Devoir R

Bryan Tchakote

11/30/2020

Contents

1 Exercice 2.2.1

```
# On considère deux variables x et y reliées par la relation
# y = (2/3)*x^3 - (1/2)*x - 5.

# 1. Fonction formule y.val prenant en entrée x et renvoyant la valeur de y correspondante.
y.val = function(x) return((2/3)*x^3 - (1/2)*x - 5)

# 2. Vecteur vecy des valeurs de y associées aux valeurs -2, 1.7, 3, 10, -7 de x.
vecy = y.val(c(-2, 1.7, 3, 10, -7))
vecy
```

```
## [1]   -9.333333   -2.574667   11.500000  656.666667 -230.166667
```

2 Exercice 2.2.2 (Nombres premiers)

```
# 1. Fonction 'premier' prenant en entrée un entier naturel n et renvoyant TRUE si n est
#    premier et FALSE sinon.
premier = function(n)
{
  if(n < 2) return (FALSE)
  if(n == 2) return (TRUE)
  for(i in 2:sqrt(n))
    if((n %% i) == 0) return (FALSE)
  return (TRUE)
}

# Vectorisation de la fonction 'premier'
# sapply(v, FUN)
```

```r
# v -> vecteur
# FUN -> fonction a appliquer a chaque element du vecteur
premier_v = function(n) return (sapply(n, premier))

# 2. Nombres premiers  100.
which(premier_v(1:100))
```

```
##  [1]  2  3  5  7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

3 Exercice 2.3.1 (is.na, tapply)

```r
# 1.1. Description du jeu de données 'airquality'.
?airquality
```

```
## starting httpd help server ... done
```

```r
## 'airquality' est un jeu de données sous R décrivant la qualité journalière de l'air
## dans l'Etat de New-York (USA) de mai à septembre 1973
```

```r
# 1.2. Premières lignes du jeu de données
head(airquality)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
## 6    28      NA 14.9   66     5   6
```

```r
# 1.3 Dimensions du tableau de données
dim(airquality)
```

```
## [1] 153   6
```

```r
## 153 lignes et 6 colonnes
```

```r
# 2. Nombre de valeurs manquantes pour la concentration d'ozone
 sum(is.na(airquality$Ozone))
```

```
## [1] 37
```

```r
## 37 valeurs relatives à la concentration d'ozone manquent dans le jeu de données 'airquality'

# Nombre de valeurs manquantes par variable
valeurs_manquantes = sapply(1:ncol(airquality), function(j){
  return(sum(is.na(airquality[, j])))
})
names(valeurs_manquantes) = names(airquality)
valeurs_manquantes
```

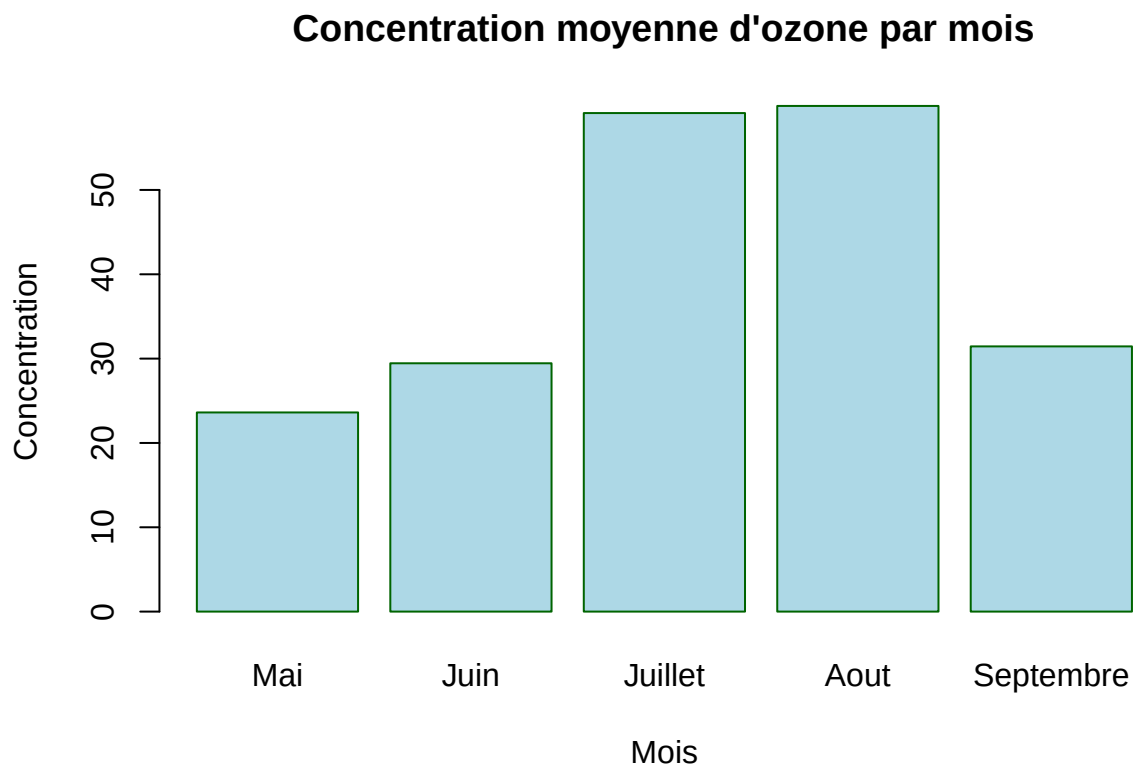
```
##   Ozone Solar.R    Wind    Temp   Month     Day
##      37       7       0       0       0       0
```

```r
# 3. Concentration d'ozone moyenne et variance pour les mois de mai, juillet et septembre
moyenne = tapply(airquality$Ozone, airquality$Month, mean,na.rm=TRUE)
variance = tapply(airquality$Ozone, airquality$Month, function(x){
```

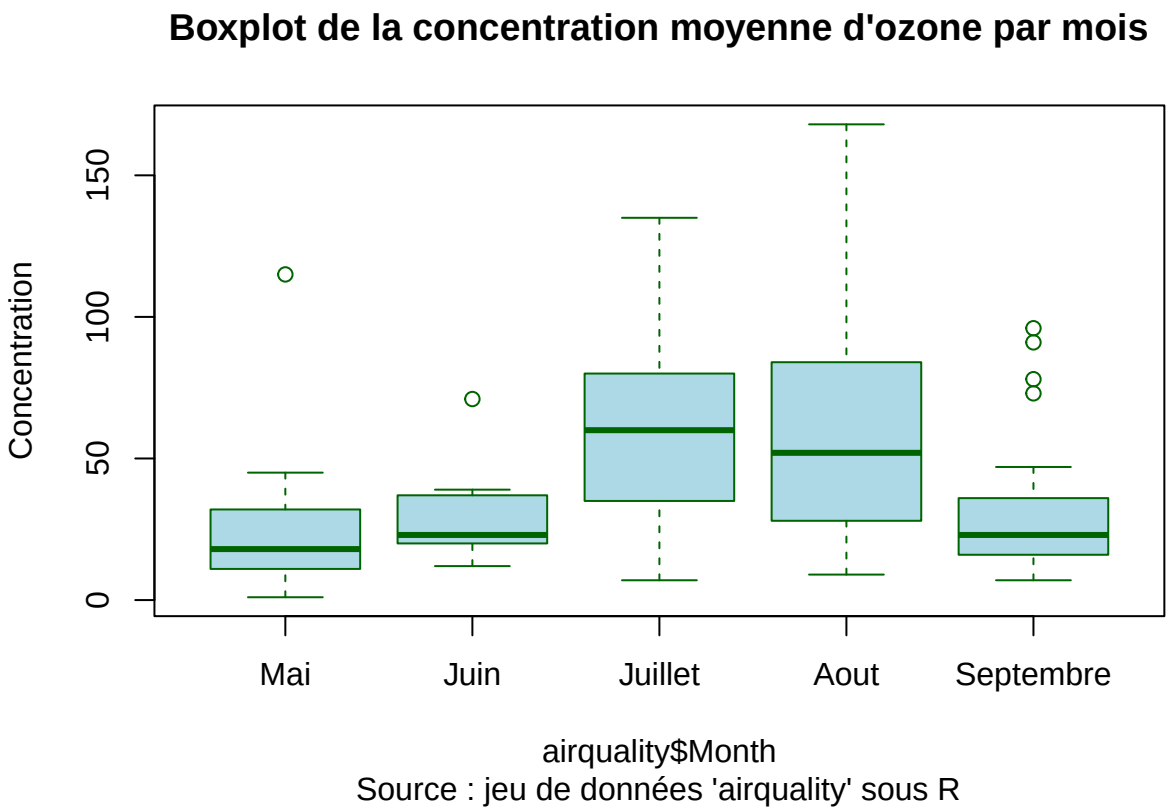
```r
  nn = length(na.omit(x))
  return (var(na.omit(x))*(nn-1)/nn)
})
mois = c("Mai", "Juin", "Juillet", "Aout", "Septembre")
data.frame(mois, moyenne, variance)
```

```
##        mois  moyenne  variance
## 5       Mai 23.61538  474.9290
## 6      Juin 29.44444  294.6914
## 7   Juillet 59.11538  962.3328
## 8      Aout 59.96154 1514.0370
## 9 Septembre 31.44828  562.7301
```

```r
# 4. Diagramme à barres de la concentration moyenne d'ozone par mois.
barplot(moyenne, names.arg = mois, col = "lightblue", border = "darkgreen",
        main = "Concentration moyenne d'ozone par mois",
        ylab = "Concentration", xlab = "Mois")
```




```r
# 5. Boxplot de la concentration moyenne d'ozone par mois.
boxplot(airquality$Ozone~airquality$Month, col = "lightblue", border = "darkgreen",
        main = "Boxplot de la concentration moyenne d'ozone par mois", names = mois,
        sub = "Source : jeu de données 'airquality' sous R", ylab = "Concentration")
```



4 Exercice 2.3.2 (Écrire et charger un jeu de données)

```
# 2.1. Description du jeu de données 'iris'
?iris
## 'iris' est un jeu de données sous R fournissant des informations sur les dimensions en
## centimètres des sépales et des pétales d'un échantillon de 3 espèces différentes d'iris

# 2.2. Variables du jeu de données
names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```
# 2.3 Dimensions du jeu de données
dim(iris)
```

```
## [1] 150   5
```

```
## 150 lignes et 5 colonnes

# 3. Écriture du jeu de données dans le fichier iris.txt
write.table(iris, file = "iris.txt", row.names = FALSE)

# 4. Chargement des données de 'iris.txt' dans 'iris2'
iris2 = read.table("iris.txt", header = TRUE)
head(iris2)
```

```
##   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
```

```
# Indicateurs statistiques par espèces pour chaque valeur quantitative
moyenne1 = sapply(1:(ncol(iris)-1), function(x) return (tapply(iris[, x], iris$Species, mean)))
colnames(moyenne1) = names(iris)[-ncol(iris)]
moyenne1
```

```
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa            5.006       3.428        1.462       0.246
## versicolor        5.936       2.770        4.260       1.326
## virginica         6.588       2.974        5.552       2.026
```

```
variance = sapply(1:(ncol(iris)-1),
                  function(x) return (tapply(iris[, x], iris$Species,
                                     function(x){
                                       n = length(x)
                                       return (var(x)*(n-1)/n)
                                     })))
colnames(variance) = colnames(moyenne1)
variance
```

```
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa         0.121764    0.140816     0.029556    0.010884
## versicolor     0.261104    0.096500     0.216400    0.038324
## virginica      0.396256    0.101924     0.298496    0.073924
```

```
ecart_type = sqrt(variance)
colnames(ecart_type) = colnames(moyenne1)
ecart_type
```

```
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa        0.3489470   0.3752546    0.1719186   0.1043264
## versicolor    0.5109834   0.3106445    0.4651881   0.1957652
## virginica     0.6294887   0.3192554    0.5463479   0.2718897
```

```
mediane = sapply(1:(ncol(iris)-1), function(x) return (tapply(iris[, x], iris$Species, median)))
colnames(mediane) = colnames(moyenne1)
mediane
```

```
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa              5.0         3.4         1.50         0.2
## versicolor          5.9         2.8         4.35         1.3
## virginica           6.5         3.0         5.55         2.0
```

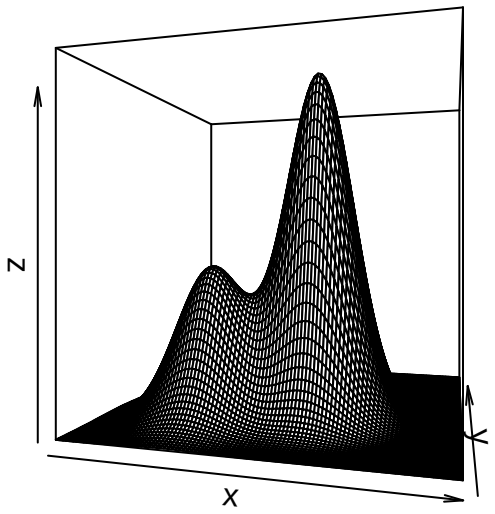
```
# apply(T, [1 ou 2], FUN)
# T -> tableau à deux dimensions
# 1 -> Parcours de chaque ligne / 2 -> Parcours de chaque colonne
# FUN -> fonction à appliquer sur la dimension choisie
moyenne2 = apply(iris[-ncol(iris)], 2, function(x) return (tapply(x, iris[ncol(iris)], mean)))
as.data.frame(moyenne2)
```

```
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa            5.006       3.428        1.462       0.246
```

```
## versicolor        5.936       2.770        4.260       1.326
## virginica         6.588       2.974        5.552       2.026
```

5 Quelques graphiques

```
x = seq(0, 10, length = 100)
y = seq(0, 10, length = 100)
f = function(x, y) return ((0.3/sqrt(2*pi))*exp(-0.5*((x-3)^2+(y-3)^2)) +
                        (0.7/sqrt(2*pi))*exp(-0.5*((x-6)^2+(y-4)^2)))
z = outer(x, y, f)
persp(x, y, z, theta = 15, phi = 0)
```




```
# contour(z, nlevels = 10)
```