



SC-Camp 2015

Introduction to R and Data Analysis

Joseph Emeras

University of Luxembourg, Luxembourg



Summary

1 Introduction to R

2 Crazy Examples

3 Practical Session

- Pre-requisites

- Objectives

- Practical Session Details



Summary

1 Introduction to R

2 Crazy Examples

3 Practical Session

Pre-requisites

Objectives

Practical Session Details



R?

R (pronounced aRgh – pirate style) is a programming language and environment for statistical computing and graphics

- oriented towards data handling analysis and storage facility
- R Base
- Packages tools and functions (user contributed)
- R Base and most R packages are available from the [Comprehensive R Archive Network \(CRAN\)](#)
- Use R console or IDE: **Rstudio**, Deducer, vim/emacs...
- Comment is #, help is ? before a function name



Using R

Installing/using packages

Install and load the ggplot2 package (even if already installed)

```
install.packages("ggplot2")  
library(ggplot2)
```

Or in one step, install if not available then load:

```
require(ggplot2) || {install.packages("ggplot2");  
                      require(ggplot2)}
```



Using R

Usefull Functions

- List all objects in memory: `ls()`
- Save an object: `save(obj, file)`
- Load an object: `load(file)`
- Set working directory: `setwd(dir)`



Data Structures

- scalar:

```
s = 3.14
```

- vector:

```
v = c(1, 2, "ron")
```

- list:

```
l = list(1:10, 'a', pi)
```

- matrix:

```
m = matrix(seq(1:6), 2)
```

- **dataframe:**

```
df = data.frame("col1" = seq(1:4), "col2" = c(5, 6, "cerveza", 6*7))
```

- ...



Entering Data

Reading CSV or text files

```
# comma separated values
dat.csv <- read.csv(<file or url>)
# tab separated values
dat.tab <- read.table(<file or url>,
  header=TRUE, sep = "\t")
```




Entering Data

Reading data from other software: Excel, SPSS...

Excel Spreadsheets – need `xlsx` package

```
read.xlsx()
```

SPSS and Stata both need the `foreign` package

```
dat.spss <- read.spss(<file or url>,  
                      to.data.frame=TRUE)
```

```
dat.dta <- read.dta(<file or url>)
```



Data Frames

Most easy structure to use, have a matrix structure.

- **Observations** are arranged as **rows** and **variables**, either numerical or categorical, are arranged as **columns**.
- Individual rows, columns, and cells in a data frame can be accessed through many methods of indexing.
- We most commonly use **object[*row,column*]** notation.



Accessing Items in a data.frame

Aside with R are provided example datasets, i.e. `mtcars` that can be used

```
data(mtcars)
```

```
head(mtcars)
```

```
colnames(mtcars)
```

```
# single cell value
```

```
mtcars[2,3]
```

```
# omitting row value implies all rows
```

```
mtcars[,3]
```

```
# omitting column values implies all columns
```

```
mtcars[2,]
```



Accessing Items in a data.frame

We can also access variables directly by using their names, either with **object[,"variable"]** notation or **object\$variable** notation.

```
# get first 10 rows of variable 'mpg' using two methods:  
mtcars[1:10, "mpg"]  
mtcars$mpg[1:10]
```



Exploring Data

Description Of Dataset

- Using **dim**, we get the number of observations(rows) and variables(columns) in the dataset.
`dim(mtcars)`
- Using **str**, we get the structure of the dataset, including the class(type) of all variables.
`str(mtcars)`
- **summary** when used on a dataset, returns distributional summaries of variables in the dataset.
`summary(mtcars)`
- **quantile** function enables to get statistical metrics on the selected data
`quantile(mtcars$mpg)`



Exploring Data

Conditional Exploration

- **subset** enables to explore data conditionally
`subset(mtcars, cyl <= 5)`
- **by** enables to call a particular function to sub-groups of data
`by(mtcars, mtcars$cyl, summary)`



Summary

1 Introduction to R

2 Crazy Examples

3 Practical Session

Pre-requisites

Objectives

Practical Session Details



Random Text Generation

```
library(XML)
stem <- "http://www.5novels.com/classics/u5688"
hobbit <- NULL
for(i in 1:74) {
  if(i==1) { url <- paste0(stem, ".html") }
  else { url <- paste0(stem, "_", i, ".html") }
  x <- htmlTreeParse(url, useInternalNodes=TRUE)
  xx <- xpathApply(x, "//p", xmlValue)
  hobbit <- c(hobbit, gsub("\r", "", xx[-length(xx)]))
  Sys.sleep(0.5)
}
hobbit = paste(hobbit, collapse=' ')
```




Random Text Generation – 2

```
library(ngram)
ng2 <- ngram(hobbit, n=2)

babble(ng2, 128, seed=987654)
```



PiRates



Summary

1 Introduction to R

2 Crazy Examples

3 Practical Session

Pre-requisites

Objectives

Practical Session Details



Install and Run R

1 On your local machine:

- Find a release that fits your distribution at CRAN Archive
- Install and launch R-Studio

<http://cran.r-project.org/>

<https://www.rstudio.com/>

2 On the cluster

First connect to the cluster, then submit a job to run R.

```
(localhost)$> ssh gaia-cluster  
(frontend)$> oarsub -I -l core=1,walltime="00:30:00"  
(node)$> module load lang/R/3.2.0-ictce-7.3.5-bare  
(node)$> R
```

3 Install and Load a Package

```
(R-shell)$> install.packages("ggplot2")  
(R-shell)$> library(ggplot2)
```



Objectives of this Practical Session

- Being able to plot data
 - histogram for data distribution
 - plot in different colors from different data sources
- Know some tips to organize your data
 - aggregate a dataset by column and apply an aggregation function
 - data.table package for binary search in datasets
 - performance in R operations
- R in parallel
 - on one machine
 - on a cluster with socket communications
 - MPI communications



Exercises

- Start the tutorial <https://github.com/ULHPC/tutorials/tree/devel/advanced/R>
 - Plot 2 graphs in section Simple Plotting
 - Answer 2 questions at the end of section Organizing your Data
 - Compare performance of aggregation operations w/wo parallelization
- Plot a speedup graph
 - with different number of cores and/or machines
 - needs: ggplot, parallel R



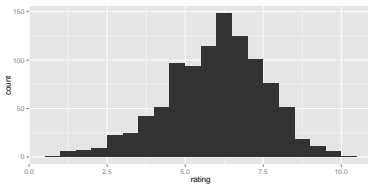
After the Practical Session: Problems

- regression models: example and exercise
- parallelization of K-means Clustering: example and exercise

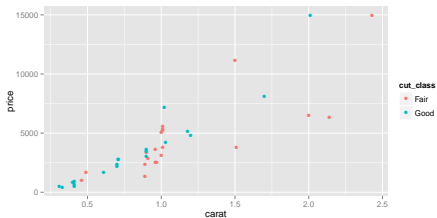


Simple Plotting

Movies Histogram:



Diamonds Plot with 2 colours:





PS Questions

Question: use ddply instead of tapply in the first example

```
ddply(DT, .(x), summarize, sum(v))
```

Question: return the min and max instead of the sum.

```
min_max = function(data){  
  c(min(data), max(data))  
}  
DT[,min_max(v),by=x]  
  
## or  
DT[,c(min(v), max(v)),by=x]
```



Thank you for your attention...

Questions?



- 1 Introduction to R
- 2 Crazy Examples
- 3 Practical Session
 - Pre-requisites
 - Objectives
 - Practical Session Details



Thank you for your attention...

Usefull links

- CRAN Archive <http://cran.r-project.org/>
- ggplot2 Documentation <http://docs.ggplot2.org/current/>
- CRAN HPC Packages <http://cran.r-project.org/web/views/HighPerformanceComputing.html>
- Advanced R programming by Hadley Wickham <http://adv-r.had.co.nz/>