do not halt the execution of the function or code
- You should generally try to fix any code that produces a warning message, particularly if you are unsure of why the warning message occurred

```
sqrt(-2)

log(-1)
```

Error Messages

Error messages indicate that a fatal error occurred while running the code

- Error messages halt the execution of the code
- You need to fix error messages in order to execute the code

```
x <- rnorm(10)
y <- rnorm(11)
cor(x, y)
```