Exercices Fouilles de données

Tom BLACHON Meyssa BEDDAR Matthieu SIMOES

Exercice 1

D'après l'axiome A1,

$$d(x,x) = 0$$

D'après l'axiome A3,

$$d(x,x) \leq d(x,y) + d(y,x)$$

Et d'après l'axiome A2,

$$d(y,x) = d(x,y)$$

On a donc :

$$d(x,x) = 0 \leq d(x,y) + d(y,x) = 2d(x,y)$$

Or

$$0 \leq 2d(x,y) \iff 0 \leq d(x,y)$$

On peut ainsi conclure que les trois axiomes impliquent la non négativité.

Exercice 2

La distance euclidienne est définie de la manière suivante :

$$d(x,y) = \sqrt{\sum_{j=1}^{m}(x_j - y_j)^2}$$

Voyons si elle répond aux trois axiomes :

Axiome 1 :

$$d(x,y) = 0 \iff x = y$$

$\Longleftarrow$

Soit x=y. Alors on a :

$$d(x,y) = \sqrt{\sum_{j=1}^{m}(x_j - y_j)^2} = \sqrt{\sum_{j=1}^{m}(x_j - x_j)^2} = 0$$

$\Longrightarrow$

$$\sqrt{\sum_{j=1}^{m}(x_j - y_j)^2} = 0 \iff \sum_{j=1}^{m}(x_j - y_j)^2 = 0 \iff x = y$$

Axiome 2 :

$$d(y,x) = d(x,y)$$

$$d(x,y) = \sqrt{\sum_{j=1}^{m}(x_j - y_j)^2} = \sqrt{\sum_{j=1}^{m}(y_j - x_j)^2} = d(y,x)$$

Axiome 3 :

$$d(x,z) \leq d(x,y) + d(y,z)$$

On a :

$$d(x,y)+d(y,z) = \sqrt{\sum_{j=1}^{m}(x_j - y_j)^2}+\sqrt{\sum_{j=1}^{m}(y_j - z_j)^2} \geq \sqrt{\sum_{j=1}^{m}((x_j - y_j) + (y_j - z_j))^2} = \sqrt{\sum_{j=1}^{m}(x_j - z_j)^2} = d(x,z)$$

Ainsi, la distance euclidienne est bien une vraie distance au regard des 3 axiomes.

Exercice 3

La distance de Canberra se définit de la manière suivante :

$$d(x,y) = \sum_{j=1}^{m} \frac{|x_j - y_j|}{|x_j| + |y_j|}$$

Axiome 1 :

$\Longleftarrow$

Posons x=y

Alors on a :

$$\sum_{j=1}^{m} \frac{|x_j - y_j|}{|x_j| + |y_j|} = \frac{|x_1 - y_1|}{|x_1| + |y_1|} + \frac{|x_2 - y_2|}{|x_2| + |y_2|} + ... + \frac{|x_m - y_m|}{|x_m| + |y_m|}$$

$$= \frac{|x_1 - x_1|}{|x_1| + |x_1|} + \frac{|x_2 - x_2|}{|x_2| + |x_2|} + ... + \frac{|x_m - x_m|}{|x_m| + |x_m|} = 0$$

$\Longrightarrow$

$$d(x,y) = 0 \iff \sum_{j=1}^{m} \frac{|x_j - y_j|}{|x_j| + |y_j|} = 0$$

$$\iff \sum_{j=1}^{m} |x_j - y_j| = 0$$

$$
\begin{aligned}
&\iff |x_1 - y_1| + ... + |x_m - y_m| = 0 \\
&\iff \begin{aligned} |x_1 - y_1| &= 0 \\ &\vdots \\ |x_m - y_m| &= 0 \end{aligned} \\
&\iff \begin{aligned} x_1 &= y_1 \\ &\vdots \\ x_m &= y_m \end{aligned}
\end{aligned}
$$

Axiome A2 :

$$d(x,y) = \sum_{j=1}^{m} \frac{|x_j - y_j|}{|x_j| + |y_j|} = \sum_{j=1}^{m} \frac{|y_j - x_j|}{|y_j| + |x_j|} = d(y,x)$$

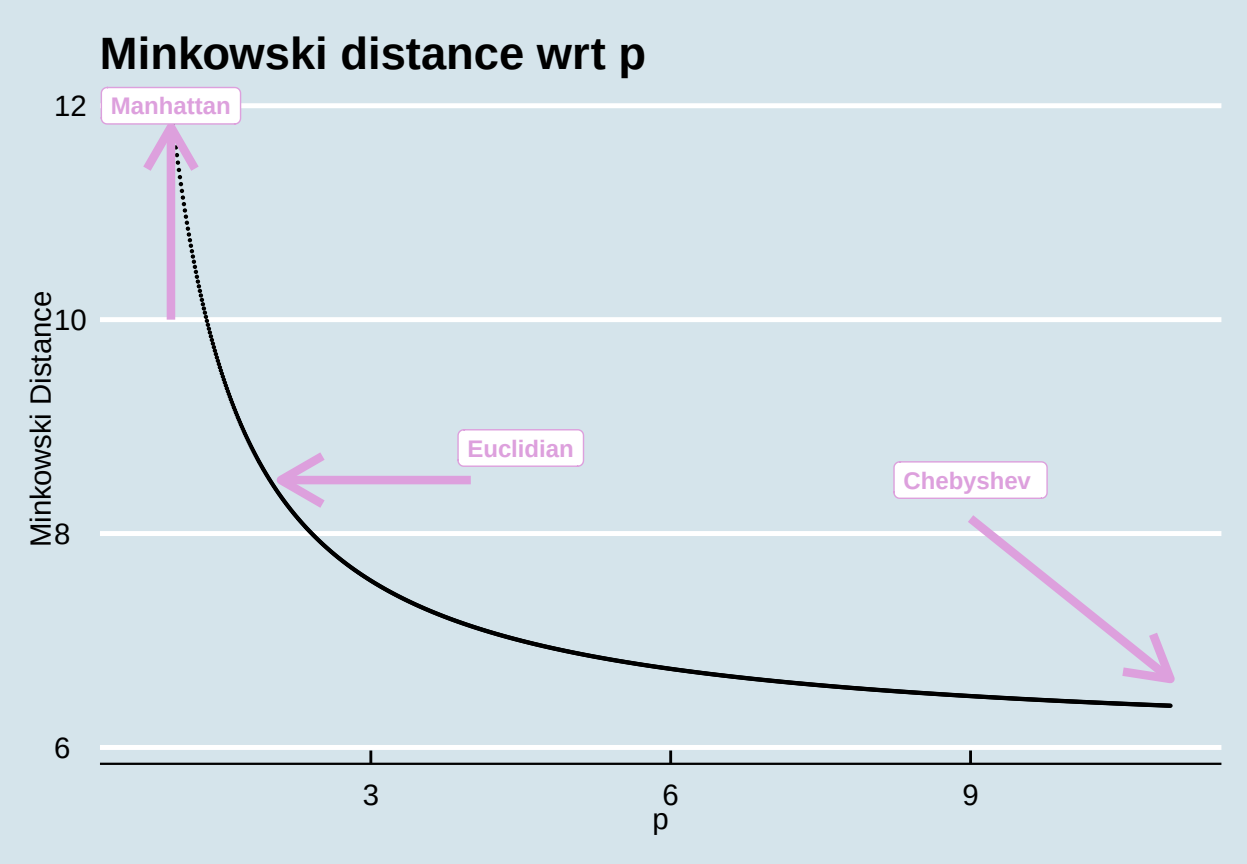
Exercice 4

```
#Création de deux vecteurs pour le calcul des distances
x = c(0, 0)
y = c(6,6)

# Calcul des distances selon p variant de 1 à 30 avec un pas de 0.01
MinkowDist=c() # Initialisation de la liste MinkowDist à vide
for (p in seq(1,30,.01))
{
  MinkowDist=c(MinkowDist,dist(rbind(x, y), method = "minkowski", p = p))
}

# Annotations pour le graphique
annotation <- data.frame(
   x = c(1,4.5,9),
   y = c(12,8.8,8.5),
   label = c("Manhattan", "Euclidian","Chebyshev ")
)

# Graphique des distances de Minkowski selon p avec annotations
ggplot(data =data.frame(x = seq(1,30,.01), y=MinkowDist) , mapping = aes( x=x, y= y))+
  geom_point(size=.1,color="black")+theme_economist()+
  xlab("p")+ylab("Minkowski Distance")+ ggtitle("Minkowski distance wrt p") +
  xlim(0.8,11) + annotate("segment", x = c(1,4,9), xend = c(1,2.1,11),
                          y = c(MinkowDist[1]-2,MinkowDist[100],min(MinkowDist)+2),
                          yend = c(MinkowDist[1]-0.2,MinkowDist[100],min(MinkowDist)+0.5),
                          colour = "plum", size = 1.5, arrow = arrow()) +
  geom_label(data=annotation, aes( x=x, y=y, label=label), color="plum",
             size=3 , angle=45, fontface="bold" )
```



Exercice 5

Ajustement pour la fonction rlnorm et création des deux listes de 100 tirages (moyenne de 1600 et écart-type de 300)

```
#Moyenne
m <- 1600
#Ecart-type
s <- 300

#Ajustement
location <- log(m^2 / sqrt(s^2 + m^2))
shape <- sqrt(log(1 + (s^2 / m^2)))

#Création des listes draws1 et draws2
draws1 <- rlnorm(n=100, location, shape)
draws2 <- rlnorm(n=100, location, shape)
# Vérification des moyennes et écarts-types
mean(draws1)
```

```
## [1] 1592.618
```

```
mean(draws2)
```

```
## [1] 1545.407
```

```
sd(draws1)
```

```
## [1] 309.1432
```

```
sd(draws2)
```

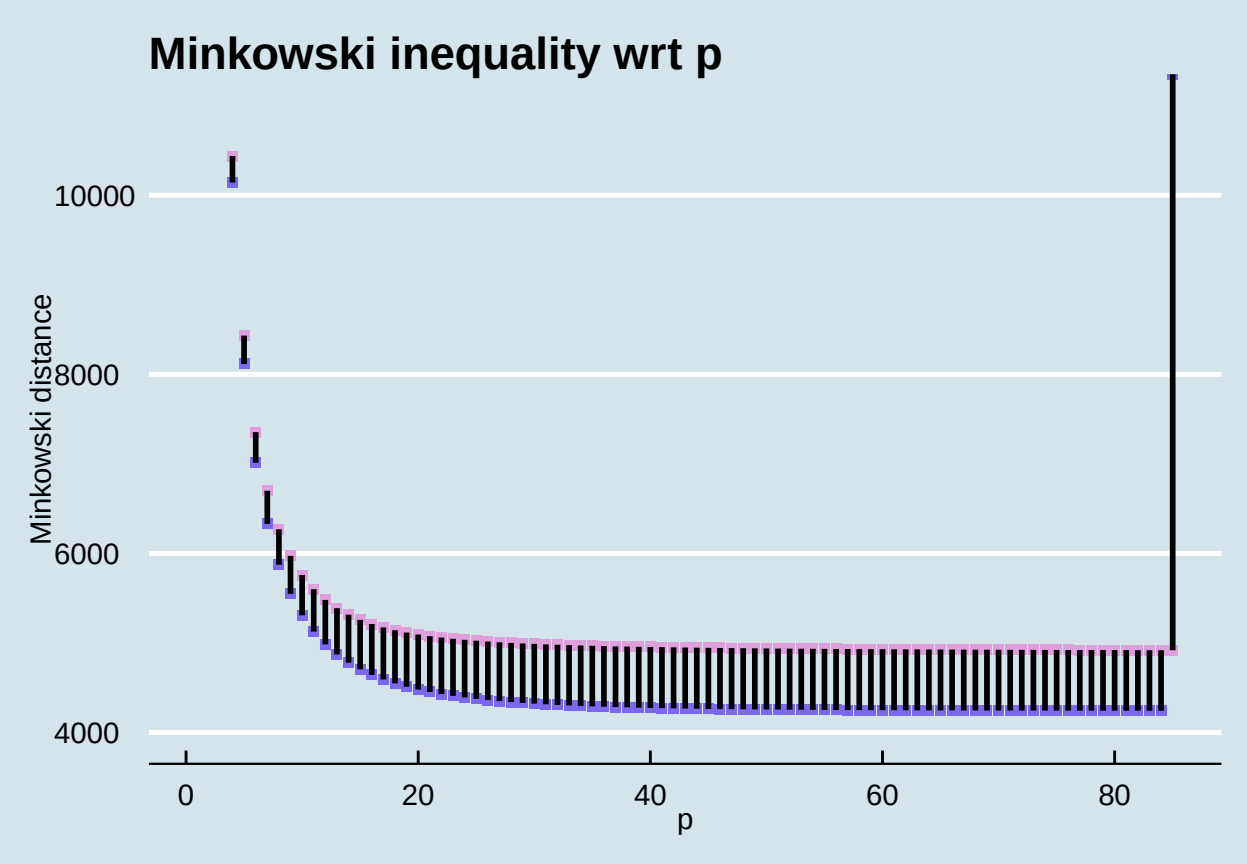
```
## [1] 271.4102
```

```
# Création des listes LHS (inégalité de gauche) et RHS (inégalité de droite) initialisées à vide
LHS<-c()
RHS<-c()

# Calcul des termes de l'inégalité selon p variant de 1 à 100 avec un pas de 1
for (p in seq(1,100,1))
    {
      LHS[p] <- (sum((draws1+draws2)^p))^(1/p)
      RHS[p] <- (sum((draws1)^p))^(1/p) + (sum((draws2)^p))^(1/p)
}

# Création du dataframe data contenant les 85 premiers termes de l'inégalité
data = data.frame(x = seq(1,85,1), LHS=LHS[1:85], RHS =RHS[1:85])

# Graphique illustrant l'inégalité de Minkowski
ggplot(data =data, aes(x = x)) +
  geom_point(aes(y = LHS),colour = "mediumslateblue",shape=15,size=1.5) +
  geom_point(aes(y = RHS),colour = "plum",shape=15,size=1.5) + geom_segment(aes(x = x,
  y = LHS,
  xend = x,
  yend = RHS),
 size = 1)+theme_economist() +
  xlab("p")+ylab("Minkowski distance")+ggtitle("Minkowski inequality wrt p") +
  ylim(4000,11000)
```



Les points roses représentent l'inégalité de droite tandis que les points bleus représentent la partie gauche de l'inégalité. Comme on pouvait s'y attendre, la partie droite de l'inégalité est supérieure à la partie gauche.

Note : les valeurs abérrantes (inf) ont été mises de côté. Aussi, afin d'avoir une meilleure visibilité des différences nous avons décidé de nous concentrer sur les distances dont les valeurs sont comprises entre 4 000 et 11 000.

Exercice 6

```
# Création des vecteurs x et y
x=c(22,34,1,12,25,56,7)
y=c(2,64,12,2,22,5,8)

#Rangs de x
rank(x)
```

```
## [1] 4 6 1 3 5 7 2
```

```
#Rangs de y
rank(y)
```

```
## [1] 1.5 7.0 5.0 1.5 6.0 3.0 4.0
```

```
#différence entre les rangs de x et y
d=rank(x)-rank(y)

# Coefficient de spearman (à partir de la corrélation de pearson sur les rangs)
cor(rank(x),rank(y))
```

```
## [1] 0.1621687
```

```
# Coefficient de spearman calculé
cor_calc <- 1-6*sum(d^2)/(7*(7^2-1))
cor_calc
```

```
## [1] 0.1696429
```

```
# Coefficient de corrélation de Spearman avec la fonction R associée
cor(x, y, method = "spearman")
```

```
## [1] 0.1621687
```

On remarque qu'on n'obtient pas le même résultat selon la méthode utilisée. Effectivement, lorsqu'on calcule le coefficient avec la formule

$$1 - \frac{6 \sum_{j=1}^{m} d_j^2}{n(n^2 - 1)}$$

le résultat est différent puisque nous avons deux rangs similaires pour le vecteur y (plus précisément, on a deux rangs 1.5). Ainsi, on obtient un coefficient de 0.169 au lieu de 0.162.

Exercice 7

Création des vecteurs x et y

```
x=c(22,34,1,12,25,56,7)
y=c(2,64,12,2,22,5,8)
```

Liste des paires d'observations de x :

```
combinaisons_x <- combn(x, 2)
combinaisons_x
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
## [1,]   22   22   22   22   22   22   34   34   34    34    34     1     1     1
## [2,]   34    1   12   25   56    7    1   12   25    56     7    12    25    56
##      [,15] [,16] [,17] [,18] [,19] [,20] [,21]
## [1,]     1    12    12    12    25    25    56
## [2,]     7    25    56     7    56     7     7
```

Liste des paires d'observations de y :

```
combinaisons_y <- combn(y, 2)
combinaisons_y
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
## [1,]    2    2    2    2    2    2   64   64   64    64    64    12    12    12
## [2,]   64   12    2   22    5    8   12    2   22     5     8     2    22     5
##      [,15] [,16] [,17] [,18] [,19] [,20] [,21]
## [1,]    12     2     2     2    22    22     5
## [2,]     8    22     5     8     5     8     8
```

Ayant 21 paires pour le vecteur x et similairement pour le vecteur y, on obient donc 42 paires.

Calcul des signes des différences - Pour le vecteur x

```
sign(combinaisons_x[2,]-combinaisons_x[1,])
```

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

- Pour le vecteur y

```r
sign(combinaisons_y[2,]-combinaisons_y[1,])
```

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

Calcul du tau de kendall “manuellement”

```r
#Calcul du taux de kendall
tau=0
for (i in 1:7)
{
tau=tau+sign(x -x[i])%*%sign(y -y[i])
}
tau=tau/(7*6)
tau
```

```
##           [,1]
## [1,] 0.0952381
```

Calcul du taux de kendall avec la fonction r associée

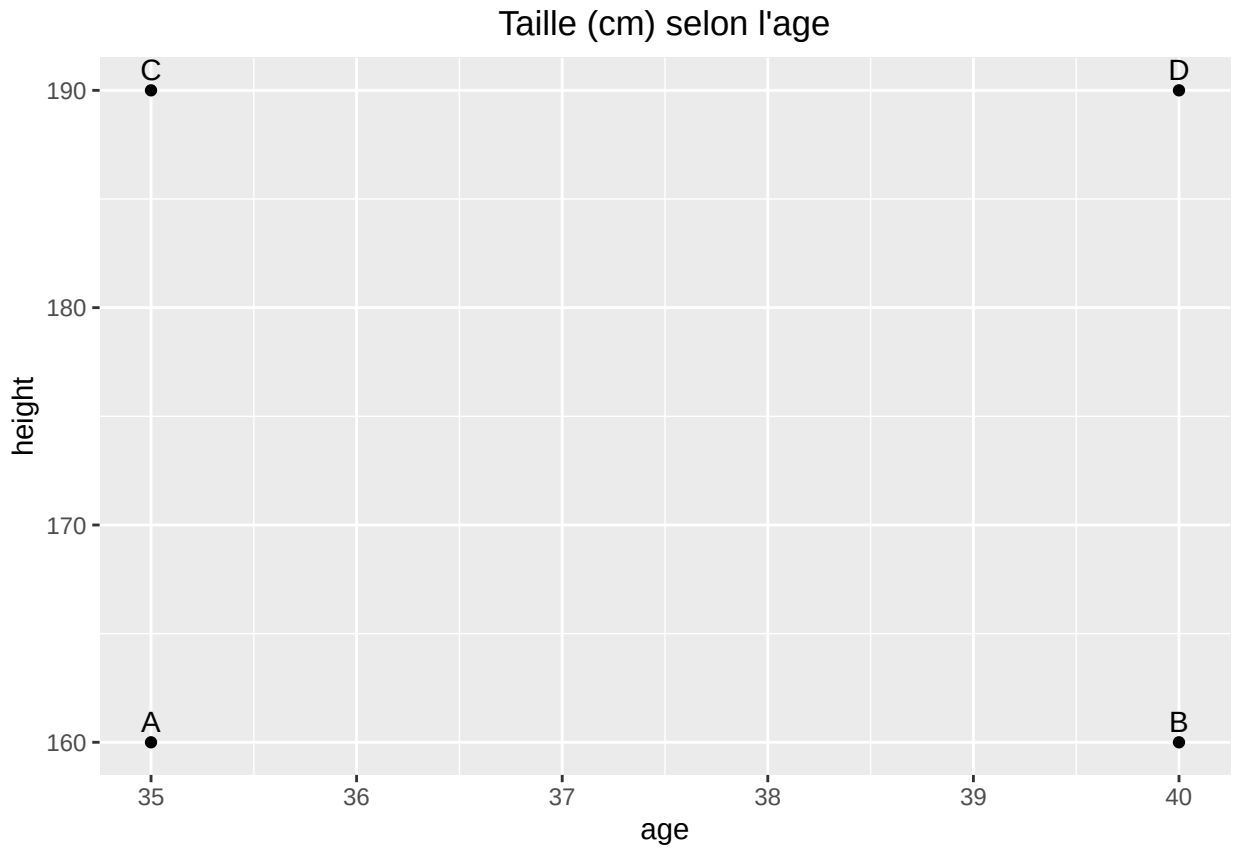
```r
# Calcul du taux de Kendall avec la fonction r associée
cor(x,y, method="kendall")
```

```
## [1] 0.09759001
```

On trouve comme on pouvait s’y attendre le même résultat via les deux manières.

Exercice 8

```r
df <- data.frame(person = c("A","B","C","D"), age = c(35, 40, 35, 40),
                 height = c(160, 160, 190, 190))
ggplot(df, aes(x = age, y = height)) + geom_point() +
  geom_text(label = df$person,vjust = -0.5) + ggtitle("Taille (cm) selon l'age") +
  theme(plot.title = element_text(hjust = 0.5))
```

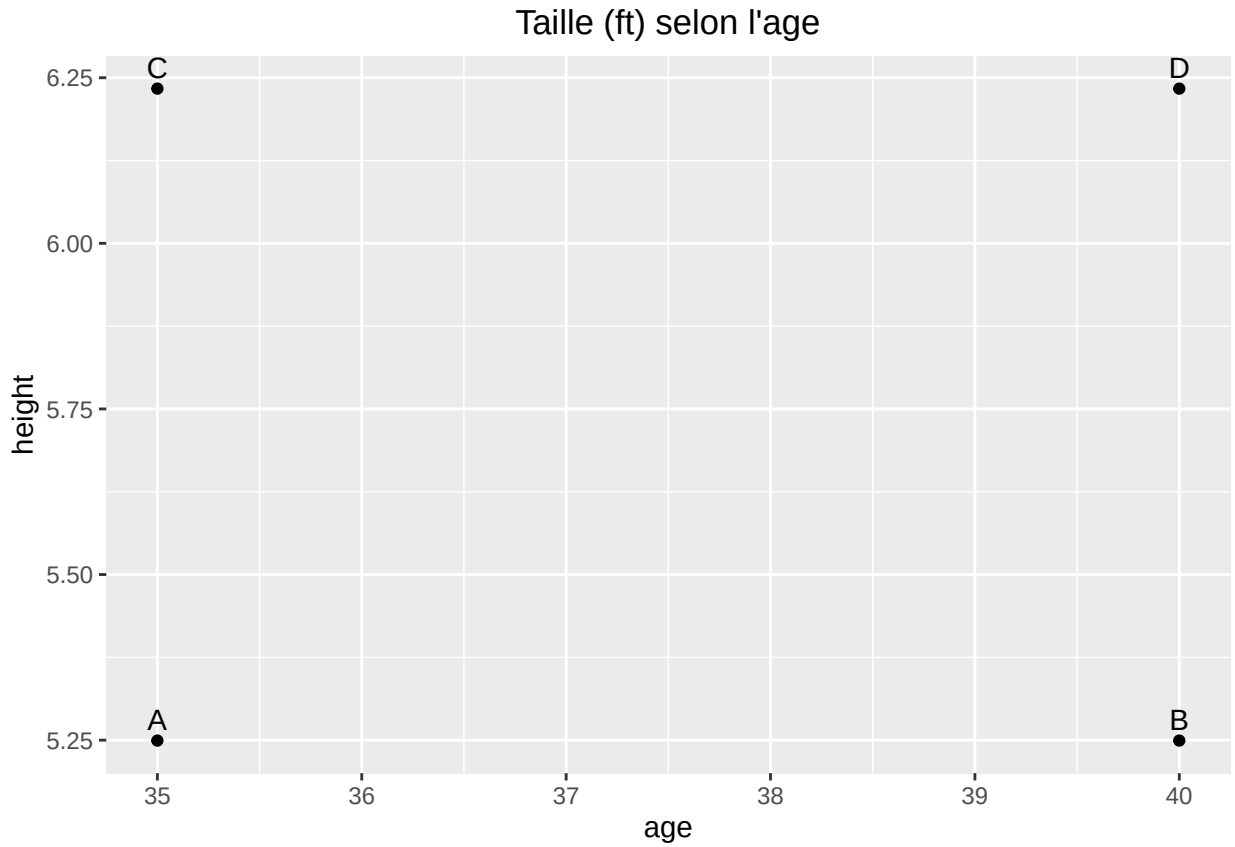


Premièrement nous avons observer les données, nous possédons quatre individus très distincts. Deux personnes ont 35 ans dont une mesure 160 cm et l'autre mesure 190cm. Les deux autres personnes ont 40 ans ont une mesure 160 cm et l'autre mesure 190cm. On a donc 2 individus de 190 cm et de 160 cm mais aussi 2 individus de 35 ans et 40 ans.

```
df_height <- df %>% mutate(height = height/30.48)
table(df_height)
```

```
## , , height = 5.249343832021
##
##       age
## person 35 40
##      A  1  0
##      B  0  1
##      C  0  0
##      D  0  0
##
## , , height = 6.23359580052493
##
##       age
## person 35 40
##      A  0  0
##      B  0  0
##      C  1  0
##      D  0  1
```

```
ggplot(df_height, aes(x = age, y = height)) + geom_point() +
  geom_text(label = df$person, vjust = -0.5) + ggtitle("Taille (ft) selon l'age") +
  theme(plot.title = element_text(hjust = 0.5))
```



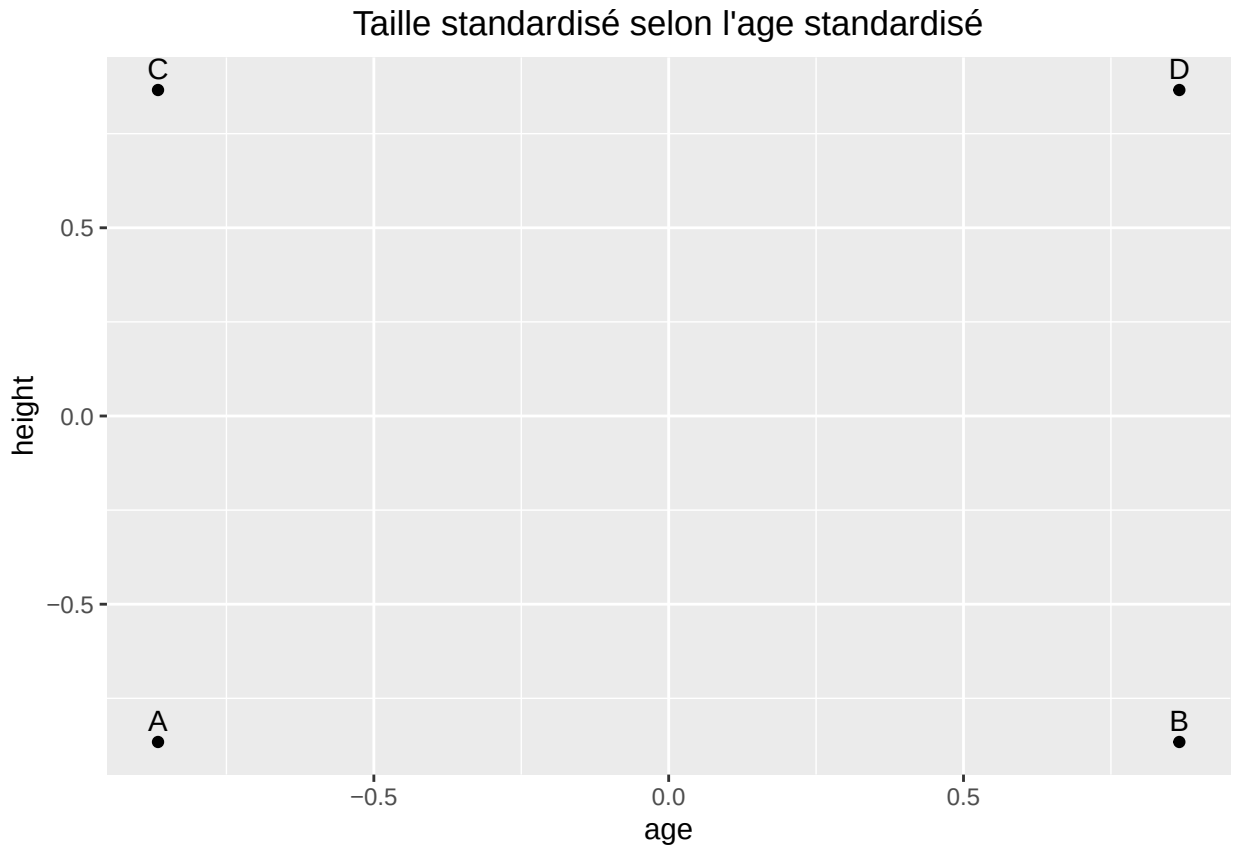
Le seul changement se remarque sur l'echelle de la taille des individus mais leur répartition reste identique.

```
moy_age=mean(df_height$age)
moy_height=mean(df_height$height)
sd_age=sd(df_height$age)
sd_height=sd(df_height$height)
df_height$age=(df_height$age-moy_age)/sd_age
df_height$height=(df_height$height-moy_height)/sd_height
table(df_height)
```

```
## , , height = -0.866025403784439
##
##       age
## person -0.866025403784439 0.866025403784439
##      A                  1                 0
##      B                  0                 1
##      C                  0                 0
##      D                  0                 0
##
## , , height = 0.866025403784439
##
##       age
## person -0.866025403784439 0.866025403784439
```

```
##      A                  0                  0
##      B                  0                  0
##      C                  1                  0
##      D                  0                  1
```

```
ggplot(df_height, aes(x = age, y = height)) + geom_point() +
  geom_text(label = df$person, vjust = -0.5) +
  ggtitle("Taille standardisé selon l'age standardisé") +
  theme(plot.title = element_text(hjust = 0.5))
```



Après avoir standardisé, on ne remaruqe pas de différence majeure. Le changement marquand réside dans le fait que pour les deux variables deux individus possèdent des valeurs opposés aux deux autres mais c'est un changement normal dû à la standardisation. De plus, on peut dire que tous les individus possèdent la même distance entre eux à part pour les cas $d(A, D)$ et $d(B, C)$.

Exercice 9

Pour traiter les NA, KAUFMAN and ROUSSEUW ont mesurer la similiarité entre la variable contenant des NA et les autres variables. Il a ensuite remplacé les NA par la valeur d'un même individu de la variable la plus proche

Voici les lignes possédant des NA

```
data("animals")

df=animals
df = df-1
df %>% filter(is.na(war)|is.na(fly)|is.na(ver)|is.na(end)|is.na(gro)|is.na(hai))
```

```
##     war fly ver end gro hai
## fro   0   0   1   1  NA   0
## lio   1   0   1  NA   1   1
## lob   0   0   0   0  NA   0
## sal   0   0   1   0  NA   0
## spi   0   0   0  NA   0   1
```

Calcul des similarités entre la variable GRO et les autres variables

```
growar <- addmargins(table(df$gro, df$war))
grofly <- addmargins(table(df$gro, df$fly))
grover <- addmargins(table(df$gro, df$ver))
groend <- addmargins(table(df$gro, df$end))
grohai <- addmargins(table(df$gro, df$hai))

growarsim <- abs((growar[1,1]*growar[2,2])-(growar[2,1]*growar[1,2]))
groflysim <- abs((grofly[1,1]*grofly[2,2])-(grofly[2,1]*grofly[1,2]))
groversim <- abs((grover[1,1]*grover[2,2])-(grover[2,1]*grover[1,2]))
groendsim <- abs((groend[1,1]*groend[2,2])-(groend[2,1]*groend[1,2]))
grohaisim <- abs((grohai[1,1]*grohai[2,2])-(grohai[2,1]*grohai[1,2]))
cbind(growarsim, groflysim, groversim, groendsim, grohaisim)
```

```
##      growarsim groflysim groversim groendsim grohaisim
## [1,]        26        10        21        10         3
```

Calcul du test d'indépendance entre les variables

```
#Essai d'identification de lien entre test du Chi2 et similarité avec "chisq.test()"
chisq.test(df$gro, df$war)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$gro and df$war
## X-squared = 1.1269, df = 1, p-value = 0.2884
```

```
chisq.test(df$gro, df$fly)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$gro and df$fly
## X-squared = 0.011145, df = 1, p-value = 0.9159
```

```
chisq.test(df$gro, df$ver)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$gro and df$ver
## X-squared = 0.67077, df = 1, p-value = 0.4128
```

```
chisq.test(df$gro, df$end)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$gro and df$end
## X-squared = 0.0375, df = 1, p-value = 0.8465
```

```
chisq.test(df$gro, df$hai)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$gro and df$hai
## X-squared = 7.1964e-32, df = 1, p-value = 1
```

Calcul de la similarité avec la fonction phi()

```
#Variante de calcul de similarité avec "phi()"
cbind(phi(table(df$gro, df$war)), phi(table(df$gro, df$fly)), phi(table(df$gro, df$ver)),
      phi(table(df$gro, df$end)), phi(table(df$gro, df$hai)))
```

```
##      [,1]  [,2] [,3] [,4] [,5]
## [1,] 0.38 -0.17 0.33  0.2 0.04
```

On remarque la variable la plus proche est la variable WAR

Remplacement des NA pour la variable GRO

```
#Remplacement des NA par les valeurs de les variables la plus proche
df[11,5]=df[11,1]
df[15,5]=df[15,1]
df[18,5]=df[18,1]
df
```

```
##     war fly ver end gro hai
## ant   0   0   0   0   1   0
## bee   0   1   0   0   1   1
## cat   1   0   1   0   0   1
## cpl   0   0   0   0   0   1
## chi   1   0   1   1   1   1
## cow   1   0   1   0   1   1
## duc   1   1   1   0   1   0
## eag   1   1   1   1   0   0
## ele   1   0   1   1   1   0
## fly   0   1   0   0   0   0
## fro   0   0   1   1   0   0
## her   0   0   1   0   1   0
## lio   1   0   1  NA   1   1
## liz   0   0   1   0   0   0
## lob   0   0   0   0   0   0
## man   1   0   1   1   1   1
## rab   1   0   1   0   1   1
## sal   0   0   1   0   0   0
## spi   0   0   0  NA   0   1
## wha   1   0   1   1   1   0
```

Réalisation de la même méthode pour la variable END

```
endwar <- addmargins(table(df$end, df$war))
endfly <- addmargins(table(df$end, df$fly))
endver <- addmargins(table(df$end, df$ver))
endgro <- addmargins(table(df$end, df$gro))
endhai <- addmargins(table(df$end, df$hai))
```

```r
#calcul de la similarité
endwarsim <- abs((endwar[1,1]*endwar[2,2])-(endwar[2,1]*endwar[1,2]))
endflysim <- abs((endfly[1,1]*endfly[2,2])-(endfly[2,1]*endfly[1,2]))
endversim <- abs((endver[1,1]*endver[2,2])-(endver[2,1]*endver[1,2]))
endgrosim <- abs((endgro[1,1]*endgro[2,2])-(endgro[2,1]*endgro[1,2]))
endhaisim <- abs((endhai[1,1]*endhai[2,2])-(endhai[2,1]*endhai[1,2]))

#identification de la variable la plus similaire : war
cbind(endwarsim, endflysim, endversim, endgrosim, endhaisim)
```

```
##      endwarsim endflysim endversim endgrosim endhaisim
## [1,]        36         6        30        12         6
```

```r
#Essai d'identification de lien entre test du Chi2 et similarité avec "chisq.test()"
chisq.test(df$end, df$war)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$end and df$war
## X-squared = 2.25, df = 1, p-value = 0.1336
```

```r
chisq.test(df$end, df$fly)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$end and df$fly
## X-squared = 1.9545e-31, df = 1, p-value = 1
```

```r
chisq.test(df$end, df$ver)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$end and df$ver
## X-squared = 1.6962, df = 1, p-value = 0.1928
```

```r
chisq.test(df$end, df$gro)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$end and df$gro
## X-squared = 0.028125, df = 1, p-value = 0.8668
```

```r
chisq.test(df$end, df$hai)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df$end and df$hai
## X-squared = 1.3831e-31, df = 1, p-value = 1
```

```r
#Variante de calcul de similarité avec "phi()"
cbind(phi(table(df$end, df$war)), phi(table(df$end, df$fly)), phi(table(df$end, df$ver)),
      phi(table(df$end, df$gro)), phi(table(df$end, df$hai)))
```

```
##       [,1]  [,2] [,3] [,4]  [,5]
## [1,] 0.47 -0.09 0.44 0.16 -0.08
```

On remarque la variable la plus proche est la variable WAR

```
#Remplacement des NA par les valeurs de les variables la plus proche
df[13,4]=df[13,1]
df[19,4]=df[19,1]
df
```

```
##     war fly ver end gro hai
## ant   0   0   0   0   1   0
## bee   0   1   0   0   1   1
## cat   1   0   1   0   0   1
## cpl   0   0   0   0   0   1
## chi   1   0   1   1   1   1
## cow   1   0   1   0   1   1
## duc   1   1   1   0   1   0
## eag   1   1   1   1   0   0
## ele   1   0   1   1   1   0
## fly   0   1   0   0   0   0
## fro   0   0   1   1   0   0
## her   0   0   1   0   1   0
## lio   1   0   1   1   1   1
## liz   0   0   1   0   0   0
## lob   0   0   0   0   0   0
## man   1   0   1   1   1   1
## rab   1   0   1   0   1   1
## sal   0   0   1   0   0   0
## spi   0   0   0   0   0   1
## wha   1   0   1   1   1   0
```

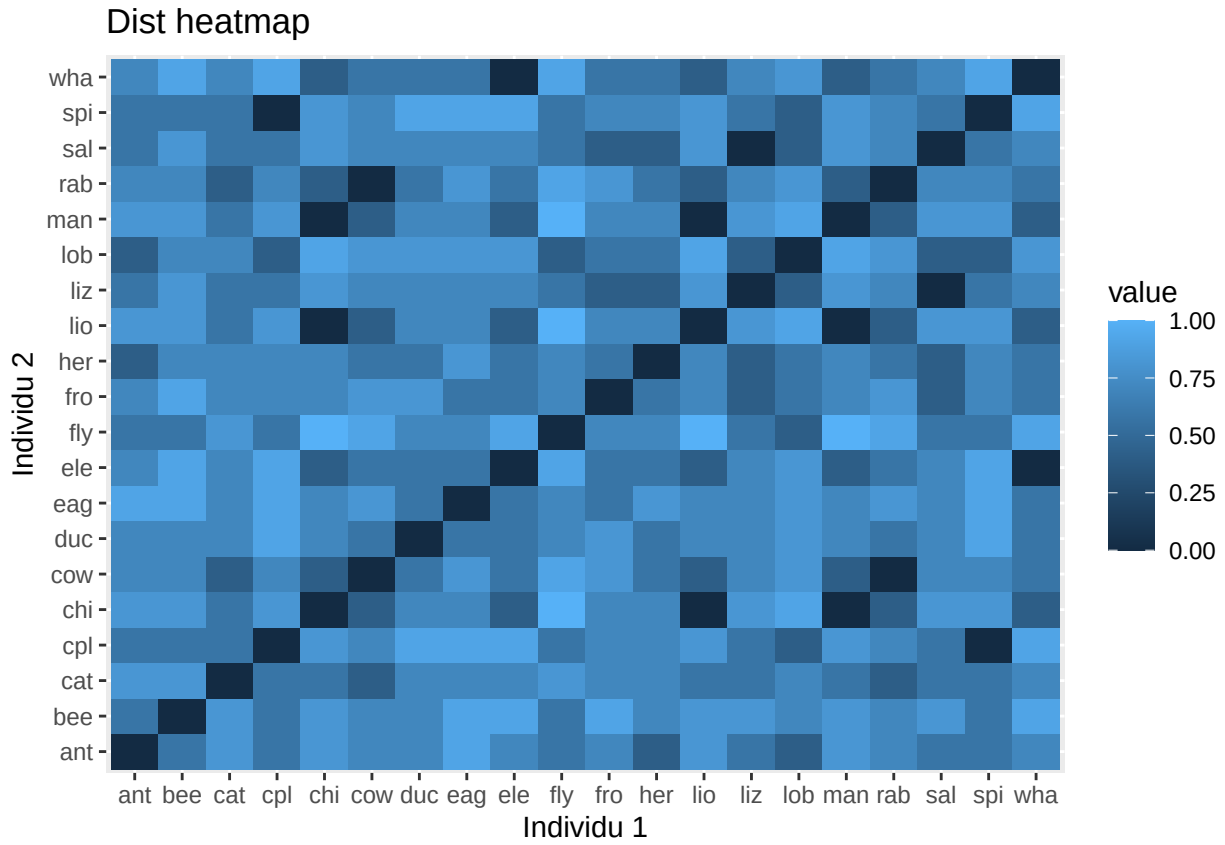
```
matrice_dist <-dist.binary(df, method = 2, diag = T, upper = T)

mat_matrice_dist <- as.matrix(matrice_dist)
#Apercu de la matrice des distances


melted_mat <- melt(mat_matrice_dist)
head(melted_mat)
```

```
##   Var1 Var2     value
## 1  ant  ant 0.0000000
## 2  bee  ant 0.5773503
## 3  cat  ant 0.8164966
## 4  cpl  ant 0.5773503
## 5  chi  ant 0.8164966
## 6  cow  ant 0.7071068
```

```
ggplot(data = melted_mat, aes(x = Var1, y = Var2, fill = value)) + geom_tile() +
  ggtitle("Dist heatmap") + xlab("Individu 1") + ylab("Individu 2")
```



On voit que le poulet, le lion et l'homme sont proches. Le lapin est proche de la vache. L'araignée est proche du CPL, la balein est proche de l'elephant.

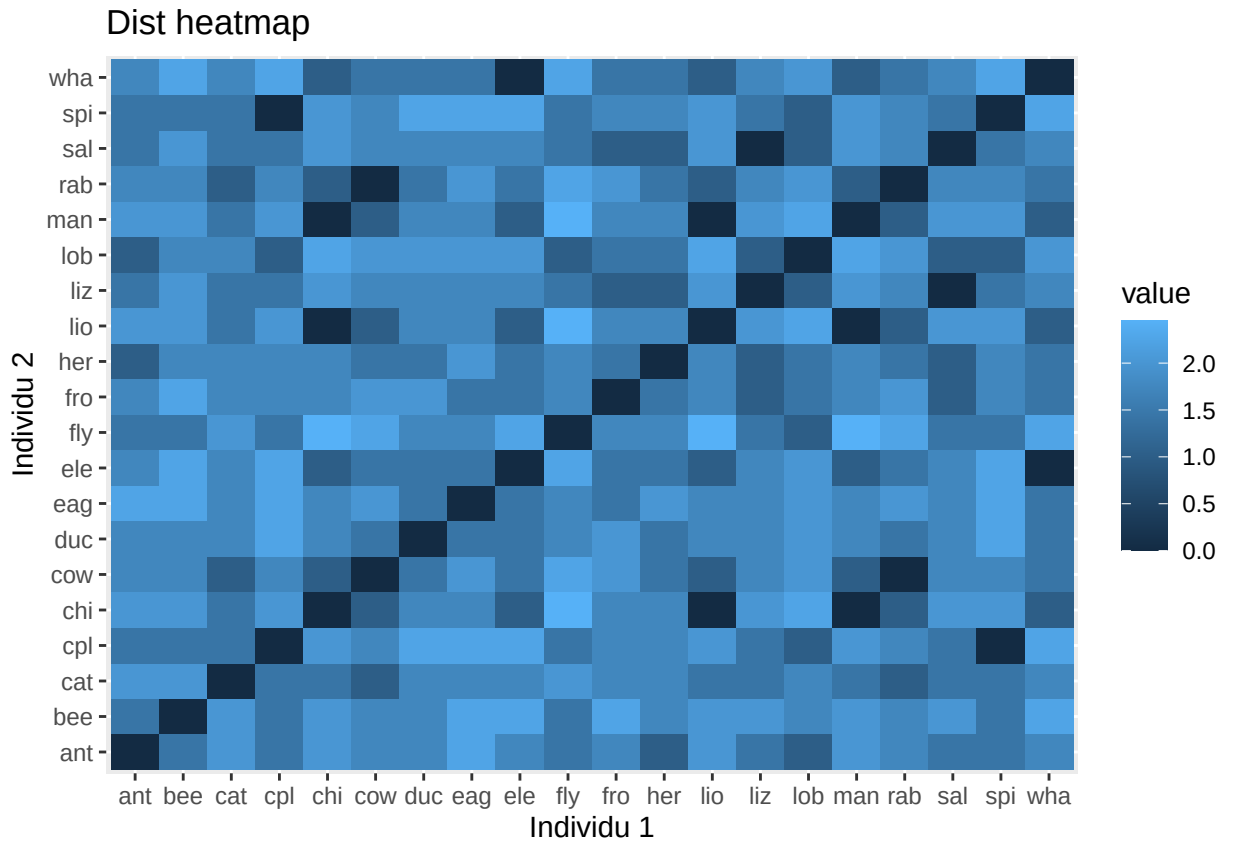
```
matrice_dist=dist(df,method = "minkowski")
mat_matrice_dist <- as.matrix(matrice_dist)
mat_matrice_dist[1:8,1:8]
```

```
##          ant      bee      cat      cpl      chi      cow      duc      eag
## ant 0.000000 1.414214 2.000000 1.414214 2.000000 1.732051 1.732051 2.236068
## bee 1.414214 0.000000 2.000000 1.414214 2.000000 1.732051 1.732051 2.236068
## cat 2.000000 2.000000 0.000000 1.414214 1.414214 1.000000 1.732051 1.732051
## cpl 1.414214 1.414214 1.414214 0.000000 2.000000 1.732051 2.236068 2.236068
## chi 2.000000 2.000000 1.414214 2.000000 0.000000 1.000000 1.732051 1.732051
## cow 1.732051 1.732051 1.000000 1.732051 1.000000 0.000000 1.414214 2.000000
## duc 1.732051 1.732051 1.732051 2.236068 1.732051 1.414214 0.000000 1.414214
## eag 2.236068 2.236068 1.732051 2.236068 1.732051 2.000000 1.414214 0.000000
```

```
melted_mat <- melt(mat_matrice_dist)
head(melted_mat)
```

```
##   Var1 Var2    value
## 1  ant  ant 0.000000
## 2  bee  ant 1.414214
## 3  cat  ant 2.000000
## 4  cpl  ant 1.414214
## 5  chi  ant 2.000000
## 6  cow  ant 1.732051
```

```
ggplot(data = melted_mat, aes(x = Var1, y = Var2, fill = value)) + geom_tile() +
  ggtitle("Dist heatmap") + xlab("Individu 1") + ylab("Individu 2")
```



Lorsque nous changons de méthode de calcul pour la distance, on remarque que ce sont les mêmes individus qui sont proches.

Exercice 11

Pour avoir le nombre de partition, il faut réaliser

$$\binom{27}{26} + 1$$

Soit le résultat du calcul est

$$\binom{27}{26} + 1 = 352$$

Exercice 14

Ci-dessous, une heatmap représentant selon le cluster associé chacun des individus.

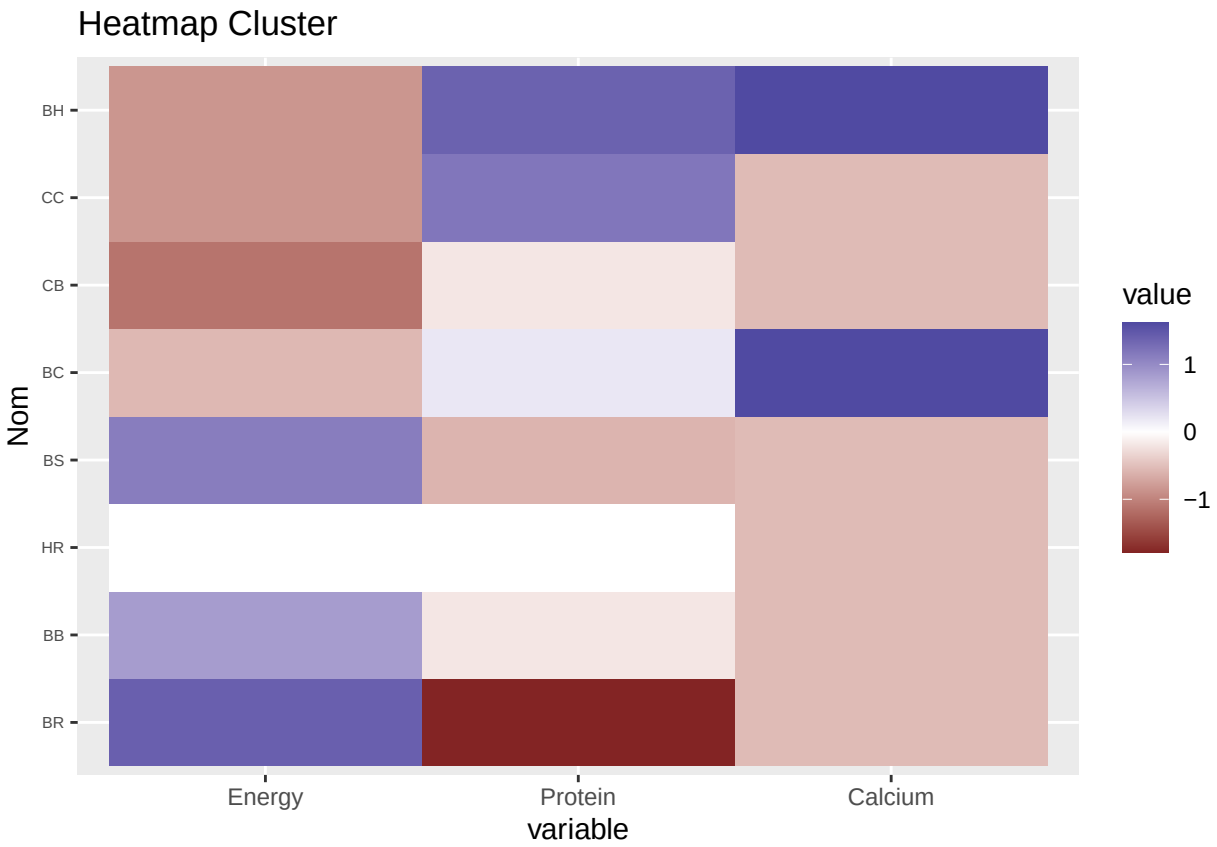
```
harti <- read.csv("https://raw.githubusercontent.com/karkil2205/Cluster-Analysis/master/Hartigandata1.cs
df<-harti[1:8,c(3,4,6)]
df[3,1]<-13 # Error in line 3
df[6,1]<-4 # Error at line 6
df[7,3]<-1 # Error at line 7
rownames(df)<-c("BB","HR","BR","BS","BC","CB","CC","BH")
```

```r
colnames(df)<-c("Energy","Protein","Calcium")
df$Name=rownames(df)
km.res<-kmeans(df[1:8,1:3],3,iter.max = 100)
df.scaled <- df
df.scaled[,c(1:3)]=scale(df[,1:3])
df.scaled=cbind(df.scaled,cluster=km.res$cluster)
df.scaled=df.scaled[order(df.scaled[,5]),]
df.scaled
```

```
##        Energy    Protein    Calcium Name cluster
## BR  1.4101902 -1.7847670 -0.5400617   BR       1
## BB  0.8461141 -0.1983074 -0.5400617   BB       2
## HR  0.0000000  0.0000000 -0.5400617   HR       2
## BS  1.1281521 -0.5949223 -0.5400617   BS       2
## BC -0.5640761  0.1983074  1.6201852   BC       3
## CB -1.1281521 -0.1983074 -0.5400617   CB       3
## CC -0.8461141  1.1898447 -0.5400617   CC       3
## BH -0.8461141  1.3881521  1.6201852   BH       3
```

```r
df.long=melt(df.scaled,id=c("Name","cluster"))
df.long=df.long[order(df.long[,"cluster"]),]
heatmap.plot <- ggplot(data = df.long, aes(x = variable, y = Name)) +
  geom_tile(aes(fill = value)) +
  scale_fill_gradient2() + ggtitle("Heatmap Cluster") + ylab("Nom") +
  theme(axis.text.y = element_text(size = 6)) +
  scale_y_discrete(limits=df.scaled[order(df.scaled[,5]),4])

# Preview the heatmap
print(heatmap.plot)
```



Ainsi on peut remarquer que leur corrélation avec les 3 variables peut être similaire s'ils appartiennent au même cluster. Les clusters peuvent se discerner grâce à cette heatmap.

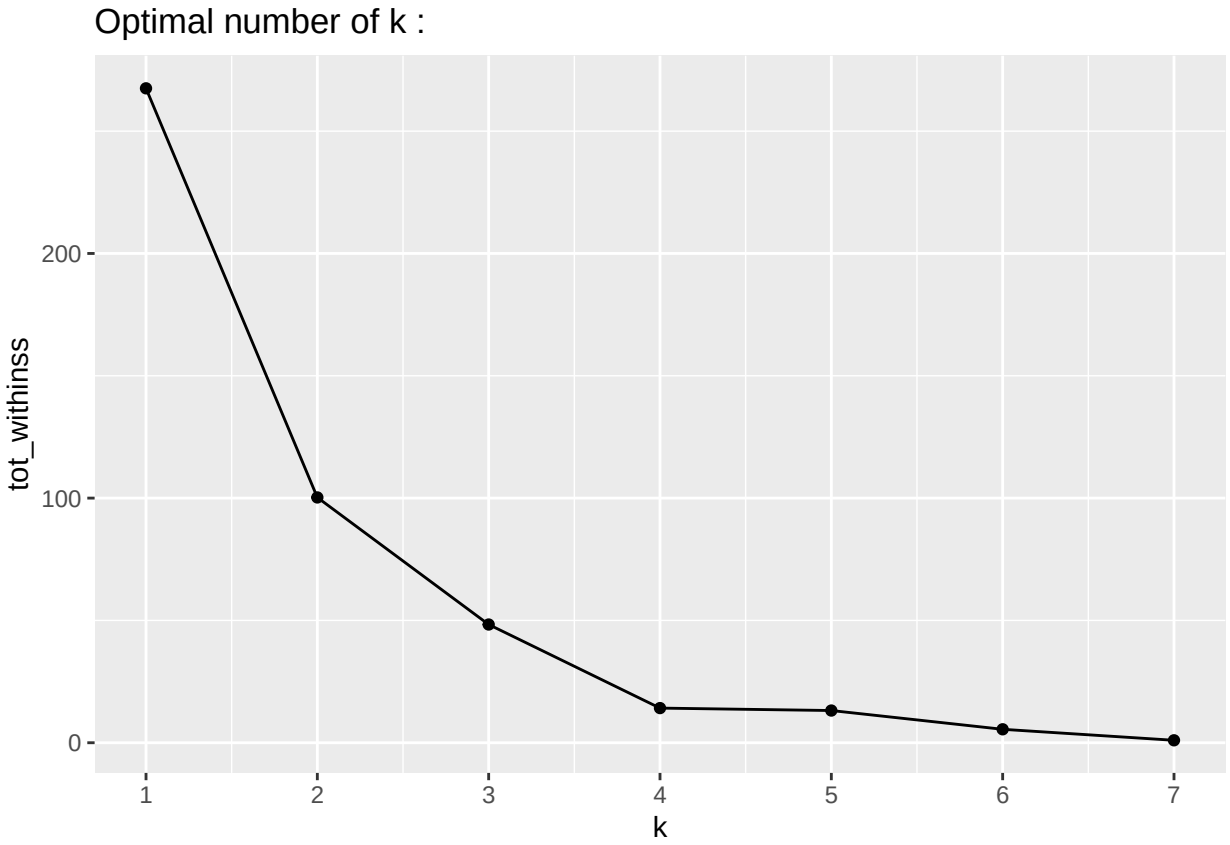
Exercice 16

D'après cet éboulis qui présente l'erreur totale selon le nombre de cluster, on voit que le coude est présent au nombre de 3 clusters. On peut donc dire qu'il faut réaliser un kmeans avec 3 clusters

```
tot_withinss <- map_dbl(1:7,  function(k){
  model <- kmeans(x = df[,1:3], centers = k, iter.max = 200)
  model$tot.withinss
})

elbow_df <- data.frame(
  k = 1:7,
  tot_withinss = tot_withinss
)

ggplot(elbow_df, aes(x = k, y = tot_withinss)) +
  geom_line() + geom_point()+
  scale_x_continuous(breaks = 1:7) + ggtitle("Optimal number of k :")
```



Exercice 18

Voici un graphe la méthode silhouette.

```
harti_pure <- df[,-4]

knitr::opts_chunk$set(echo = TRUE)
km.res<-kmeans(df[1:8,1:3],3,iter.max = 100)
slobj<-silhouette(km.res$cluster,dist(df[1:8,]))
row.names(slobj)<-row.names(df[1:8,])

slobj=slobj[order(slobj[,"cluster"]),]

knitr::kable(slobj[,], caption = "Silhouettes values for the food example",
             digits = 4,col.names = c("Cluster","Neighbor","Silhouette width"),
             align=c("c","c","c"),booktabs = TRUE)%>%
  kable_classic(latex_options = "hold_position")%>%
  row_spec(1:2,background = "#2EFEF7")%>%
  row_spec(3:5,background = "#F3F781")%>%
  row_spec(6:8,background = "#FA58F4")%>%
  kable_styling(latex_options = "hold_position")
```

```
silhouette(km.res$cluster,dist(df[1:8,]))%>%
  fviz_silhouette()
```

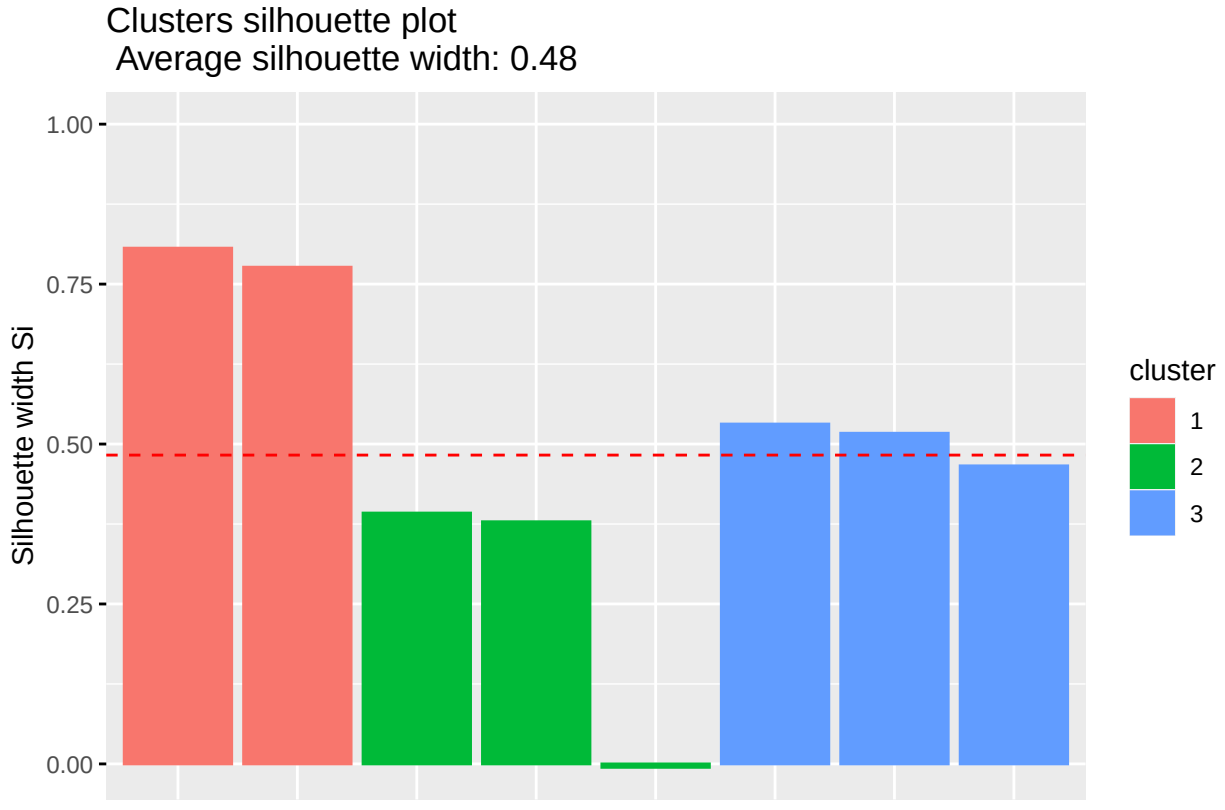
```
##   cluster size ave.sil.width
## 1       1    2          0.79
```

[!h]

Table 1: Silhouettes values for the food example

	Cluster	Neighbor	Silhouette width
CC	1	3	0.7764
BH	1	3	0.8062
BB	2	3	-0.0053
BR	2	3	0.3785
BS	2	3	0.3921
HR	3	2	0.4659
BC	3	1	0.5168
CB	3	1	0.5312

```
## 2       2    3          0.26
## 3       3    3          0.50
```

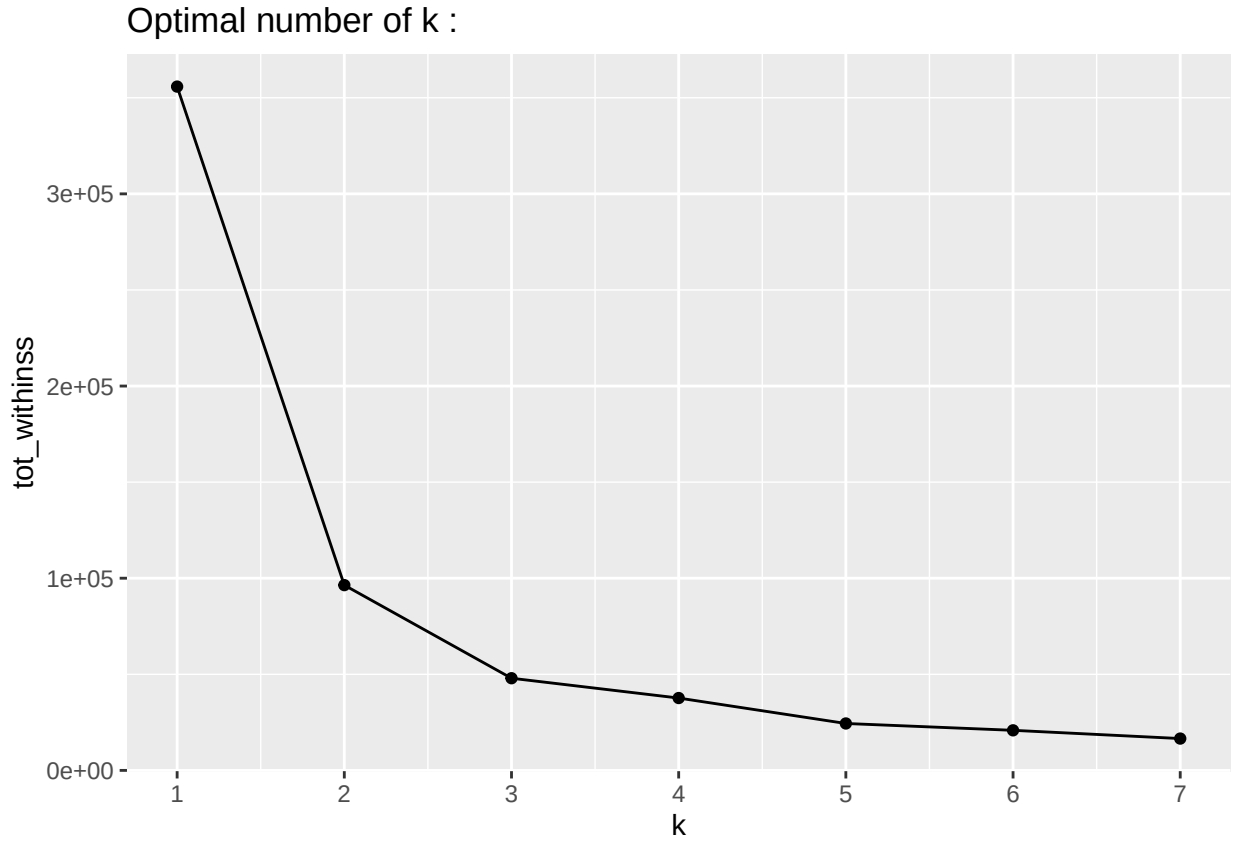


Exercice 19

```
tot_withinss <- map_dbl(1:7,  function(k){
  model <- kmeans(x = USArrests, centers = k, iter.max = 200)
  model$tot.withinss
})
elbow_df <- data.frame(
```

```
  k = 1:7,
  tot_withinss = tot_withinss
)

ggplot(elbow_df, aes(x = k, y = tot_withinss)) +
  geom_line() + geom_point()+
  scale_x_continuous(breaks = 1:7) + ggtitle("Optimal number of k :")
```



Comme pour l'eboulis sur les données Hartigan, 3 clusters est le nombre le plus appropriés

```
k_mean <- kmeans(USArrests,3, iter.max = 500)
k_mean$centers
```

```
##      Murder  Assault UrbanPop     Rape
## 1  4.270000  87.5500 59.75000 14.39000
## 2  8.214286 173.2857 70.64286 22.84286
## 3 11.812500 272.5625 68.31250 28.37500
```

```
table(k_mean$cluster)
```

```
##
##  1  2  3
## 20 14 16
```

Lorsque nous réalisons un clustering Kmeans avec 3 clusters, on remarque le nombre d'états par cluster n'est pas homogène. Du cluster 1 à 3, nous avons de moins en moins d'états présent. Ensuite si nous croisons cela avec la moyenne des variables par clusters, on voit que le cluster 1 possède des moyennes très différents des deux autres clusters. Tout d'abord le cluster est celui qui possède la moyenne la plus faible de chacune des

variables. Pour le nombre de meurtre et d'assault, sa moyenne vaut environ la moitié de celle du cluster 2. Ensuite on voit que le cluster 3 possède les moyennes les plus hautes de manière non négligeable pour 3 des variables. Il possède une moyenne un peu plus inférieur pour le pourcentage d'urbanisation par rapport au cluster 2. Ainsi on peut conclure que le cluster 1 contient le plus d'états mais aussi celui le plus "pacifique". Son contraire est donc le cluster 3 qui possède le moins d'états mais c'est celui qui contient les états les plus "dangereux".

```
k_mean$withinss
```

```
## [1] 19263.760  9136.643 19563.863
```

```
k_mean$tot.withinss
```

```
## [1] 47964.27
```

```
k_mean$betweenss
```

```
## [1] 307843.6
```

```
k_mean$totss
```

```
## [1] 355807.8
```

Voici la somme des carrés de la variance par cluster:(19263,76;19563,86;9136,64) En sommant ces 3 trois sommes nous obtenons la somme des carrés de la variance intra-cluster: 47964.27 Son complémentaire est la somme des carrés de la variance inter-cluster: 307843.6 Ainsi en sommant ces deux dernières sommes, nous obtenons la somme des carrés de la variance totale : 355807.8
