The R programming language

Short course

Modelling for sustainable management of crop health

Part 1. Monday 13th January 2014

Part 2. Tuesday 14th January 2014

François Brun (ACTA)

First,
Installation of the material for the course on your computer

a thumb (USB) drive

- With a directory « CourseSmach » containing all what we will need for the courses.

  – Copy it to D:
  – Or to C: or elsewhere (you will need to adapt the path in the scripts)

Installation of R

- In D:\CourseSmach\software
  - R-3.0.2-win.exe for windows
  - R-3.0.2.pkg for Mac
- Install with default settings
- Optional : R studio (a popular R editor)

Objectives of this initiation with R

- **Objectives:** To familiarize yourself with the R language and the software to follow the course on modeling. Basic operation, to handle and represent data, programming.
- **For beginner**
- **To learn and practice:**
  - follow the presentation
  - practice on your computer
    - Type the commands
    - use an existing script

Program

- **Part 1. Monday 13th**
  - **1. Introduction - 0.25h**
  - **2. Types of data and objects - 0.5h**
  - **3. Other structures of data (matrix, data.frame, list,…) - 0.5h**
  - **4. load external data - 0.25h**

  Change of pace lecture : M. Pautasso & Break
  - **5. Graphs under R and parameters graphic - 1h**
  - **Practical work 1. Brown rust data -1h**

- **Part 2. Tuesday**
  - **6. Basic statistics with R - 0.5h**
  - **7. Safeguards of script, graphs, data and results of analysis - 0.25h**
  - **8. Programming with R (function, loop, conditions,…) - 0.5h**
  - **9. Complements: to install a package, helps, given missing - 0.25h**
  - **Practical work 2. Sum of temperature function - 1h**

- **Practical work 3. SEIR model of Zadoks (Tuesday afternoon)**

1. Introduction
Use of R like a calculator. Creation of the your first script.

What is what R?

- environment/system of statistical analysis
  - a programming language AND a software
  - distributed freely under the Public GNU Licence
  - R Development Core TEAM dialect of the language S (software S-PLUS)
- a simple language of very high level
  - Graphic
  - statistical analyses
  - matrix algebra
  - But interpreted (a little slow…)

Where does we find R?

http://www.r-project.org/

- exe of installation of the software (current version: 3.0.1)
- zip of the complements (=package) to software
- pdf of documentation
- FAQ
- Web links

Use of USB thundrive

- Content:

USB:/courseSMACH

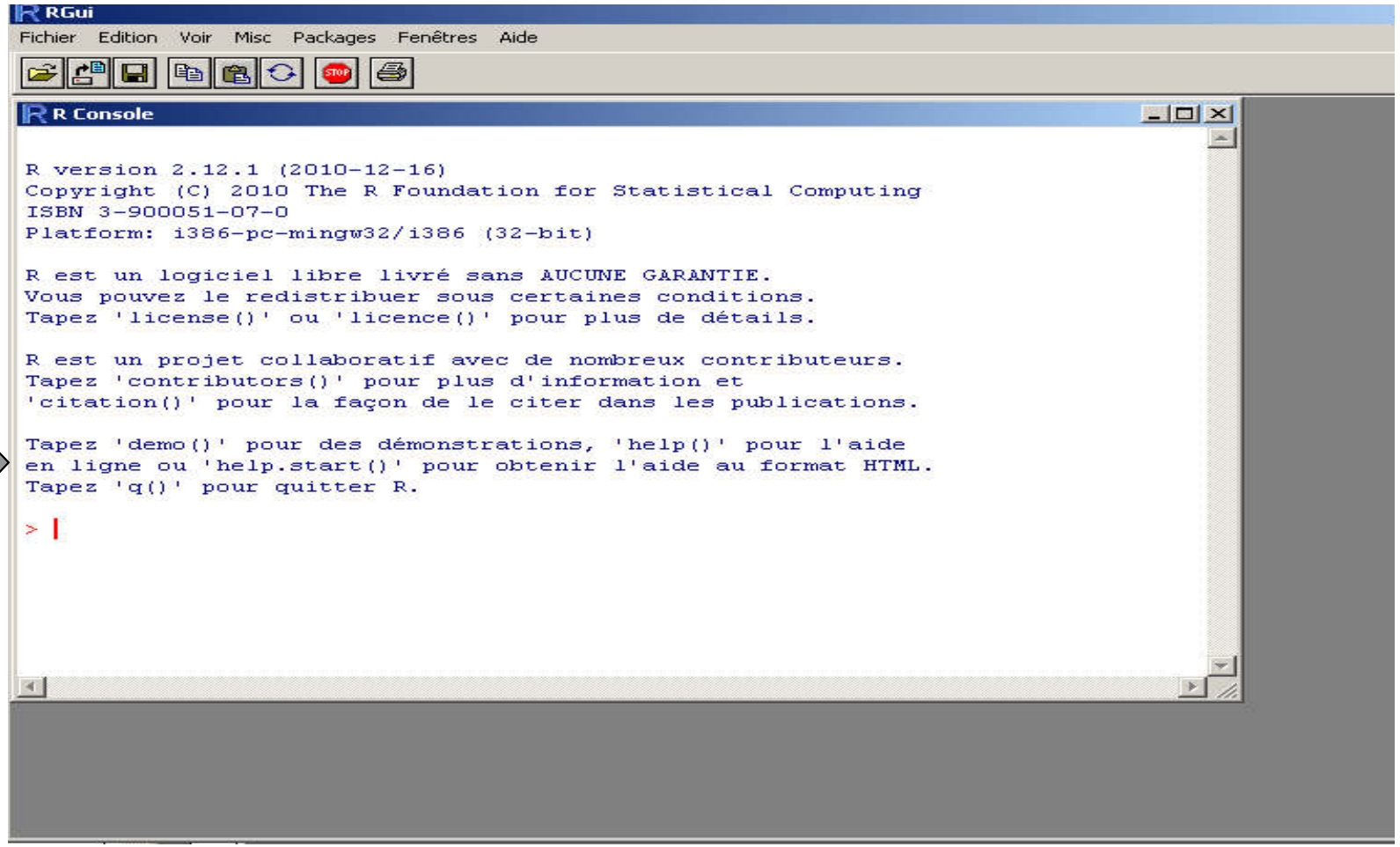
… or zip courseSMACH.zip

To copy (if possible) in

D:/courseSMACH/

(or in D:/courseSMACH or where you can, you will need to adapt some script for the path then)

Open R software



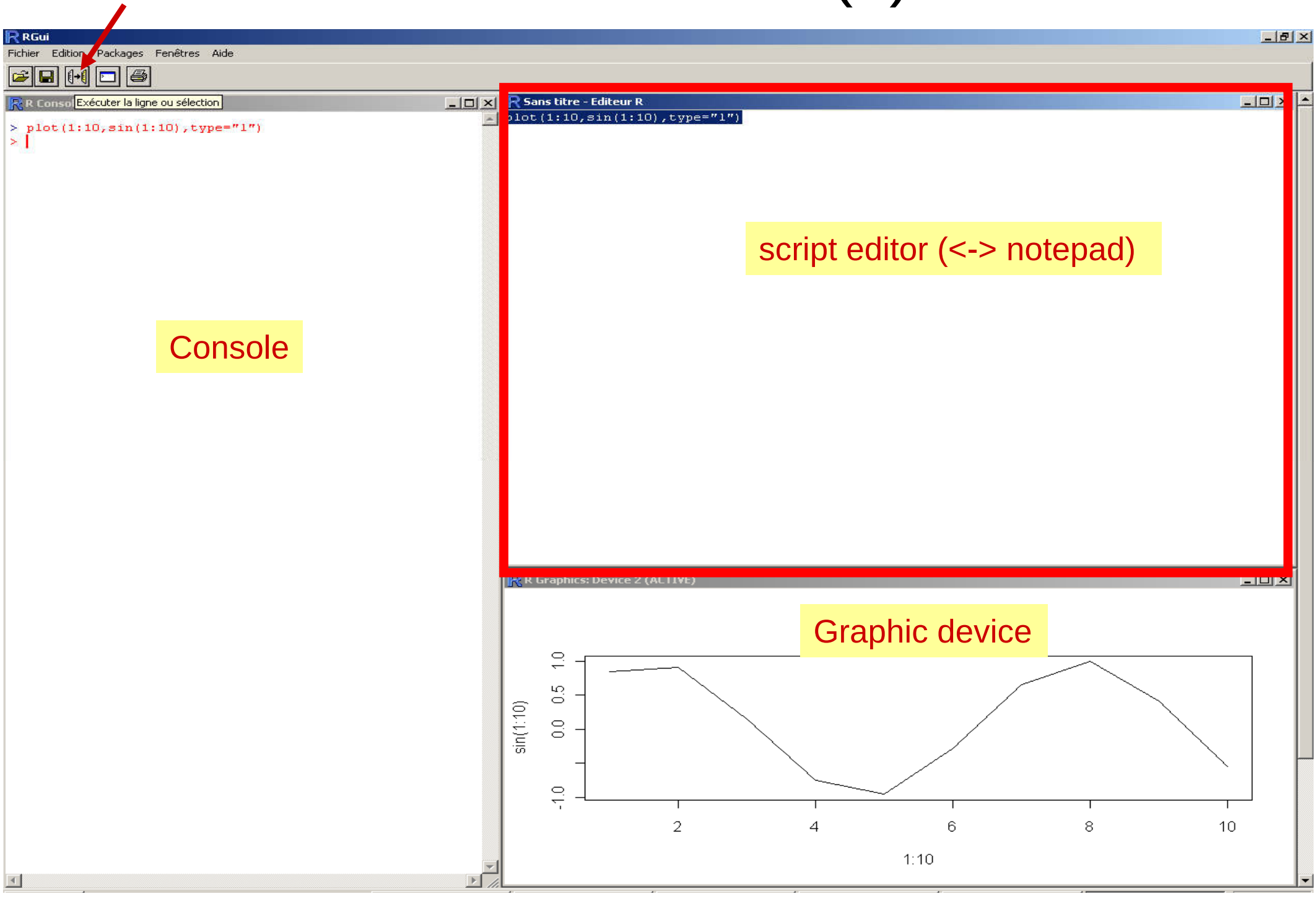
- Each line of code is typed (take attention to the syntax!)
- One types the lines of code (a script) in an editor (R one = notepad, Rstudio, tinnR,…) then lines of code are copied-pasted into R consol.

User interface (1)

- Menu and icons for the management of files, of windows, …

- Three sub-windows
  - R console: type and execute your code, print the results
  - Text editor : useful to create scripts (possibility of using an external editor)
  - graphics

User interface (2)



Code color for the course

- In red: what is to be typed in the console (after « > »)
- In blue: the printed result

Ex :

```
1+5
[1] 6
```

Look at the help

?seq # open an help windows (using your web navigator, but you do not need internet connection)

help(seq)

alternative forms

arguments

value

examples

Use as a calculator

```
7+12
[1] 19

6-8
[1] -2

3*5
[1] 15

4/2
[1] 2

2^6
[1] 64
```

```
sqrt(15.7)
[1] 3.962323

exp(4.6)
[1] 99.48432

log(10)
[1] 2.302585
```

Operator <- or =

- “Object oriented” language
  - variables, data, matrices, functions, results,… **are stored** in the RAM memory of the computer in the form of **objects** which have a **name**

```
x<-  4    # ou x=4 # ou 4->x
x
[1] 4
X        # sensitive to case (lower case / capital) !
Error : objet 'X' not found

y<-  7
x+y
[1] 11
```

-> is equivalent to <- but not to =

Print results

- On screen (into the console windows)

```
print(pi)  # ou pi
[1] 3.141593

options(digits=2)
print(pi)
[1] 3.1
```

R objects

```
objects()   or  ls()
[1] "x" "y"

rm(x)  # it removes x
```

My first script

- In the editor, write the 3 lines:

```
# my first script
x = 1:10
plot(x,x^2)
```

- Save it to a file « file > save to file »

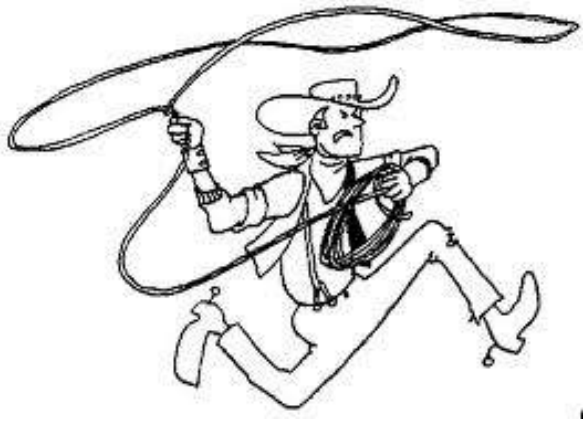
- Select the whole script, and click on the icon “send the line or the selection”

2. Types of data and objects

Handle simple vectors and operations on these vectors (assignment, selection).

Types of storages of objects

numeric, character, factor, ordered, logical



1,2,3, …

"T"

"a", "b", …

"a" < "b" < "c" < …

F

2012

TRUE

Main objects

- **vector**
- **matrix**
- **list** of objects
- table of data : **data.frame**
- constant

numerical vector

- a collection of values
- missing value : **NA** (“not available”)
- set up with function **c()** or **seq()**

```
c(10,6,5.7,1)

x<- c(10,6,5.7,1)
x

x= c(10,6,5.7,1)
x

y<- 1:10
y<- seq(1,10, by=1)
y
```

boolean vector

- To handle logical quantities
- Elements: TRUE (T), FALSE (F) or NA
- Bolean operation: **&** : and, **|** : or, ! : not

```
y<- x<- 2

w<- x>5
w
[1] FALSE

z1<- y==3
z1
[1] FALSE

z2<- y>=1
z2
[1] TRUE
```

```
w<- T
w2<- c(T,F)
```

Character vector

- String vector
- Each element is specified with " or ‘

```
x<- c("I","like",“Volterra","very","much")
x
[1] "I"    "like"    "Volterra"   "very"    "much"
```

Manipulation of vectors

- subscript

```
y<- c(1,2,5,10,100,200,500)
y[2]
y[2:6]
y[-1]
```

- Alteration

```
y[2]<- 100
y
y[y>=100]<- 2
y
```

- Arithmetic

```
z<- 2*y
y2<- y+z
y2
```

Operations on vectors

```
x= c(1, 5, 8, -2, 7)
x
[1]  1  5  8 -2  7
sqrt(x)
log(x)
abs(x)
exp(x)
sin(x)

length(x)
[1] 5
# count the number of elements ≥5 in x
length(x[x>=5])  # or: sum(x>=5)
[1] 3
```

Basic statistical operations

```
mean(x)
sum(x)
var(x)
sd(x)
median(x)
range(x)
min(x)
max(x)
quantile(x, probs=seq(0,1, by=0.1))
```

counting

```
table(x)
```

Sorting

```
x<- c(8, 1, -2, 7, 5)
sort(x)
[1] -2  1  5  7  8

sort(x, decreasing=TRUE)
```

Caution: R tries to make things work !

- It works (but it might be an error…)

```
vector4<- c(2,7,3,4)
vector2<- c(3,6)
vector4+vector2
[1]  5 13  6 10
```

- It works too … but with a warning message…

```
vector5<- c(2,7,3,4,8)
vector5+vector2
[1]  5 13  6 10 11
Warning message:
In vector5 + vector2 :
  longer object length is not a multiple of
  shorter object length
```

3. Other structures of data (matrix, data.frame, list,…)

Creation and handling of the structures.

Matrix

- A two-dimensional array
- An unique type of data
- matrix()

```
M1 <- matrix(0, nrow=2, ncol=3)
M1
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
```

Table of data : data.frame

- It looks like a matrix, but columns can be of different types
- columns are named
- Very common and usefull for data such as experimental data for example.

```
City<- c("Brussels", "Volterra", "Paris")
Rank<- c(13,9,1)
Weather<- c("Rainy", "Sunny", NA)

TAB<- data.frame(City, Rank, Weather)
TAB
       City Rank Weather
1 Brussels   13   Rainy
2 Volterra    9   Sunny
3    Paris    1    <NA>
TAB$Rank  # ou TAB[,2]  ou TAB[,"Rank"]
[1] 13  9  1
```

data.frame

```
TAB[TAB$City=="Paris", ]

   City Rank Weather
3 Paris    1    <NA>


TAB[2,]
          City Rank Weather
2 Volterra    9   Sunny
```

data.frame

```
class(TAB)      [1] "data.frame"

ncol(TAB)     [1] 3

nrow(TAB)     [1] 3

summary(TAB)
                City          Rank          Weather
           Lille      :1   Min.    : 1.000   Rainy:1
           Volterra:1      1st Qu.: 5.000   Sunny:1
           Paris      :1   Median : 9.000   NA's :1
                           Mean    : 7.667
                           3rd Qu.:11.000
                           Max.    :13.000

head(TAB,2)
                 City Rank Weather
           1       Lille   13   Rainy
           2 Volterra       9   Sunny
```

data.frame : sort data

```
TAB[order(TAB$Rank), ]
```

```
        City Rank Weather
3      Paris    1    <NA>
2   Volterra    9   Sunny
1      Lille   13   Rainy
```

List

- An ordered collection of objects (types can be different)
- Useful to store results of calculation
- list() to create
- elements subscripted using **[[…]] (double bracket)**

```
MyList<-list(FALSE,7,M1, TAB)
MyList
[[1]]
[1] FALSE

[[2]]
[1] 7

[[3]]
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0

[[4]]
         City Rank Weather
1       Lille   13   Rainy
2    Volterra    9   Sunny
3       Paris    1    <NA>
```

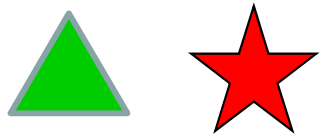
```
MyList[[3]]
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
```

4. Access to external data. Read data from files.

Read external data

- R reads text files, but there are other possibilities…
- For Excel® files:
  - solution 1. save the sheets (one by one) under a text file (TXT, CSV,…)
  - solution 2. use a specific package (xlsx)
- If you use a database system
  - you can direct write your query in SQL and use library to connect to the database.

Example, read file



- **read.table** (or variant : read.csv2 or read.delim)
- With specific options

Ex : 01_read.file.r

- Or directly xlsx file (Excel®) (2nd part of script)

Change of pace lecture
Break

5. Graphs under R and parameters graphic

graphisme

- Many types of possible graphs
- basic Functions: plot, lines, points, hist, barplot

- Functions of personalization

  par(mfrow=c(.,.))
- Packages with advanced functions

- A first graph :

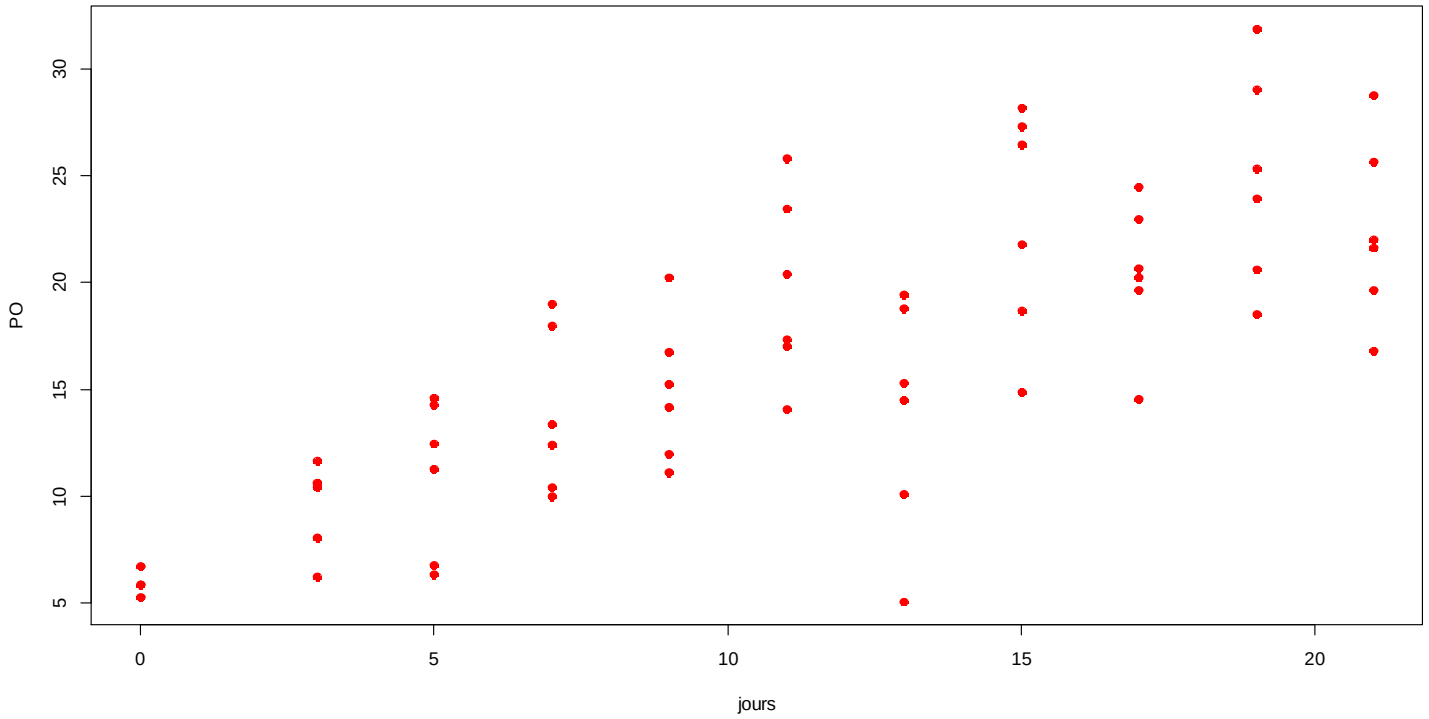
```
x=c(0,1,2,3,4,5)
y=x^2
plot(x, y)
```

Scatter plot

Ex : 03_graphics.r

PO versus day for v1

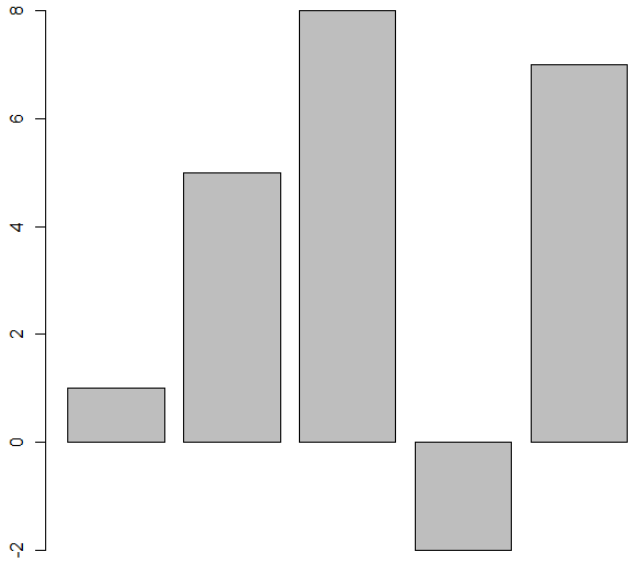
```
df1 <- statenz[statenz$cultivar=="v1", c("PO", « days")]
plot(PO~days, df1, pch=19, col="red")
```



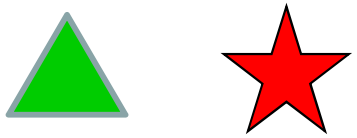
Other graphs

Barplot. Each value represented by a bar

```
x <- c(1,5,8,-2,7)
barplot(x)
```



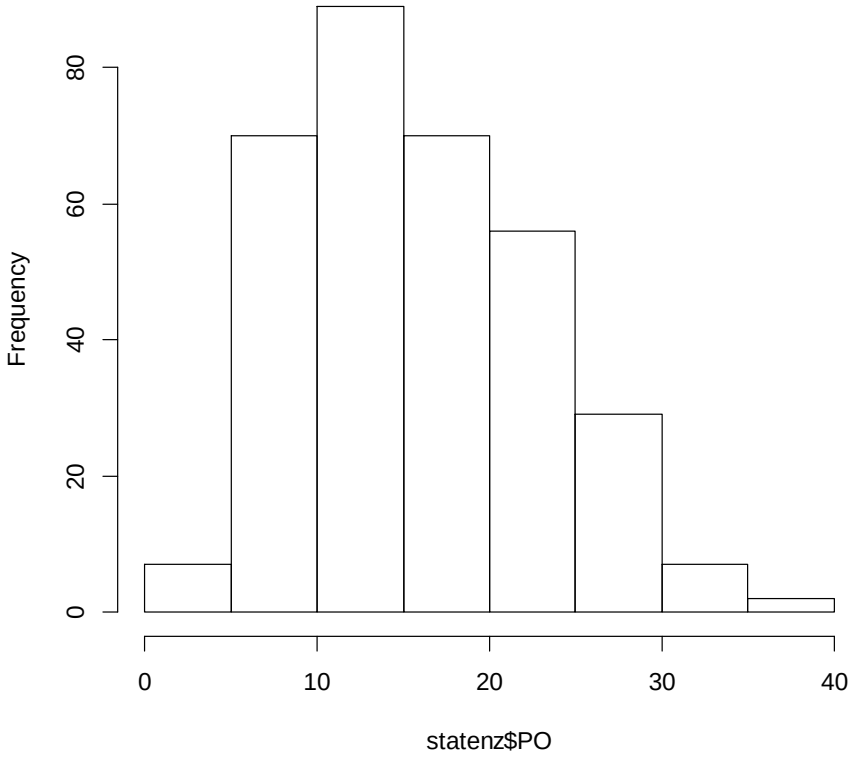
Other graphs



Histogram. Each category represented by a bar according to the frequency

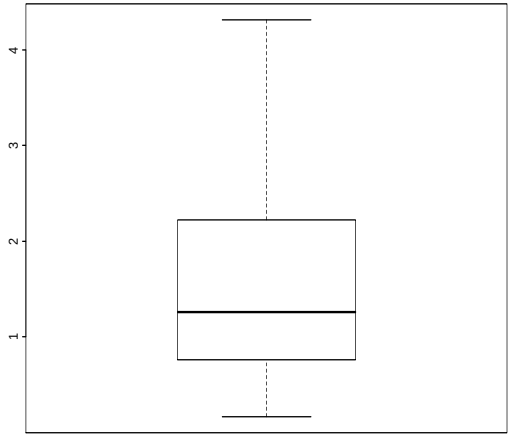
```
hist(statenz$PO, xlab="statenz$PO", main=" ")
```

Ex : 03_graphics.r

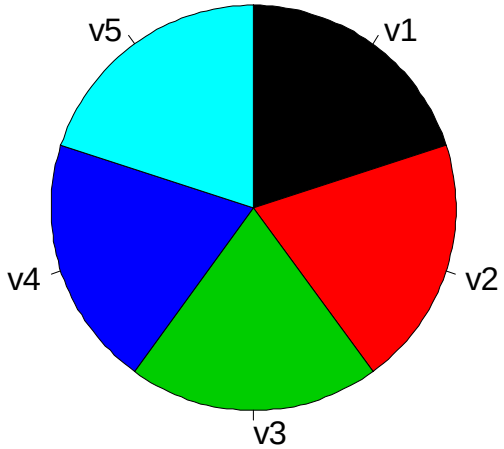


Other graphs

```
boxplot(statenz$GST)
```

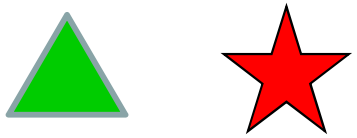



```
pie(table(statenz$cultivar), col=1:5, clockwise=TRUE, cex=2)
```



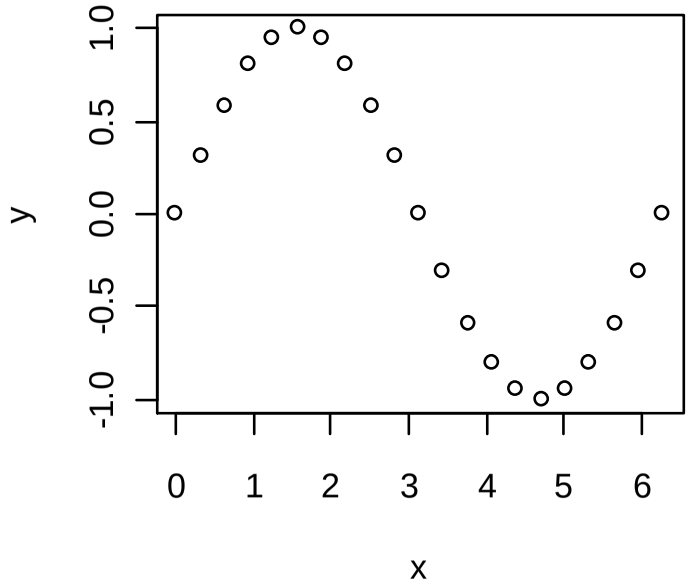
personalization

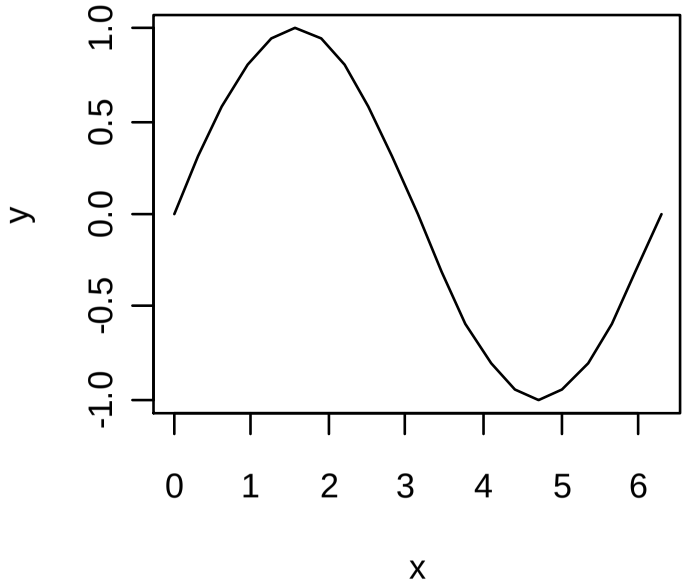
- You can personalize the graphs with **par()** or in **the graphic functions** themselves…
- several graph on a page **mfrow=c(2, 2)**
- margins: **mar=c(bottom, left, signal, right)**
- size of the characters and symbols: **cex=1**
- Log scale : **xlog**
- …. See **help (par)**.

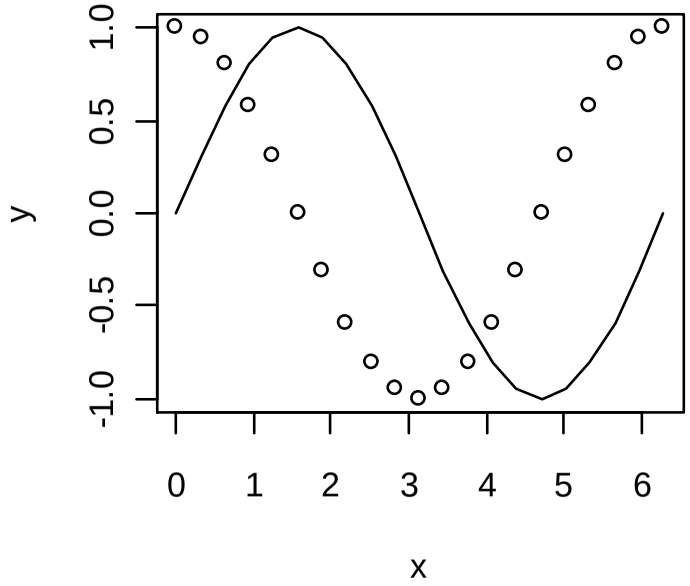


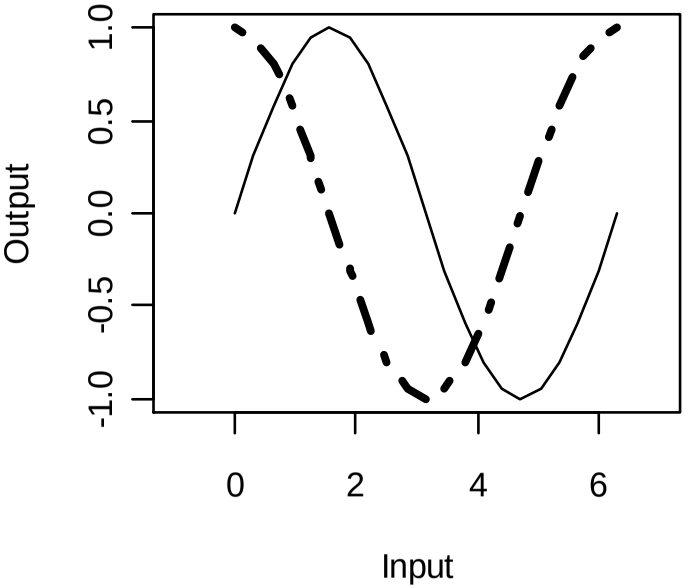
Ex : 03_graphics.r

```
par(mfrow=c(2,2))

x<- seq(0,2,by=0.1)*pi
y<- sin(x)
z<- cos(x)

plot(x,y)
plot(x,y,type="l")
plot(x,y,type="l")
   points(x,z)
plot(x,y,type="l",xlab="Input", ylab="Output",
xlim=c(-1,7))
   lines(x,z,lwd=3,lty=4)
```


y
1.0
0.5
0.0
-0.5
-1.0
0
1
2
3
4
5
6
x


y
1.0
0.5
0.0
-0.5
-1.0
0
1
2
3
4
5
6
x


y
1.0
0.5
0.0
-0.5
-1.0
0
1
2
3
4
5
6
x


Output
1.0
0.5
0.0
-0.5
-1.0
0
2
4
6
Input

Program

- **Part 1. Monday 13th**
  - **1. Introduction - 0.25h**
  - **2. Types of data and objects - 0.5h**
  - **3. Other structures of data (matrix, data.frame, list,…) - 0.5h**
  - **4. To see external data - 0.25h**
  - **5. Graphs under R and parameters graphic - 1h**
  - **Practical work 1. Brown rust data -1h**

- **Part 2. Tuesday**
  - **6. Basic statistics with R - 0.5h**
  - **7. Safeguards of script, graphs, data and results of analysis - 0.5h**
  - **8. Programming with R (function, loop, conditions,…) - 0.5h**
  - **9. Complements: to install a package, helps, given missing - 0.5h**
  - **Practical work 2. Sum of temperature function - 1h30**

- **Practical work 3. SEIR model of Zadoks (Tuesday afternoon)**

Practical Work 1. Load a file and explore a data set on Brown Rust.

See booklet

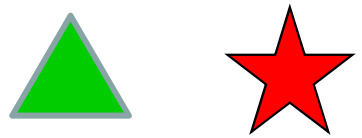
End of day 1 (first part on R)

6. Statistics functions

ANOVA et regression

- Many R functions are available for ANOVA and regression
- Linear regression: lm()
- ANOVA: aov()
- Generalized linear model: glm()
- Mixed-effect model: lme()
- Non linear regression: nls()

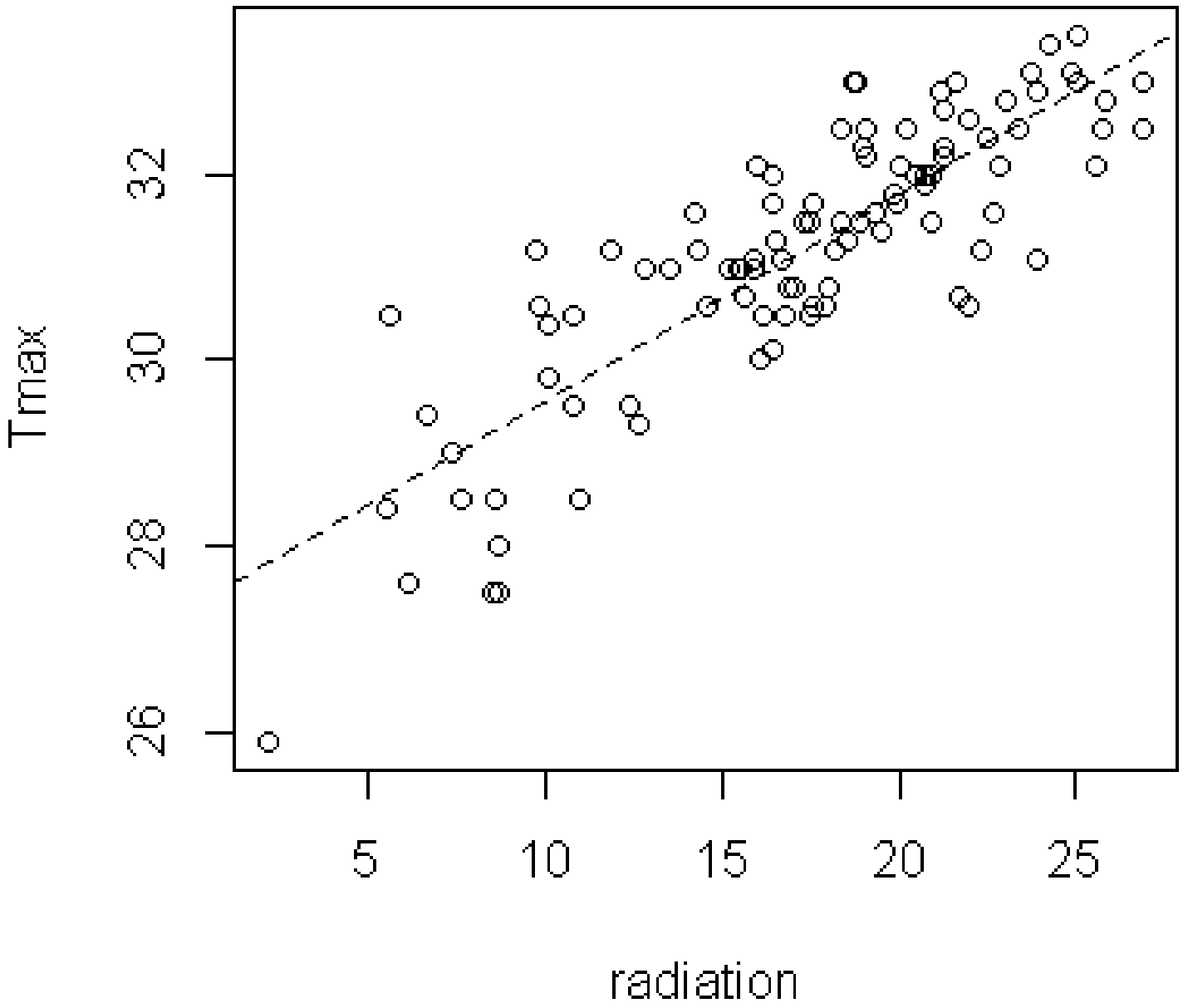
Model of simple linear regression



Ex : 04_stat.r

```
y<- weather$Tmax
x<- weather$Radiation

Fit<- lm(y~x)
print(Fit)
summary(Fit)
COEF <- Fit$coefficients

plot(x,y)
abline(a=COEF[1], b=COEF[2],lty=2)
```

```
> print(Fit)
Call:
lm(formula = weather$Tmax ~ weather$Radiation)

Coefficients:
      (Intercept)  weather$Radiation
          27.2527            0.2277

> summary(Fit)
Call:
lm(formula = weather$Tmax ~ weather$Radiation)

Residuals:
     Min       1Q   Median       3Q      Max
-1.85269 -0.54191  0.03522  0.52091  1.96859

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      27.25268    0.27787   98.08   <2e-16 ***
weather$Radiation  0.22769    0.01543   14.75   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8126 on 88 degrees of freedom
Multiple R-squared: 0.7121,     Adjusted R-squared: 0.7088
F-statistic: 217.6 on 1 and 88 DF,  p-value: < 2.2e-16
```


Tmax
26
28
30
32
5
10
15
20
25
radiation

7. Save scripts, graphs, data and results of analysis

Save a script

File, save to….

Save a graphics

Copy-paste to word, powerpoint, …

Or to a file

```
png("filename.png")
  barplot(1:5)
dev.off()
```

See help(png)

Export data from R

Create a file from R with write.table instruction; The exact path may be specified

```
write.table(iris, file="iris.txt", row.names=F, sep="\t")
```

Save result

Copy-paste form the console to any editor…

Or save to a file…

```
sink("synthese.txt")
     summary(weather)
sink()
```

Save all (or almost all)

All the on going work (= all the variable of the session)
Menu > File > Save workspace

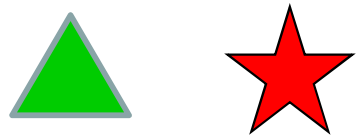
```
save.image("my_file.RData")
```

To load an existing workspace

```
load("my_file.RData")
```

8. Programming with R (function, loop, conditions,...)

loops



- To repeat a set of instructions
- different ways: **for**, while, repeat, apply

Ex : 06_prog.r

```
X<- matrix(seq(1,60,by=1), nrow=10, ncol=6)
X
Y<- rep(NA, 10)

for (i in 1:10) {
  Y[i]<- sum(X[i,]) }
Y

apply(X, 1, sum)
apply(X, 2, sum)
```

Performance : avoid loops … when possible

- Often, we can take advantage of the vector and matrix structure

- Or use **apply** instruction (that are more efficient)…

Conditional instructions


- Execute instructions depending on conditions
- Syntax: **if** (condition) expression_1 **else** expression_2

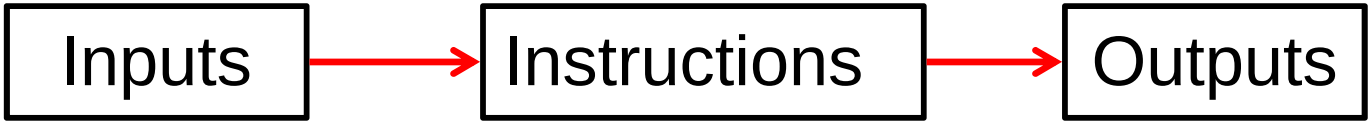
```
X=4
cond <- x>5
if (cond) print("OK") else print(“not OK")
#or
if (x>5) print("OK") else print(“not OK")
```

```
TMIN <- weather$Tmin
for (i in 1:length(TMIN)) {
 if (TMIN[i] < 24) TMIN[i]=0
}  # Correct
#but prefer this more efficient way :
TMIN[TMIN < 24] <- 0
```

Ex : 06_prog.r

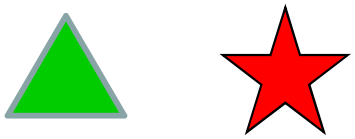
function (definition)

- A R user can create his own R function containing a set of instructions useful to a specific calculation.
- A function is characterized by it inputs (optional), the set of instructions, and it output (optional too).




```
functionName<- function(paramètres/arguments)  {
   commandes
   return(valeur/résultat de la fonction)
}
```

function



```
MeanTemp<- function(Tmin, Tmax) {
      Tmean<-(Tmin+Tmax)/2
      return(Tmean)
}

MeanTemp(weather$Tmin, weather$Tmax)
```

Ex : 06_prog.r

```
 [1] 28.60 28.70 28.20 27.70 27.05 27.55 27.90 27.95 27.85 27.25 26.05 27.15
[13] 27.95 28.15 27.70 27.65 28.05 27.60 27.85 28.60 28.45 28.00 27.75 27.95
[25] 26.00 27.30 27.00 27.30 29.00 28.15 28.00 28.00 27.45 29.05 29.00 29.15
[37] 28.25 28.15 28.30 27.90 27.60 27.45 27.75 28.40 28.85 27.75 24.85 25.95
[49] 26.45 27.10 26.25 26.25 26.90 27.40 28.05 29.80 29.65 28.15 27.50 27.95
[61] 28.05 27.45 27.25 27.40 28.55 28.65 28.85 28.30 27.50 26.80 27.95 28.90
[73] 28.00 28.25 27.50 27.30 28.40 27.85 27.75 25.50 27.25 26.80 27.15 26.00
[85] 26.75 26.80 25.85 27.40 27.05 27.90 27.70 28.20 28.00 27.20 27.95 27.85
[97] 26.95 27.90 28.35
```

9. install a package
ZeBook
sensitivity
triangle

package

- R packages contain supplementary functions, data and documentation
- R basic software includes about 30 packages
- there are over 4000 additional packages available.
- each package dedicated to some particular type of method
- for example the sensitivity package for sensitivity analysis
- a list of packages at http://www.r-project.org on page *CRAN*.
- # to see all packages installed

```
library()
```

Examples of useful package

graphics : lattice, plotrix, maptools, ggplot2

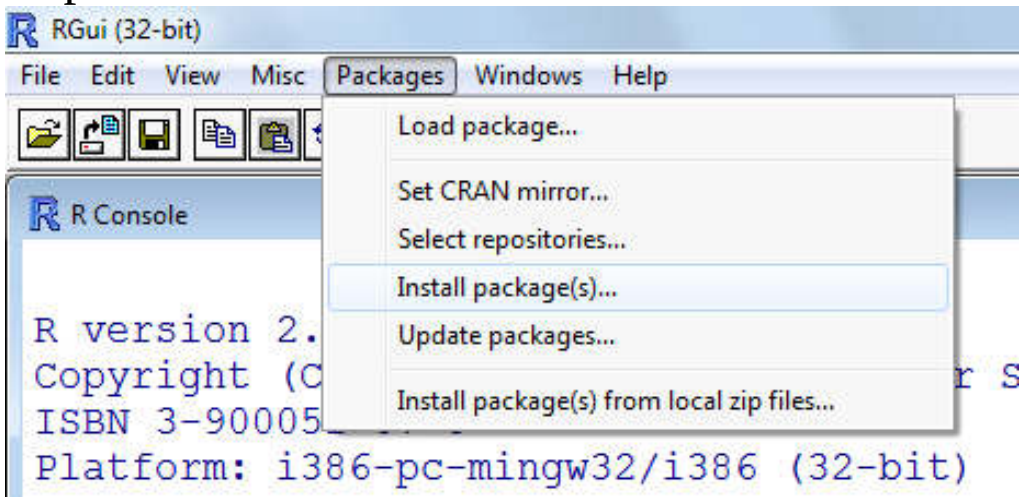
Geographical information system : maps, maptools, shapefiles

Statistic analysis of experiment : agricolae

Multidimensional analysis : FactoMineR, rpart, randomForest

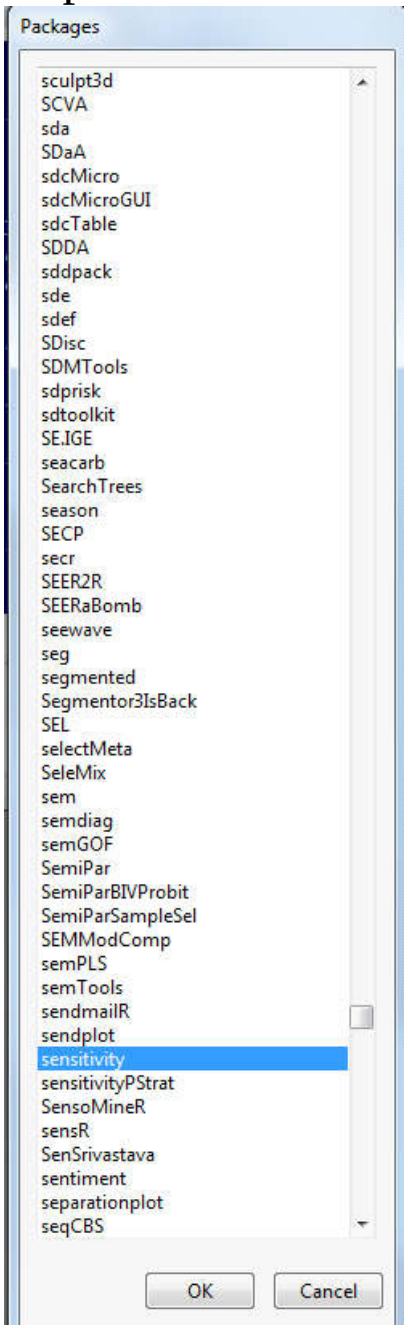
Installation of a package : ZeBook

step 1



step 2

step 3



step 4

```
> utils:::menuInstallPkgs()
--- Please select a CRAN mirror for use in this session ---
Error in contrib.url(repos, type) :
  trying to use CRAN without setting a mirror
> utils:::menuInstallPkgs()
--- Please select a CRAN mirror for use in this session ---
trying URL 'http://cran.cict.fr/bin/windows/contrib/2.15/sensitivity_1.4-1.zip'
Content type 'application/zip' length 92984 bytes (90 Kb)
opened URL
downloaded 90 Kb

package 'sensitivity' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\brun\AppData\Local\Temp\Rtmp6XMjtd\downloaded_packages
> |
```

Use a package

- load a package (already installed)

```
library(ZeBook)
```

- show documentation on a package (already installed)

```
library(help=ZeBook)
```

or

```
help(ZeBook)
```

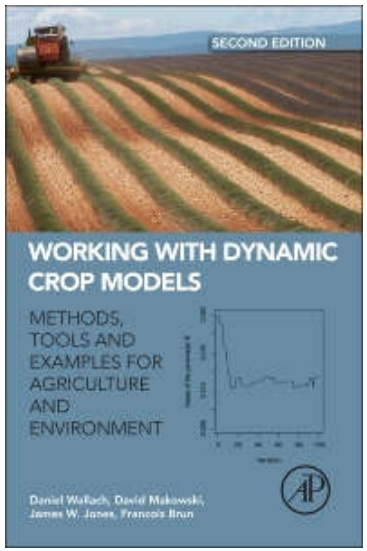
- Then, use an embedded function…

More information and documentation

- Many references
- Venables W.N., Smith D.M. and the R Development Core Team 2010. An introduction to R (available online)
- Chapter 3 in

Introduction to R,

oriented for modeling



- http://cran.r-project.org/manuals.html
- …. practice and practice….

help

?name_fonction
??name_fonction

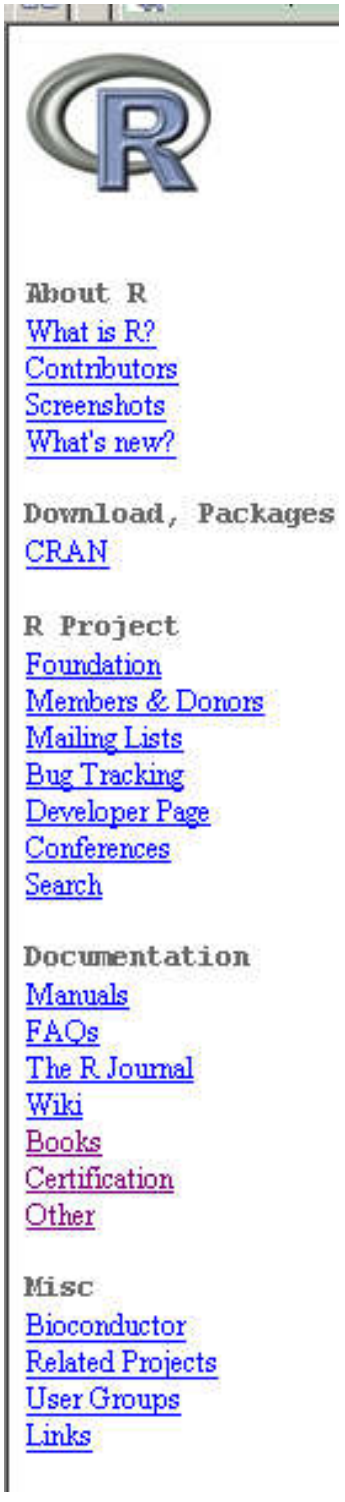
R site search

http://finzi.psych.upenn.edu/search.html

http://r-project.markmail.org/search/

Forums

http://forums.cirad.fr/logiciel-R/



Useful links ….

http://www.r-project.org/

http://finzi.psych.upenn.edu/search.html

http://pbil.univ-lyon1.fr/R/enseignement.html

http://forums.cirad.fr/logiciel-R/

http://www.oga-lab.net/RGM2/images.php?show=all&pageID=299

http://dirk.eddelbuettel.com/cranberries/

http://r-project.markmail.org/search/

http://research.stowers-institute.org/efg/R/Color/Chart/

http://www.springer.com/series/6991

http://www.crcpress.com/ecommerce_product/browse_book_categories.jsf?category=STA