R Documentation

1. Installation

To install R please go to the webpage cran.r-project.org and download the executable for MS windows or the source code. The self installation file will guide you through the installation procedure. You can also get a direct link to the windows folder going to my website and to the data mining page: Www.rci.rutgers.edu/~cabrera/

2. R

- R is an interactive programming language.
- You can easily write programs in R.
- It has a GUI interface
- Utilize interactive graphics
- Access various data types and formats.
- Organized in packages to make it easier to share your software and to document the software better.
- Capacity to create communities like Bioconductor to support an activity.

3. Note of caution

R can crash due to bugs inside R or in the user code. It could also crash due to excessive use of resources.
As a precaution please use with frequency the saving actions: "Save History" y also "Save Workspace" in order to save code and data in disk. If R crashes you will be able to recover only until your last save.

4. Libraries

To load a library use the command
> library(MASS) or the newer function > require(MASS)
This library contains lots of different functions.

5. Options

getwd() tells current working disk directory # These can be set using the R menus
setwd([path]) modifies working disk directory
global options list manipulated by options
global graphical options by par

6. Data Structures

atoms: numeric or character
vectors: linear structure made of atoms of the same type
factor: categorical data
matrix: structure of dimension 2 mode of numbers or characters
array: matrix generalization
list: linear structure of any collection of objects
data.frame: data table with columns = variables, rows=observations. A data.frame is also a list.

7. Operators and common functions in R

+ , - , * , /
%/% division by integers
%% operator modulus
^ Exponentiation
- unary minus
Precedence ^ * / + -
Sequence ":"

Example:

```
7 %% 3 + 4%/% 3
7 %% 3^2
1:5                # Sequence
27:3
1:5 + 1:5          # vectors
1:5 + 3            # precedence
2^(1:5)
c(1,2*4,4, 10:12) # function that combines vectors
c("dog","cat","mouse")
```

Exercise 1: Produce the sequence 2 3 4 5 1 2 3 4 5 1 2 3 4 5
Exercise 2: Produce the vector "Monday", "Tuesday", "Wednessday"
Data in R are of mode: logical, integer, real , character.

Common R functions: abs, cos, exp, help, length, lm, log, log10, lsfit, max, mean, median, min, plot, prod range, rnorm, round, runif, sample, seq, sin, sort, sort.list, sqrt, sum, var.

8. The assignment operator '<-' or '='

```
Parimes = c(1,2,3,5,7,11)
greeting = "Hello World"
x = rnorm(1000,mean=2,sd=2)
x = rnorm(1000,mean=(m <- 2),sd=m) # Differences between '<-' and '='
runif(10)                  # Random numbers
sqrt(var(x)) or sd(x)      # Ex compute std. dev. of x
```

9. Reading data files:

9.1 Text files

Create a file with a data set in your favorite editor. I use "Emacs" or "Wordpad" should be an array of numbers 6 by 4 separated by spaces. The first Row should have the name of the four columns namely x y u v. Read the file using "read.table":

```
X = read.table(path…fname…,head=T)
```

If you don't like to use the path just go to the menu "File->Change Dir" and set the directory to the folder with the data file (could be desktop or somewhere else)

Content of `"test.txt"`:

```
x y u v
1 1 5 2
2 3 4 3
1 3 3 2
2 4 5 1
3 6 1 1
5 7 2 4
```

Now read it with this code:

```
X = read.table("test.txt",head=T)
attach(X)
z <- sin(x/5)
plot(x)            #simple scatter plot
plot(x,y)          #
plot(x,z,type="l") # line plot , connected lines
plot(x,z,type="n") # do not plot, then
text(x,z,seq(x))   # use text to label each point from 1 to n
```

9.2 Reading CSV files

This is the easiest way to import files from MS Excel. If the first row of the data has variable names say head=T

X = read.csv(path…fname.cs, head=T)

9.3 Reading SAS export files

This is the easiest way to import files from MS Excel. If the first row of the data has variable names say head=T

```
require(foreign)
hospital = read.xport("hosp.xpt")
hospital[1:2,] # Shows the first two rows of the data
```

SAS Code to create a SAS export file from a SAS dataset. Sas dataset name"sasuser.hospital". Export file name "hosp.xpt".

```
OPTIONS NOFMTERR VALIDVARNAME=V7;
LIBNAME EXPTR XPORT "hosp.xpt";
PROC COPY IN=SASUSER OUT=EXPTR;
SELECT HOSPITAL;
RUN;
```

9.4 Output Data.

Use `write.table` to output an R data frame or an R matrix into a text file data table

```
write.table(pima,file="c:\\Users\\a\\MyDocuments\\587\\PIMA1.txt",row.names=F)
```

9.5 save and load.

```
save(x,y,z, file="C:/saved.txt", compress=F)
save.image(file="C:/.RData", compress=F)
load( file="C:/saved.txt" )
```

10. Linear Regression

Another dataset is the "faithful" data from R's own library. It has two variables representing the time length of an eruption and the weiting time between eruptions.

```
        data(faithful)
        plot(faithful)
# Statistical model notation  y ~ x1 + x2  means model y = a + b x1 + c x2
#  y ~ x1 + x2 + x1:x2   is  y = a + b x1 + c x2 + d x1 x2  (interaction)
        lm(waiting ~ eruptions, data=faithful)
        abline(33,10)
        abline(v=3.1)
        title(sub="This is the faithful data")
```

11. Matrices:

```
        X
        X[ 3,1:2]
        rownum <- 1:4  # get odd rows
        rownum %% 2
        X[rownum %%2 ==1 , ]
        X[ -1,-c(1,3)]
        nrow(X)
        ncol(X)
        Y <- 1:4
        XX <- as.matrix(X)
        XX %*%Y
        Y <-  rnorm(6)
        solve(t(XX) %*% XX) %*% (t(XX) %*% Y)
        X <- matrix(rnorm(18),6,3)
        Y <-  X%*% c(1,2,3) + rnorm(6)/4
        solve(t(X) %*% X) %*% (t(X) %*% Y)
        lsfit(X,Y,int=F)
```

12. LOGICAL OPERATORS: > , >= , < , <= , == , != , & , | or ! Not

```
        x <- rnorm(10)
        u <- x > 0
        u
        x[u]
        x
```

13. CONDITIONAL

```
        x <- rnorm(10)
        u <- sum(x)
      if( u < 0) {  x <-  -x  }
```

14. LOOPS

```
      z <- matrix(rnorm(200),20,10)
      mean.samp <- NULL
      for(i in 1:10) { mean.samp[i] <- mean(z[,i]) }
      stem(mean.samp)
      apply(z,2,mean)  # does the same thing
while loop:
      while(condition is true) { do this }
      repeat  { do this ; if (this) break  }
      apply( x, 1, fun)
      sapply( x, fun)
      tapply( x, y, fun)
      It is very important to vectorize the code.
```

15. FUNCTIONS

```
        fourmom <- function(x) {
           m1 <- c(mean(x))
           if(is.na(m1)) return("Error: There are NA's")
           m2 <- mean(x^2)
```

```
    m3 <- mean(x^3)
    m4 <- mean(x^4)
    c(m1=m1,m2=m2,m3=m3,m4=m4)
}
x <- rnorm(10)
fourmom(x)
```

Homework:
Write your own function that takes the median of the rows of a matrix without using loops or apply.
Exercise: Take the following function. Can you tell me what it does?

```
cmean <- function(x) {
n <- nrow(x)
p <- ncol(x)
nax = is.na(x)
nna <- rep(1,n) %*% nax
i <- nna > 0
result = rep(NA,p)
result[!i] <- rep(1,n) %*% x[,!i] / n
result
}
```

Modify the function so it will allow you to remove missing values.
Answer:

```
cmean <- function(x, na.rm=T) {
n <- nrow(x)
if( na.rm) {
nax = is.na(x)
nna <- rep(1,n) %*% (!nax)
x[nax] <- 0
return( c(rep(1,n) %*% x / nna))
}
else return(c( rep(1,n) %*% x /n))
}
```

16. PLOTS:
16.1 High level:
barplot() to draw bar-plots

```
data(VADeaths, package = "base")
barplot(VADeaths, plot = FALSE)
barplot(VADeaths, plot = FALSE, beside = TRUE)
mp <- barplot(VADeaths) # default
tot <- colMeans(VADeaths)
text(mp, tot + 3, format(tot), xpd = TRUE, col = "blue")
#
barplot(VADeaths, beside = TRUE,
col = c("lightblue", "mistyrose", "lightcyan",
"lavender", "cornsilk"),
legend = rownames(VADeaths), ylim = c(0, 100))
title(main = "Death Rates in Virginia", font.main = 4)
#
hh <- t(VADeaths)[, 5:1]
mybarcol <- "gray20"
mp <- barplot(hh, beside = TRUE,
col = c("lightblue", "mistyrose",
"lightcyan", "lavender"),
legend = colnames(VADeaths), ylim= c(0,100),
main = "Death Rates in Virginia", font.main = 4,
sub = "Faked upper 2*sigma error bars", col.sub = mybarcol,
cex.names = 1.5)
segments(mp, hh, mp, hh + 2*sqrt(1000*hh/100), col = mybarcol, lwd = 1.5)
stopifnot(dim(mp) == dim(hh))# corresponding matrices
mtext(side = 1, at = colMeans(mp), line = -2,
text = paste("Mean", formatC(colMeans(hh))), col = "red")
```

Use plot() to draw scatter-plots

```
x <- rnorm(30)
y <- rcauchy(30)
abline(0,1)
plot(x,y, pch=16, cex=0.7, xlab="from Normal",ylab="from Cauchy",main="Random Data")
abline(0,1)
```

16.2 LOW LEVEL COMMANDS

```
par(mfrow=c(2,2))
mm <- array(c(1, 2), dim = 2)
nf <- layout(mm, 4, c(5, 3), TRUE)
plot(x,y)
hist(x)
par(fig=0,0.5,0,1)
par( mar=c(2,2,1,1))
```

Pairwise scatter plots:

pairs(state.x77)

3D Plots:Are sometimes useful but may need animation

16.3 Conditional plots

```
(In R) data(state)
attach(data.frame(state.x77))#> don't need `data' arg. below
coplot(Life.Exp ~ Income | Illiteracy * state.region, number = 3,
panel = function(x, y, ...) panel.smooth(x, y, span = .8, ...))
detach() # data.frame(state.x77)
```

16.4 Parallel Plot: Graph of a multivariate dataset where the observations are represented by lines.

Objectives:

- To visualize comparisons between multivariate data groups.
- Help asses the quality of classification tools
- To find data clusters and outliers.

```
parallel( ~ state.x77 | state.region )
                        Using the Crime dataset:parallel(~X[,1:4])
```

16.5 Building complicated plots with *layout* functions. Example of scatter plot with marginal histograms.

```
The following command sets the layout of the page.

nf <- layout(matrix(c(2,1,0,3),2,2,byrow=TRUE), c(1,3), c(3,1), respect=TRUE)
layout.show(nf)


hist.inv <- function(x,side=1) {
gg <- hist(x,plot=F)
con <- max(gg$counts)*1.05
if( side==1) {
plot(range(gg$breaks),c(0,con),type="n", axes=F,xlab="",ylab="")
axis(2,con -( i <- pretty(gg$counts)),i)
for(i in 1:length(gg$counts)) rect(gg$breaks[i],con-gg$counts[i],gg$breaks[i+1],con)
}
if(side==2) {
plot(c(0,con),range(gg$breaks),type="n", axes=F,xlab="",ylab="")
axis(1,con -( i <- pretty(gg$counts)),i)
for(i in 1:length(gg$counts)) rect(con-gg$counts[i],gg$breaks[i],con,gg$breaks[i+1])
}
}

superplot <- function(x,y) {
par(mar=c(1.1,1.1,1,1))
nf <- layout(array(c(2,4,1,3), dim=c(2,2)), c(1,3), c(3, 1), TRUE)
plot(x,y)
hist.inv(y,2)
```

```
hist.inv(x) ; frame()
}
x = pima[,2]; y= pima[,3]; superplot(x,y)

##Another way of doing it
superplot2 = function(x,y) {
xhist <- hist(x,  plot=FALSE)
yhist <- hist(y, plot=FALSE)
top <- max(c(xhist$counts, yhist$counts))
xrange <- range(x)
yrange <- range(y)
nf <- layout(matrix(c(2,1,0,3),2,2,byrow=TRUE), c(1,3), c(3,1), TRUE)
layout.show(nf)
par(mar=c(1,1,1,1))
plot(x, y, xlim=xrange, ylim=yrange, xlab="", ylab="")
par(mar=c(1,1,1,1))
barplot(yhist$counts, xlim=c(top, 0), space=0, horiz=TRUE) # !
par(mar=c(1,1,1,1))
barplot(xhist$counts, ylim=c(top,0), space=0) #!ylim!
invisible()
}
x <- pmin(3, pmax(-3, rnorm(50)))
y <- pmin(3, pmax(-3, rnorm(50)))
```

17. LIBRARIES.

mva- Multivariate Analysis. Principal Components

```
library()
library(cluster)
## the variances of the variables in the
## USArrests data vary by orders of magnitude
data(USArrests)
(pc.cr <- princomp(USArrests))
princomp(USArrests, cor = TRUE)
princomp(scale(USArrests, scale = TRUE, center = TRUE), cor = FALSE)
summary(pc.cr <- princomp(USArrests))
loadings(pc.cr)
plot(pc.cr) # does a screeplot.
biplot(pc.cr)
```

18. Principal Components

```
library(mva)
## the variances of the variables in the
## USArrests data vary by orders of magnitude
data(USArrests)
(pc.cr <- princomp(USArrests))
princomp(USArrests, cor = TRUE)
princomp(scale(USArrests, scale = TRUE, center = TRUE), cor = FALSE)
summary(pc.cr <- princomp(USArrests))
loadings(pc.cr)
plot(pc.cr) # does a screeplot.
biplot(pc.cr)
```

19. Linear Models, ANOVA

Annette Dobson (1990) "An Introduction to Generalized Linear Models". Page 9: Plant Weight Data.

```
ctl = c(4.17,5.58,5.18,6.11,4.50,4.61,5.17,4.53,5.33,5.14)
trt = c(4.81,4.17,4.41,3.59,5.87,3.83,6.03,4.89,4.32,4.69)
group <- gl(2,10,20, labels=c("Ctl","Trt"))
weight <- c(ctl, trt)
anova(lm.D9 <- lm(weight ~ group))
summary(lm.D90 <- lm(weight ~ group - 1))# omitting intercept
summary(resid(lm.D9) - resid(lm.D90)) #- residuals almost identical
opar <- par(mfrow = c(2,2), oma = c(0, 0, 1.1, 0))
```

```
plot(lm.D9, las = 1)      # Residuals, Fitted, ...
par(opar)
```

20. Smoothing Splines

```
data(cars)
attach(cars)
plot(speed, dist, main = "data(cars)  &  smoothing splines")
cars.spl <- smooth.spline(speed, dist)
(cars.spl)
all(cars.spl $ w == table(speed)) # TRUE (weights = multiplicities)
lines(cars.spl, col = "blue")
lines(smooth.spline(speed, dist, df=10), lty=2, col = "red")
legend(5,120,c(paste("default [C.V.] => df =",round(cars.spl$df,1)),
         "s( * , df = 10)"), col = c("blue","red"), lty = 1:2, bg='bisque')
detach()
```

21. Hospital Data

Please download the file hospital.csv that can be read using the function *read.csv*. This is a very useful version of *read.table* for excel comma delimited files.

```
> hosp  = read.csv( path.. "hospital.csv")
> pairs(hosp[,8:12])
The next step in analyzing this data set is doing transformations.
> group = hosp$TH +hosp$TRAUMA*2 +hosp$REHAB*4
> table(group)
> boxplot( split(hosp$SALES12,group))
> boxplot( split(log(1+hosp$SALES12),group))
> hosp.lm = lm(SALES12 ~ TH + TRAUMA+REHAB,data=hosp)
> anova(hosp.lm)
> summary(hosp.lm)
> hosp.lm1 = lm(SALES12 ~ factor(group),data=hosp)
> anova(hosp.lm1)
```

> summary(hosp.lm1)

22. Writing R – packages:

These should work for not very old versions of windows like Windows 7 32 bit and 64 bit, Windows Vista, Windows XP with regular updates. It may not work for most Windows 2000 versions and older but I would try it anyway.

1. Create a package skeleton in R, using the command
   package.skeleton(name="mypackage",list=c('superplot','superplot2','hist.inv'))
   Then edit the DESCRIPTION file and .Rd help files appropriately. You will need to follow the instructions of the manual (Help =>Manuals=> Writing R Extensions

2. Download and install the following software:
   1. Mike TeX: http://www.miktex.org
   2. Rtools: http://www.murdoch-sutherland.com/Rtools/.
      Rtools will install: Vanilla Perl, Tool set, MinGW, Cygwin, TCL/TK

3. Check PATH VARIABLE: (Start->Settings->)Control Panel->System->Advanced. Click on the Environment Variables button, which should be in the middle. The previous software in Rtools should be in the path. If your version of windows is not very old the previous installation on step 2. should make the appropriate modifications to the path. In my windows 7 - 64 bits my path like this at the beginning of the path:
   c:\Rtools\bin;c:\Rtools\perl\bin; C:\R\bin;c:\Rtools\MinGW64\bin;C:\Program Files\R\R-2.15.1\bin\x64;C:\Program Files (x86)\MiKTeX 2.9\miktex\bin; and here continues with other addresses for other non R software. For a 32 bit windows is not identical either, and there will be modifications

related to your installation folder addresses.
You can check in the CMD window if R is already in the path by just typing R.
You can also check if MikeTeX is in the path by typing PATH in the CMD window.

4. Download the newest version of the R batch files http://cran.r-project.org/contrib/extra/batchfiles/
   Extract the files to the folder `C:\Rbatch`. And pasted `C:\Rbatch;` to the PATH as before.

5. OPEN cmd window with administrator permission. Then go to the folder root to where the package was created
   Cd ….

6. RUN the command to create the R-package.
   R CMD build mypackage

7. RUN the command to install the R-package.
   R CMD INSTALL mypackage

8. Alternative to 7. RUN the command to make a "zip" file that can be loaded from the menu inside Rgui.
   R CMD INSTALL --build mypackage.tar.gz
9. If there are any error messages that you can't resolve please bring them to class and we will deal with them.

Homework 3:
Take a look at the superplot2 function on the notes above. The bottom left box of the resulting plot is empty. Please fill it up with a summary of the two variables including means, medians, standard deviations and correlation coefficient. If you have room add also the MAD's. (Median absolute deviation)
Homework:

1. Fix *superplot* and *superplot2*. There is a problem with the scales on the side. The histogram scale does not exactly correspond to the scales of the axes of the scatter plot. There is a small displacement please try to fix this is you can but only after you succeed with part 2.
2. Write an R-package with these functions. *superplot, superplot2,*`hist.inv.`