R Introductory Course

Louis-Claude CANON

FEMTO-ST
Université de Franche-Comté
SMAsHpc

February, 2016
Outline

1. Quick Overview
2. Basics
3. Learning R
Outline

1. Quick Overview
2. Basics
3. Learning R
Data analysis tools

- linear and nonlinear modeling (regression)
- classical statistical tests (e.g., normality test)
- time-series analysis
- classification
- clustering
- principal component analysis

Large collection of user-submitted packages
Comprehensive R Archive Network (CRAN)

Review website: http://crantastic.org/
Quick Overview

Statistical Functions

Data analysis tools
- linear and nonlinear modeling (regression)
- classical statistical tests (e.g., normality test)
- time-series analysis
- classification
- clustering
- principal component analysis

Large collection of user-submitted packages
Comprehensive R Archive Network (CRAN)
Review website: http://cran.tastic.org/
Quick Overview

Graphics (1)

PCA 5 vars
princomp(x = data, cor = cor)

Fertility
Catholic
Examination
Education
Agriculture
(1-3) 60%

Clustering 4 groups

Factor 1 [41%]
Factor 3 [19%]
> qplot(depth, ..density.., data = diamonds, geom = "freqpoly",
+     xlim = c(58, 68), binwidth = 0.1, colour = cut)
Example

```r
> X <- c(3, 4, 5)
> X^2
[1]  9 16 25
> X > 4 & X^2 > 4
[1] FALSE FALSE TRUE
> sqrt(X^2)
[1]  3  4  5
```
Vector (Operations)

Example

```r
> X <- c(3, 4, 5)
> X^2
[1] 9 16 25
> X > 4 & X^2 > 4
[1] FALSE FALSE TRUE
> sqrt(X^2)
[1] 3 4 5
```

Indexes

```r
> X[c(2, 3)]
[1] 4 5
> X[-c(2, 3)]
[1] 3
> X[X > 4]
[1] 5
```
Vector (Speed)

Comparable to Numerical Analysis Software

Most internals are coded in C and Fortran. Benchmarks put it on par with Matlab.
Integrated Help

Manual

> ?length
Outline

1. Quick Overview
2. Basics
3. Learning R
Implementation of the S programming language (started in 1996).
Lexical scoping semantics inspired by Scheme.
Part of the GNU project (GPL).
Objectives: statistical and graphical techniques.
Command line interface & several graphical user interfaces
Assignment

```r
> R <- 4
> R
[1] 4
```
Syntax

Assignment

```r
> R <- 4
> R
[1] 4
```

Functions

```r
> test <- function(X) {
    return(X + 1)
}
> test(R)
[1] 5
```
# Syntax

## Assignment

```r
> R <- 4
> R
[1] 4
```

## Functions

```r
> test <- function(X) {
    return(X + 1)
}
> test(R)
[1] 5
```

## Special values

NA, Inf, NaN  
Related functions: is.na, is.infinite, is.nan
Common structure for storing several basic values

```r
> R <- c(4, 3)
```

Vector

Common structure for storing several basic values

```r
> R <- c(4, 3)
```

Related functions

- sum, max/min, range, mean/var, length, sort/order
- summary, fivenum (give summary of the repartition of the values)
Vector

Common structure for storing several basic values

```r
> R <- c(4, 3)
```

Related functions

- `sum`, `max/min`, `range`, `mean/var`, `length`, `sort/order`
- `summary`, `fivenum` (give summary of the repartition of the values)

Generating functions

```r
> seq(from = 2, to = 3, by = 0.1)
[1] 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 3.0
> 1:6
[1] 1 2 3 4 5 6
> rep(5, 3)
[1] 5 5 5
```
Vector Operations

Vector arithmetic

\[ 3 \times X^2 + \sqrt{Y / Z} + 1 \]
Vector Operations

Vector arithmetic

\[ 3 \times X^2 + \sqrt{Y / Z} + 1 \]

Logical vector

Boolean: TRUE or FALSE

\& vectorized and

| vectorized or

any  test if any element of an array is TRUE

all  test if all elements of an array are TRUE
Vector Operations

Vector arithmetic

\[ 3 \times X^2 + \sqrt{Y / Z} + 1 \]

Logical vector

Boolean: TRUE or FALSE

- & vectorized and
- | vectorized or

- any test if any element of an array is TRUE
- all test if all elements of an array are TRUE

Vector indexes

\[ > (X + 1)[3:5] \]
\[ [1] 4 5 6 \]

\[ > X[\text{is.na}(X)] <- 0 \]
### Basics

#### String Manipulation functions

```r
> paste("X", "Y", sep = "")
[1] "XY"

> substr("abcde", start = 2, stop = 4)
[1] "bcd"
```
Basics

**String Manipulation functions**

```r
> paste("X", "Y", sep = "")
[1] "XY"
> substr("abcde", start = 2, stop = 4)
[1] "bcd"
```

**Vector names**

The function *names* returns the labels of the values of a vector.

```r
> X <- 1:10
> names(X) <- rep("ee", 10)
> X
  ee  ee  ee  ee  ee  ee  ee  ee  ee  ee
  1  2  3  4  5  6  7  8  9 10
```
Data types

**Primitive**

- Numeric (integer and double) and complex
- Character
- Logical
- Factor (nominal value or level)
Data types

**Primitive**
- Numeric (integer and double) and complex
- Character
- Logical
- Factor (nominal value or level)

**Collections**
- Vector
- Array (multi-dimensional, same type)
- List (several elements of any type)
- Data frame (several collections having the same size and any type)
# Data types

## Primitive
- Numeric (integer and double) and complex
- Character
- Logical
- Factor (nominal value or level)

## Collections
- Vector
- Array (multi-dimensional, same type)
- List (several elements of any type)
- Data frame (several collections having the same size and any type)

The functions *attributes*, *class* and *mode* gives information on the data.
Arrays

Basic operations

- **dim** returns the dimensions as a vector (or `nrow` and `ncol`)
- **cbind** combines two elements by columns
- **rbind** combines two elements by rows
- **t** transpose of a matrix
Arrays

Basic operations

- **dim** returns the dimensions as a vector (or nrow and ncol)
- **cbind** combines two elements by columns
- **rbind** combines two elements by rows
- **t** transpose of a matrix

Array indexes (on T, a matrix or data frame)

- **T[2,1]** is the element in the second row and first column
- **T[,1]** is the first column (also T[,-(2:ncol(T))])
- **T[,-1]** is T without the first column
- **T[-(2:3),]** is T without the second and third rows
## Arrays

### Basic operations

- `dim` returns the dimensions as a vector (or `nrow` and `ncol`)
- `cbind` combines two elements by columns
- `rbind` combines two elements by rows
- `t` transpose of a matrix

### Array indexes (on T, a matrix or data frame)

- `T[2,1]` is the element in the second row and first column
- `T[,1]` is the first column (also `T[,-(2:ncol(T))]`)
- `T[-1]` is `T` without the first column
- `T[-(2:3),]` is `T` without the second and third rows

### Matrix multiplication

`A %*% B`
**read.table function**

Read data (numeric and character) put by column in a file. Many parameters: separator between fields (`sep`), number of lines to skip (`skip`), number of lines to read (`nrows`), character comment (`comment.char`), ...
**read.table function**

Read data (numeric and character) put by column in a file. Many parameters: separator between fields (sep), number of lines to skip (skip), number of lines to read (nrows), character comment (comment.char), ...

**Example**

```r
> read.table("data.txt")
V1 V2
1 r 1
2 a 2
```
Example

```r
> X <- 1:100
> Y <- sqrt(X)
> plot(X, Y)
```
Flow Control

Condition

```r
if (all(X > 5) && i == 1) {
    # Code
}
```
Flow Control

Condition

```r
if (all(X > 5) && i == 1) {
    # Code
}
```

Loop

```r
for (i in 1:10) {
    # Code
}
```

Better to use vectorized operations and pre-compiled routines, otherwise everything is interpreted and slow.
Outline

1. Quick Overview
2. Basics
3. Learning R
Documentation within the environment

Search within manual pages: "??" before any string.
Start the name of a function and enter tab.
Familiarity with a Programming Language

Documentation within the environment

Search within manual pages: "??" before any string.
Start the name of a function and enter tab.

Practical session

Many great exercises!
swirl package
Long tutorial explaining everything.
Familiarity with R

ggplot2 and dplyr

Take a look to modern features of R.