TP2: MCA with R

Exercice 1. The pre-processing in MCA

1. Load the dataset **dogs.rda** of the $n = 27$ dogs described on $p = 6$ categorical variables.

```
load("../data/dogs.rda")
print(data[1:5,])
```

```
##                 Size Weight Velocity Intelligence Affectivity Aggressivness
## Beauceron        S++     W+      V++           I+         Af+           Ag+
## BassetHound       S-     W-       V-           I-         Af-           Ag+
## GermanShepherd   S++     W+      V++          I++         Af+           Ag+
## Boxer             S+     W+       V+           I+         Af+           Ag+
## Bulldog           S-     W-       V-           I+         Af+           Ag-
```

2. Check the class of the object **data**. Check the class of the first column of **data**. Use the function **levels** to get the levels of the variable *Size*.

```
class(data)
data$Size #first columns
class(data$Size)
levels(data$Size)
```

```
## [1] "data.frame"
##  [1] S++ S-  S++ S+  S-  S++ S-  S-  S+  S++ S+  S++ S++ S+  S++ S++ S-
## [18] S++ S+  S++ S++ S-  S++ S++ S++ S-  S++
## Levels: S- S+ S++
## [1] "factor"
## [1] "S-"  "S+"  "S++"
```

3. Use the functions **lapply** to find the number m_j of levels of each variable j and the total number of levels $\ell = \sum_{j=1}^{p} \ell_j$.

```
lj <- unlist(lapply(data,function(x){length(levels(x))}))
l <- sum(lj)
```

4. Build the matrix K of the disjonctive table using the function **tab.disjonctif** of the R package **FactoMineR**.

```
library(FactoMineR)
K <- tab.disjonctif(data)
print(K[1:4,])
```

```
##                 S- S+ S++ W- W+ W++ V- V+ V++ I- I+ I++ Af- Af+ Ag- Ag+
## Beauceron        0  0   1  0  1   0  0  0   1  0  1   0   0   1   0   1
## BassetHound      1  0   0  1  0   0  1  0   0  1  0   0   1   0   0   1
## GermanShepherd   0  0   1  0  1   0  0  0   1  0  0   1   0   1   0   1
## Boxer            0  1   0  0  1   0  0  1   0  0  1   0   0   1   0   1
```

5. Compute the frequencies n_s and the relative frequencies $\frac{n_s}{n}$ of the levels ?

```
ns <- apply(K,2,sum)
print(ns)
```

```
##  S-  S+ S++  W-  W+ W++  V-  V+ V++  I-  I+ I++ Af- Af+ Ag- Ag+
##   7   5  15   8  14   5  10   8   9   8  13   6  13  14  14  13
```

```r
n <- nrow(K)
fs <- ns/n
print(fs)
```

```
##        S-        S+       S++        W-        W+       W++        V-
## 0.2592593 0.1851852 0.5555556 0.2962963 0.5185185 0.1851852 0.3703704
##        V+       V++        I-        I+       I++       Af-       Af+
## 0.2962963 0.3333333 0.2962963 0.4814815 0.2222222 0.4814815 0.5185185
##       Ag-       Ag+
## 0.5185185 0.4814815
```

6. Build the matrix Z of the centered disjonctive table.

```r
Z <- sweep(K,2,fs,FUN="-")
apply(Z,2,mean)
```

```
##            S-            S+           S++            W-            W+
##  0.000000e+00  2.056270e-17 -2.466158e-17  1.644674e-17  2.877151e-17
##           W++            V-            V+           V++            I-
##  2.055969e-17  2.055165e-17  1.644173e-17  3.700944e-17  1.644976e-17
##            I+           I++           Af-           Af+           Ag-
##  5.345518e-17  1.232276e-17  5.348731e-17  2.875144e-17  2.876147e-17
##           Ag+
##  5.345920e-17
```

7. Perform the variance of the columns of the disjonctive table with the function **var** and then from the formula $\frac{n_s}{n}(1 - \frac{n_s}{n})$.

```r
apply(Z,2,var)*(n-1)/n
fs*(1-fs)
```

```
##        S-        S+       S++        W-        W+       W++        V-
## 0.1920439 0.1508916 0.2469136 0.2085048 0.2496571 0.1508916 0.2331962
##        V+       V++        I-        I+       I++       Af-       Af+
## 0.2085048 0.2222222 0.2085048 0.2496571 0.1728395 0.2496571 0.2496571
##       Ag-       Ag+
## 0.2496571 0.2496571
##        S-        S+       S++        W-        W+       W++        V-
## 0.1920439 0.1508916 0.2469136 0.2085048 0.2496571 0.1508916 0.2331962
##        V+       V++        I-        I+       I++       Af-       Af+
## 0.2085048 0.2222222 0.2085048 0.2496571 0.1728395 0.2496571 0.2496571
##       Ag-       Ag+
## 0.2496571 0.2496571
```

8. Perform the distance between the two breeds *Pekingese* and *Doberman* descibed in disjonctive table using the metric $M = diag(\frac{n}{n_s})$. Check that the result is the same when using the centered disjonctive table.

```r
# Pekingese row 22 and Doberman row 12
i <- 12; j=22
sqrt(sum((K[i,]-K[j,])^2/fs))
sqrt(sum((Z[i,]-Z[j,])^2/fs))
```

```
## [1] 5.704972
## [1] 5.704972
```

9. Perform $I(K)$, the total inertia of the 27 dogs descrived in the disjonctive table.

```r
p <- ncol(data)
total <- l-p
```

Exercice 2. The GSVD of Z.

The GSVD of a real matrix Z of dimension $n \times p$ with metrics N on $\mathbb{R}^n$ and M on $\mathbb{R}^p$ gives the following decomposition:

$$Z = U\Lambda V^t,$$

where

- $\Lambda = \text{diag}(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_r})$ is the $r \times r$ diagonal matrix of the singular values of ZMZ^tN and Z^tNZM, and r denotes the rank of Z;
- U is the $n \times r$ matrix of the first r eigenvectors of ZMZ^tN such that $U^tNU = \mathbb{I}_r$, with $\mathbb{I}_r$ the identity matrix of size r;
- V is the $p \times r$ matrix of the first r eigenvectors of Z^tNZM such that $V^tMV = \mathbb{I}_r$.

The idea is to perform the GSVD of the centered disjonctive table Z of the dogs data with metrics $N = \frac{1}{n}\mathbb{I}_n$ and $M = diag(\frac{n}{n_s}, s = 1, \ldots, \ell)$ used in MCA.

1. Build with the two metrics M and N using the function **diag**.

```r
N <- diag(rep(1/n,n))
M <- diag(n/ns)
```

2. The GSVD of Z can be obtained by performing the standard SVD of the matrix $\tilde{Z} = N^{1/2}ZM^{1/2}$, that is a GSVD with metrics $\mathbb{I}_n$ on $\mathbb{R}^n$ and $\mathbb{I}_p$ on $\mathbb{R}^p$. It gives:

$$\tilde{Z} = \tilde{U}\tilde{\Lambda}\tilde{V}^t$$

and transformation back to the original scale gives:

$$\Lambda = \tilde{\Lambda} \quad , \quad U = N^{-1/2}\tilde{U} \quad , \quad V = M^{-1/2}\tilde{V} \quad .$$

This procedure has been implemented in a function **gsvd** avalaible in the file **gsvd.R**. Open this file and read the description of the function and its R code.

3. Perform the GSVD of the centered disjonctive table Z with the metrics M and N.

```r
source("gsvd.R")
```

```r
w <- rep(1/n,n)
c <- n/ns
res <- gsvd(Z,w,c)
d <- res$d
U <- res$U
V <- res$V
```

4. Check that the rank of the centered disjonctive table is is r=min$(n-1, \ell-p)$. Check using %*% (matrix product in R) that the matrix U is N-orthonormal and that the matrix V is M-orthonormal.

```r
length(d) # r=10 singular values
t(U)%*%N%*%U
t(V)%*%M%*%V
```

Exercice 3. GSVD and MCA.

We want to perform MCA using the GSVD of the disjonctive table performed in the previous exercice.

1. Build the matrix F of dimension $n \times r$ of the factor coordinates of the dogs.

```r
F <- U%*%diag(d)
colnames(F) <- paste("dim",1:length(d),sep="")
print(F[1:5,1:2])
```

```
##                      dim1      dim2
## Beauceron      -0.7769783  1.023155
## BassetHound     0.6224395 -2.697444
## GermanShepherd -1.1914209  1.137664
## Boxer           1.0958158  2.159906
## Bulldog         2.4821958 -1.346924
```

2. Build the matrix A of dimension $m \times r$ of the factor coordinates of the levels.

```r
A <- M%*%V%*%diag(d)
rownames(A) <- rownames(V)
colnames(A) <- paste("dim",1:length(d),sep="")
print(A[1:5,1:2])
```

```
##           dim1        dim2
## S-   1.1849557 -0.92389650
## S+   0.8510880  1.23171972
## S++ -0.8366753  0.02057846
## W-   1.1689180 -0.82434462
## W+  -0.3054053  0.81887572
```

2. Perform the variance of the columns of F and check that you get the eigenvalues of the GSVD.

```r
apply(F,2,function(x){sum(x^2)/n})
d^2
```

```
##      dim1      dim2      dim3      dim4      dim5      dim6      dim7
## 2.8896370 2.3084237 1.2657243 0.9453242 0.9007960 0.7397718 0.4887748
##      dim8      dim9     dim10
## 0.2740185 0.1412515 0.0462782
##  [1] 2.8896370 2.3084237 1.2657243 0.9453242 0.9007960 0.7397718 0.4887748
##  [8] 0.2740185 0.1412515 0.0462782
```

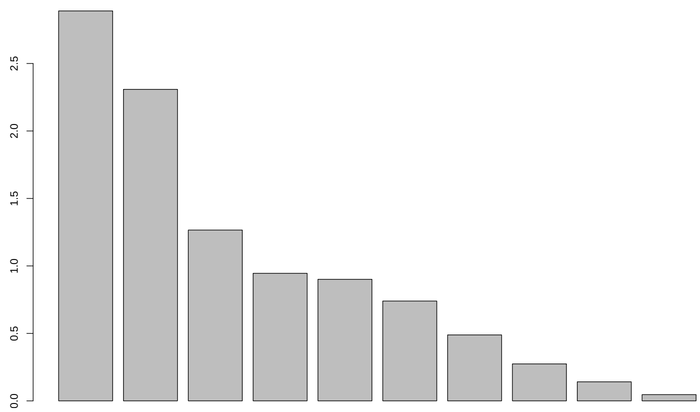
3. Check that the sum of all the eigenvalues is equal to the total inertia.

```r
sum(d^2)
```

```
## [1] 10
```

4. Plot the eigenvalues with the function **barplot**. How many dimension $q \leq r$ would you keep here ?

```r
barplot(d^2)
```




```
q <- 3 # keep two or 3 dimensions
```

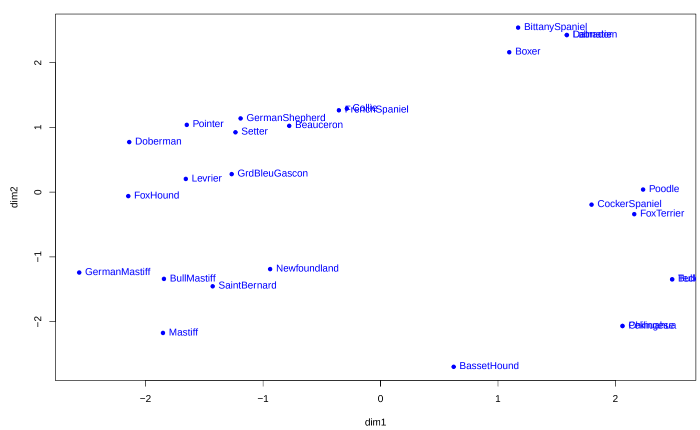
5. Perform the proportion of inertia explained by the q first principal components.

```
d^2/total
sum(d[1:q]^2/total)
```

```
##  [1] 0.28896370 0.23084237 0.12657243 0.09453242 0.09007960 0.07397718
##  [7] 0.04887748 0.02740185 0.01412515 0.00462782
## [1] 0.6463785
```

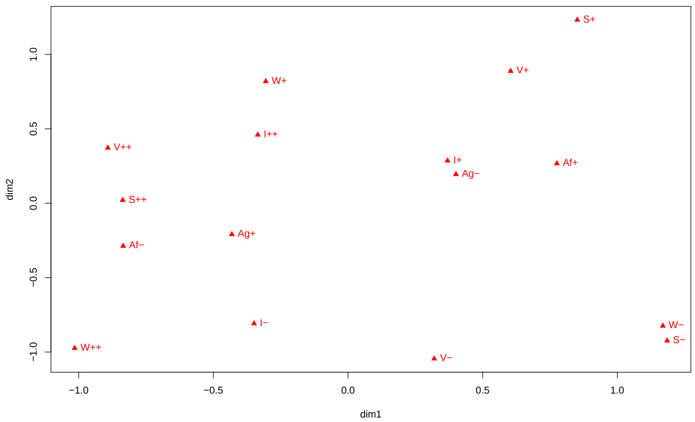
6. Plot of the dogs according to their factor coordinates on dim1-2.

```
plot(F[,1:2],col=4,pch=16,
     xlab="dim1", ylab="dim2")
text(F[,1:2],rownames(F),pos=4,cex=1,col=4)
```



7. Plot of the levels according to their factor coordinates on dim1-2.

```
plot(A[,1:2],col=2,pch=17,xlab="dim1", ylab="dim2")
text(A[,1:2],rownames(A),pos=4,cex=1,col=2)
```

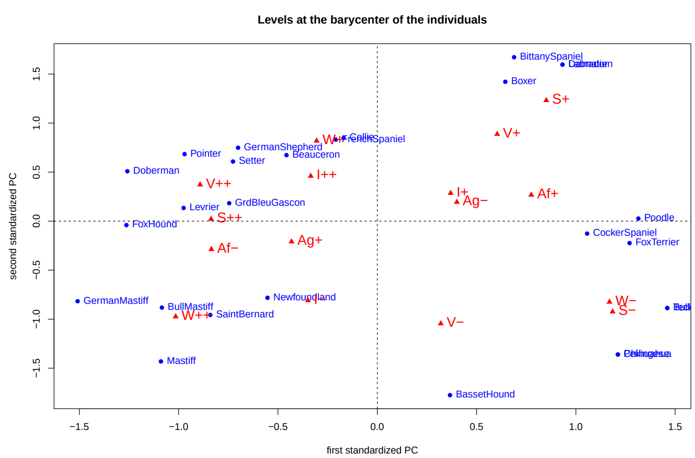


8. Check the barycentric property for the level S-.

```
which(K[,1]==1) #dogs of size S-
apply(U[which(K[,1]==1),1:2],2,mean) #mean of the standardized coordinates of the dogs of size S++ on d
A[1,1:2] #factor coordinates of S++
```

9. Plot the levels at the barycenter of the dogs on dim1-2.

```
plot(U[,1:2],main="Levels at the barycenter of the individuals",col=4,pch=16,xlab="first standardized PC
text(U[,1:2],rownames(U),pos=4,cex=1,col=4)
points(A[,1:2],pch=17,col=2)
text(A[,1:2],rownames(A),pos=4,col=2,cex=1.4)
abline(h=0,lty=2)
abline(v=0,lty=2)
```



10. Perform the matrix C of dimension $p \times 2$ of the contributions of the categorical variables to the inertia of the two first principal components.

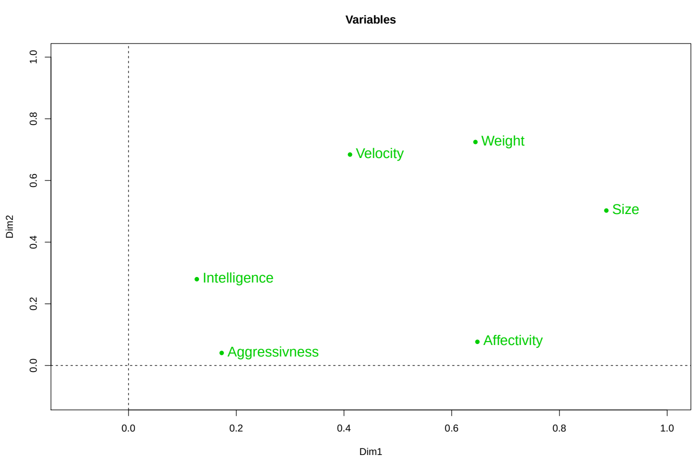
```
eta2 <- function(x, gpe) {
  moyennes <- tapply(x, gpe, mean)
  effectifs <- tapply(x, gpe, length)
  varinter <- (sum(effectifs * (moyennes - mean(x))^2))
  vartot <- (var(x) * (length(x) - 1))
  res <- varinter/vartot
  return(res)
}
```

```
C <- matrix(NA,p,2)
C[,1] <- apply(data,2,function(x){eta2(F[,1],x)})
C[,2] <- apply(data,2,function(x){eta2(F[,2],x)})
rownames(C) <- colnames(data)
colnames(C) <-colnames(F)[1:2]
print(C,digit=2)
```

```
##              dim1  dim2
## Size         0.89 0.502
## Weight       0.64 0.725
## Velocity     0.41 0.684
## Intelligence 0.13 0.280
## Affectivity  0.65 0.077
## Aggressivness 0.17 0.041
```

11. Plot the variables according to their contributions to the two first principal components.

```
plot(C,main="Variables",col=3,pch=16,xlab="Dim1", ylab="Dim2",
     xlim=c(-0.1,1), ylim=c(-0.1,1))
text(C[,1:2],rownames(C),pos=4,cex=1.4,col=3)
abline(h=0,lty=2)
abline(v=0,lty=2)
```



Exercice 3. MCA with R functions.

We want now to perform MCA using the functions **MCA** of the R package **FactoMineR** and **PCAmix** of the R package **PCAmixdata**.

1. Apply the function **MCA** to the dogs dataset. Explain and comment the three graphical output obtained by default.

```
library(FactoMineR)
res <- MCA(data,graph=FALSE)
```

2. Put the result in an object **res**. What is the class of this R object ? Two functions (methods) are associated with this class of R objects : **plot.MCA** and **print.MCA**. Check that is is equivalent to execute:
   a. **res** or **print.MCA(res)**
   b. **plot(res)** or **plot.MCA(res)**
3. Find in the object **res**:
   a. the numerical results used to build the previous 3 graphical representations.
   b. the numerical results used to interpret these graphics.

4. With the method **plot** associated with the objects of class **MCA**, plot on the map 1-2 the dogs, then the levels, then the levels and the dogs on the same map, then the variables.

```
?plot.MCA
plot(res) #both levels and individuals
plot(res,choix="ind",invisible="var")
plot(res,choix="ind",invisible="ind")
plot(res,choix="var")
```

5. Compare the factor coordinates obtained via the GSVD (in the exercice 3) and via the function **MCA** of **FactoMineR**. More precisely:
   a. Check that the factor coordinates of the levels and variables are identical.
   b. Check that the factor coordinates of the individuals are identical up to a multiplicative constant (to be defined).
   c. Check the consequence on the inertia of the principal components and on the total ineria.

```
#Comparison of the levels coordinates
res$var$coord[1:3,1:2]
A[1:3,1:2]
#Comparison of the individuals coordinates
res$ind$coord[1:3,1:2]
F[1:3,1:2]
res$ind$coord[1:3,1:2]*sqrt(p)
#Comparison of the variance of the PCs
res$eig[,1]
d^2
res$eig[,1]*p
#Comparison of the total inertia
sum(res$eig[,1]) #(m-p)/p
sum(d^2) #m-p
```

6. Apply now the function **PCAmix** of the R package **PCAmixdata**. Answer the same questions as previously with the function **MCA** of the package **FactoMineR**.

```
library(PCAmixdata)
?PCAmix
res2 <- PCAmix(X.quali=data,graph=FALSE)
class(res2)
names(res2)

res2$ind$coord[1:5,]
res2$levels$coord[1:5,]
res2$sqload
res2$eig

?plot.PCAmix
plot(res2,choice="ind")
plot(res2,choice="levels")
plot(res2,choice="sqload")
```

Exercice 4. PCA of a mixture of quantitative and qualitative data.

We want now to use a method called **PCAmix** wich performs a Principal Component Analysis of a mixture of numerical and categorical data. This function is implemented in the R package **PCAmixdata**.

1. First check that the function **PCAmix** performs a **PCA** if all the data are numerical and an **MCA** if all the data are categorical. Use the examples provided in the help of the function.

2. Use the vignette of the package to see the main possibilities of the function **PCAmix** (prediction and supplementary variables for instance).
3. Use the vignette to discover the possibilities of the functions **PCArot** and **MFAmix** of the package.