Chapter 1

Matrix calculations

1.1 Matrix definition

A matrix is a two-dimension collection of elements of the same type, very useful in mathmatics. Often a matrix is thought of as a number of vectors of the same length. A matrix is thus like a data frame with columns of the same type where names are not needed.

```
m<-matrix(1:10,nrow=4)
m
```

```
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7    1
[4,]    4    8    2
```

```
class(m)
```

```
[1] "matrix" "array"
```

```
as.data.frame(m)
```

```
  V1 V2 V3
1  1  5  9
2  2  6 10
3  3  7  1
4  4  8  2
```

For the example above we can observe

- nrow=4 decides the numer of rows
- the data 1:10 are filled column by column
- the number of coumns is decided by the data (input numbers)
- if the length of the input numbers is shorter than rows x columns the numbers are filled starting from the beginning

```
# we can define a matrix of characters but it is not so useful for calculations
matrix(letters[1:8],ncol=2)
```

```
     [,1] [,2]
[1,] "a"  "e"
[2,] "b"  "f"
[3,] "c"  "g"
[4,] "d"  "h"

m1<-matrix(1:12,nrow=4)
m1

     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12

nrow(m1)

[1] 4

ncol(m1)

[1] 3

dim(m1)

[1] 4 3

m2<-m1[2:4,]  # row 2-4 of m2
# specifying rows and columns work like as for a data frame
m2

     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    3    7   11
[3,]    4    8   12

dim(m2)

[1] 3 3
```

Own experimentation

Create matrixes of varying dimensions. Instead of specifying nrow you can use ncol. The dimension of the matrix is then decided by the number of columns. Try to change the order of filling a matrix using the argument byrow = TRUE. Can you create a matrix of logics (TRUE/-FALSE)?

1.2 Calculations

Matrixes can be transposed i.e. rows and columns are switched. Two matrixes can be added or subtracted if they have the same dimension. The product of two matrixes A and B with dimensions $(R_A \text{ x } C_A)$ and $(R_B \text{ x } C_B)$ written AB is only possible if $C_A = R_B$. The resulting product written AB has dimension $(R_A \text{ x } C_B)$. OBS that BA has dimension $(R_B \text{ x } C_A)$. It is therefore not obvious that AB=BA or that BA is even defined (necessary that $(C_B{=}R_A)$). Observe the operator for multiplication. The (i, j)th element in the resulting matrix product AB is the inner product of the vectors from row i in A and column j in B.

```
m1<-matrix(1:12,nrow=4)
m1
```

```
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

```
dim(m1)
```

```
[1] 4 3
```

```
t(m1) # this is the transpose of the matrix
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

```
dim(t(m1))
```

```
[1] 3 4
```

```
m2<-matrix(11:25,nrow=3)
m2
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   11   14   17   20   23
[2,]   12   15   18   21   24
[3,]   13   16   19   22   25
```

```
dim(m2)
```

```
[1] 3 5
```

```
m1+m1 # the numer of rows and column must match
```

```
     [,1] [,2] [,3]
[1,]    2   10   18
[2,]    4   12   20
[3,]    6   14   22
[4,]    8   16   24
```

```
m1+m2  # this does not work
```

```
Error in m1 + m2:  non-conformable arrays
```

```
3*m2
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   33   42   51   60   69
[2,]   36   45   54   63   72
[3,]   39   48   57   66   75
```

```
m1%*%m2   # the product of two matrixes
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]  188  233  278  323  368
[2,]  224  278  332  386  440
[3,]  260  323  386  449  512
[4,]  296  368  440  512  584
```

```
m1[3,]

[1]  3  7 11

m2[,2]

[1] 14 15 16

sum(m1[3,]*m2[,2])  # the element in row 3 column 2 of the product m1%*%m2

[1] 323

(m1%*%m2)[3,2]

[1] 323

m2%*%m1

Error in m2 %*% m1:  non-conformable arguments
```

The last did not work because the number of columns in the first and the number of rows in the second do not match

1.3 Matrix inverse and equation systems

Assume we want to solve the equations $\begin{Bmatrix} x+3y=4 \\ 2x-y=3 \end{Bmatrix}$
This can be written in matrix form as $AX = b$ where X is the vector (x, y), b the vector $(4, 3)$ and $A = \begin{pmatrix} 1 & 3 \\ 2 & -1 \end{pmatrix}$
The equation system can be solved by taking the inverse of A and the solution is the result of $A^{-1}b$. This is an extension to solving single equation $ax = b$ where the solution is b/a or as it can be written $a^{-1}b$.

```
A<-matrix(c(1,2,3,-1),nrow=2)
A

     [,1] [,2]
[1,]    1    3
[2,]    2   -1

b<-matrix(c(4,3),nrow=2)
b

     [,1]
[1,]    4
[2,]    3
```

The function for matrix inverse is **solve**. If the matrix inverse is calculated we only have to multiply with the 1x2 matrix `b` . However, this is not the recommended solution but we still use **solve**.

```
Am1<-solve(A)
Am1

           [,1]       [,2]
[1,] 0.1428571  0.4285714
[2,] 0.2857143 -0.1428571
```

```
Am1%*%b     # this is the straightforward solution

          [,1]
[1,] 1.8571429
[2,] 0.7142857

X<-solve(A,b)  # this is the recommended solution
X

          [,1]
[1,] 1.8571429
[2,] 0.7142857
```

Let us check if the solution is correct by multiplying A and x. Similar to $a^{-1}a = 1$ for scalars we get a special result if a matrix is multiplied with its inverse. It is called identity matrix.

```
A%*%X

     [,1]
[1,]    4
[2,]    3

round(Am1%*%A,10)  # The identity matrix

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

diag(2)   # it can easily be generated

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

diag(10)

      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    1    0    0    0    0    0    0    0    0     0
 [2,]    0    1    0    0    0    0    0    0    0     0
 [3,]    0    0    1    0    0    0    0    0    0     0
 [4,]    0    0    0    1    0    0    0    0    0     0
 [5,]    0    0    0    0    1    0    0    0    0     0
 [6,]    0    0    0    0    0    1    0    0    0     0
 [7,]    0    0    0    0    0    0    1    0    0     0
 [8,]    0    0    0    0    0    0    0    1    0     0
 [9,]    0    0    0    0    0    0    0    0    1     0
[10,]    0    0    0    0    0    0    0    0    0     1
```

1.4 Arrays

An array is a generalisation of a matrix. The dimension of a matrix is $(R$ x $C)$.An array is a generalisation of a matrix to more than two dimensions. As for a matrix the elements have to be of the same type, usually numeric.

```r
x<-array(1,dim=c(3,3))
x
```

```
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
```

```r
class(x)
```

```
[1] "matrix" "array"
```

```r
y<-array(1:18,dim=c(3,3,2))
y
```

```
, , 1

     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

, , 2

     [,1] [,2] [,3]
[1,]   10   13   16
[2,]   11   14   17
[3,]   12   15   18
```

```r
class(y)
```

```
[1] "array"
```

```r
dim(x)
```

```
[1] 3 3
```

```r
dim(y)
```

```
[1] 3 3 2
```

```r
y[,,1]
```

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```r
y[,,2]
```

```
     [,1] [,2] [,3]
[1,]   10   13   16
[2,]   11   14   17
[3,]   12   15   18
```

```r
y[1,,2]   # the first dimension=1 (first row) and third dimension=2
```

```
[1] 10 13 16
```

```r
is.vector(y[1,,2])
```

```
[1] TRUE
```

Let us import the data file norsjo86.

agegrp:	Age group	(30, 40, 50 ,60 years)
health:	Health status	(0=good, 1=not quite good/bad)
sex:	Sex	(1=man, 2=woman)
height:	Body height	(cm)
weight:	Body weight	(kg)
sbp:	Systolic blood pressure	
dbp:	Diastolic blood pressure	
kolester:	Cholesterol	
smoker:	Smoking status	(0=non-smoker, 1=smoker)
bmi:	Body mass index	(kg/m^2)

```
# import spss file norsjo86
library(haven)
norsjo86 <- read_sav("../data/norsjo86.sav")
head(norsjo86,3)

# A tibble: 3 x 10
         agegrp         health     sex height weight   sbp   dbp kolester     smoker   bmi
      <dbl+lbl>      <dbl+lbl> <dbl+l>  <dbl>  <dbl> <dbl> <dbl>    <dbl>  <dbl+lbl> <dbl>
1 60 [60 years~ 0 [good]       2 [Wom~    157     61   110    70      6.7 0 [Non-sm~  24.7
2 60 [60 years~ 1 [not quite ~ 2 [Wom~    157     97   150   100      6.6 0 [Non-sm~  39.4
3 60 [60 years~ 0 [good]       1 [Man]    170     74   136    96      8.2 0 [Non-sm~  25.6

xtabs(~sex+agegrp,data=norsjo86)

   agegrp
sex 30 40 50 60
  1 36 31 27 34
  2 27 37 35 33

tab<-xtabs(~sex+agegrp+smoker,data=norsjo86)

tab

, , smoker = 0

   agegrp
sex 30 40 50 60
  1 31 25 24 25
  2 20 27 27 27

, , smoker = 1

   agegrp
sex 30 40 50 60
  1  5  6  3  9
  2  7 10  8  6

is.array(tab)

[1] TRUE

tab[,2,]    # gives the table for 40 year

   smoker
sex  0  1
  1 25  6
  2 27 10
```