Chapter 3

R tutorial

Introduction

- The website of R is http://www.r-project.org/ where you can download the packages, documentation, etc.
- A full tutorial can be found in http://www.cyclismo.org/tutorial/R/.
- Command help.search("keyword") helps to find the proper packages
- install.packges("package name") is used to install the necessary packages
- library(package name) loads the package in memory
- help(command) or ? command displays the help of that command.
- source("filename.R") loads a batch file with R commands and execute them serially.

Easy commands

- There are three different way to assign the value 3 to the variable a.
- The command c() concatenates elements and therefore it can be used to create a vector
- The command ls() shows the variables in memory.
- The command rm() remove the variables inside the brackets.

```
>a<-3
>3->a
>a=3
>b<-"lorry"
>b
[1] "lorry"
>w
[1] 1 2 3 4 5
> z<-c(x, 0.9, x)
> z
[1] 2.0 3.0 0.9 2.0 3.0

> ls()
[1] "a" "b" "w" "x" "y" "z"
> rm("a")
> ls()
[1] "b" "w" "x" "y" "z"
```


Arithmetic

The arithmetic operations are +, -, *, /, ; %%.

```
>x<-c(2,3)
>y<-c(1,4,5,6)
>x
[1] 2 3
> 2*x
[1] 4 6
> x-1
[1] 1 2
> y
[1] 1 4 5 6
> x-y
[1]  1 -1 -3 -3  # Beware!!
#Reminder
> 9%%3
[1] 0
```

Vector arithmetic

Notice the difference between * and %*%.

```
> x
[1] 2 3
> x*x
[1] 4 9
> x%*%x
     [,1]
[1,]   13
> x^2
[1] 4 9
```


Matrices

Elements are access by indexes that start at 1.

```
#matrix

> w<-matrix(1, 3,3)
> w
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
#To change the diagonal
> diag(w)=c(3,2,4)
> w
     [,1] [,2] [,3]
[1,]    3    1    1
[2,]    1    2    1
[3,]    1    1    4

#access row 1
> w[1,]
[1] 3 1 1

#access column 2
> w[,2]
[1] 1 2 1

#access element 1, 2
> w[1,2]
[1] 1

#access rows 1 and 3
> w[c(1,3),]
     [,1] [,2] [,3]
[1,]    3    1    1
[2,]    1    1    4
```

Matrices

- Element by element product, sum, division, etc. with the classical arithmetic commands
- Matrix product with %*%
- Inverse using solve()

```
> w*w
     [,1] [,2] [,3]
[1,]    9    1    1
[2,]    1    4    1
[3,]    1    1   16
> w%*%w
     [,1] [,2] [,3]
[1,]   11    6    8
[2,]    6    6    7
[3,]    8    7   18

#matrix determinant
> det(w)
[1] 17

#inverse of w
> solve(w)
            [,1]       [,2]        [,3]
[1,]  0.41176471 -0.1764706 -0.05882353
[2,] -0.17647059  0.6470588 -0.11764706
[3,] -0.05882353 -0.1176471  0.29411765
```


Sequences

- The expression 1:n denotes the sequence 1,2, . . ., n.
- More generally, seq(i, j, k)

```
> w<-1:5
> w
[1] 1 2 3 4 5
> seq(3,10,2)
[1] 3 5 7 9
```

Types of vectors

There can be vectors that are non numeric:

```
> mywords<-c("This", "is", "a", "line")
> mywords
[1] "This" "is"   "a"    "line"
> mywords[3]
[1] "a"
> mywords=="is"
[1] FALSE  TRUE FALSE FALSE
> TF<-(mywords=="is")
> TF
[1] FALSE  TRUE FALSE FALSE
> myfactor<-factor(c("Low", "Medium", "High"))
> myfactor
[1] Low    Medium High
Levels: High Low Medium
```


Indexing vector elements

We use the square brackets [] to obtain the elements of a vector or matrix.

```
> x<-c(1,3,4,7)
> x[1]
[1] 1
#Shows elements in position 2,3 and 4
> x[2:4]
[1] 3 4 7
#Show every element but the one in position 2
> x[-2]
[1] 1 4 7
> my.index<-c(2,4)
> x[my.index]
[1] 3 7
> x[c(2,4)]
[1] 3 7
> x[x<3.5]
[1] 1 3
```


Data frames

Vectors of elements of different types can be put together without losing their type in data.frames.

```
> cartoon<-c("Pondus", "Hagard", "Steen og Stoffer")
> nat<-factor(c("Norwegian", "Denmark", "US"))
> x<-c(4,NA, 3)
> miscellaneous<-data.frame(cartoon, nat, x)
> miscellaneous
          cartoon       nat  x
1          Pondus Norwegian  4
2          Hagard   Denmark NA
3 Steen og Stoffer        US  3

> rm(cartoon, nat, x)
> x
Error: object 'x' not found
> miscellaneous$x
[1]  4 NA  3
> attach(miscellaneous)
> x
[1]  4 NA  3
> detach(miscellaneous)
> x
Error: object 'x' not found
```