Session 01 - Exercise solutions

João Gonçalves

14 de Julho de 2018

Contents

Using the help system

Exercise 1

```
#1)

# Mean function
?mean

# standard-deviation
?sd

# Calculate quantiles
?quantile

# Calculate the Median Absolute Deviation (MAD)
?mad
```

Exercise 2

2a)

```
# 2a)

??'Weighted Arithmetic Mean'
help.search("Weighted Arithmetic Mean")
```

2b)

```
# 2b)
??'correlation'
help.search("correlation")
```

2c)

```
# 2c)
??'linear model'
help.search("linear model")
```

Operations with vectors

Basic operations

Exercise 3

Define the variables first:

```
x <- 10.2
y <- 5.7
```

3a)

```
# 3a)
(x + y) / 2
```

```
## [1] 7.95
```

3b)

```
# 3b)
(1/2) * x^2 - log10(y)
```

```
## [1] 51.26413
```

3c)

```
# 3c)
z <- sqrt(cos(pi/4)) + 2*x
print(z)
```

```
## [1] 21.2409
```

3d)

```r
# 3d)
round(z, 1)
```

```
## [1] 21.2
```

Concatenating multiple values in vectors

Exercise 4

Start by defining the data vectors:

```r
# t.max vector
t.max <- c(20.7,18.9, 20.8, 18.8, 19.2, 18.6, 20.5,
         20, 19.1, 21.3, 16.9, 18.6)

# est.clim vector
est.clim <- c("Anadia", "Dunas de Mira", "Nelas", "Guarda","Caramulo", "S. Jacinto",
            "Viseu", "Serra da Muna", "Estarreja", "Fig. Cast. Rodrigo", "Arouca/S.Freita",
            "Moimenta da Beira")
```

4a)

```r
# 4a)
sqrt(t.max + 100)
```

```
##  [1] 10.98636 10.90413 10.99091 10.89954 10.91788 10.89036 10.97725
##  [8] 10.95445 10.91329 11.01363 10.81203 10.89036
```

The function was applied to all elements - in R terminology this is a property of base functions called 'recycling'

4b)

```r
# 4b)
length(t.max)
```

```
## [1] 12
```

4c)

```r
# 4c)
mean(t.max)
```

```
## [1] 19.45
```

```r
sd(t.max)
```

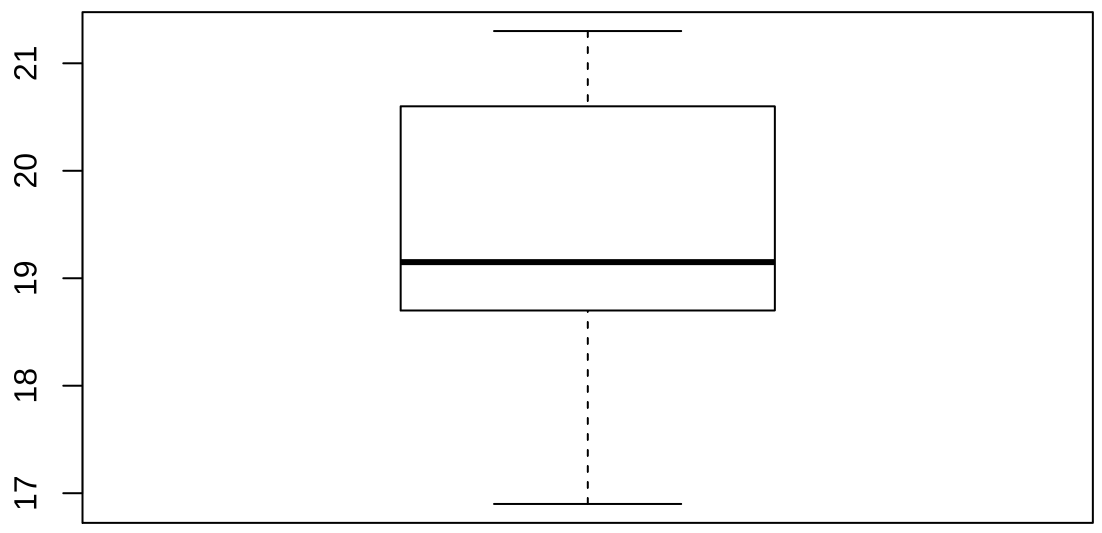
```
## [1] 1.24572
```

4d)

```r
# 4d)

# Calculate quartiles (the default behaviour of quantile function)
quantile(t.max)
```

```
##    0%   25%   50%   75%  100% 
## 16.90 18.75 19.15 20.55 21.30
```

```
# Make a boxplot
boxplot(t.max)
```



```
# Calculate the median
quantile(t.max)["50%"]
```

```
##   50%
## 19.15
```

```
# or, more directly
median(t.max)
```

```
## [1] 19.15
```

Generating sequences of values

Exercise 5

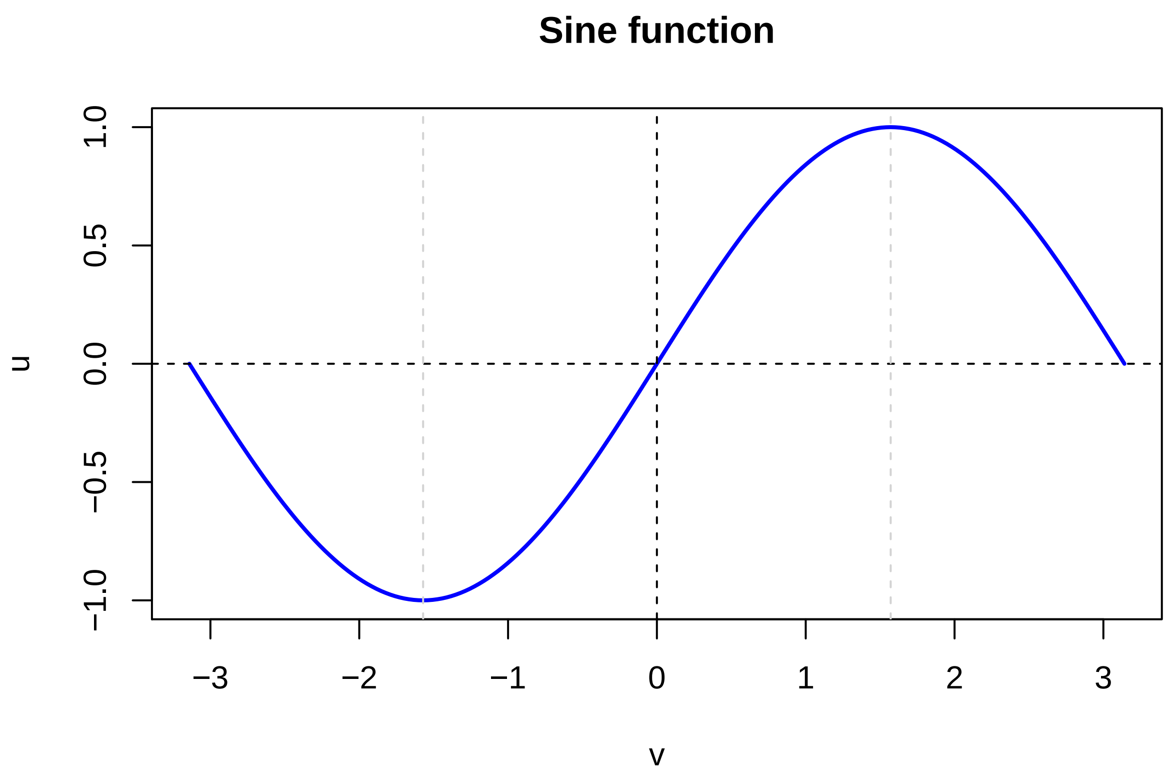
5a)

```
# 5a)

k <- 10:100

sum(k)
```

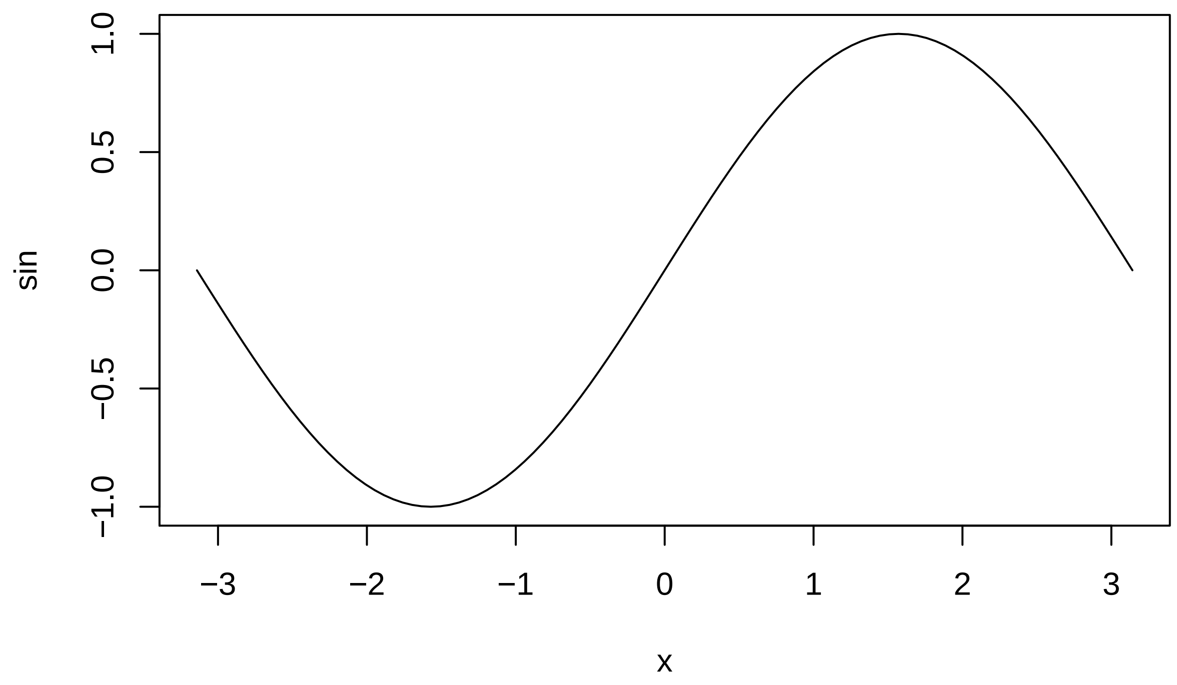
```
## [1] 5005
```

5b)

```r
# 5b)

# Make v vector
v <- seq(-pi, pi, by = pi/100)

# Compute the sin as u
u <- sin(v)

# Now let´s plot the two vectors in x and y
plot(v, u, type="l", lwd=2, col="blue", main="Sine function")

# Make the plot more pretty with some grid lines
abline(h=0,v=0,lty=2,col="black")
abline(v=pi/2,col="light grey",lty=2)
abline(v=-pi/2,col="light grey",lty=2)
```



```r
# Alternatively, plotting a function can be accomplished more directly like this:
plot(sin, -pi, pi) # see ?plot.function
```



Logical operations

Quick exercises

QE 1)

```r
# correct answer:
print("c")
```

```
## [1] "c"
```

```r
# Let's see one example of logical operations recycling:
a <- 1:20
a >= 10
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
## [12]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

QE 2)

```r
# correct answer:
print("c")
```

```
## [1] "c"
```

```r
# Let's check
print(as.numeric(FALSE))
```

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

```r
print(as.numeric(TRUE))
```

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

QE 3)

```r
# correct answer:
print("e")
```

```
## [1] "e"
```

```r
# But let's check:
sum( (1:10) > 5 )
```

```
## [1] 5
```

Vector indexation

Exercise 6 (logical operations and vector indexation)

6a)

```r
# Input the data again (just in case... ;-)

t.max <- c(20.7, 18.9, 20.8, 18.8, 19.2, 18.6, 20.5, 20, 19.1, 21.3, 16.9, 18.6)

est.clim <- c("Anadia", "Mira", "Nelas", "Guarda", "Caramulo", "S. Jacinto", "Viseu",
              "S. Muna", "Estarreja", "Fig. Cast. Rodrigo", "Arouca", "Moimenta")


## 6a) Now define the names attributed to each value in the t.max vector
names(t.max) <- est.clim
print(t.max)
```

```
##             Anadia               Mira              Nelas
##               20.7               18.9               20.8
##             Guarda           Caramulo         S. Jacinto
##               18.8               19.2               18.6
##              Viseu            S. Muna          Estarreja
##               20.5               20.0               19.1
## Fig. Cast. Rodrigo             Arouca           Moimenta
##               21.3               16.9               18.6
```

6b)

```r
# 6b)
t.max[1:5]
```

```
##   Anadia     Mira    Nelas   Guarda Caramulo
##     20.7     18.9     20.8     18.8     19.2
```

6c)

```r
# 6c)
t.max[c("Anadia","Nelas")]
```

```
## Anadia  Nelas
##   20.7   20.8
```

6d)

```
# 6d) Combine two logical conditions separated by () and combined by the & (AND) operator
t.max[(t.max > 20) & (t.max <= 21)]
```

```
## Anadia  Nelas  Viseu
##   20.7   20.8   20.5
```

6e)

```
# 6e)
# Notice that we accessing one element in the left-hand side and attributing a value
# in the right-hand side
t.max["Guarda"] <- 19

# or using position indexation:
t.max[4] <- 19
```

Matrix operations

Matrix indexing

Exercise 7

7a)

```
# 7a) The correct option is:
print("iv")
```

```
## [1] "iv"
```

7b)

```
# 7b) Input the data into R
smokers<-matrix(c(50.3, 60.6, 71.5, 82.3, 59.9, 79.3, 41.4, 80.9, 72.1, 59.1,
                  54.4, 52.4, 67.1, 78.3, 59.2, 65.1, 86.3, 81.3, 57.3, 61.3),
                ncol=10,nrow=2,byrow=TRUE)

# Add row and column names
rownames(smokers) <- c("W","M")
colnames(smokers) <- c("Aveiro", "Braga", "Bragança", "Porto", "Coimbra",
                       "Covilhã", "Leiria", "Lisboa", "Setúbal", "Faro")
print(smokers)
```

```
##   Aveiro Braga Bragança Porto Coimbra Covilhã Leiria Lisboa Setúbal Faro
## W   50.3  60.6     71.5  82.3    59.9    79.3   41.4   80.9    72.1 59.1
## M   54.4  52.4     67.1  78.3    59.2    65.1   86.3   81.3    57.3 61.3
```

7c)

```
# 7c) Index the column by name
mean(smokers["M",])
```

```
## [1] 66.27
```

```r
# or, by row number
mean(smokers[2,])
```

```
## [1] 66.27
```

7d)

```r
# 7d) Use sorting based on the first row (women data by city)
#
sort(smokers[1,], decreasing=TRUE)[1:2]
```

```
##  Porto Lisboa
##   82.3   80.9
```

7e)

```r
# 7e) Use sorting based on the absolute difference between groups
#
sort(abs(smokers[2,] - smokers[1,]), decreasing=TRUE)[1]
```

```
## Leiria
##   44.9
```

7f)

```r
# 7f) calculate by column the index of where the maximum value is 1 - women and 2 - men
#
smkStats <- apply(smokers, 2, which.max)

# Now change the resulting vector from above
smkStats[smkStats == 1] <- "Women"
smkStats[smkStats == 2] <- "Men"

print(smkStats)
```

```
##   Aveiro    Braga Bragança    Porto  Coimbra  Covilhã   Leiria   Lisboa
##    "Men"  "Women"  "Women"  "Women"  "Women"  "Women"    "Men"    "Men"
##  Setúbal     Faro
##  "Women"    "Men"
```