A Simple Introduction to R

Outline

1. What is R
2. Why use R
3. Installing R in your own computer
4. R studio
5. Data Structures and Manipulation
6. Data Import
7. Exploratory Data Analysis
8. Loops
9. If/Else Statements
10. Data Export

What is R?

- R is a powerful, versatile, and free statistical programming language.

- Data analysis is done in R by writing or using built in scripts and functions in the R language.

- The R environment is not only equipped with all the standard methods, but also some of the most recent cutting-edge techniques.

- R is open source. This means that you can download and use R for free, and additionally the source code is open and available for inspection and modification.

Why Use R?

1. R is free and open.

2. R is a language. You learn much more than just point and click.

3. R has excellent tools for graphics and data visualization.

4. R is flexible. You are not restricted to the built in set of functions, you can use them and extend them with your own.

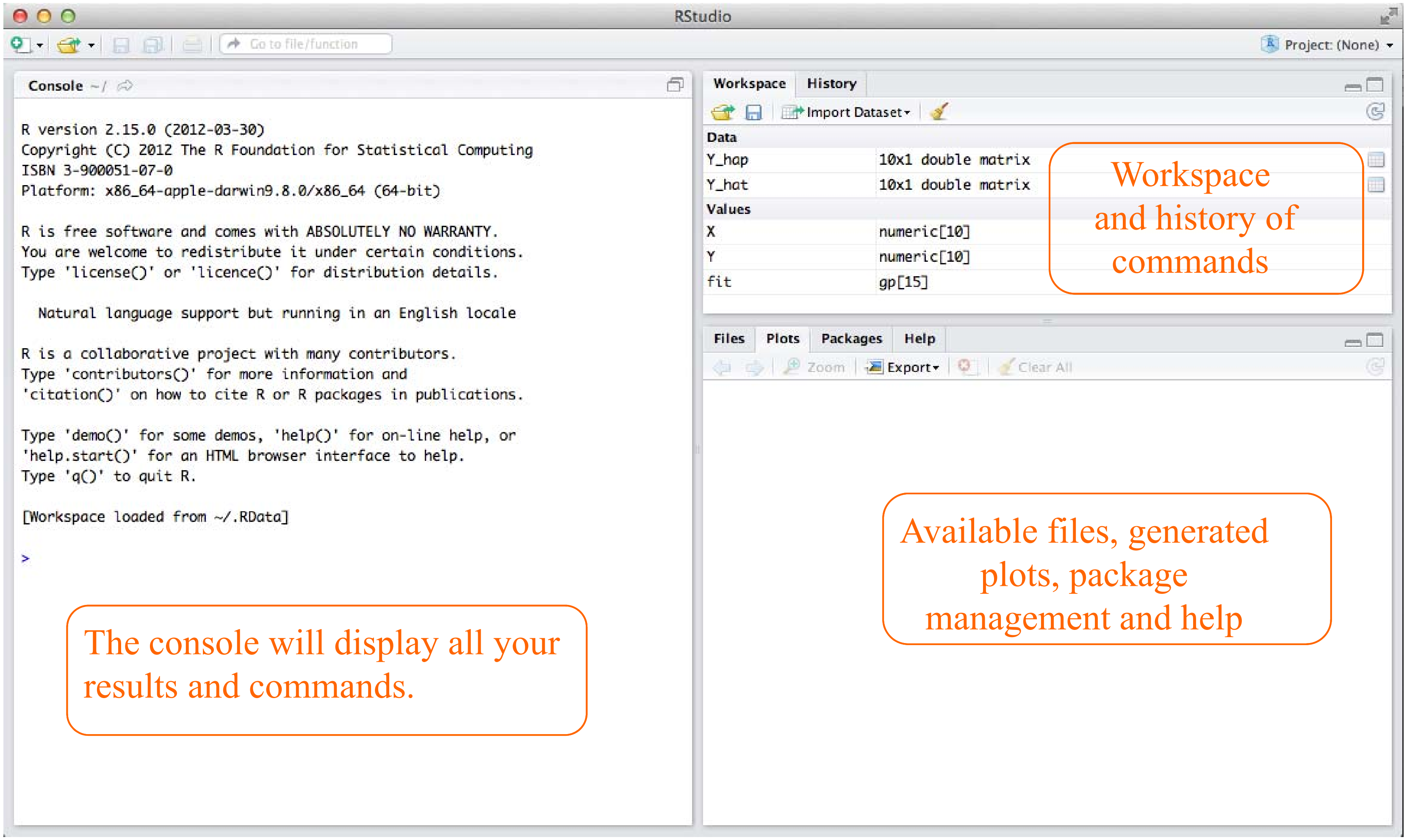
You can make your analysis your own!

How to Get R?

Windows:
http://cran.r-project.org/bin/windows/base/

MacOs X:
http://cran.r-project.org/bin/macosx/

R Studio



```
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.

[Workspace loaded from ~/.RData]

>
```

Data	
Y_hap	10x1 double matrix
Y_hat	10x1 double matrix
Values	
X	numeric[10]
Y	numeric[10]
fit	gp[15]

Workspace and history of commands

Available files, generated plots, package management and help

The console will display all your results and commands.

Rstudio can be downloaded from *https://www.**rstudio**.com/*

R Help Manual

- Point and Click at the interface and search.
- Type ?word at the console and R will search for help pages.
- Google: How to do ________ in R.
- UCLA website: http://www.ats.ucla.edu/stat/

Data Structures and Manipulation

1. Object Creation
   Expression: A command is given, evaluated and the result is printed on the screen.
   Assignment: Storing the results of expressions.

2. Vectors:
   The basic data structure in R. (Scalars are vectors of dimension 1).
   a. Creating sequences:
      - : command. Creates a sequence incrementing/decrementing by 1
      - seq() command.
   b. Vectors with no pattern. c() function.
   c. Vectors of characters. Also use c() function with the help of “”
   d. Repeating values. rep() function.
   e. Arithmetic with vectors: All basic operations can be performed with vectors.
   f. Subsets: The basic syntax for subsetting vectors is: vector[index]

Data Structures and Manipulation

3. Matrices: Objects in two dimensions.
   a. Creating Matrices
      Command: matrix(data, nrow, ncol, byrow).
      data: list of elements that will fill the matrix.
      nrow, ncol: number of elements in the rows and the columns respectively.
      byrow: filling the matrix by row. The default is FALSE.

   b. Some Matrix Functions
      - dim(): Lists the dimensions of the matrix.
      - cbind(): Creating matrix by putting columns together.
      - rbind(): Creating matrix by putting rows together.
      - diag(d): Creates identity matrix of dimension d.

Data Structures and Manipulation

c. Some Matrix computations
   - Addition
   - Subtraction
   - Inverse: function solve()
   - Transpose: function t()
   - Element-wise multiplication: *
   - Matrix multiplication: %*%

d. Subsets
   - Referencing a cell: matrix[r,c], where r represents the row and c represents the column.
   - Referencing a row: matrix[r,]
   - Referencing a column: matrix[,c]

Data Import

We need to set the working directory. For this we use the function setwd:

setwd(“location”)

1. Comma Separated Values:

   Use the function read.table

   mydatacsv<- read.table('Iris.csv', sep=',', header=T)

2. Text File:

   Use the function read.table:

   mydata = read.table(file = "iris.txt", sep = " ", header = TRUE)

Exploratory Data Analysis: Summary

Quantitative summary of variable Petal Width. We van calculate the mean, variance, median for that variable, as well as minimum, maximum, etc.

```
PW = iris$Petal.Width;
mean(PW)
var(PW)
median(PW)
min(PW)
max(PW)
```

You can obtain the 5 number summary for the variable by using the command:

```
summary(PW)
```

Exploratory Data Analysis: Grpah

1. Histogram of PW.

```
hist(PW, main="Histogram of Petal Width",
col="dodgerblue", breaks=10)
```

2. Boxplot of PW

```
boxplot(PW, main="Boxplot of Petal Width",
col="khaki1", ylab="Petal Width”)
```

3. Boxplot of PW by Species.

```
boxplot(PW~mydata[,5])
```

4. Normal Quantile-Quantile Plot

```
qqnorm(PW, main="Normal QQ Plot Petal Length")
```

For Loop

This statement allows for code to be executed repeatedly.

```
for(i in 1:n){
        statement
}
```

While Loop

This statement allows for code to be executed repeatedly while a condition holds true.

```
while(condition){
   statement
}
```

If/Else Statement

if statement – use this statement to execute some code only if a specified condition is true:

```
if(condition){
  statement
}
```

If/Else Statement

if...else statement – use this statement to execute some code if the condition is true and another code if the condition is false.

```
if ( condition )
 statement
else
statement2
```

If/Else Statement

if...else if....else statement – use this statement to select one of many blocks of code to be executed

```
if (condition){
statement
} else{
if (condition2){
statement2
} else {
Statement4
}
}
```

Data Export

If you have modified your dataset in R you can export it as a .csv file using the following code:

```
write.csv(mydatacsv,file="mydatacsv.csv")
```

or

```
write.table(file = "iris.txt", iris)
```

Can also export vectors or other objects that you have created to .csv file:

```
 write.csv(vec2,file="vec2.csv")
```

Data Export

If you have modified your dataset in R you can export it as a space delimited .txt file using the following code:

```
write.table(mydatacsv,file="mydatatxt.txt", sep=" ")
```

You can export it as a tab delimited .txt file using the following code:

```
write.table(mydatacsv,file="mydatatxt2.txt",
sep="\t")
```

Remarks and Comments

Thank you!