An Introduction to R

Anupama Lakshmanan

I must confess…

- I don’t have all the answers!
- There are more 13000 packages in R (as of 2018)
- *“Even the most experienced **R** user is surprised to learn about features they were **unaware** of”* – Robert Kabacoff, author of *R in Action*

What is R?

- Free and open source software to carry out statistical data analysis
  - Runs on Windows, Mac OSX, Linux
- Can also run on the cloud
- User contributed packages (> 13000)
- Read in data in multiple formats
  - Excel, csv, sas, spss, text…
- Share your analysis

History

- Before **R**, there was **S**! Data R Us!
  - Developed by people at Bell Labs, starting1976.
- Then in 1993, Robert Gentleman and Ross Ihaka from University of Auckland, NZ.

Pictures from Wikipedia

- Now maintained by a core team (including a statistician at ISI Delhi!)

Get This

- Install R
  - Download from http://cran.r-project.org/mirrors.html
  - **Basic console**, where you can type in **text commands**
- We will use Rstudio, a user-friendly environment
  - Download from http://rstudio.org/download/
  - Critical operations are **menu-driven**
  - help() or **?**
  - Try out **?lm** for help on a linear model

Ready to **R**umble?

- demo()
- Try out a couple
  - demo(graphics)
  - demo(Hershey)

Basics

- How is **data** represented?
  - Usually, as a **vector**
  - c(1, 2, 3, 4) is the same as c(1:4)
- Later, we shall study a matrix, data frame, list, . . .
- Try it in the **Console** window
- What if we wish to save our work?
- Select File → New File → R Script file
- Type the same things in and hit File → Save

Some basic stats

- # This is how you declare a set of numbers
- # In this case, from 1 to 10.
- nums <- c(1:10)

- # Explicitly declaring a set of numbers
- primes <- c(2,3,5,7,11,13,17,19,23,29)

- # Some stats on sets of numbers
- myMean <- mean(primes)
- cat("The mean of primes is", myMean,
  "\n") # This is an end-of-line character

Run or Source

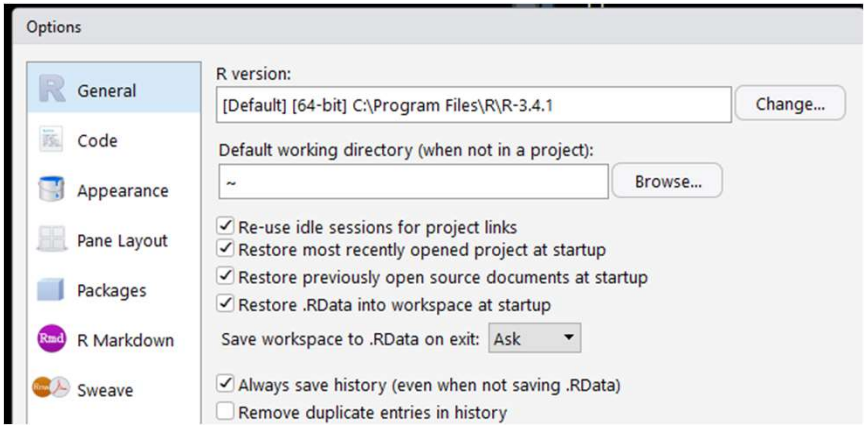
- Type these into a new file

```
normvar <- rnorm(1000)
hist(normvar)
unifvar <- runif(1000)
hist(unifvar)
```

- Highlight a few statements and hit **Run**
- Contrast this with **Source**
- Bring down 1000 to, say, 25, and watch what happens

Directory settings

- You want to access a file somewhere else
  - Like when reading in a data set from a file
- **Working directory**
  - Use getwd() & setwd("your_directory")
    - Make good use of **tab-completion**
- Optionally via a menu: Tools > Options



Read and Write (I/O)

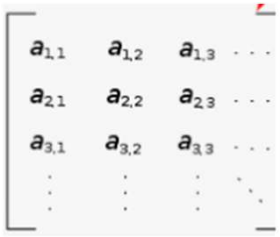
- Input from Console
  - scan(what = 'character')
- Input from a file
  - scan(file.choose())
- Output results to a file
  - **Text**: sink(“filename”,append=TRUE/FALSE)
  - **Graphics**: Use pdf/png/jpeg(“fileName”)

Packages

- When base R is not enough
- Install an external package
  - install.packages("data.table")
- Reference the package within your R session
  - library(data.table)

Datasets

- We often look at **spreadsheets** and databases
  - Different data *modes*: numeric, character, logical
  - Tabulated data has *variables* and *observations*
- **Vectors**
  - Condition: All items are of the same kind
    - Typically used to store a variable column
- v <- c(1, 3, 5); w <- c(“blue”, “red”)
- Refer to elements as v[1], w[2], v[c(1,3)]
- Similar to spreadsheets, you can try v[c(1:3)]



Matrices

- Rectangular arrays of data with a common mode
  - Can only have two dimensions
  - Note: If the dataset has mixed mode, use a **data frame**
- A <- matrix(c(1,2,3,4,5,6), nrow=2)
  - Can supply row and column names with dimnames
  - Access a row/column as A[rowNum,]/ A[,colNum]
  - To access 2nd row 3rd column element, use A[2, 3]

Exercise: How do you create a **submatrix** of A?

File: 2-Matrices.R

More Data Structures - Lists

- List - ordered collection of objects which allows to gather a variety of (possibly unrelated) objects under one name.
- List with 4 components: string, numeric vector, matrix and a scalar

```
w <- list(name="Fred", mynumbers=c(1,2,3), mymatrix=matrix(1:10,nrow=2), age=5.3)
```

- w
- $name

```
[1] "Fred"
```

- $mynumbers

```
[1] 1 2 3
```

- $mymatrix

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
```

- $age

```
[1] 5.3
```

Understanding Data Types - factors

- A factor stores the nominal values as a vector of integers in the range [1... k]

```
gender <- c(rep("male",20), rep("female", 30))
gender <- factor(gender) # stores gender as 20 1s and 30 2s and associates. 1=female, 2=male. #R now treats gender as a nominal variable
summary(gender)
```

Data frame

- Grid where the data can be of **mixed mode**
  - True of most datasets in real life
  - Cannot have mixed mode within the same column
  - Access columns with frame$<columnName>
- You can **summarise** numerical data values
  - Try summary(frame$serialNums) for example
- You can **order** categorical values in a column
  - Check out the example of the **factor** function

File: 3-Frames.R

Build a Dataframe

- Suppose you wish to build a data frame of **mixed mode**

```
col1 <- c(12, 32, 445, 13, -5)
col2 <- col1 + 3
col3 <- c(“Aman”,”Neha","Prabu","Sunil","Deb")
myFrame <- data.frame(col1, col2, col3)
View(myFrame)
print(sort(myFrame$col1))
print(sort(col3))
```

Import (Export) Data

- Data can be imported in multiple ways
- Common ways
  - csv file

```
myData <- read.table(file.choose(),header=TRUE, sep=",")
# first row of the csv being #read, contains variable #names, comma is separator
```

(write.table() for output)

- data.table is an R package that provides **an enhanced version** of data.frames
  - With this package, can read in large tables using fread()
  - Similarly, to write to a file, use fwrite()

Let's Play

- Google string: "AAPL stock historical prices Yahoo"
- Dates are tricky to work with
  - First **convert** the Date column to the correct format
  - as.Date function
  - Mac users, alter the format: "%d/%m/%Y"
- Standard Date functionality shipped with R is **limited**
  - We use a **library** called **lubridate** to play with dates
  - To access the month, day, year fields

File: 4-Plot.R

Plot Parameters

- Default list of modifiables: par(no.readonly=TRUE)
- Declare a parameter with par(name=value)

bg/fg	Back/foreground	Look up colors()
cex	Scaling factor	Used on text/symbols
col	Plotting color	By "name" or number
lty	Line type	"solid" "dashed" …
mfcol,mfrow	Grid size	To combine graphs
pch	Point symbol	See below!
pty	Type of plot region	"s" square "m" maximal

Try this

- Good practice: Capture the **original** parameter settings, modify them, and **restore** them once you are done
  - See how this is done in **R**
- Consider the **Apple** time series plot
  - Modify the **subset** to focus on August 2013 data only
  - Change the **line** type to ----
  - Change the **symbol** type to a diamond (pch=5)
  - **Scale** the symbol to 1.5 times the size (cex)
  - Adjust the **margins** of the plot
- Overlay plots

File: 5-Overlay.R

Charts

- For **categorical** data, we want to depict **counts**
  - Append a Month variable in the Apple stock data
  - Plot the **number** of trading days by month over years
    - barplot
    - pie
- Figure out which trading month is the "best"
  - aggregate **by month**, and apply the **mean** FUNction
  - Barplot the numbers

File: 6-Charts.R

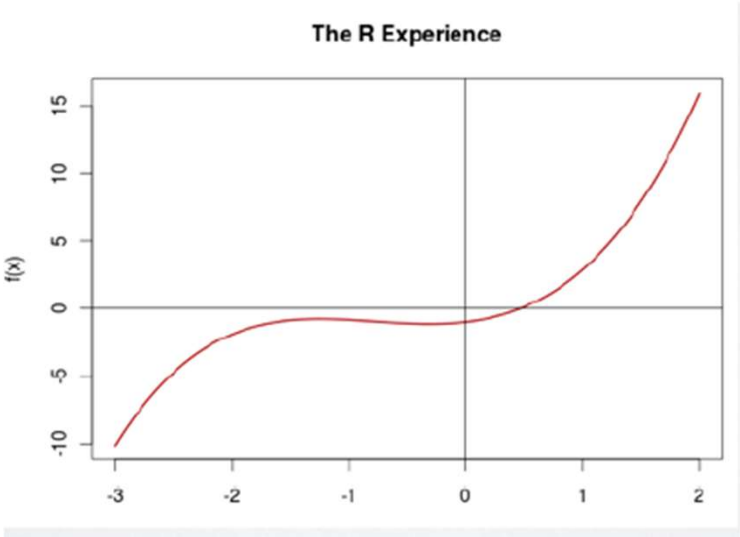
Boxplots

- **Box-and-whisker** plots
  - boxplot(numbers) draws a **box** from Q1 to Q3
  - Plus **whiskers** extending to either min/max or to **outliers** =**1.5 IQRs** from the quartile marks
  - summary(numbers) returns min, Q1, Q2, Q3, max
  - boxplot.stats(numbers) returns Q1-2-3 + hinges

File: 7-Boxplots.R

Functions

- Repeating some operations many times? Write your own
  - Name the function - it's like a variable
- Think of it as (x, f(x)), where x ranges over a seq of values



File: 8-Functions.R

Data Analysis

- Life Cycle Savings
- Under the life-cycle savings hypothesis as developed by Franco Modigliani, the savings ratio (aggregate personal saving divided by disposable income) is explained by
  - per-capita disposable income,
  - the percentage rate of change in per-capita disposable income
  - and two demographic variables:
    - the percentage of population less than 15 years old
    - the percentage of the population over 75 years old.
- The data are averaged over the decade 1960–1970 to remove the business cycle or other short-term fluctuations.
- Source:

  The data were obtained from Belsley, Kuh and Welsch (1980).

Thank You