Tutorial 3 Operations in R

R has many operations for most statistical analyses. For specialised applications it is necessary to install additional packages. For the purpose of this workshop, the default installation is sufficient. Please see the final lecture for instructions on installing packages and extensions.

Basic mathematical operations on single variables

R can be used as a calculator. Use +, -, * and / for basic operations. For example: (Note that the items should be numbers and not strings)

```
a <- 1
b <- 2

a + b
```

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

```
c <- b - a

a * 4
```

```
## [1] 4
```

```
a/4
```

```
## [1] 0.25
```

```
4^2  #where the ^ is 'to the power of'
```

```
## [1] 16
```

Some other operations can be used to compare values, such as >, <, <=, >=, and == (equal to). These return items of class "logical". We will not work with these in depth, but they are useful for some analyses.

```
a < b
```

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

```
a <= b
```

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

```
a == b
```

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

```r
a == 1
```

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

```r
class(a == 1)
```

```
## [1] "logical"
```

These operations are also possible for vectors, matrices and data frames in which all items are numeric:

```r
mymatrix1 <- matrix(1:9, 3, 3)
mymatrix1 * 2
```

```
##      [,1] [,2] [,3]
## [1,]    2    8   14
## [2,]    4   10   16
## [3,]    6   12   18
```

```r
mymatrix1 + 1
```

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

Individual exercise: create a vector and perform some basic operations. Now create an other vector and add them (note that the two vectors should have the same length, and in the case of matrices, they should have the same dimensions). Inspect the results and notice that the operations are conducted individually for every item using the indexing (for vector1 + vector2, vector1[1] is added to vector2[1])

Descriptive statistics

In this exercise we will work with a preloaded data set in R. To obtain the data set:

```r
data(iris)
class(iris)
```

```
## [1] "data.frame"
```

```r
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

The function data() is used to load the data sets available in the default R installation. the function head() prints the first six items of an object, in this case it is the first six rows.

This data set consists of measurements of floral parts and the species.

We will obtain the sum, the mean, the median, the standard deviation and the variance of the sepal length (column Sepal.Length)

```
sum(iris$Sepal.Length)
```

```
## [1] 876.5
```

```
mean(iris$Sepal.Length)
```

```
## [1] 5.843
```

```
median(iris$Sepal.Length)
```

```
## [1] 5.8
```

```
sd(iris$Sepal.Length)
```

```
## [1] 0.8281
```

```
var(iris$Sepal.Length)
```

```
## [1] 0.6857
```

We can also obtain the maximum and minimum values as follows:

```
max(iris$Sepal.Length)
```

```
## [1] 7.9
```

```
min(iris$Sepal.Length)
```

```
## [1] 4.3
```

To get the highest and lowest quartiles use:

```
quantile(iris$Sepal.Length, c(0.25, 0.75))
```

```
## 25% 75%
## 5.1 6.4
```

Where the vector c(0.25, 0.75) refers to the first and last quartiles, respectively.

For a vector, we can obtain some summary statistics with the summary() function.

```r
summary(iris$Sepal.Length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4.30    5.10    5.80    5.84    6.40    7.90
```

Concatenating and combining objects

We have used the function c() to concatenate numbers and characters. This is also possible for vectors:

```r
vector1 <- 1:10
vector2 <- 11:20
vector3 <- c(vector1, vector2)
vector3
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

For matrices the function c() collapses all the items:

```r
mat1 <- matrix(1:9, 3, 3)
mat2 <- matrix(10:18, 3, 3)
mat1
```

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

```r
mat2
```

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

```r
c(mat1)
```

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

```r
c(mat1, mat2)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
```

These two matrices have three columns and three rows, so we could also bind them using the rows or columns. We can use the functions rbind and cbind respectively:

```r
rbind(mat1, mat2)
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
## [4,]   10   13   16
## [5,]   11   14   17
## [6,]   12   15   18
```

```
cbind(mat1, mat2)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    4    7   10   13   16
## [2,]    2    5    8   11   14   17
## [3,]    3    6    9   12   15   18
```