Intro to R Programming: Assignment 2

You can obtain a total of 30 marks for this problem sheet. Please upload your answers **before 20th November 2018, 12noon**. To upload your answers, log on to Moodle and go to *Introduction to R Programming*. Under *Assignments*, there is a link *Upload answers for assignment 2*. Click on this link to upload the R file containing your R code and your comments. You do not need to upload R output. Please do not upload Word documents or zip files. The name of the file **must** be your student ID number.

Under *Assignments* on Moodle there is also a folder *Data files for assignment 2* which includes all data files needed for this assignment.

Task 1

The effective life (in hours) of batteries is compared by material type (A, B or C) and operating temperature: Low, Medium or High. Eighteen batteries are randomly selected from each material type and are then randomly allocated to each temperature level. We assume that two batteries where allocated to each combination of material type and temperature. The following table reports the average life of the batteries for each combination of material type and temperature.

		Temperature		
		Low	Medium	High
Material	A	134.75	57.25	57.5
	B	155.75	119.75	198
	C	144	145.75	85.5

In this question we fit the model

$$y_{ij} = \mu + \alpha_i + \beta_j + \varepsilon_{ij},$$

which will explain the average battery life y_{ij} for material type i and temperature j as stemming from an overall mean μ, a material type effect α_i and a temperature effect β_j. ε_{ij} is the residual, modelling the noise in the experiment. This model is sometimes called a "two-way ANOVA model".

There are simple formulae for computing the estimates of the parameters

Mean: $$\hat{\mu} = \frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=1}^{n} y_{ij}$$

Material effect: $$\hat{\alpha}_i = \frac{1}{n} \sum_{j=1}^{n} y_{ij} - \hat{\mu}$$

Temperature effect: $$\hat{\beta}_j = \frac{1}{m} \sum_{i=1}^{m} y_{ij} - \hat{\mu},$$

where m is the number of rows (i.e. types of material) and n is the number of columns (i.e. temperature levels).

1. **[1 mark]** Create the matrix `battery` reporting the numbers in the above table.
2. **[1 mark]** Give the associated names to the rows and columns of `battery`.
3. **[1 mark]** Compute $\hat{\mu}$.
4. **[1 mark]** Use `apply` to compute $\hat{\boldsymbol{\alpha}} = (\hat{\alpha}_1, \dots, \hat{\alpha}_m)$ and $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \dots, \hat{\beta}_n)$.

5. **[2 marks]** Use `sweep` to compute the matrix of residuals

$$\hat{\varepsilon}_{ij} = y_{ij} - \hat{\mu} - \hat{\alpha}_i - \hat{\beta}_j.$$

You have to apply the function `sweep` *twice when computing the residuals: once to remove the material type effect and once to remove the temperature effect.*

6. **[1 mark]** Use the matrix of point 5 to compute the residual sum of squares

$$RSS = \sum_{i=1}^{m} \sum_{j=1}^{n} \varepsilon_{ij}^2.$$

7. **[1 mark]** Compute the test statistic for the F-test whether the material type is affecting the life of the batteries. The test statistic is

$$F_\alpha = \frac{\dfrac{SS_\alpha}{m-1}}{\dfrac{RSS}{(m-1)\cdot(n-1)}},$$

where $SS_\alpha = n \cdot \sum_{i=1}^{m} \alpha_i^2$. If F_α is greater than the corresponding critical value (4.46), then there is significant evidence for an effect of the material.

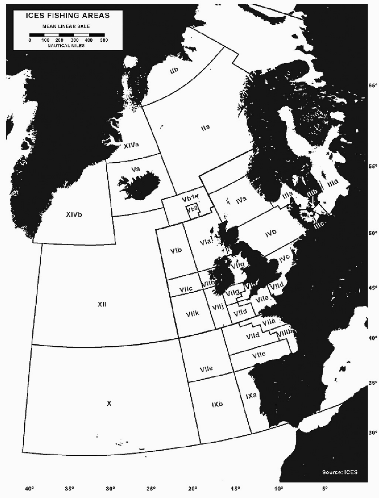
8. **[1 mark]** Compute the test statistic for the F-test whether the temperature is affecting the life of the batteries. The test statistic is

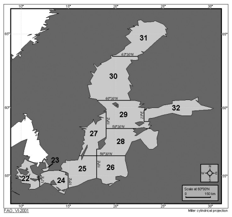
$$F_\beta = \frac{\dfrac{SS_\beta}{n-1}}{\dfrac{RSS}{(m-1)\cdot(n-1)}},$$

where $SS_\beta = m \cdot \sum_{j=1}^{n} \beta_j^2$. If F_β is greater than the corresponding critical value (4.46), then there is significant evidence for an effect of the temperature.

Task 2

The International Council for the Exploitation of the Sea (ICES) is an organisation that coordinates and promotes marine research in the North Atlantic. One of the main tasks of ICES is collecting and consolidating data. One of the data collected by ICES are the yearly catch records for over 200 species in the Northeast Atlantic Ocean. The data frame `ices` (in file `ices.RDS`) contains the (more or less) raw catch records of the year 2000. The data frame consists of the yearly catch (in 1,000 tons) per species and per subdivision (region). Maps of the relevant subdivisions of the Northeast Atlantic are shown below.





1. [**1 mark**] Species not abundant in a certain subdivisions are encoded using `NA`. In order to be able to analyse this data further, replace all `NA`'s by `0`.
2. [**1 mark**] Determine the total catch for each subdivision.
3. [**2 marks**] Remove the subdivisions in which the total catch is 0.
4. [**1 mark**] Determine the total catch for each species.
5. [**1 mark**] Which three species are the most abundant?
6. [**1 mark**] To analyse the types of fishery operating in different subdivisions it is useful to convert the table of total catches to a table that contains the composition of the catches in percent. This can be done by dividing each row of the data frame by its sum. *Hint: Use your result from part 3 and the function* `sweep` *or* `scale`.
7. [**2 marks**] For each subdivision determine the species that is most abundant. *Hint: The function* `which.max` *might be of use.*

Task 3

Suppose you are given a data set consisting of observations $x_1, \ldots, x_n$ and you want to find the mode of the distribution of the x_i's. This can be achieved using a method called the *mean shift*. The mean shift algorithm is guaranteed to converge to a local mode of the distribution of the x_i's. The algorithm is as follows:

1. Set $\mu^{(0)}$ to the median of $x_1, \ldots, x_n$.
2. For $k = 1, 2, \ldots$:

- Compute weights $w_i^{(k)} = \exp\left(-\frac{(x_i-\mu^{(k-1)})^2}{2h^2}\right)$ for $i = 1, \ldots, n$.
- Set $\mu^{(k)} = \frac{\sum_{i=1}^n w_i^{(k)} x_i}{\sum_{i=1}^n w_i^{(k)}}$.

Write a function `mean.shift` which takes as input the vector of observations `x`, the parameter `h` and the number of iterations `it`, carries out the above algorithm and returns the final $\mu^{(k)}$. The algorithm should stop after the given number of iterations or if $\mu^{(k)}$ changes by less than 10^{-8}, whichever happens first. The default value of `h` should be $\frac{\hat{\sigma}}{n^{1/5}}$, where $\hat{\sigma}$ is the standard deviation of $x_1, \ldots, x_n$, and the default value of `it` should be 100. The function should return the error `"h cannot be zero"` if $h = 0$ and `"it must be a positive integer"` if the number of iterations is not a positive integer.

Your function will be marked depending on the number of unit tests it will pass. You will receive 1 mark each time the output of your function equals the ones below (this way you can predict the score for this question). We will use the `gamma.RDS` R file:

```
x <- readRDS("gamma.RDS")
mean.shift(x)
```

```
## [1] 4.040046
```

```
mean.shift(x,h=0)
```

```
Error in mean.shift(x, h = 0) :  h must be different to zero
```

```
 mean.shift(x,h=1.2)
```

```
## [1] 4.240553
```

```
mean.shift(x,it=-10)
```

```
Error in mean.shift(x, it = -10) :  it must be a positive integer
```

```
mean.shift(x,it=4)
```

```
## [1] 4.571908
```

```r
mean.shift(c(1,1,2),it=1)
```

```
## [1] 1.04649
```

Task 4

The Collatz conjecture is a conjecture in mathematics that concerns a sequence defined as follows: start with any positive integer n. Then each term is obtained from the previous term as follows: if the previous term is even, the next term is one half the previous term. If the previous term is odd, the next term is 3 times the previous term plus 1. The conjecture is that no matter what value of n, the sequence will always reach 1.

Using the rule above and starting with 13, we generate the following sequence:

$$13 \to 40 \to 20 \to 10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1$$

Write the function `collatz` which takes as input the starting value of the sequence `n` and the length of the sequence `length` and computes the sequence described above. The default value `length` is 30. The algorithm should stop whenever the number 1 is reached. The output of `collatz` is the sequence from the starting value `n` to either `1` if this is reached, or the sequence starting from `n` of length `length`. The function should report the error `Input not positive` if $n \leq 0$ and the error `Input not integer` if n is not an integer.

Again, your function will be marked depending on the number of unit tests it will pass. You will receive 1 mark each time the output of your function equals the ones below.

```r
collatz(14)
```

```
##  [1] 14  7 22 11 34 17 52 26 13 40 20 10  5 16  8  4  2  1
```

```r
collatz(2)
```

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

```r
collatz(17,5)
```

```
## [1] 17 52 26 13 40
```

```r
collatz(10001)
```

```
##  [1] 10001 30004 15002  7501 22504 11252  5626  2813  8440  4220  2110
## [12]  1055  3166  1583  4750  2375  7126  3563 10690  5345 16036  8018
## [23]  4009 12028  6014  3007  9022  4511 13534  6767
```

```r
collatz(0.8)
```

```
Error in collatz(0.8) :  Input not integer
```

```r
collatz(-2)
```

```
Error in collatz(-2) :  Input not positive
```