Summary - Maths & Statistics

0. Basics

0.1 Managing Packages

```r
# install.packages("ggplot2")
library(ggplot2)
```

0.2 Data Types

```r
typeof(5)
typeof("text")
typeof(TRUE)
```

```
## [1] "double"
## [1] "character"
## [1] "logical"
```

0.3 Arithmetic with R

```r
5 + 5 # An addition
5 - 4 # A subtraction
3 * 5 # A multiplication
9 / 3 # A division
2 ^ 5 # Exponentiation
7 %% 3 # Modulo
```

```
## [1] 10
## [1] 1
## [1] 15
## [1] 3
## [1] 32
## [1] 1
```

0.4 Variable Assignment

```r
var1 <- 5
var2 <- 13
var1 + var2
```

```
## [1] 18
```

1. Vectors & Matrices

1.1 Vectors

1.1.1 Creating a vector

```
v1 <- c(1, 2, 3, 4, 5)
v1
length(v1)
```

```
## [1] 1 2 3 4 5
## [1] 5
```

1.1.2 Getting the components of a vector (slicing)

```
v1[1]
v1[1:3]
```

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

1.1.3 Auto population of a vector

```
c(rep(0, times = 5))
c(rep(c(1, 2), times = 3))
c(5:9)
c(seq(from = 1, to = 7, by = 2))
c(seq(from = 1, to = 7, length = 5))
```

```
## [1] 0 0 0 0 0
## [1] 1 2 1 2 1 2
## [1] 5 6 7 8 9
## [1] 1 3 5 7
## [1] 1.0 2.5 4.0 5.5 7.0
```

1.1.4 Arithmetic with vectors

```
a <- 1:5
b <- 3:7
a+b
a-b
a*b
a/b
a^(b-a)
```

```
## [1]  4  6  8 10 12
## [1] -2 -2 -2 -2 -2
## [1]  3  8 15 24 35
## [1] 0.3333333 0.5000000 0.6000000 0.6666667 0.7142857
## [1]  1  4  9 16 25
```

And the scalar

```
a%*%b
```

```
##      [,1]
## [1,]   85
```

1.2 Matrices

1.2.1 Creating a matrix

```r
m1 <- matrix(c(1:6), ncol = 3, byrow = TRUE)
m1
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
```

```r
diag(c(1:4))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    2    0    0
## [3,]    0    0    3    0
## [4,]    0    0    0    4
```

```r
diag(3)
```

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

1.2.2 Length, Dimensions and Slicing

```r
length(m1)
dim(m1)
```

```
## [1] 6
## [1] 2 3
```

```r
m1[2,3]
m1[,2]
```

```
## [1] 6
## [1] 2 5
```

1.2.3 Arithmetic with matrices

```r
m2 <- matrix(c(1:6), ncol = 2, byrow = TRUE)
m1
m2
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
## [3,]    5    6
```

```r
3*m1 # scalar multiplication
```

```
##      [,1] [,2] [,3]
## [1,]    3    6    9
## [2,]   12   15   18
```

```r
m1%*%m2
```

```
##      [,1] [,2]
## [1,]   22   28
## [2,]   49   64
```

```r
m2%*%m1
```

```
##      [,1] [,2] [,3]
## [1,]    9   12   15
## [2,]   19   26   33
## [3,]   29   40   51
```

```r
m1*m2
```

```
## Error in m1 * m2: nicht passende Arrays
```

1.2.4 Transpose matrices

Let $\mathbf{M1} = [a_{ij}]$ then $\mathbf{M1}' = [a_{ji}]$

```r
m1
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
```

```r
t(m1)
```

```
##      [,1] [,2]
## [1,]    1    4
## [2,]    2    5
## [3,]    3    6
```

$(\mathbf{M1}')' = \mathbf{M1}$

```r
t(t(m1))
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
```

$\mathbf{M1M1}'$ always yields a symmetric matrix $\mathbf{Q}$

```r
m1%*%t(m1)
```

```
##      [,1] [,2]
## [1,]   14   32
## [2,]   32   77
```

1.2.5 Inverse of matrix

An $n \times n$ matrix $\mathbf{M}$ has an inverse, denoted $\mathbf{M^{-1}}$, provided that $\mathbf{MM^{-1}} = \mathbf{I_n}$

```r
m3 <- matrix(c(2, 5, 1, 3), nrow = 2, byrow = TRUE)
m3
```

```
##      [,1] [,2]
## [1,]    2    5
## [2,]    1    3
```

```r
solve(m3)
```

```
##      [,1] [,2]
## [1,]    3   -5
## [2,]   -1    2
```

```
m3%*%solve(m3)
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    0    1
```

Otherwise it is said to be noninvertible or singular:

```
solve(matrix(c(1:9), nrow = 3))
```

```
## Error in solve.default(matrix(c(1:9), nrow = 3)): Lapackroutine dgesv: System ist genau singulär: U[3
```

1.2.6 Rank of a matrix

```
m4 <- matrix(c(1,2,3,2,4,6,10,20,30), ncol = 3, byrow = FALSE)
m4
```

```
##      [,1] [,2] [,3]
## [1,]    1    2   10
## [2,]    2    4   20
## [3,]    3    6   30
```

```
qr(m4)$rank
```

```
## [1] 1
```

The rank of the matrix is 1, because $Col2 = 2 * Col1$ and $Col3 = 10 * Col1$. This attribute can be used to reduce the size of matrices and data.

1.2.7 Eigenvalues

We use the example of a 3×3 matrix. This could be considered as the variance-covariance matrix of three variables, but the main thing is that the matrix is square and symmetric, which guarrantees that the eigenvalues λ_i are real numbers.

```
m5 <- matrix(c(13, -4, 2, -4, 11, -2, 2, -2, 8), ncol = 3, byrow = TRUE)
m5
```

```
##      [,1] [,2] [,3]
## [1,]   13   -4    2
## [2,]   -4   11   -2
## [3,]    2   -2    8
```

```
eigen(m5)
```

```
## eigen() decomposition
## $values
## [1] 17  8  7
##
## $vectors
##            [,1]       [,2]      [,3]
## [1,]  0.7453560  0.6666667 0.0000000
## [2,] -0.5962848  0.6666667 0.4472136
## [3,]  0.2981424 -0.3333333 0.8944272
```

As shown above, this returns a namend list, containing the eigenvalues and eigenvectors. One may test the result by calculating the required equation for this eigenvalues:

$(A - \lambda I_n)x = 0$

which leads to:

$det(A - \lambda I_n) = 0$

```
m_test1 = m5 - eigen(m5)$values[1] * diag(3)
m_test1
```

```
##      [,1] [,2] [,3]
## [1,]   -4   -4    2
## [2,]   -4   -6   -2
## [3,]    2   -2   -9
```

```
round(det(m_test1), digits = 2)
```

```
## [1] 0
```

Which should also be true for the second eigenvalue:

```
m_test2 = m5 - eigen(m5)$values[2] * diag(3)
round(det(m_test2), digits = 2)
```

```
## [1] 0
```