CT5102: Programing for Data Analytics

Problem Sheet – Lists and Functions

1. Write a function that:

   - Accepts a variable number of vectors
   - For each vector, calculates the minimum, maximum, mean and standard deviation
   - Returns the information in a list, so that the input data and output results are contained in successive list elements.

All operations should be vectorized, and no loops are permitted. In order to process the vectors and generate a list, R's **lapply** function should be used (lapply returns a list). The general form of the **lapply(x,f,fargs)** function is as follows:

- **x** is the target vector or list
- **f** is the function to be called
- **fargs** are the optional set of arguments that can be applied to the function **f**.

Sample output from the function is shown:

```
> ans<-process.vectors(c(1,2,3),34:78,c(8,9,10))
> str(ans)
List of 3
 $ :List of 5
  ..$ data: num [1:3] 1 2 3
  ..$ min : num 1
  ..$ max : num 3
  ..$ mean: num 2
  ..$ sd  : num 1
 $ :List of 5
  ..$ data: int [1:45] 34 35 36 37 38 39 40 41 42 43 ...
  ..$ min : int 34
  ..$ max : int 78
  ..$ mean: num 56
  ..$ sd  : num 13.1
 $ :List of 5
  ..$ data: num [1:3] 8 9 10
  ..$ min : num 8
  ..$ max : num 10
  ..$ mean: num 9
  ..$ sd  : num 1
```

2. Write a function called my_rmse(v1, v2, na.rm=F) that calculates the root mean square error between two numeric vectors. The function should have the following behaviours:

   - It must check that both vectors entered are numeric. If not, the function should halt.

   - If the user specifies that na.rm=F *AND* there are any NA values in either vector, the function should halt with an appropriate message.

   - If the user specifies that na.rm = TRUE, then the function must filter out all NA values from both vectors. If a vector v1 has an NA in position *i*, then the corresponding *ith* position in v2 must also be omitted (and vice versa).

   - Test the function so that the following scenarios are evaluated, namely:

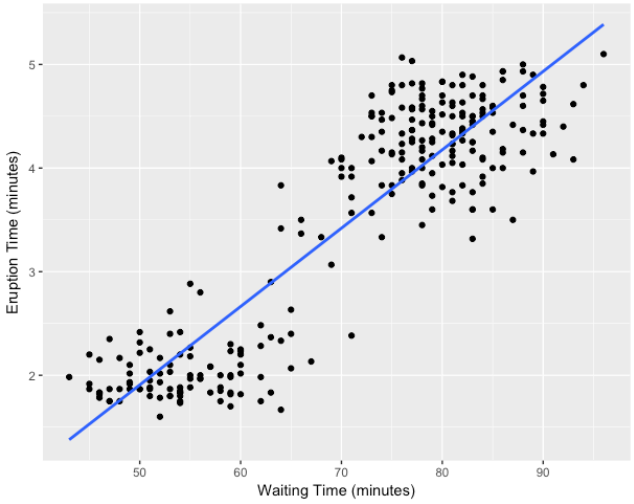
Test	**v1**	**v2**	**na.rm**	**Desired Output**
1	Non-numeric	Numeric	F	Invalid v1 vector
2	Numeric	Non-Numeric	F	Invalid v2 vector
3	Numeric (with NAs)	Numeric (no NAs)	F	Invalid v1 with NA
4	Numeric (no NAs)	Numeric (with NAs)	F	Invalid v2 with NA
5	Numeric (no NAs)	Numeric (no NAs)	F	Correct RSME
6	Numeric (with NAs)	Numeric (with NAs)	T	Correct RSME

Note, for successful cases, the function **rsme()** from the CRAN package Metrics[1] should be used to validate output.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

[1] https://cran.r-project.org/web/packages/Metrics/Metrics.pdf

3. In R, the list data structure can be used to return data mining solutions from learning algorithms. A good example is the **lm()** function which performs regression analysis on a data set. For example, the **faithful** data set in R contains information on an independent variable (x - waiting time), and a dependent variable (y - eruption time) for the Old Faithful geyser. The data (and linear model) is shown in the chart below (using the ggplot2 library).




```
ggplot(faithful,aes(x=waiting,y=eruptions))+
       geom_point()+stat_smooth(method = "lm", se = F)+
       xlab("Waiting Time (minutes)")+
       ylab("Eruption Time (minutes)")
```

The linear model can be directly calculated using the **lm()** function, as follows:

```
er_mod <- lm(eruptions ~ waiting, data=faithful)
```

The **er_mod** variable is a complex list structure.

From this structure you t extract the two model coefficients (i.e. the intercept and the slope). They should have the following values.

```
(Intercept)                    waiting
  -1.874016                  0.07562795
```

4. Write a function (here, the function is called **f()**), which takes in a numeric vector, and returns (in a list) four elements:

    - The original data vector
    - The minimum value in the vector
    - The maximum value in the vector
    - The mean value in the vector

The function should be robust and return error messages under two conditions:

- If the vector is not numeric, it should return immediately
- If there are any NA values in a numeric vector, it should return immediately.

Here are sample test cases for the function:

```
> f(c(1:9,"Test"))
Error in f(c(1:9, "Test")) : Error, type should be numeric!


> f(c(1:9,NA))
Error in f(c(1:9, NA)) : Error, Cannot have NAs!

> f(c(1:10))
$Data
 [1]  1  2  3  4  5  6  7  8  9 10

$Min
[1] 1

$Max
[1] 10

$Mean
[1] 5.5
```

5. Write a function that takes in a variable number of vectors, and returns a list of aggregate information on each vector. An apply function should be used to generate the result. The returned information should include the following:
    - vector length
    - maximum value
    - minimum value
    - median
    - standard deviation.

6. For the function f(), what will each of the calls below evaluate to?

```
f <- function(abc, ac, def){
  list(First=abc, Second=ac, Third=def)
}

f(1,2,3)
f(1,2,abc=3)
f(3,2,d=1)
f(1,2,a=3)
```

7. Create a vector of 100 random numbers in the range 1:10 (numbers can be repeated). Write a function to remove all duplicates from this vector, using the R function duplicated().

8. For the list l <- list(c(1,2),list(4:5,6:7))

   - Visualise the list
   - Explain the difference between l[1] and l[[1]]
   - Visualise the results of the following:
     - l[2]
     - l[[2]]
     - l[[1]]
     - l[1:2]