Introduction to R

Osama Mahmoud

Website: ***http://osmahmoud.com***

E-mail: ***o.mahmoud@bristol.ac.uk***

A part of the course:

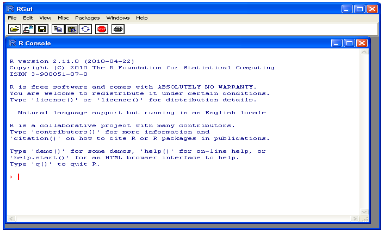
Introduction to Data Visualisation & Web-applications, 14-15 Nov. 2019

Contents

What is R?

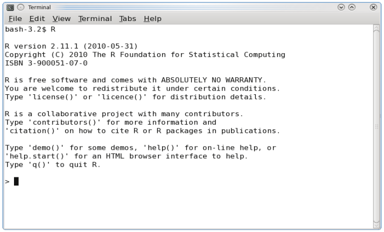
- R is a language (environment) for data analysis.
- Command-line oriantated available for Windows, Linux and Mac OS X.

Windows



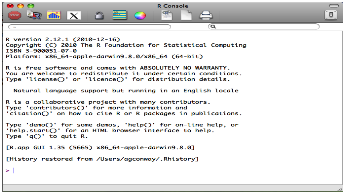
```
R version 2.11.0 (2010-04-22)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

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.

> |
```

Linux



```
bash-3.2$ R

R version 2.11.1 (2010-05-31)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

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.

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.

> █
```

Mac OS X



```
R version 2.12.1 (2010-12-16)
Copyright (C) 2010 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 1.35 (5665) x86_64-apple-darwin9.8.0]

[History restored from /Users/agconway/.Rhistory]

> |
```

Why R?

- R is free, and available on every major platform (www.cran.r-project.org).
- Massive set of packages for statistical modelling, visulaisation, AI, ML, and data science.
- Powerful tools for communicating results:
  - RMarkdown - turns your results into HTML, PDF, Word documents, PowerPoint presentations, and more.
  - Shiny - makes beautiful interactive apps without any knowledge of HTML or Javascript.
- Many IDE, e.g. RStudio which tailored to the needs of data science and statistical programming.
- Cutting edge tools - Researchers do often publish an R package with their technical articles.
- Strongly supported open-source communities (e.g. over 50% of software engineers at RStudio work on open source projects)

Why R?

- R is free, and available on every major platform (www.cran.r-project.org).
- Massive set of packages for statistical modelling, visulaisation, AI, ML, and data science.
- Powerful tools for communicating results:
  - RMarkdown - turns your results into HTML, PDF, Word documents, PowerPoint presentations, and more.
  - Shiny - makes beautiful interactive apps without any knowledge of HTML or Javascript.
- Many IDE, e.g. RStudio which tailored to the needs of data science and statistical programming.
- Cutting edge tools - Researchers do often publish an R package with their technical articles.
- Strongly supported open-source communities (e.g. over 50% of software engineers at RStudio work on open source projects)

Why R?

- R is free, and available on every major platform (www.cran.r-project.org).
- Massive set of packages for statistical modelling, visulaisation, AI, ML, and data science.
- Powerful tools for communicating results:
    - RMarkdown - turns your results into HTML, PDF, Word documents, PowerPoint presentations, and more.
    - Shiny - makes beautiful interactive apps without any knowledge of HTML or Javascript.
- Many IDE, e.g. RStudio which tailored to the needs of data science and statistical programming.
- Cutting edge tools - Researchers do often publish an R package with their technical articles.
- Strongly supported open-source communities (e.g. over 50% of software engineers at RStudio work on open source projects)

Why R?

- R is free, and available on every major platform (www.cran.r-project.org).
- Massive set of packages for statistical modelling, visulaisation, AI, ML, and data science.
- Powerful tools for communicating results:
    - RMarkdown - turns your results into HTML, PDF, Word documents, PowerPoint presentations, and more.
    - Shiny - makes beautiful interactive apps without any knowledge of HTML or Javascript.
- Many IDE, e.g. RStudio which tailored to the needs of data science and statistical programming.
- Cutting edge tools - Researchers do often publish an R package with their technical articles.
- Strongly supported open-source communities (e.g. over 50% of software engineers at RStudio work on open source projects)

Why R?

- R is free, and available on every major platform (www.cran.r-project.org).
- Massive set of packages for statistical modelling, visulaisation, AI, ML, and data science.
- Powerful tools for communicating results:
  - RMarkdown - turns your results into HTML, PDF, Word documents, PowerPoint presentations, and more.
  - Shiny - makes beautiful interactive apps without any knowledge of HTML or Javascript.
- Many IDE, e.g. RStudio which tailored to the needs of data science and statistical programming.
- Cutting edge tools - Researchers do often publish an R package with their technical articles.
- Strongly supported open-source communities (e.g. over 50% of software engineers at RStudio work on open source projects)

Why R?

- R is free, and available on every major platform (www.cran.r-project.org).
- Massive set of packages for statistical modelling, visulaisation, AI, ML, and data science.
- Powerful tools for communicating results:
  - RMarkdown - turns your results into HTML, PDF, Word documents, PowerPoint presentations, and more.
  - Shiny - makes beautiful interactive apps without any knowledge of HTML or Javascript.
- Many IDE, e.g. RStudio which tailored to the needs of data science and statistical programming.
- Cutting edge tools - Researchers do often publish an R package with their technical articles.
- Strongly supported open-source communities (e.g. over 50% of software engineers at RStudio work on open source projects)

Why R?

- R is free, and available on every major platform (www.cran.r-project.org).
- Massive set of packages for statistical modelling, visulaisation, AI, ML, and data science.
- Powerful tools for communicating results:
    - RMarkdown - turns your results into HTML, PDF, Word documents, PowerPoint presentations, and more.
    - Shiny - makes beautiful interactive apps without any knowledge of HTML or Javascript.
- Many IDE, e.g. RStudio which tailored to the needs of data science and statistical programming.
- Cutting edge tools - Researchers do often publish an R package with their technical articles.
- Strongly supported open-source communities (e.g. over 50% of software engineers at RStudio work on open source projects)

Editors

Functionality of editor softwares

- Easy tools for writing, managing and organizing R scripts
- offers syntax Auto-completion
- An effective interface to communicate with R

Examples of Editors

• **Windows**	• **Linux**	• **Mac OS X**
✓ RStudio	✓ RStudio	✓ RStudio
✓ RWinEdt	✓ JGR	

RStudio is highly recommended

- Multi-platform IDE (Download: www.rstudio.com)
- Provides excellent handy features and professional products.

Concepts of R

- In an interactive session, R works with Read-Eval-Print mode:
  - Input: `5 + 2`
  - Evaluation: R evaluates the input expression `5 + 2 = 7`
  - Output: R prints the resulted output(s) `[1] 7`

- Comments are identifyed by #
- Input prompt: >
- Prompt turns to '+' when an expression needs to be completed.

Sources for R help

Embeded Help System

- Search help for a topic `help.search("sum")`
- Search help for a function `help("sum")` or `?sum`
- Search help for usages of a function `example("sum")` or `demo()`

Integrated Help System

- HTML help system via your web browser `help.start()`

Online Sources & Communities

- rweekly: www.rweekly.org
- R seek: http://www.rseek.org
- Twitter: #rstats twitter community

Assign values

Example: BMI Calculation

Calculation of the Body Mass Index for a person whose weight in kilograms and height in meters are w and h respectively.

$$BMI = \frac{w}{h^2}$$

```
> w <- 100              # assignment of weight value
> h <- 1.75             # assignment of height value
> BMI <- w/(h^2)        # calculation of BMI
> BMI
[1] 32.65306
```

Vectors

- Combinations of elements from the same data type

```
> x <- c(2,4,6)
> x
[1] 2 4 6
```

```
> days <- c("Monday", "Thursday")
> days
[1] "Monday" "Thursday"
```

- Sequences

```
> x <- 3:5
> x
[1] 3 4 5
```

```
> x <- seq(from=10, to=20, by=2)
> x
[1] 10 12 14 16 18 20
```

- Replications

```
> x <- rep(5, times=3)
> x
[1] 5 5 5
```

```
> x <- rep(c(1,2,3), each=2)
> x
[1] 1 1 2 2 3 3
```

Indexing elements of a vector

- Vector elements are selected with a set of indices locating in square brackets:

  ✓ *Regular indexing*

```
> x <- 11 : 20
> x [3]
[1] 13
```

```
> x <- seq(0, 1, by=0.1)
> x [c(1, 2, 5)]
[1] 0.0 0.1 0.4
```

  ✓ *Reverse indexing*

```
> x <- 1 : 5
> x [-2]
[1] 1 3 4 5
```

```
> x <- c(1, 3, 5, 7, 11, 13)
> x [-(1 : 3)]
[1] 7 11 13
```

Logical values

A boolean expression, e.g. `a <= b`, returns logical value: `TRUE` or `FALSE`. R treats `TRUE` as 1 and `FALSE` as 0 when they are used in an arithmetic operation.

```
> 3 <= 5
[1] TRUE
> TRUE + TRUE + FALSE
[1] 2
> x <- c(1,2,3,4,5); x < 4
[1] TRUE TRUE TRUE FALSE FALSE
sum(x < 4)
[1] 3
```

Logical functions: `any()`, `all()`, `which()`

Missing values

NA (Not Available) represents missing values

```
> x <- c(2, -1, NA, 5, 0); any(is.na(x))
[1] TRUE
```

```
> x[10]
[1] NA
```

Proceses with missing values result in `NA` unless the answer is clear.

```
> sum(x)
[1] NA
```

```
> NA & TRUE
[1] NA
```

```
> NA | TRUE
[1] TRUE
```

However, some functions can exclude missing values from calculations. Users can extract data without missing values using the function `na.omit()`

```
> sum(x, na.rm = TRUE)
[1] 6
```

```
> na.omit(x)
[1] 2 -1 5 0
attr(,"na.action")
[1] 3
attr(,"class")
[1] "omit"
```

Factor class

The factor data type is used for categorical data. It can be produced by the function factor().

```
> Groups <- factor(c("Treatment", "Control", "Treatment",
+ "Control", "Control"))
> Groups
```

```
[1] Treatment Control Treatment Control Control
Levels:  Control Treatment
```

Remark

The factor data is coded with numbers. Note, mode() leads to numeric. Nevertheless, factors can be identified by class()

Practical Example

- Check the mode and class of the Groups vector.
- Convert the Groups vector to a numerical form.

Types of data structures

Structures of R's data can be organised by their dimensionality and whether they're homogeneous or heterogeneous:

dim.	Homogeneous	Heterogeneous
1d	`vector`	`data frame`
2d	`matrix`	`list`
nd	`array`	

```
> ?array
> ?data.frame
```

Concatenating of structures

The functions rbind() and cbind() concatenate data structures by row and by column respectively.

```
> Patient <- c(102, 105); gender <- factor(c("F", "M"))
> Heart.R <- c(83, 78)
> (Rates <- cbind(Patient, Heart.R))

     Patient  Heart.R
[1,]     102       83
[2,]     105       78

> class(Rates)
[1] "matrix"

> HeartData <- as.data.frame(Rates)
> class(HeartData)
[1] "data.frame"
```

Practical Example

Add the vector gender to the matrix Rates and to the data frame HeartData. What do you observe?

Data frames

Data frames consist of vectors (of equal sizes) which, unlike to matrices, may represent different types of data (heterogeneous). The function data.frame() can generate them.

```
> participant <- c(1:4); group <- c("T", "C", "C", "T")
> age <- c(22, 18, 33, 45); BMI <- c(25, 18, 32, 36)
> (MyData <- data.frame(participant, group, age, BMI))
```

```
   participant  group  age  BMI
1            1      T   22   25
2            2      C   18   18
3            3      C   33   32
4            4      T   45   36
```

Practical Example

Show the structure and the attributes of the created object MyData.

Data frames

- Edit elements, row and/or column names

```
> MyData$participant <- MyData$participant + 100
> MyData
  participant group age BMI
1         101     T  22  25
2         102     C  18  18
3         103     C  33  32
4         104     T  45  36

> colnames(MyData) <- c("Patient", "Treat.", "Age", "BMI")
# Similarly, use rownames() for row names
> MyData[1:2,]

  Patient Treat. Age BMI
1     101      T  22  25
2     102      C  18  18
```

Merging of data frames

```
> merge(MyData, HeartData)
  Patient Treat. Age BMI Heart.R
1     102      C  18  18      83

> merge(MyData, HeartData, all = TRUE)
  Patient Treat. Age BMI Heart.R
1     101      T  22  25      NA
2     102      C  18  18      83
3     103      C  33  32      NA
4     104      T  45  36      NA
5     105     NA  NA  NA      78
```

Practical Example

Merge the data frames `MyData` and `HeartData` together: (a) with the parameter `all.x=TRUE`; (b) with the parameter `all.y=TRUE`. What do you observe ?

Import data sets

Data sets that internally stored with R do not need to be loaded

```
> data(iris); head(iris)
```

- view specific rows, e.g. rows number 7 and 11

```
> iris[c(7,11),]
```

- view specific columns, e.g. columns number 3, 4 and 5

```
> iris[,3:5]
```

To import and/or export data from/to an external file - on your local drive - you need to specify the path to the folder containing/that will contain the file.

- Show current working directory

```
> getwd()
```

- Change working directory

```
> setwd("C:/R-courses/RIntro")
```

Import and export data sets

- List all files in a given path

```
> list.files(getwd())
```

- export data sets to csv or txt file formats can be done using:

```
> write.csv(iris, "Mydata.csv", row.names = FALSE)
> write.table(iris, "Mydata.txt", row.names = FALSE)
```

- import data from csv or txt files:

```
> Im.data1 = read.csv(file = "Mydata.csv", header = TRUE)
> Im.data2 = read.table("Mydata.txt", header = TRUE)
```

- Get more information on data set

```
> attributes(iris); str(iris)
```

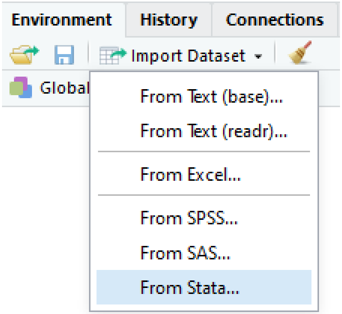
- Show class of each variable

```
> sapply(iris, class); lapply(iris, class)
```

Import and export data sets

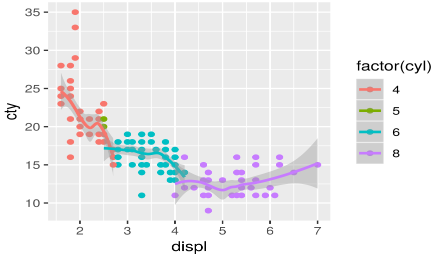
- Import from other data formats using haven function:
  - `> install.packages("haven")`
  - `> library(haven)`

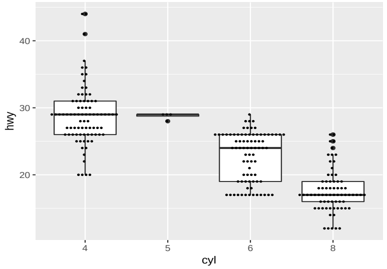
- SPSS:
  - `> read_sav()`
- Excel sheets:
  - `> read_excel()`
- Stata:
  - `> read_stata()`

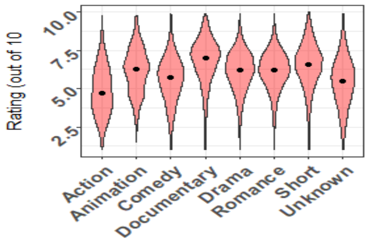


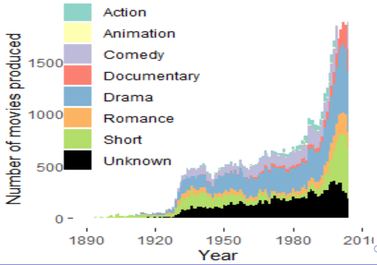
Getting More - High quality plots

ggplot2 allows you to create plots with publication quality



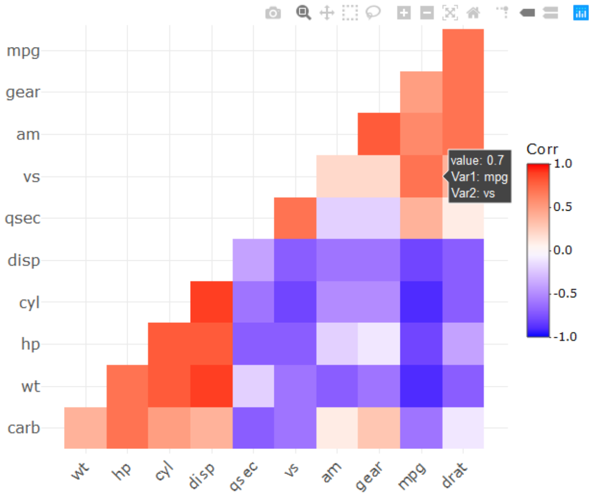






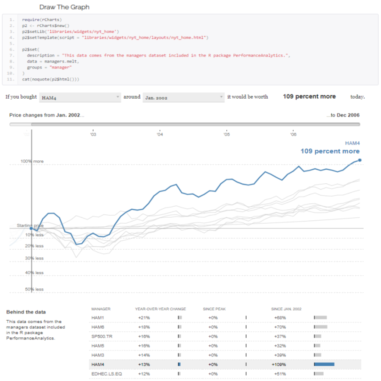
Getting More - Interactive plots

plotly allows you to create interactive plots



Getting More - Dynamic reports

R Markdown allows you to create: dynamic/interactive reports including narrative text, script, and output; dash, handout tutorials; presentations; dashboards; and more.



Draw The Graph

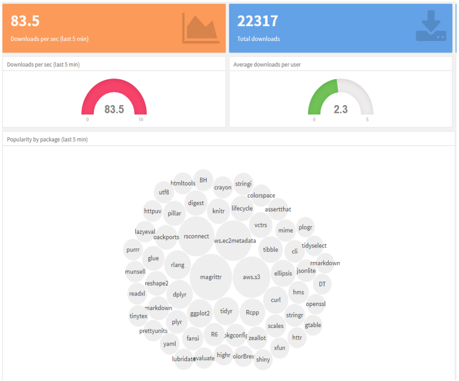
```r
require(rCharts)
p2 <- rCharts$new()
p2$setLib('libraries/widgets/nyt_home')
p2$setTemplate(script = "libraries/widgets/nyt_home/layouts/nyt_home.html")

p2$set(
  description = "This data comes from the managers dataset included in the R package PerformanceAnalytics.",
  data = managers.melt,
  groups = "manager"
)
cat(noquote(p2$html()))
```

If you bought HAM4 around Jan. 2002 it would be worth **109 percent more** today.

Behind the data

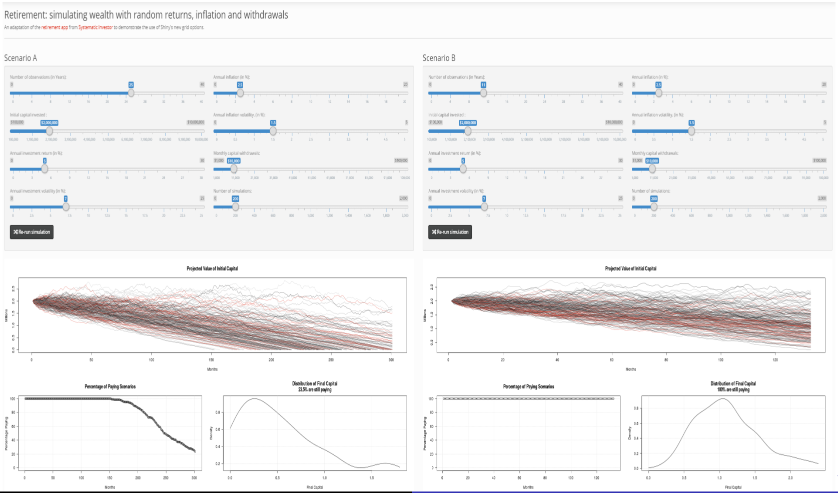
This data comes from the managers dataset included in the R package PerformanceAnalytics.

MANAGER	YEAR-OVER-YEAR CHANGE	SINCE PEAK	SINCE JAN. 2002
HAM1	+21%	+0%	+68%
HAM6	+18%	+0%	+70%
SP500.TR	+16%	+0%	+37%
HAM5	+16%	+0%	+32%
HAM3	+14%	+0%	+39%
HAM4	+13%	+0%	+109%
EDHEC.LS.EQ	+12%	+0%	+51%



Getting More - Web apps

Shiny allows you to create: interactive apps without any knowledge of HTML or Javascript!



Retirement: simulating wealth with random returns, inflation and withdrawals

An adaptation of the retirement app from Systematic Investor to demonstrate the use of Shiny's new grid options.

Scenario A

Number of observations (in Years): 25 (0–40)
Annual inflation (in %): 2.5 (0–20)
Initial capital invested : $2,000,000 ($100,000–$10,000,000)
Annual inflation volatility. (in %): 1.5 (0–5)
Annual investment return (in %): 5 (0–30)
Monthly capital withdrawals: $10,000 ($1,000–$100,000)
Annual investment volatility (in %): 7 (0–25)
Number of simulations: 200 (0–2,000)

Re-run simulation

Scenario B

Number of observations (in Years): 11 (0–40)
Annual inflation (in %): 2.5 (0–20)
Initial capital invested : $2,000,000 ($100,000–$10,000,000)
Annual inflation volatility. (in %): 1.5 (0–5)
Annual investment return (in %): 5 (0–30)
Monthly capital withdrawals: $10,000 ($1,000–$100,000)
Annual investment volatility (in %): 7 (0–25)
Number of simulations: 200 (0–2,000)

Re-run simulation

Projected Value of Initial Capital
Millions
Months
Percentage of Paying Scenarios
Percentage Paying
Distribution of Final Capital
23.5% are still paying
Density
Final Capital
Projected Value of Initial Capital
Percentage of Paying Scenarios
Distribution of Final Capital
100% are still paying

Practical ‘prac_introR’