

Introduction to R: Statistical Tools for Biologists

Petr Nazarov

petr.nazarov@crp-sante.lu

10-09-2009

◆ General introduction and information package

◆ Data manipulation and visualization

- ◆ basic operations
- ◆ import and export
- ◆ visualization

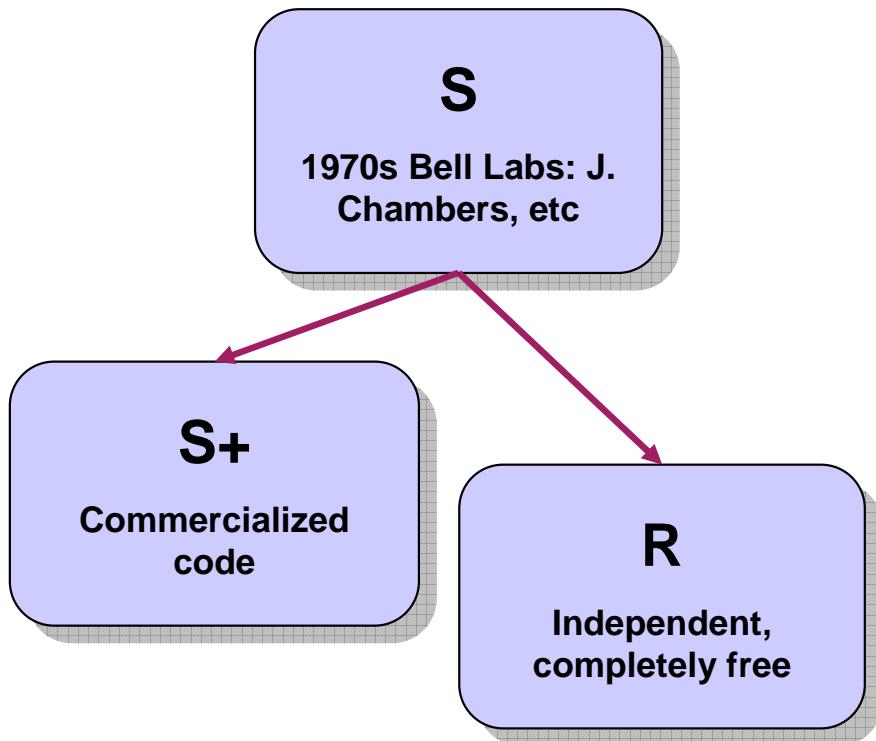
◆ Simple statistical analysis

- ◆ descriptive statistics
- ◆ statistical tests
- ◆ simple ANOVA (to be continued by Thomas)

◆ Demonstration of Bioconductor

1. GENERAL INTRODUCTION

History



R was created by **Ross Ihaka** and **Robert Gentleman** at the University of Auckland, New Zealand, and is now developed by the **R Development Core Team**.

Positive Features

- ◆ Scripting language of high level
- ◆ Interactive work with data (similar to MATLAB)
- ◆ Works under Windows, Linux, Mac OS
- ◆ Free code + verified algorithms
- ◆ Fast developing: new version each 2 month
- ◆ Extremely wide application: from biology to theoretical physics and computer sciences
- ◆ Very good information support
- ◆ Fast installation (with some exclusions)

Negative

- ◆ Not memory efficient (comparing to C)
- ◆ Slower than C/C++
- ◆ No advanced built-in GUI development tools

2. INFORMATION PACKAGE

Main Web-page:

cran.r-project.org

cran.r-project.org/manuals.html

cran.r-project.org/web/packages/

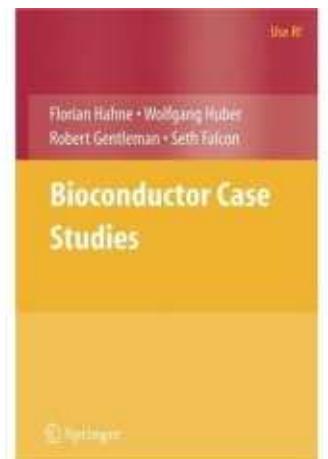
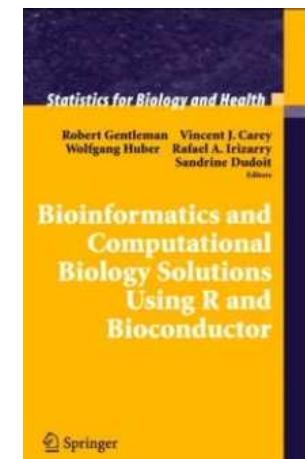
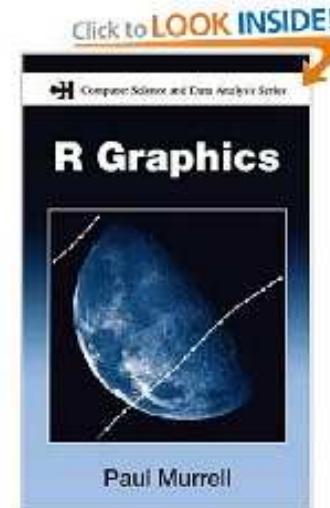
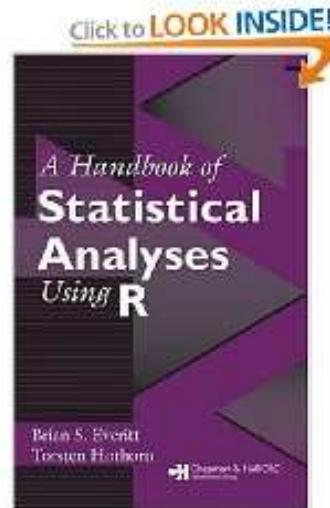
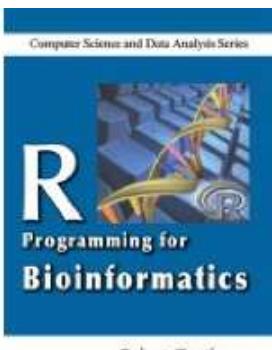
cran.r-project.org/other-docs.html

R-Project Seek Engine:

www.rseek.org

R/Bioconductor

www.bioconductor.org

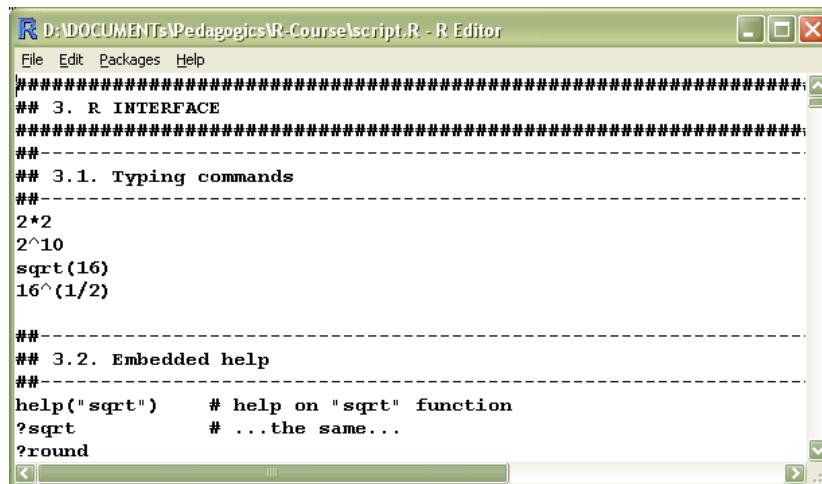


2. INFORMATION PACKAGE

All the materials and data are available here

<http://edu.sablab.net/r>

Built-in Script Editor



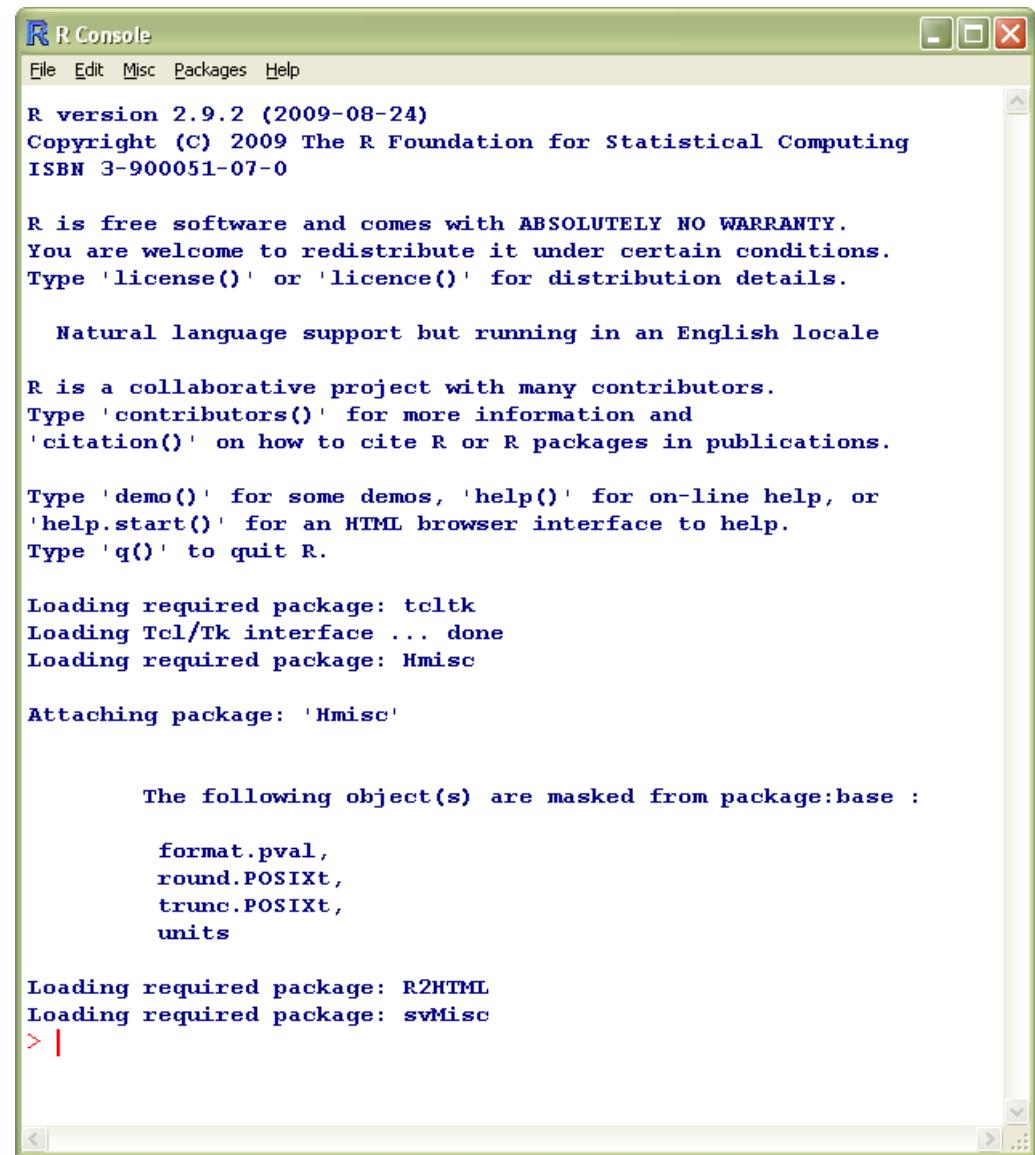
```

R D:\DOCUMENTS\Pedagogics\R-Courses\script.R - R Editor
File Edit Packages Help
#####
## 3. R INTERFACE
#####
##-----#
## 3.1. Typing commands
##-----#
2^2
2^10
sqrt(16)
16^(1/2)

##-----#
## 3.2. Embedded help
##-----#
help("sqrt")      # help on "sqrt" function
?sqrt              # ...the same...
?round

```

Console



```

R version 2.9.2 (2009-08-24)
Copyright (C) 2009 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.

Loading required package: tcltk
Loading Tcl/Tk interface ... done
Loading required package: Hmisc

Attaching package: 'Hmisc'

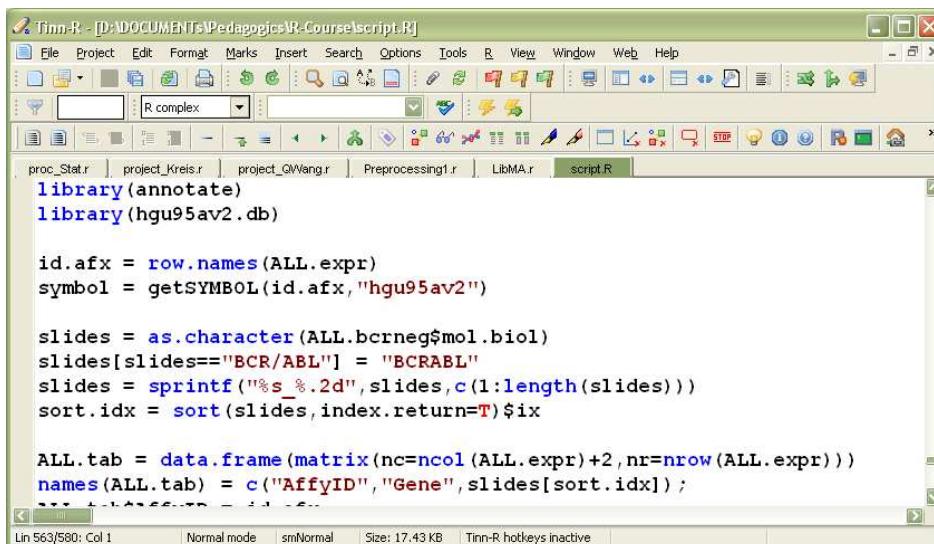
The following object(s) are masked from package:base :

  format.pval,
  round.POSIXt,
  trunc.POSIXt,
  units

Loading required package: R2HTML
Loading required package: svMisc
> |

```

Alternative: Tinn-R (highly recommended for win-users)



```

Tinn-R - [D:\DOCUMENTS\Pedagogics\R-Courses\script.R]
File Project Edit Format Marks Insert Search Options Tools R View Window Web Help
R complex
proc_Start | project_Kreis.r | project_QWang.r | Preprocessing.r | LibM.R | script.R
library(annotate)
library(hgu95av2.db)

id.afx = row.names(ALL.expr)
symbol = getSYMBOL(id.afx,"hgu95av2")

slides = as.character(ALL.bcrneg$mol.biol)
slides[slides=="BCR/ABL"] = "BCRABL"
slides = sprintf("%s %.2d",slides,c(1:length(slides)))
sort.idx = sort(slides,index.return=T)$ix

ALL.tab = data.frame(matrix(nc=ncol(ALL.expr)+2,nr=nrow(ALL.expr)))
names(ALL.tab) = c("AffyID","Gene",slides[sort.idx]);

```

Lin 563/580: Col 1 Normal mode smNormal Size: 17.43 KB Tinn-R hotkeys inactive

```
##-----
## 3.1. Typing commands
##-----
```

```
2*2
2^10
sqrt(16)
16^(1/2)
```

type in the
Console

```
> 2*2
[1] 4

> 2^10
[1] 1024

> sqrt(16)
[1] 4

> 16^(1/2)
[1] 4
> |
```

select in Editor
and press Ctrl-R

```
2*2
2^10
sqrt(16)
16^(1/2)
```

```
##-----
## 3.2. Embedded help
##-----
```

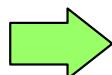
```
help("sqrt")      # help on "sqrt" function
?sqrt             # ...the same...
?round
??round          # fuzzy search for "round" in all help topics
apropos("plot")  # propose commands with the word "plot" inside the name

demo()           # show available demos
demo("image")    # start demo "image"
```

4. BASIC OPERATIONS (4.1)

```
##-----
## 4.1. Mathematical operations and variables
##-----
```

```
x = 2
x
y <- 3
y
x + y -> z
#one more way to show the data
print(z)
#show variables in memory
ls()
```

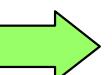


```
> x = 2
> x
[1] 2
> y <- 3
> y
[1] 3
> x + y -> z
> print(z)
[1] 5
> ls()
[1] "x" "y" "z"
```

See
“R Reference Card”
 for more functions
 and details

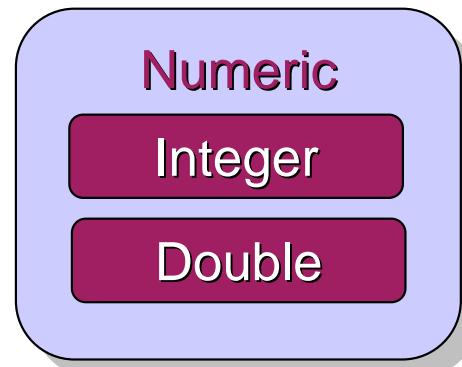
```
# remove all variables from memory
```

```
rm(list=ls())
ls()
```



```
> rm(list=ls())
> ls()
character(0)
>
```

4. BASIC OPERATIONS (4.2)



1
3.141593



TRUE
FALSE

Scalar Data



"Hello, world!"

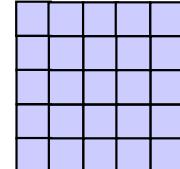
has a sense to use
only in vectors or
data frames



```
> answer=factor(c("yes", "no"))
> answer
[1] yes no
Levels: no yes
```

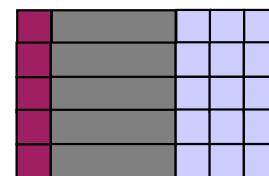


```
> x
[1] 1 2 3 4 5
```

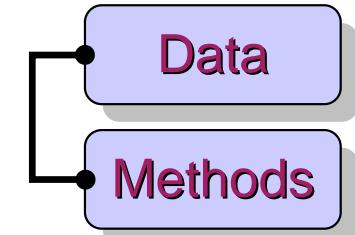
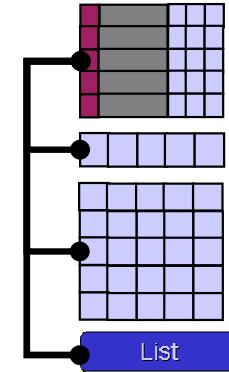


```
> A
[,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    1
```

Data Containers



	name	marks
1	Alex	10
2	Jean	8
3	David	7



4. BASIC OPERATIONS (4.2)

```
##-----
## 4.2. Types of data
##-----

##-----
## Numeric (integer, double)

i=5
i
i*2
i/2
i%2 # integer division
i%%2 # remainder of integer division
round(1.5)

r=1.5
r
l=pi*2*r # let us calculate the circumference
l
```

```
##-----
## Boolean

b1=TRUE # try b1=T
b2=FALSE # try b2=F
b1 & b2 # logical AND
b1 | b2 # logical OR
!b1 # logical NOT
xor(b1,b2) # logical XOR
r==1
r<1
```

For integer bitwise operation install and use
 "bitops" package
 bitAnd(), bitOr(), ...

```
> l=pi*2*r # let us calculate the circumference
> l
[1] 9.424778
> b1=TRUE # try b1=T
> b2=FALSE # try b2=F
> b1 & b2 # logical AND
[1] FALSE
> b1 | b2 # logical OR
[1] TRUE
> !b1 # logical NOT
[1] FALSE
> xor(b1,b2) # logical XOR
[1] TRUE
> r==1
[1] FALSE
> r<1
[1] TRUE
> |
```

4. BASIC OPERATIONS (4.2)

```
##-----
## Character (strings)
st = "Hello, world!"
st
paste("We say:",st) # concatenation
sprintf("We say for the %d-rd time: %s...",3,st) # a more powerfull method a-la C
sprintf("By the way pi=%f, and e=%f",pi,exp(1))

sub(", world","",st) # replace a part of the sting
# (*) in R the regular expression are used to define the pattern
casefold(st, upper=T) # change the case
nchar(st) # number of characters
strsplit(st,"")[[1]] # (*) transforms a string into the vector of single characters
```

```
> st
[1] "Hello, world!"

> paste("We say:",st) # concatenation
[1] "We say: Hello, world!"

> sprintf("We say for the %d-rd time: %s...",3,st) # a more powerfu$ 
[1] "We say for the 3-nd time: Hello, world!.."

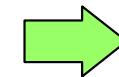
> sprintf("By the way pi=%f, and e=%f",pi,exp(1))
[1] "By the way pi=3.141593, and e=2.718282"
```

```
## how to check who is who?
is.character(st)
is.numeric(st)
is.numeric(l)
```

```
> sub(", world","",st) # replace a part of the sting
[1] "Hello!"
```

4. BASIC OPERATIONS (4.3)

```
##-----  
## 4.3. Special values  
##-----  
## NA - Not-Available (missing data)  
na = NA  
na + 1  
100>na  
na==na  
is.na(na)  
  
## Inf - Infinity (+/- infinite data)  
0*1/0  
-1/0  
is.infinite(1/0)  
  
## NaN - Not-A-Number  
0/0  
is.nan(sqrt(-1))
```



```
> na = NA  
> na + 1  
[1] NA  
  
> 100>na  
[1] NA  
  
> na==na  
[1] NA  
  
> is.na(na)  
[1] TRUE  
  
> 0*1/0  
[1] NaN  
  
> -1/0  
[1] -Inf  
  
> is.infinite(1/0)  
[1] TRUE  
  
> 0/0  
[1] NaN  
  
> is.nan(sqrt(-1))  
[1] TRUE  
Warning message:  
In sqrt(-1) : созданы NaN  
> |
```

4. BASIC OPERATIONS (4.4)

```

##-----
## 4.4. Vectors, matrixes and lists
##-----

## Vector creation
a = c(1,2,3,4,5)
a
a[1]+a[4]

b=5:9
a+b  #(*) try b=5:10. Can you explain the effect?

seq(from=1,to=10,by=0.5) #sequence
rep(1:4, 2)      # same as rep(1:4, times=2)
rep(1:4, each=2) # not the same

txt = c(st, "Let's try vectors", "bla-bla-bla")
txt

boo = c(T,F,T,F,T)
boo

## Vector indexes
a
a[1:3] # take a part of vector by index numbers
a[boo] # take a part of vector by logical vector
a[a>2] # take a part by a condition (!!!)
```

```

> a = c(1,2,3,4,5)

> a
[1] 1 2 3 4 5

> a[1]+a[4]
[1] 5

> b=5:9

> a+b  #(*) try b=5:10. Can you explain the effect
[1] 6 8 10 12 14

> seq(from=1,to=10,by=0.5) #sequence
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
[11] 6.0 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0

> rep(1:4, 2)      # same as rep(1:4, times=2)
[1] 1 2 3 4 1 2 3 4

> rep(1:4, each=2) # not the same
[1] 1 1 2 2 3 3 4 4

> txt = c(st, "Let's try vectors", "bla-bla-bla")

> txt
[1] "Hello, world!"      "Let's try vectors"
[3] "bla-bla-bla"

> boo = c(T,F,T,F,T)

> boo
[1] TRUE FALSE  TRUE FALSE  TRUE

> a
[1] 1 2 3 4 5

> a[1:3] # take a part of vector by index numbers
[1] 1 2 3
```

4. BASIC OPERATIONS (4.4)

```
##-----  
## Matrix  
A=matrix(0,nrow=5, ncol=5)  
A  
A=A-1 # add scalar  
A  
A=A+a # add vector  
A  
t(A) # transpose  
B=A+t(A) # add matrix  
B  
B*B # by-element product  
B%*%B # matrix product
```

> Data

	name	sex	weight	age	survival	code
1	Mouse_1	Male	21	160	TRUE	0
2	Mouse_2	Female	17	131	FALSE	1
3	Mouse_3	Female	20	149	TRUE	2
4	Mouse_4	Male	22	187	FALSE	3
5	Mouse_5	Male	19	141	TRUE	4

> |

4. BASIC OPERATIONS (4.4)

```

##-----#
## Matrix
A=matrix(0,nrow=5, ncol=5)
A
A=A-1 # add scalar
A
A=A+a # add vector
A
t(A) #
B=A+t(A) #
B
B*B #
B%*%B #
##-----#
## Data frame
Data=data.frame(A) # alternatively: D=data.frame(matrix(nr=5,nc=5))
Data

## let us add a column to D
mice = sprintf("Mouse_%d",1:5)
Data=cbind(mice,Data)
## put the names to the variables
names(Data)=c("name","sex","weight","age","survival","code")
Data
## put in the data manually
Data$name=sprintf("Mouse_%d",1:5)
Data$sex=c("Male","Female","Female","Male","Male")
Data$weight=c(21,17,20,22,19)
Data$age=c(160,131,149,187,141)
Data$survival=c(T,F,T,F,T)
Data
  
```

	name	sex	weight	age	survival	code
1	Mouse_1	Male	21	160	TRUE	0
2	Mouse_2	Female	17	131	FALSE	1
3	Mouse_3	Female	20	149	TRUE	2
4	Mouse_4	Male	22	187	FALSE	3
5	Mouse_5	Male	19	141	TRUE	4

4. BASIC OPERATIONS (4.4)

```

## (!!!) see the structure of the objects
str(Data)

## see the structure of the objects
head(Data)

## summary on the data
summary(Data)

## Let's use factors
Data$sex = factor(Data$sex)
summary(Data)

## usefull commands when working with factors:
levels(Data$sex)           # returns levels of the factor
nlevels(Data$sex)           # returns number of levels
as.character(Data$sex)       # transform into strings
  
```

```

> Data
  name   sex weight age survival code
1 Mouse_1 Male    21 160   TRUE     0
2 Mouse_2 Female  17 131  FALSE     1
3 Mouse_3 Female  20 149   TRUE     2
4 Mouse_4 Male    22 187  FALSE     3
5 Mouse_5 Male    19 141   TRUE     4
> 
  
```

```

> str(Data)
'data.frame': 5 obs. of 6 variables:
 $ name    : chr "Mouse_1" "Mouse_2" "Mouse_3" "Mouse_4" "Mouse_5"
 $ sex     : chr "Male" "Female" "Female" "Male" "Male"
 $ weight  : num 21 17 20 22 19
 $ age     : num 160 131 149 187 141
 $ survival: logi TRUE FALSE TRUE FALSE
 $ code    : num 0 1 2 3 4

> head(Data)
  name   sex weight age survival co
1 Mouse_1 Male    21 160   TRUE
2 Mouse_2 Female  17 131  FALSE
3 Mouse_3 Female  20 149   TRUE
4 Mouse_4 Male    22 187  FALSE
5 Mouse_5 Male    19 141   TRUE

> summary(Data)
  name
Length:5
Class :character
Mode  :character
  sex
Length:5
Class :character
Mode  :character
  age
Min.   :131.0
1st Qu.:141.0
Median :149.0
Mean   :153.6
3rd Qu.:160.0
Max.   :187.0
  survival
Mode :logical
  Min.   :FALSE
  1st Qu.:2
  Median :3
  Mean   :0
  NA's   :0
  3rd Qu.:1
  Max.   :TRUE
  code
Mode :integer
  Min.   :0
  1st Qu.:1
  Median :2
  Mean   :2
  3rd Qu.:3
  Max.   :4
  
```

4. BASIC OPERATIONS (4.4)

```

##-----
## Lists

L=list()
L$Data=Data
L$descr = "A fake experiment with virtual mice"
L$num = nrow(Data)
str(L)

## how to access the fields? Simple!
L$Data # try also L$"Data"
L$num
## or
L[[1]]
L[[3]]

```

```

## clear all
ls()
rm(list=ls())
ls()

```

```

> str(L)
List of 3
$ Data : 'data.frame': 5 obs. of 6 variables:
..$ name    : chr [1:5] "Mouse_1" "Mouse_2" "Mouse_3" "Mouse_4" ...
..$ sex     : Factor w/ 2 levels "Female" "Male"
..$ weight   : num [1:5] 21 17 20 22 19
..$ age      : num [1:5] 160 131 149 187 141
..$ survival: logi [1:5] TRUE FALSE TRUE FALSE TRUE
..$ code     : num [1:5] 0 1 2 3 4
$ descr: chr "A fake experiment with virtual mice"
$ num : int 5

> L$Data # try also L$"Data"
      name    sex weight age survival code
1 Mouse_1  Male    21 160   TRUE    0
2 Mouse_2 Female  17 131  FALSE    1
3 Mouse_3 Female  20 149   TRUE    2
4 Mouse_4  Male    22 187  FALSE    3
5 Mouse_5  Male    19 141   TRUE    4

> L$num
[1] 5

> L[[1]]
      name    sex weight age survival code
1 Mouse_1  Male    21 160   TRUE    0
2 Mouse_2 Female  17 131  FALSE    1
3 Mouse_3 Female  20 149   TRUE    2
4 Mouse_4  Male    22 187  FALSE    3
5 Mouse_5  Male    19 141   TRUE    4

> L[[3]]
[1] 5

```

5. DATA IMPORT AND EXPORT (5.1)

```

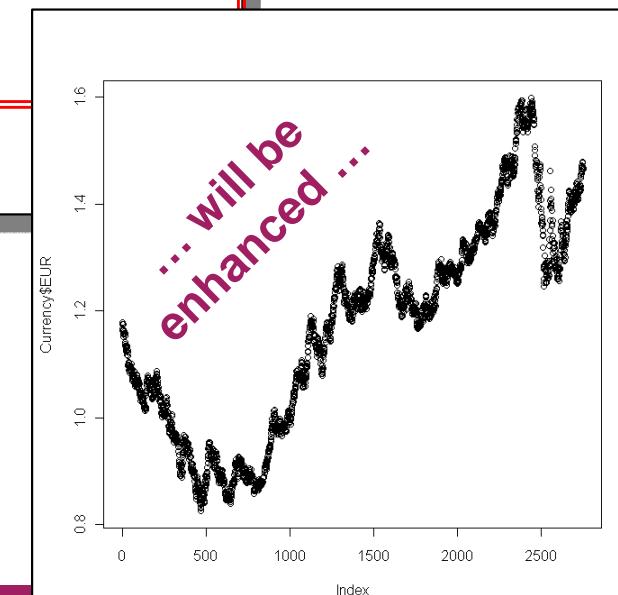
##-----
## "Currency" dataset: learn how to use "header" and "as.is"
##-----
Currency = read.table("http://edu.sablab.net/r/data/currency.txt")
str(Currency)
head(Currency)
## hmm.. wrong... let's ask to add header
Currency = read.table("http://edu.sablab.net/r/data/currency.txt",
                      header=T)
str(Currency)
head(Currency)
## still, it's better to consider dates as strings, not as factors
Currency = read.table("http://edu.sablab.net/r/data/currency.txt",
                      header=T, as.is=T)
str(Currency)
summary(Currency)
## correct!
plot(Currency$EUR)

```

```

> str(Currency)
'data.frame': 2749 obs. of 2 variables:
 $ Date: chr "1999-01-04" "1999-01-05" "1999-01-06" "1999-01-07" ...
 $ EUR : num 1.18 1.18 1.17 1.16 1.17 ...

```



5. DATA IMPORT AND EXPORT (5.2)

```
##-----  
## 5.2. "Shop" dataset: learn how to use "sep"  
##-----  
  
Shop = read.table("http://edu.sablab.net/r/data/shop.txt",  
                  header=T)  
## hm... an error!  
  
Shop = read.table("http://edu.sablab.net/r/data/shop.txt",  
                  header=T, sep="\t")  
## correct!  
  
str(Shop)  
head(Shop)
```

```
> summary(Shop)
      Customer          Payment          Items          Discount
Min.    : 1.00  American Express: 2  Min.    : 1.00  Min.    : 0.00
1st Qu.: 25.75  Discover        : 4  1st Qu.: 1.00  1st Qu.: 0.00
Median  : 50.50  Mastercard       :14  Median  : 2.00  Median  :15.00
Mean    : 50.50  Proprietary Card:70  Mean    : 3.22  Mean    :22.45
3rd Qu.: 75.25  Visa            :10  3rd Qu.: 4.00  3rd Qu.:31.25
Max.    :100.00                               Max.    :17.00  Max.    :158.30
      Sales           Gender          Status          Age
Min.    : 13.23  Female:93   Married:84   Min.    :20.00
1st Qu.: 39.60  Male  : 7    Single :16    1st Qu.:32.00
Median  : 59.70                               Median  :42.00
Mean    : 77.60                               Mean    :43.08
3rd Qu.:100.90                               3rd Qu.:50.00
Max.    :287.59                               Max.    :78.00
```

> |

5. DATA IMPORT AND EXPORT (5.3)

Index of /r/data - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Most Visited GTranslate GMaps V

Index of /r/data

Index of /r/data

- Parent Directory
- • [all.Rdata \(7 MB\)](#)
- [all.txt \(17 MB\)](#)
- [cars.txt](#)
- [currency.txt](#)
- [data for PCA1.xls](#)
- [data for PCA2.xls](#)
- [mice.pdf](#)
- [mice.txt](#)
- [nfl.txt](#)
- [shop.txt](#)

```
## DO NOT TRY ! loading of a big matrix
GE = read.table("http://edu.sablab.net/r/data/all.txt",
                 header=T, sep="\t")
## Better use this:
## download
download.file("http://edu.sablab.net/r/data/all.Rdata",
              destfile="all.Rdata", mode = "wb")
## check the current folder for ".Rdata" file
getwd()                      # show current folder
dir(pattern=".Rdata")        # show files in the current folder
load("all.RData")            # load the data
ls()
str(GE.matrix)

> str(GE.matrix)
num [1:12625, 1:79] 7.6 5.05 3.9 5.9 5.93 ...
- attr(*, "dimnames")=List of 2
..$ : chr [1:12625] "MAPK3" "TIE1" "CYP2C19" "CXCR5"
..$ : chr [1:79] "BCRABL_01" "BCRABL_03" "BCRABL_09"
> |
```

5. DATA IMPORT AND EXPORT (5.4)

```
##-----  
## 5.4. Data export  
##-----  
  
write.table(Shop, "shop.txt", sep = "\t",  
            eol = "\n", na = "NA", dec = ".",  
            row.names = F,  
            qmethod = c("escape", "double"));  
  
save(Shop, file="shop.Rdata")  
  
getwd()  
dir()  
## if you need to set working folder, use  
setwd("...put here desired path...")  
  
## clear all  
rm(list=ls())
```

```
> write.table(Shop, "shop.txt", sep = "\t",  
+             eol = "\n", na = "NA", dec = ".",  
+             row.names = F,  
+             qmethod = c("escape", "double"));  
  
> save(Shop, file="shop.Rdata")  
  
> getwd()  
[1] "D:/DOCUMENTS/Pedagogics/R-Course"  
  
> dir()  
[1] "all.Rdata"  
[2] "Data"  
[3] "Handout"  
[4] "Info"  
[5] "Introduction_to_R.doc"  
[6] "Nazarov_091009_RCourse.ppt"  
[7] "plan.doc"  
[8] "script.R"  
[9] "shop.Rdata"  
[10] "shop.txt"  
> |
```

6. DATA VISUALIZATION (6.1)

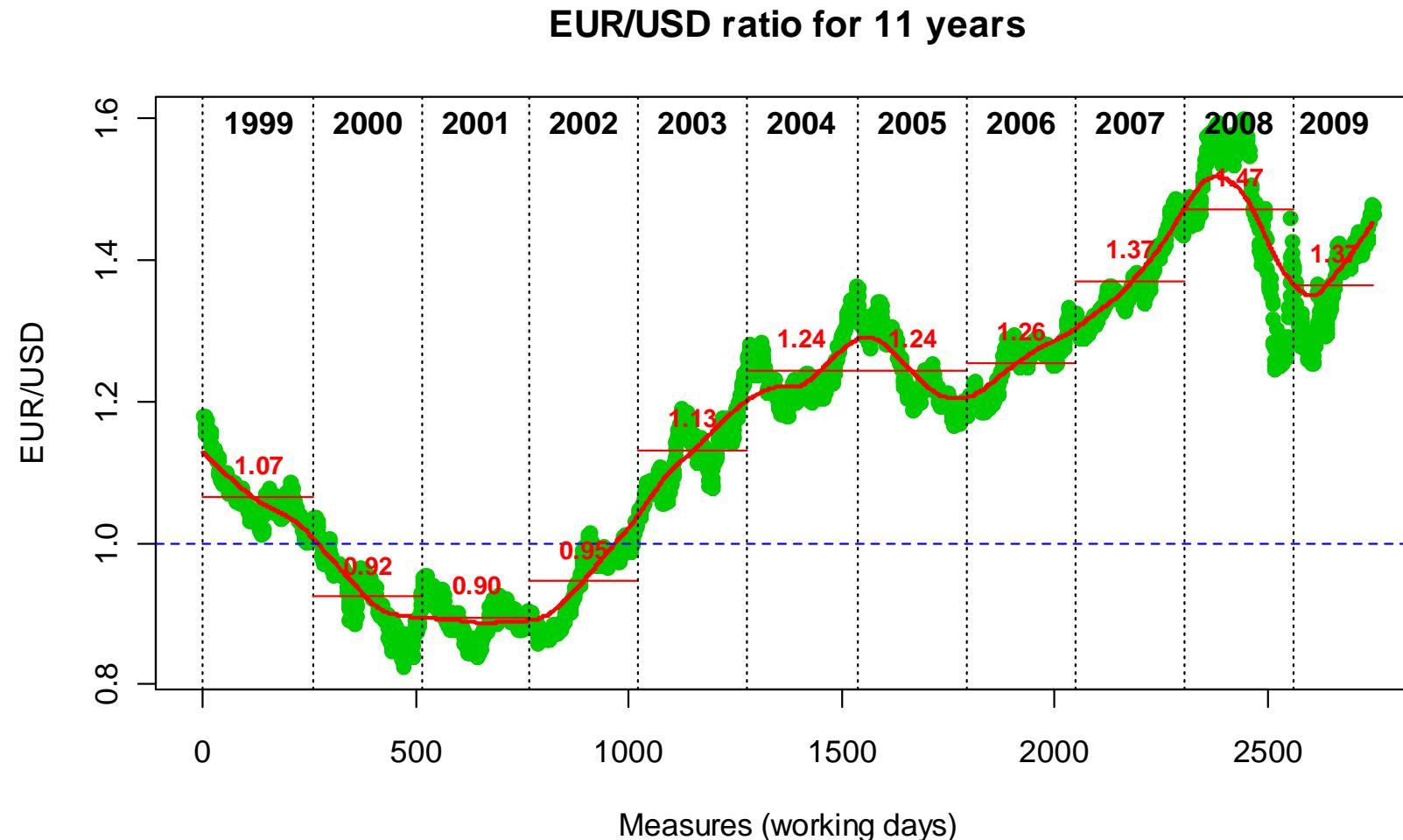
```

##-----
## 6.1. Plot time-series and smooth
##-----
## initiate window
windows(8,5) # try x11()
## get data
Currency = read.table("http://edu.sablab.net/r/data/currency.txt",
                       header=T,as.is=T)
## plot the currency behaviour for the last 10 years
plot(Currency$EUR)
## let's make it more beautiful
windows(8,5)
plot(Currency$EUR,col=3,pch=19,
      main="EUR/USD ratio for 11 years",
      ylab="EUR/USD",
      xlab="Measures (working days)")
## add smoothing. Try different "f"
smooth = lowess(Currency$EUR,f=0.1)
lines(smooth,col=2,lwd=2)
## add 1 level
abline(h=1,col=4,lty=2)

## (*) add years
year=1999 # an initial year
while (year<=2009){ # loop for all the years up to now
  idx=grep(paste("^",year,sep=""),Currency$date) # take the indexes of the meas
  average=mean(Currency$EUR[idx]) # calculate the average ratio for the "year"
  abline(v=min(idx),col=1,lty=3) # draw the year separator
  lines(x=c(min(idx),max(idx)),y=c(average,average),col=2) # draw the average r
  text(median(idx),max(Currency$EUR),sprintf("%d",year),font=2) # write the year
  text(median(idx),average+0.05,sprintf("%.2f",average),col=2,font=2,cex=0.8) #
  year=year+1;
}

```

6. DATA VISUALIZATION (6.1)



6. DATA VISUALIZATION (6.2)

```

##-----
## 6.2. 3D visualization and custom functions
##-----
## define the function to be plotted
sinc = function(x) {
  y = sin(x)/x
  y[is.na(y)] = 1
  return(y)
}
rotsinc = function(x,y) {
  10*sinc(sqrt(x^2+y^2))
}
## generate values
x = seq(-10, 10, length = 50)
y = x
z = outer(x, y, rotsinc)
windows()
## plot 3D image rotated by "theta" and "phi"
persp(x, y, z, theta = 30, phi = 30, expand = 0.5,
       col = "lightblue", ltheta = 120, shade = 0.75,
       ticktype = "detailed",
       xlab = "X", ylab = "Y", zlab = "Z")
## put the title and subscript
title(main = expression(z == sinc(sqrt(x^2 + y^2))),
       sub="Note: sinc(x) = sin(x) / x")

```

```

> sinc = function(x) {
+   y = sin(x)/x
+   y[is.na(y)] = 1
+   return(y)
+ }

> rotsinc = function(x,y) {
+   10*sinc(sqrt(x^2+y^2))
+ }

> x = seq(-10, 10, length = 50)

> y = x

> z = outer(x, y, rotsinc)

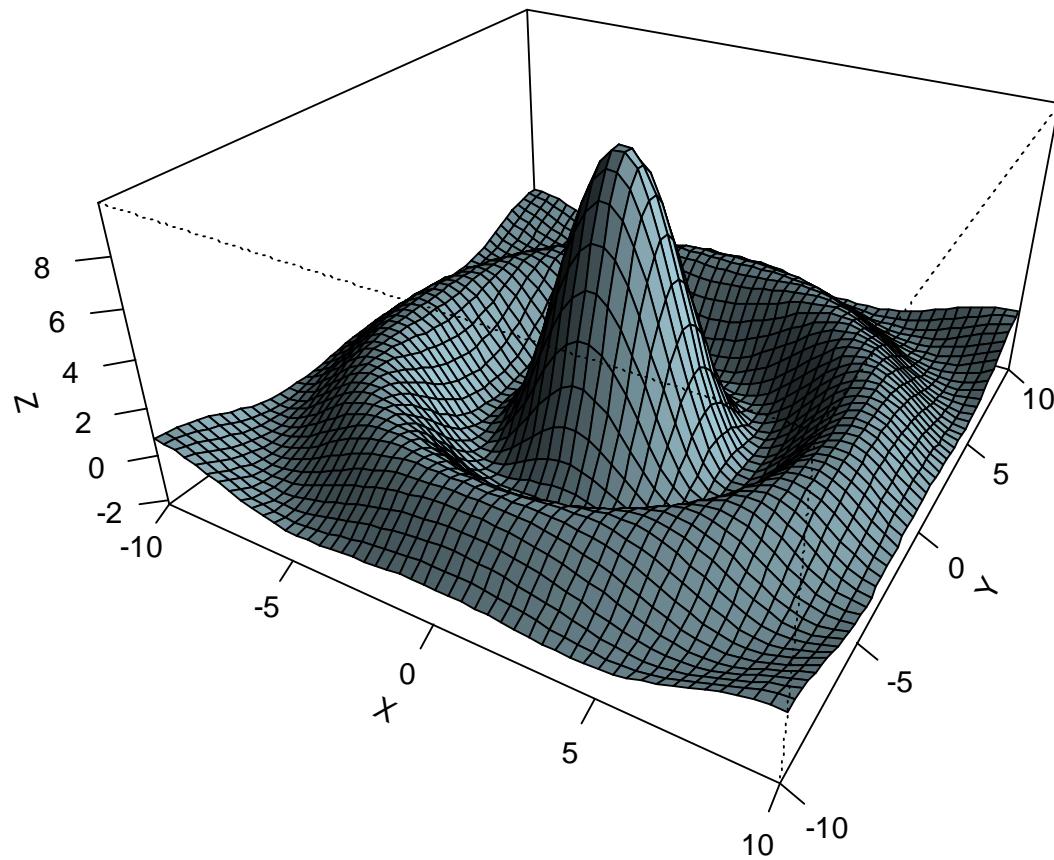
> persp(x, y, z, theta = 30, phi = 30, expand = 0.5,
+         col = "lightblue", ltheta = 120, shade = 0.75,
+         ticktype = "detailed",
+         x .... [TRUNCATED]

> title(main = expression(z == sinc(sqrt(x^2 + y^2))),
+        sub="Note: sinc(x) = sin(x) / x")
>

```

6. DATA VISUALIZATION (6.2)

$$z = \text{sinc}(\sqrt{x^2 + y^2})$$



Note: $\text{sinc}(x) = \sin(x) / x$

6. DATA VISUALIZATION (6.3)

MPD - Home Page - Mouse Phenome Database - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://phenome.jax.org/pub-cgi/phenome/mpdcgi?rtn=docs/home

Most Visited GTranslate GMaps Wikipedia TravelService - Query... The Comprehensive R... Bioinformatics.Org: Th...

MPD - Home Page - Mouse Phenome ...

About News FAQ Downloads Preferences Your flagged measurements

Welcome to the Mouse Phenome Database

Search: or do a Google MPD search

Help

Available phenotype strain survey data

How to contribute data

By subject area Expand this list

- aging
- appearance and coat color
- behavior
- blood chemistry
- blood coagulation
- blood hematology
- blood lipids
- body composition
- body weight size and growth
- bone
- brain
- cancer
- cardiovascular
- cell and tissue damage
- drinking preference
- ear
- endocrine

By mouse strain

- select a strain
- compare 2 strains
- by amount of data in MPD

By project / investigator

- list of all projects
- large phenotyping initiatives
- browse experimental protocols

By composite / consensus view

- click here for list

By intervention

- APAP (acetaminophen)
- diazepam
- ENU
- EtOH (ethanol)
- high-fat diet

Mouse SNPs

- Data sources
- News

Did you know?

You can see which MPD measurements are **correlated to your experimental data**.

Mouse QTL archives

Suggestion box

About MPD and the Mouse Phenome Project

Done

Tordoff MG, Bachmanov AA

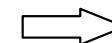
Survey of calcium & sodium intake and metabolism with bone and body composition data

Project symbol: **Tordoff3**

Accession number: **MPD:103**

Index of /r/data

- [Parent Directory](#)
- [all.Rdata](#)
- [all.txt](#)
- [cars.txt](#)
- [currency.txt](#)
- [data for PCA1.xls](#)
- [data for PCA2.xls](#)
- [mice.pdf](#)
- [mice.txt](#)
- [nfl.txt](#)
- [shop.txt](#)



6. DATA VISUALIZATION (6.3)

```

##-----
## 6.3. Mouse phenom :)

##-----  

## load data
Mice=read.table("http://edu.sablab.net/r/data/mice.txt",
                 header=T,as.is=F)
str(Mice)
## initiate window
windows(10,8)
par(mfrow=c(2,2))
## plot a factorial data
plot(Mice$strain,las=2,
      col=rainbow(nlevels(Mice$strain)),cex.names =0.7)
title("Number of mice from each strain")
## plot factorial data as pie
pie(summary(Mice$sex), col=c("pink","lightblue"))
title("Gender composition (f:female, m:male)")
## try to use special command "barplot" as well
## a histogram
hist(Mice$bw_start,probability = T,
      main="Histogram and p.d.f. approximation",
      xlab="weight, g")
lines(density(Mice$bw_start),lwd=2,col=4)

## (!) a box-plot of the population on the basis of sex
boxplot(bw_end~sex,data=Mice,col=c("pink","lightblue"))
title("Weight by sex (f:female, m:male)",
      ylab="weight, g",xlab="sex")

```

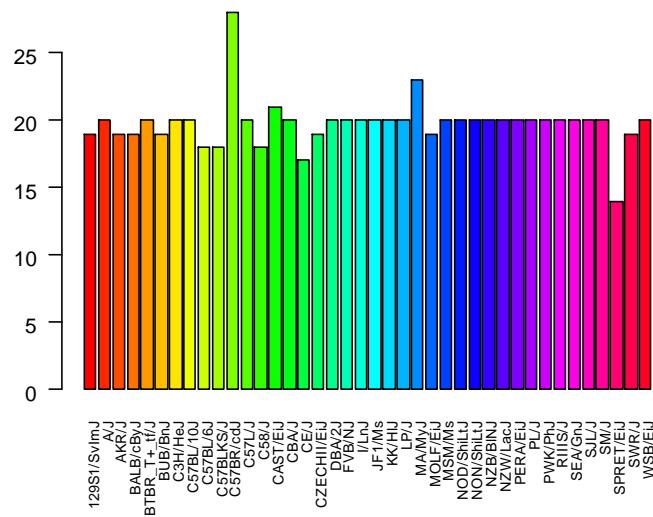
```

> str(Mice)
'data.frame': 790 obs. of 32 variables
 $ strain       : Factor w/ 40 levels 'A/J'...
 $ sex          : Factor w/ 2 levels "f"...
 $ id           : int  1 2 3 368 369 37...
 $ age          : int  66 66 66 72 72 7...
 $ bw_start     : num  19.3 19.1 17.9 1...
 $ bw_end       : num  20.5 20.8 19.8 2...
 $ bw_chg       : num  1.06 1.09 1.11 1...
 $ CaCl2_pref7 : num  33.3 58.3 63.3 5...
 $ CaCl2_pref25: num  66.7 47.1 55.8 5...
 $ CaCl2_pref75: num  38.8 27.9 41.9 5...
 $ CaLa_pref7  : num  65.2 68 57.1 58...
 $ CaLa_pref25 : num  61.9 93.3 65.9 6...
 $ CaLa_pref75 : num  45.5 23.3 73.3 4...
 $ NaCl_pref25 : num  72.2 91.1 64.9 5...
 $ NaCl_pref75 : num  80.2 94.3 86.6 4...
 $ NaCl_pref225: num  88.7 95.8 93.4 1...
 $ NaLa_pref25 : num  79.5 97.6 91 21...
 $ NaLa_pref75 : num  86.7 97.2 91.5 3...
 $ NaLa_pref225: num  89.8 97.6 95.1 8...
 $ bleeding_time: num  64 78 90 65 55 1...
 $ ionized_Ca  : num  1.2 1.15 1.16 1...
 $ pH           : num  7.24 7.27 7.26 7...
 $ adj_ionized_Ca: num  1.12 1.09 1.09 1...
 $ total_calcium: num  2.34 2.17 2.37 2...
 $ age_end      : int  116 116 108 114...
 $ BMD          : num  0.0605 0.0553 0...
 $ BMC          : num  0.536 0.493 0.51...
 $ lean_wt      : num  14.5 13.9 13.8 1...
 $ fat_wt       : num  4.4 4.4 2.9 4.2...
 $ total_wt     : num  18.9 18.3 16.7 1...
 $ pct_fat      : num  23.3 24 17.4 21...
 $ pct_lean     : num  76.7 76 82.6 78...

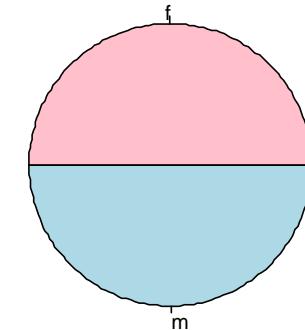
```

6. DATA VISUALIZATION (6.3)

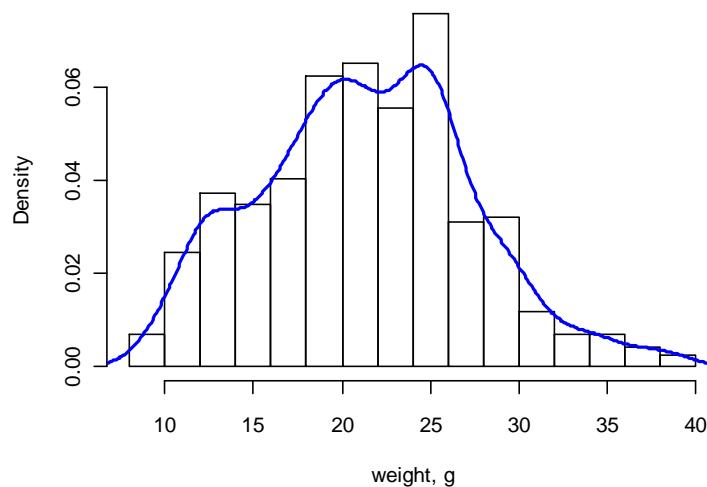
Number of mice from each strain



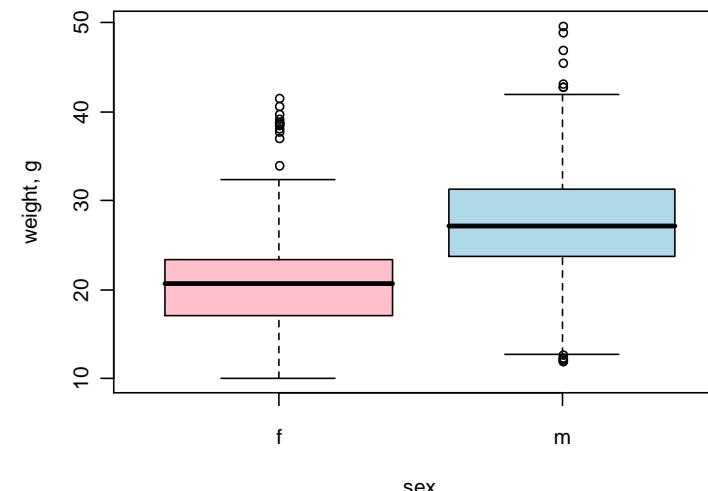
Gender composition (f:female, m:male)



Histogram and p.d.f. approximation



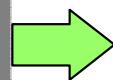
Weight by sex (f:female, m:male)



7. STATISTICAL ANALYSIS

```

##-----
## 7.1. Descriptive statistics
##-----
str(Mice)
## summary on all the variables
summary(Mice)
## descriptive statistics on weight
bw=Mice$bw_end # to shorten typing
## let us put all information into a list
ms = list()
ms$mean = mean(bw)           # average
ms$median = median(bw)        # robust average esti
ms$st.deviation = sd(bw)     # measure of data var
ms$variance = var(bw)         # st.deviation ^ 2
ms$median.absolute.deviation =
  mad(bw) # robust measure of data variabili
ms$min = min(bw) # min
ms$max = max(bw) # max
ms$correlation.with.initial.weight =
  cor(bw,Mice$bw_start)
ms
  
```

 > ms
\$mean
[1] 23.69114

\$median
[1] 23.5

\$st.deviation
[1] 7.068181

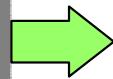
\$variance
[1] 49.95919

\$median.absolute.deviation
[1] 6.96822

\$min
[1] 10

\$max
[1] 49.6

\$correlation.with.initial.weight
[1] 0.942258

 > mean(Mice\$bleeding_time)
[1] NA

> mean(Mice\$bleeding_time,na.rm=T)
[1] 60.99868

7. STATISTICAL ANALYSIS

```

##-----#
## 7.2. Looking for potential effects
##-----#
## Let us look for potentially linked parameters.
## To do so, we can use correlation between
## parameters as a measure of their mutual effect.

## first, transform all data to numbers use data.matrix
str(data.matrix(Mice))

## now calculate the corelation matrix
xc=cor(data.matrix(Mice),data.matrix(Mice),
       use="pairwise.complete.obs")

## create a pallete
bwr.palette =
  colorRampPalette(c("blue", "white", "red"))
## plot a heatmap
heatmap(xc,scale="none",col=bwr.palette(1000))

```

```

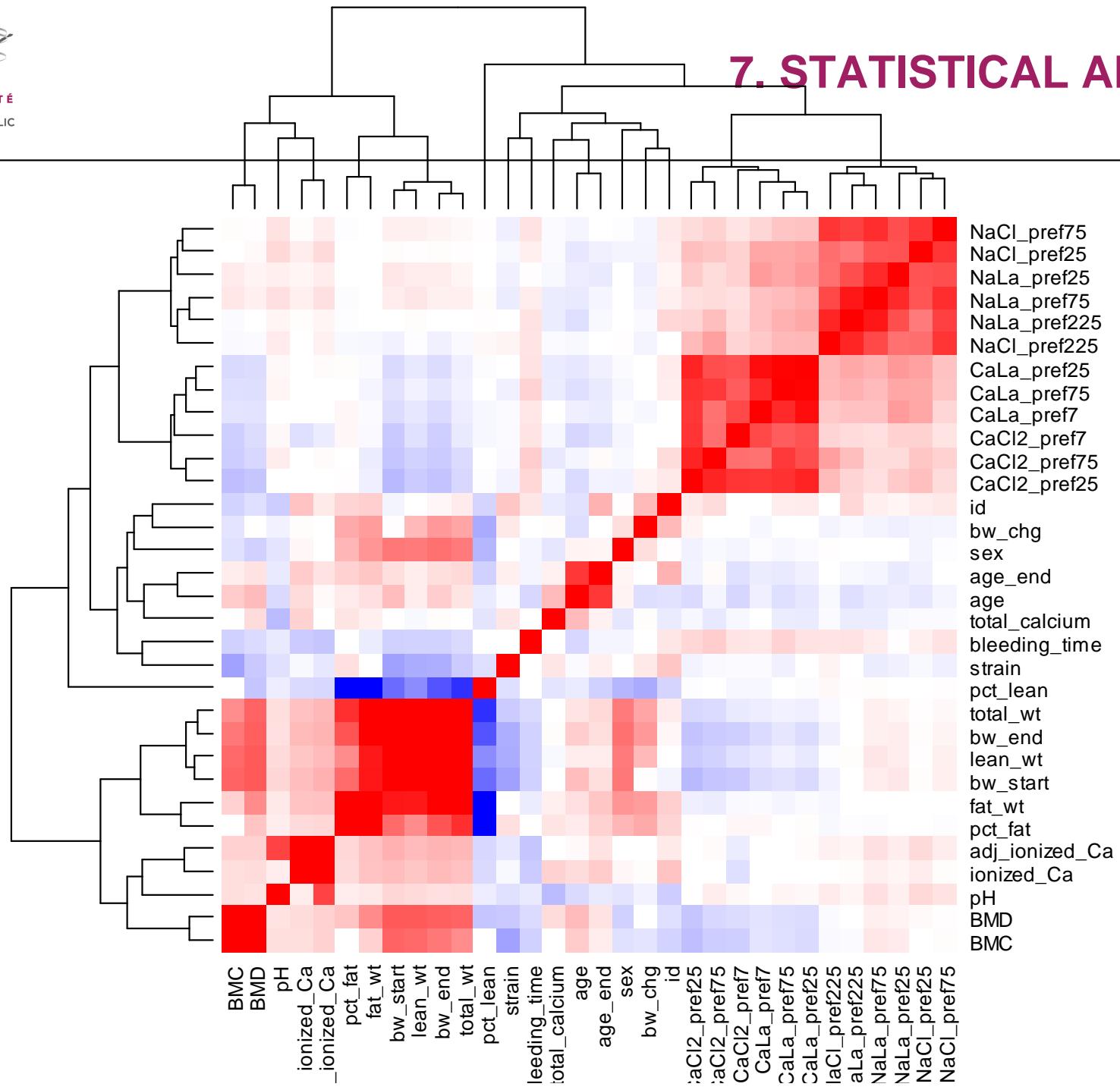
> str(data.matrix(Mice))
num [1:790, 1:32] 1 1 1 1 1 ...
- attr(*, "dimnames")=List(
+   ..$ : NULL
+   ..$ : chr [1:32] "strain" ...

> xc=cor(data.matrix(Mice),data.matrix(Mice),
+          use="pairwise.complete.obs")
> bwr.palette =
+   colorRampPalette(c("blue", "white", "red"))

> heatmap(xc,scale="none",col=bwr.palette(1000))
> |

```

7. STATISTICAL ANALYSIS

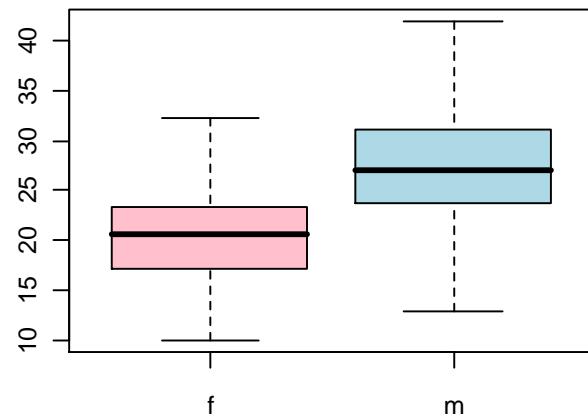


7. STATISTICAL ANALYSIS

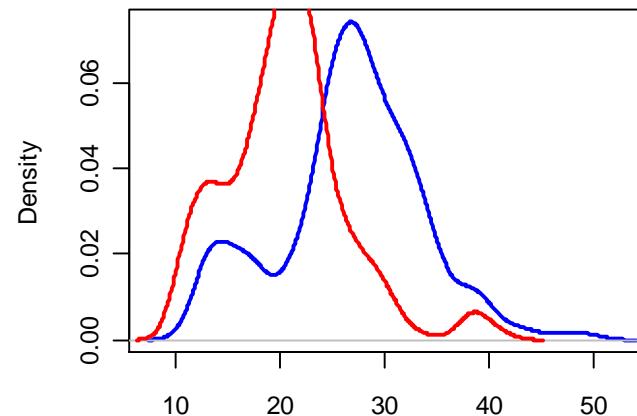
```
##-----  
## 7.3. Statistical tests  
##-----  
windows()  
par(mfrow=c(2,2))  
## Let us plot the distributions of the final weights  
## of mice in a nice way  
boxplot(bw_end~sex,data=Mice,col=c("pink","lightblue"),outline=F)  
title("Final body weights (g)")  
plot(density(Mice$bw_end[Mice$sex=="m"]),col="blue",,lwd=2, main="")  
lines(density(Mice$bw_end[Mice$sex=="f"]),col="red",lwd=2)  
title("Body weight distributions")  
## And the distributions of weight increase (after experiment minus  
## before experiment)  
boxplot(bw_chg~sex,data=Mice,col=c("pink","lightblue"),outline=F)  
title("Weights increase (g)")  
plot(density(Mice$bw_chg[Mice$sex=="m"]),col="blue",,lwd=2, main="")  
lines(density(Mice$bw_chg[Mice$sex=="f"]),col="red",lwd=2)  
title("Distributions of weight increase")  
## What can you say, looking on this pictures? :) Let's test...
```

7. STATISTICAL ANALYSIS

Final body weights (g)

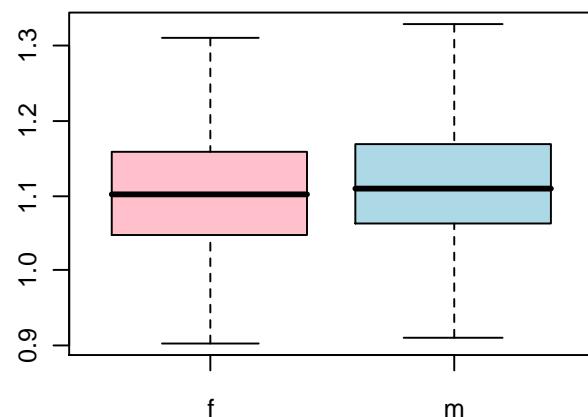


Body weight distributions

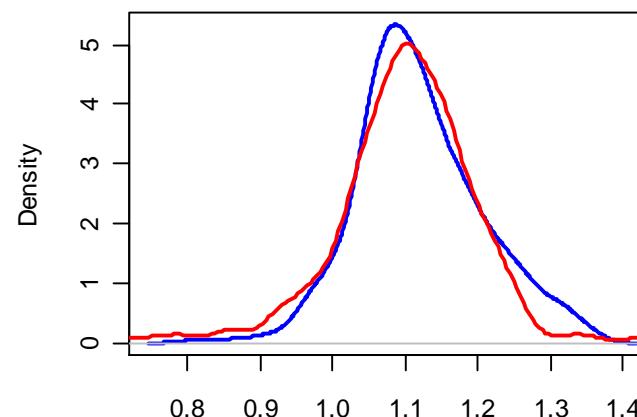


What can you say looking on these plots?

Weights increase (g)



Distributions of weight increase



N = 394 Bandwidth = 0.02154

Well, in fact almost nothing...

```
## Perform t.tests (by default - unpaired, two.sided)
t.test(Mice$bw_end[Mice$sex=="m"],Mice$bw_end[Mice$sex=="f"])
t.test(Mice$bw_chg[Mice$sex=="m"],Mice$bw_chg[Mice$sex=="f"])

## Perform Wilcoxon Rank Sum (equiv. Mann-Whitney test)
wilcox.test(Mice$bw_end[Mice$sex=="m"],Mice$bw_end[Mice$sex=="f"])
wilcox.test(Mice$bw_chg[Mice$sex=="m"],Mice$bw_chg[Mice$sex=="f"])
```

p-value = 2.2e-16

p-value = 0.0014

p-value = 2.2e-16

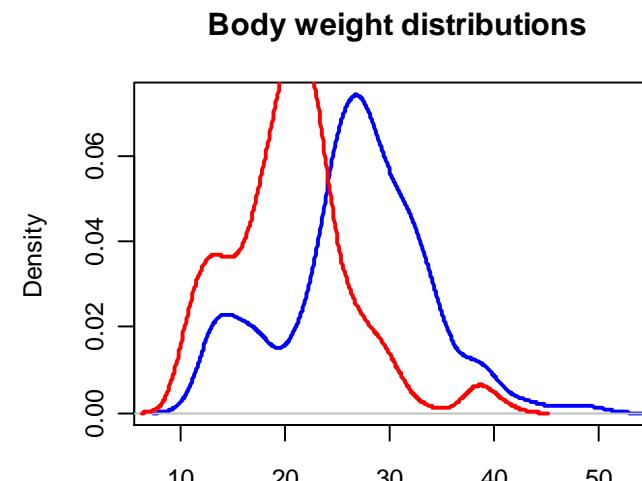
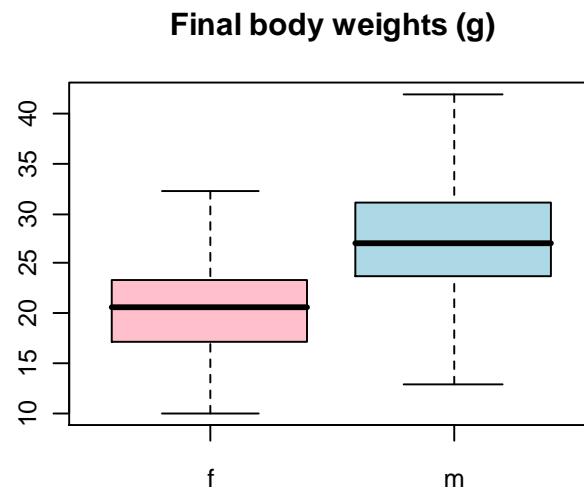
p-value = 0.029

```
> t.test(Mice$bw_chg[Mice$sex=="m"],Mice$bw_chg[Mice$sex=="f"])
```

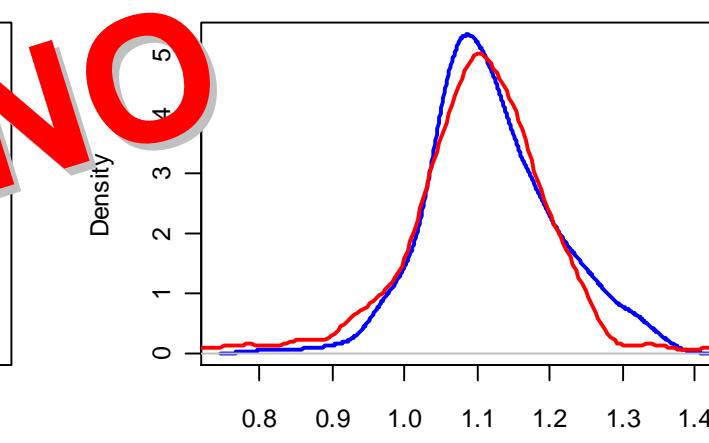
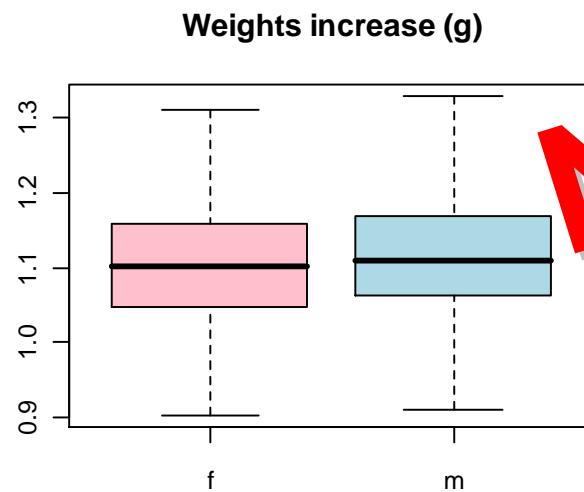
Welch Two Sample t-test

```
data: Mice$bw_chg[Mice$sex == "m"] and Mice$bw_chg[Mice$sex == "f"]
t = 3.2067, df = 682.856, p-value = 0.001405
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.009873477 0.041059866
sample estimates:
mean of x mean of y
1.119401 1.093934
```

7. STATISTICAL ANALYSIS



By the way, can you distinguish M and F by weight?



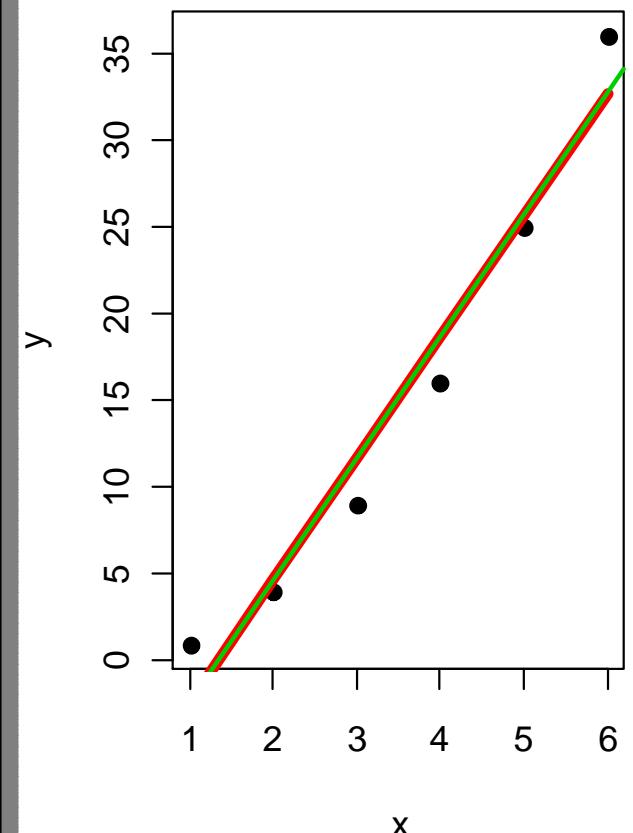
```
#####
# 8. LINEAR MODELS
#####
x = c(1,2,3,4,5,6)      # Create ordered collection
y = x^2                  # Square the elements of x
mean(y)                  # Calculate arithmetic mean of y
var(y)                   # Calculate sample variance

# Let us see the data
windows()
plot(y ~ x)

# Build a linear model
model=lm(y ~ x)          # Fit a linear regression model
summary(model)            # See the results
str(model)                # Look inside.....

# two way of representation of the model:
lines(x,model$fitted.values,col=2,lwd=5)
abline(a=coef(model)[1],b=coef(model)[2],col=3,lwd=2)

windows()
par(mfcol=c(2,2))
plot(lm(y ~ x))
```



8. MODEL FITTING

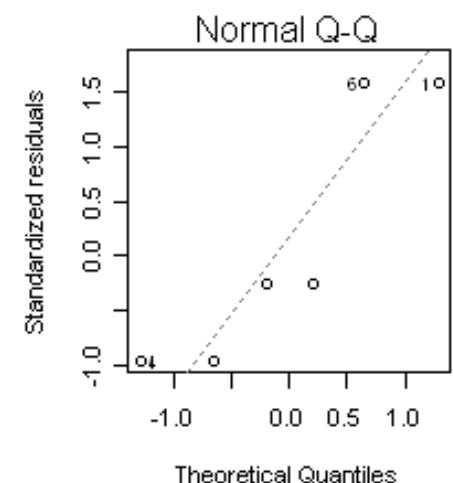
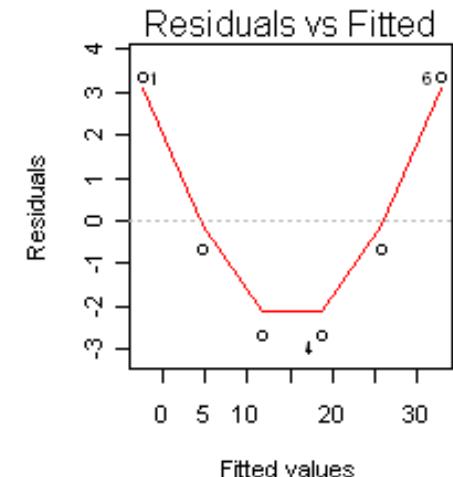
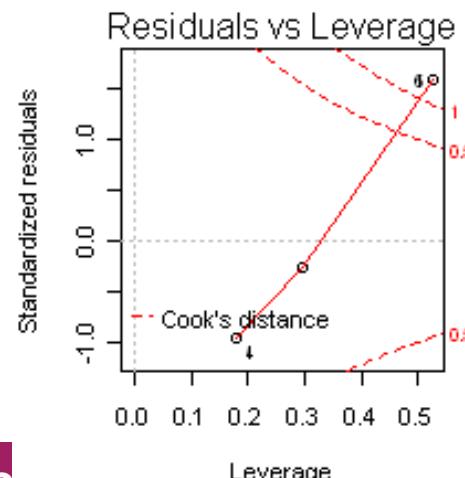
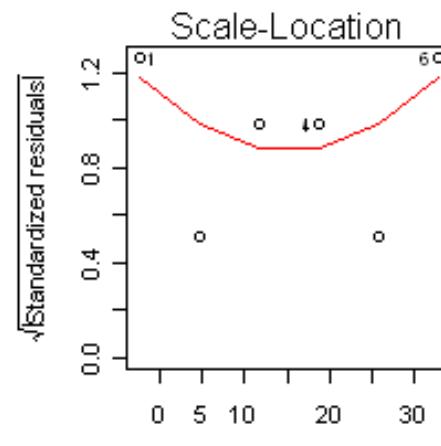
```
> summary(model)          # See the results

Call:
lm(formula = y ~ x)

Residuals:
    1     2     3     4     5     6 
 3.3333 -0.6667 -2.6667 -2.6667 -0.6667  3.3333 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -9.3333    2.8441  -3.282 0.030453 *  
x             7.0000    0.7303   9.585 0.000662 *** 
---
Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Residual standard error: 3.055 on 4 degrees of freedom
 Multiple R-squared: 0.9583, Adjusted R-squared: 0.9478
 F-statistic: 91.87 on 1 and 4 DF, p-value: 0.000662



 **BIOCONDUCTOR**
open source software for bioinformatics

home getting started overview downloads documentation publications workshops cabix

[Getting Started](#)
[Overview](#)
[Downloads](#)
[Documentation](#)
[Workflows](#)
[Installation](#)
[FAQ](#)
[Package Slides](#)
[Annual Reports](#)
[Monograph](#)
[Publications](#)
[Workshops](#)
[Developers](#)
[News](#)

Installation Instructions

Install R

1. Download the most recent version of [R](#) from [The Comprehensive R Archive Network \(CRAN\)](#). The [R FAQ](#) and the [R Installation and Administration Manual](#) contain detailed instructions for installing R on various platforms (Linux, OS X, and Windows being the main ones).
2. Start the R program; on Windows and OS X, this will usually mean double-clicking on the R application, on UNIX-like systems, type "R" at a shell prompt.
3. As a first step with R, start the R help browser by typing "help.start()" in the R command window. For help on any function, e.g. the "mean" function, type "? mean".

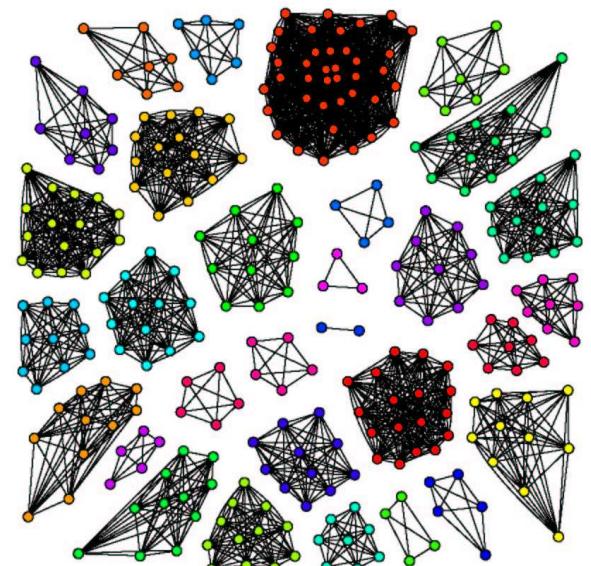
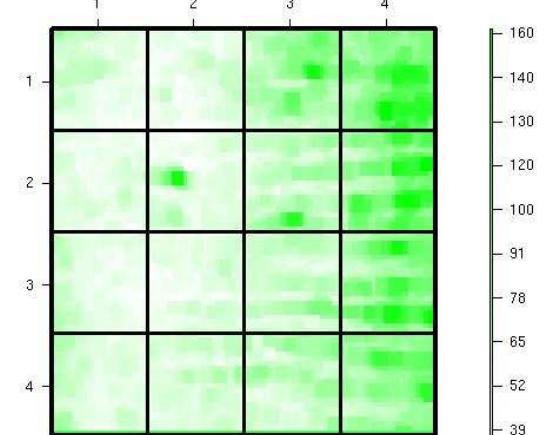
Install standard Bioconductor packages

Install BioConductor packages using the [biocLite.R](#) installation script. In an R command window, type the following:

```
source("http://bioconductor.org/biocLite.R")
biocLite()
```

This installs the following packages: affy, affydata, affyPLM, annaffy, annotate, Biobase, Biostrings, DynDoc, gcrma, genefilter, geneplotter, hgu95av2.db, limma, marray, matchprobes, multtest, ROC, vsn, xtable, affyQCReport. After downloading and installing these packages, the script prints "Installation complete" and TRUE.

Swirl array 93: Image of Cy3 background intensities



**Thank you for your
attention**

to be continued...