06-23856 Evaluation Methods and Statistics
Autumn Semester 2019

# Lab Class 2: The Basics of Using R

## 1 Open a terminal

There should be a shortcut icon in the taskbar.
Or, crtl-alt-t (three keys simultaneously)
Or, Applications-> System Tools - > Terminal

## 2 Interacting with UNIX shell

pwd tells you where you are in the directory tree
ls lists the files in that directory

Create a new directory for these lab sessions: mkdir R

Move to the new directory: cd R

Create a new directory for this lab class:

mkdir lab_1

cd lab_1

## 3 Basics of R

Based on http://cran.r-project.org/doc/manuals/r-release/R-intro.html (or use the 'help' menu)

From the shell, start the R program: R -> enter

To close R: q()

### 3.1 Simple expressions: type a formula -> enter.

```
> 3^2 + 4^2
[1] 25
```

The answer is shown after the [1]
You can include arithmetic operators (+ - * / ^) and functions (sin, log, exp, sqrt…)

### 3.2 Editing

If there is a mistake in your typing or you get an error message, use the up arrow to recall the previous command. You can step back through several previous lines by continued use of the up arrow.

### 3.3 Assigning to variables

Variables contain data. They do not need to be declared but are created by the system when they are first used.

```
> x<-3.14
```

This will not produce a visible effect but will just have a prompt arrow on the next line

```
>
```

…unless you have mistyped it, e.g.,

```
> x < 3.14
[1] FALSE
```

Now that you have assigned 3.14 to variable x, you can use this is other formula

```
> x+5
[1] 8.14
```

If you assign a new value to x, the system over-writes the previous value. It is a good idea to keep a note of which variables you are using and what values you have assigned to them (particularly when we start using more complex formulae).

# 4. Vectors

## 4.1 Vectors contain sets of data. In a vector, each data point is converted to the same type.

```
> x<-c(1,2,3,4)
> x
[1] 1 2 3 4


> x<-c(3,4,5,4e17)
> x
[1] 3e+00 4e+00 5e+00 4e+17


> x<-c(3, "hello")
> x
[1] "3"     "hello"
```

## 4.2 Applying Operations to vectors

```
> x<-c(1,2,3)
> y<-c(4,5,6)
> x+y
[1] 5 7 9
```

If the vectors are different sizes: i. R will just repeat items in the smaller vector as often as it needs to; ii. you will get a warning message

```
> z<-c(-1, 1)
> x*z
[1] -1  2 -3
Warning message:
In x * z : longer object length is not a multiple of shorter object length
```

You can create sequences easily:

```
> x<-c(1:10)
> x
 [1]  1  2  3  4  5  6  7  8  9 10
```

You can create more specialised sequences:

```
> x<-c(seq(length=10,from=-5, by=.2))
> x
 [1] -5.0 -4.8 -4.6 -4.4 -4.2 -4.0 -3.8 -3.6 -3.4 -3.2
```

# 5. Creating and using Lists

```
> a<-list("a", "B", "C")
> a
[[1]]
[1] "a"

[[2]]
[1] "B"

[[3]]
[1] "C"
```

We can address individual items in the list by their location

```
> a[2]
[[1]]
[1] "B"
```

# 6. Data Frames

R uses ‘data frames’ to deal with tables.

## 6.1 Creating a data frame

So, if we have a table with the following data:

| Name | Number | Position |
|---|---|---|
| Leno | 1 | Goalkeeper |
| Martinez | 26 | Goalkeeper |
| Bellerin | 2 | Defender |
| Sokratis | 5 | Defender |

…we create define each column

```
> n<-c("Leno", " Martinez", " Bellerin", "Sokratis")
> num<-c(1, 26, 2, 5)
> p<-c("goalkeeper", "goalkeeper", "defender", "defender")
```

We can then define the data frame…

```
> squad<-data.frame(name = n, number = num, position = p)
```

…and explore this in terms of rows…

```
> squad[2,]
          name number   position
2 Martinez     26 goalkeeper
```

…or in terms of columns

```
> squad [,2]
[1]  1 26  2  5
```

We can combine the commands to create the data frame into a single command:

```
> Squad.data<-data.frame(n=c("Leno", " Martinez", " Bellerin", "Sokratis"), num=c(1, 26, 2, 5),
p=c("goalkeeper", "goalkeeper", "defender", "defender"))
```

We can display the table:

```
> print(Squad.data)
                  n         num         p
1       Leno        1         goalkeeper
2 Martinez        26        goalkeeper
3   Bellerin      2         defender
4         Sokratis                    5         defender
```

We can interrogate the structure of the data frame:

```
> str(Squad.data)
'data.frame':   4 obs. of  3 variables:
 $ n  : Factor w/ 4 levels " Leno"," Martinez",..: 1 2 3 4
 $ num: num  1 26 2 5
 $ p  : Factor w/ 2 levels "defender","goalkeeper": 2 2 1 1
```

We can produce a summary of the data frame:

```
> print(summary(Squad.data))
                  n         num               p
 Leno             :1        Min.   : 1.00     defender  :2
 Martinez         :1        1st Qu.: 1.75     goalkeeper:2
 Bellerin         :1        Median : 3.50
 Sokratis         :1        Mean   : 8.50
                            3rd Qu.:10.25
                            Max.   :26.00
```

In this example, R is treating the player's shirt numbers as if they were integers and doing some maths with them. This is a bit odd, so we might want to exclude the 'num' column from the data frame:

```
> result<-data.frame(Squad.data$n,Squad.data$p)
> print(result)
      Squad.data.n          Squad.data.p
1       Leno                goalkeeper
2 Martinez                  goalkeeper
3   Bellerin                 defender
4        Sokratis           defender
```

Or we could extract only the first and second row of the data frame:

```
> result<-Squad.data[1:2,]
> result
             n              num        p
1       Leno        1        goalkeeper
2 Martinez        26       goalkeeper
```

We might want to add another column, say the number of matches played (m):

```
> Squad.data$m<-c(6, 2, 3, 5)
> Squad.data
             n              num        p              m
1          Leno             1        goalkeeper       6
2          Martinez         26       goalkeeper       2
3          Bellerin         2        defender         3
4          Sokratis         5        defender         5
```

We might also want to add more players. This requires us to create an additional table and then merge with the first one, using the command 'rbind'.

```
> Squad.newdata<-data.frame(n=c("Ozil", " Torreira", " Lacazette", " Nelson"), num=c(10, 11,9, 24),
p=c ("midfield", "midfield", "forward", "forward"), m=c(2,4,8,3))
> print(Squad.newdata)
             n              num       p         m
1          Ozil       10       midfield 2
2          Torreira           11       midfield 4
3          Lacazette          9        forward 8
4          Nelson  24         forward 3

> Squad.fulldata<-rbind(Squad.data,Squad.newdata)
> print(Squad.fulldata)
                            n num        p              m
1          Leno             1        goalkeeper       6
2          Martinez         26       goalkeeper       2
3          Bellerin         2        defender         3
```

| | | | | | |
|---|---|---|---|---|---|
| 4 | Sokratis | 5 | defender | 5 | |
| 5 | Ozil | 10 | midfield | | 2 |
| 6 | Torreira | 11 | midfield | | 4 |
| 7 | Lacazette | 9 | forward | | 8 |
| 8 | Nelson | 24 | forward | | 3 |

Suppose we want to know whether players with lower squad number play more matches (because they are first team players). We can extract data from the specific columns and make scatterplot.

```
> x<-(Squad,fulldata[,2])
> y<-c(Squad.fulldata[,4])
> plot (x,y)
```

You might need to switch window to see the plot, but it should look like this: