Principle Component Analysis - PCA

Bealy MECH

11/15/2021

Exercice 1 : Coding PCA

1) Code my own PCA:

On centre sans réduire les données, puis calcul de la variance de X, ensuite la valeur propre de la matrice de variance.

```
# Data input
X <- read.table(text = "
             math  scie  fran  lati   d-m
jean          6.0   6.0   5.0   5.5   8.0
aline         8.0   8.0   8.0   8.0   9.0
annie         6.0   7.0  11.0   9.5  11.0
monique      14.5  14.5  15.5  15.0   8.0
didier       14.0  14.0  12.0  12.5  10.0
andre        11.0  10.0   5.5   7.0  13.0
pierre        5.5   7.0  14.0  11.5  10.0
brigitte     13.0  12.5   8.5   9.5  12.0
evelyne       9.0   9.5  12.5  12.0  18.0
")
# create a table in pdf
knitr::kable(X, format = "latex", caption = "Tableau centré", digits = 2)
```

```
X
```

Table 1: Tableau centré

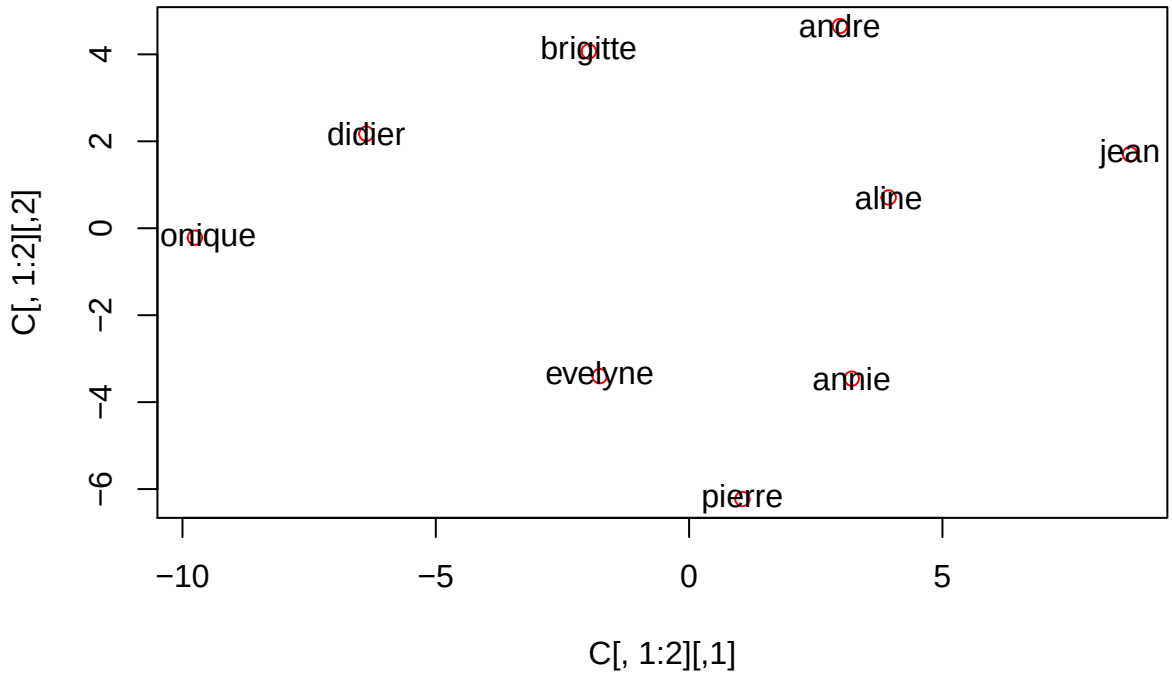
	math	scie	fran	lati	d.m
jean	6.0	6.0	5.0	5.5	8
aline	8.0	8.0	8.0	8.0	9
annie	6.0	7.0	11.0	9.5	11
monique	14.5	14.5	15.5	15.0	8
didier	14.0	14.0	12.0	12.5	10
andre	11.0	10.0	5.5	7.0	13
pierre	5.5	7.0	14.0	11.5	10
brigitte	13.0	12.5	8.5	9.5	12
evelyne	9.0	9.5	12.5	12.0	18

```
##          math scie fran lati d.m
## jean      6.0  6.0  5.0  5.5   8
## aline     8.0  8.0  8.0  8.0   9
## annie     6.0  7.0 11.0  9.5  11
## monique  14.5 14.5 15.5 15.0   8
## didier   14.0 14.0 12.0 12.5  10
## andre    11.0 10.0  5.5  7.0  13
## pierre    5.5  7.0 14.0 11.5  10
## brigitte 13.0 12.5  8.5  9.5  12
## evelyne   9.0  9.5 12.5 12.0  18
```

```
X <- scale(X, center = TRUE, scale = FALSE)
S <- var(X) # la matrice de variance
res.mypca <- eigen(S) # compute the eigenvalues and eigenvectors of S
Lambda <- res.mypca$values # the eigenvalues of S
Lambda
```

```
## [1] 31.78490603 13.58406368  9.69270028  0.02444871  0.01110353
```

```
U <- res.mypca$vectors # the eigenvector of S
C <- X%*%U
plot(C[,1:2], col = "red") # run plot together with text
text(C[,1:2], row.names(X))
```




```
# Compare with code
res.pca <- prcomp(X) # Principale Components Analysis
summary(res.pca)
```

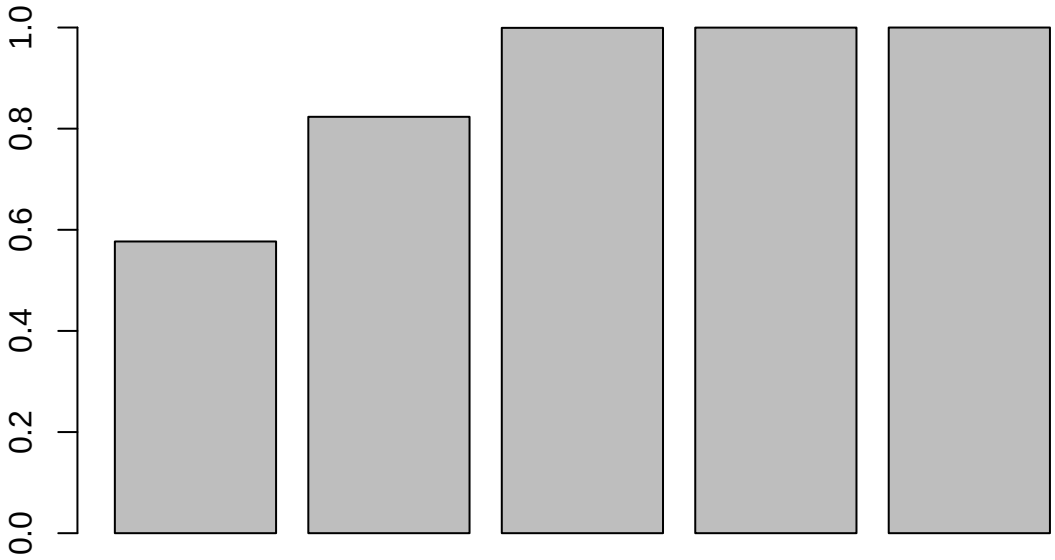
```
## Importance of components:
##                           PC1     PC2     PC3      PC4     PC5
```

```
## Standard deviation     5.6378 3.6857 3.1133 0.15636 0.1054
## Proportion of Variance 0.5769 0.2465 0.1759 0.00044 0.0002
## Cumulative Proportion  0.5769 0.8234 0.9993 0.99980 1.0000
```

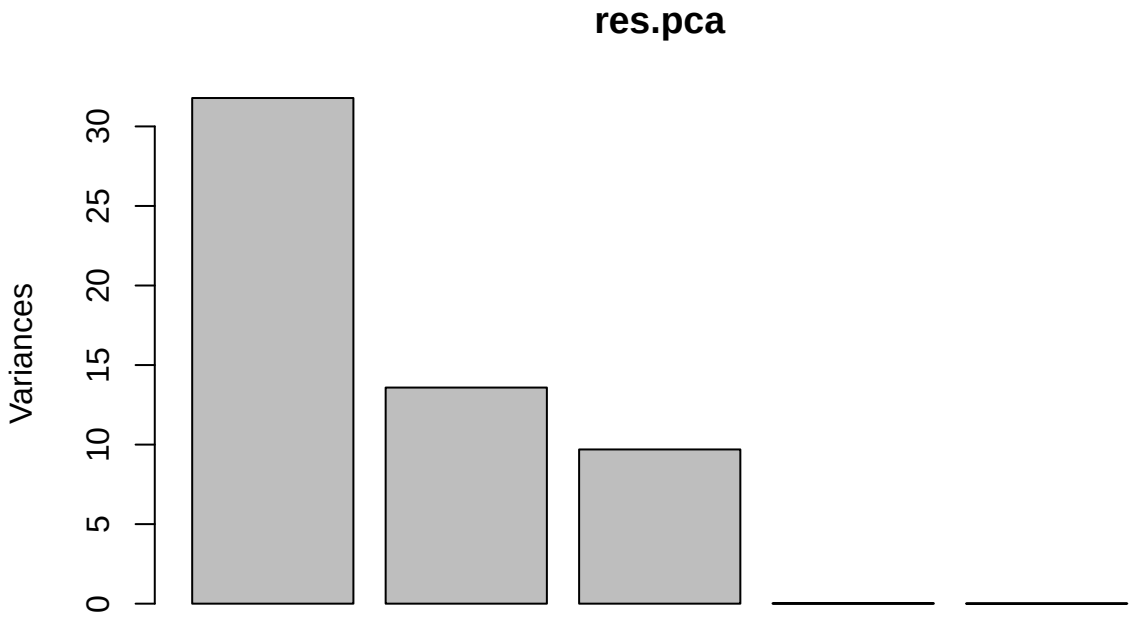
```
lamb <- res.pca$sdev**2
lamb
```

```
## [1] 31.78490603 13.58406368  9.69270028  0.02444871  0.01110353
```

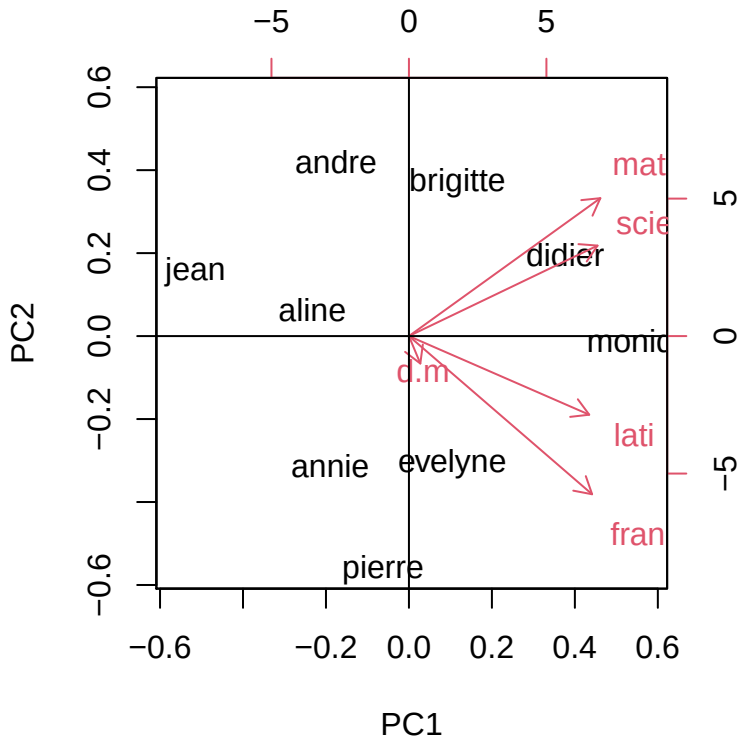
```
barplot(cumsum(lamb)/sum(lamb)) # we see last 3 variables are almost equal to 1,
```



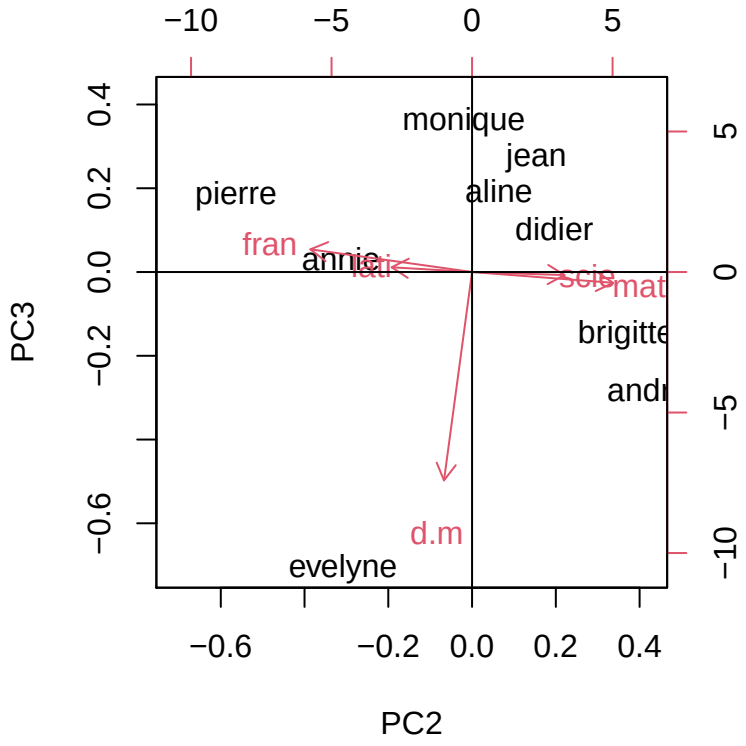

```
# which means that only these 3 variables will give us almost 100% of their information
plot(res.pca) #give the variance of each variables
```



```
biplot(res.pca) # give the direction of 5 variables into 2 new PCs
abline(h=0, v=0)
```




```
biplot(res.pca, choices = c(2,3)) #PC2 = C2, PC1 = C2 where PC is Principale Component
abline(h=0, v=0)
```



Exercice 2 : PCA and size effect

```
library(MASS)
data(crabs)
crabsquant <- crabs[,4:8] # we don't want those 3 first variables qualitative
dim(crabsquant) # the dataset now consists of 200 crabs, with 5 variables quantitative
```

```
## [1] 200   5
```

let's see the correlation matrix:

```
cor(crabsquant)
```

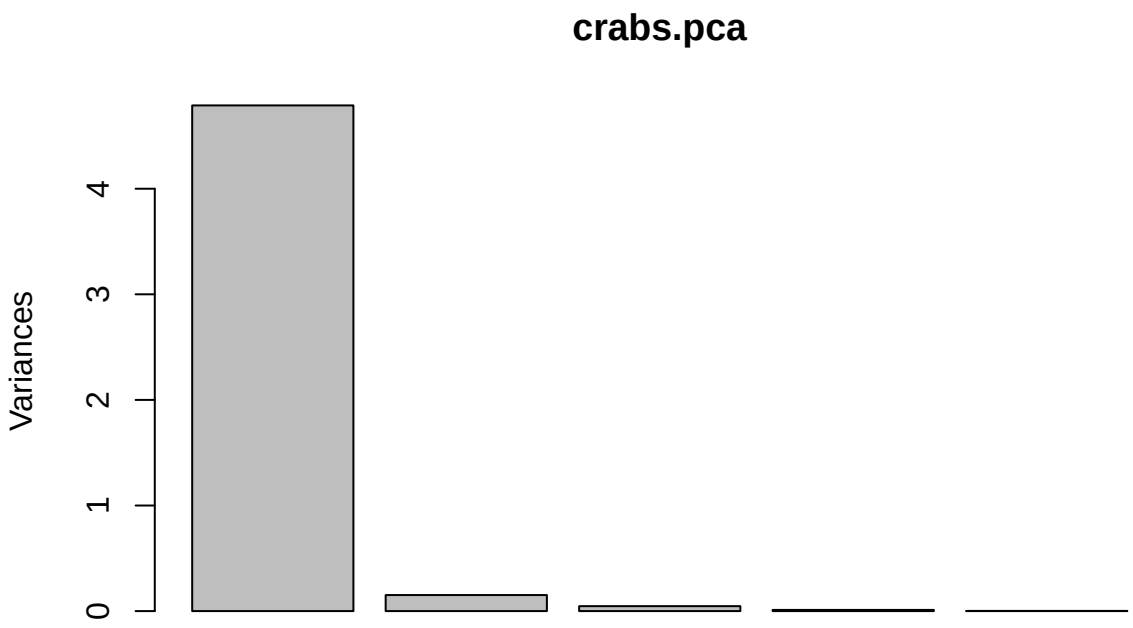
```
##           FL        RW        CL        CW        BD
## FL 1.0000000 0.9069876 0.9788418 0.9649558 0.9876272
## RW 0.9069876 1.0000000 0.8927430 0.9004021 0.8892054
## CL 0.9788418 0.8927430 1.0000000 0.9950225 0.9832038
## CW 0.9649558 0.9004021 0.9950225 1.0000000 0.9678117
## BD 0.9876272 0.8892054 0.9832038 0.9678117 1.0000000
```

Those variables are extremely correlated.

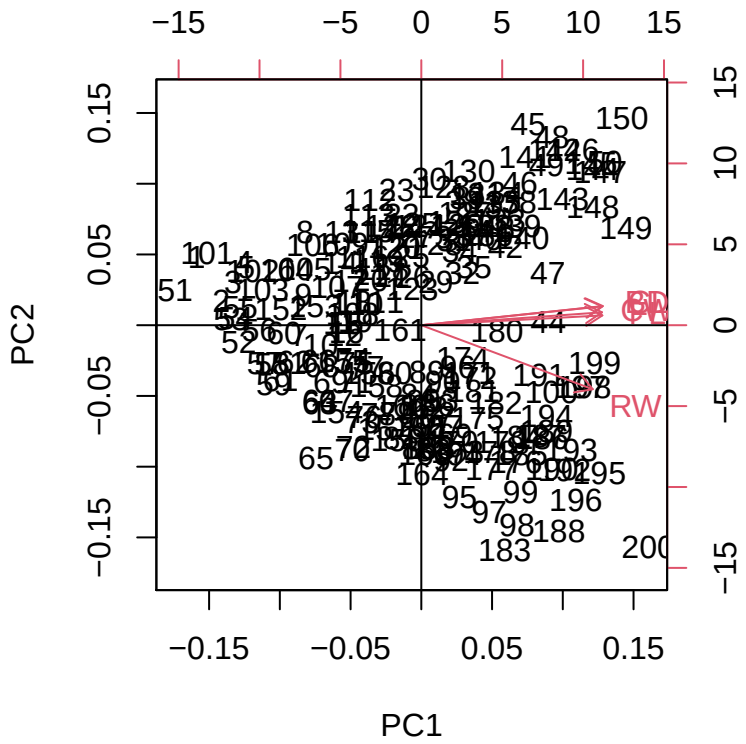
```
# 1) Test a PCA crabsquant without any preliminary transformation
crabs.pca <- prcomp(scale(crabsquant))
summary(crabs.pca) # the proportion of variance of PC1 is: 98.25% and PC2 is: 0.906%
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5
## Standard deviation     2.1883 0.38947 0.21595 0.10552 0.04137
## Proportion of Variance 0.9578 0.03034 0.00933 0.00223 0.00034
## Cumulative Proportion  0.9578 0.98810 0.99743 0.99966 1.00000
```

```
plot(crabs.pca) # the variance of each variables
```



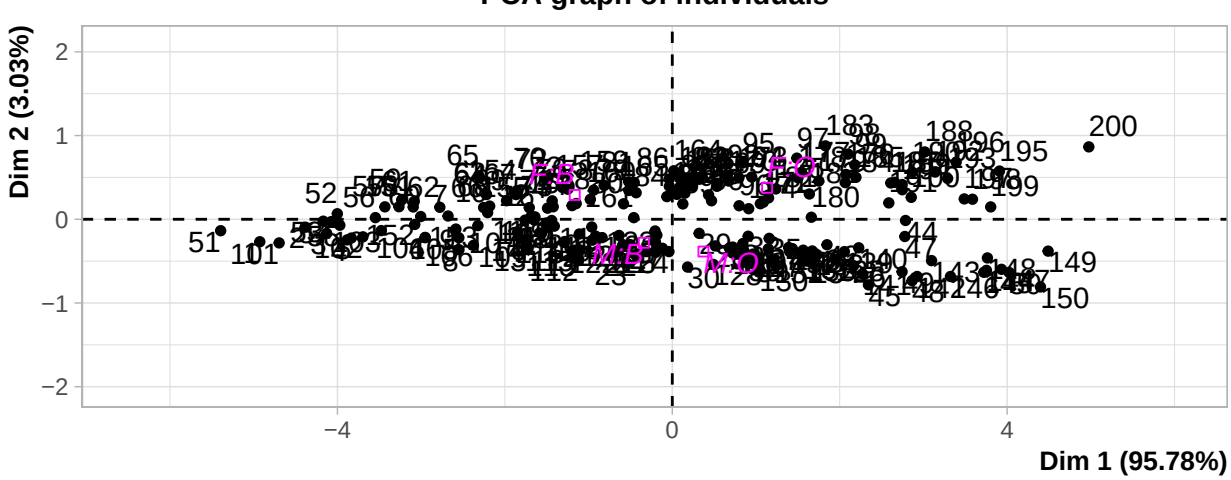
```
biplot(crabs.pca) # the direction of 5 variables into PC1 and PC2
abline(h=0, v=0) # add the line h=0 and v=0
```

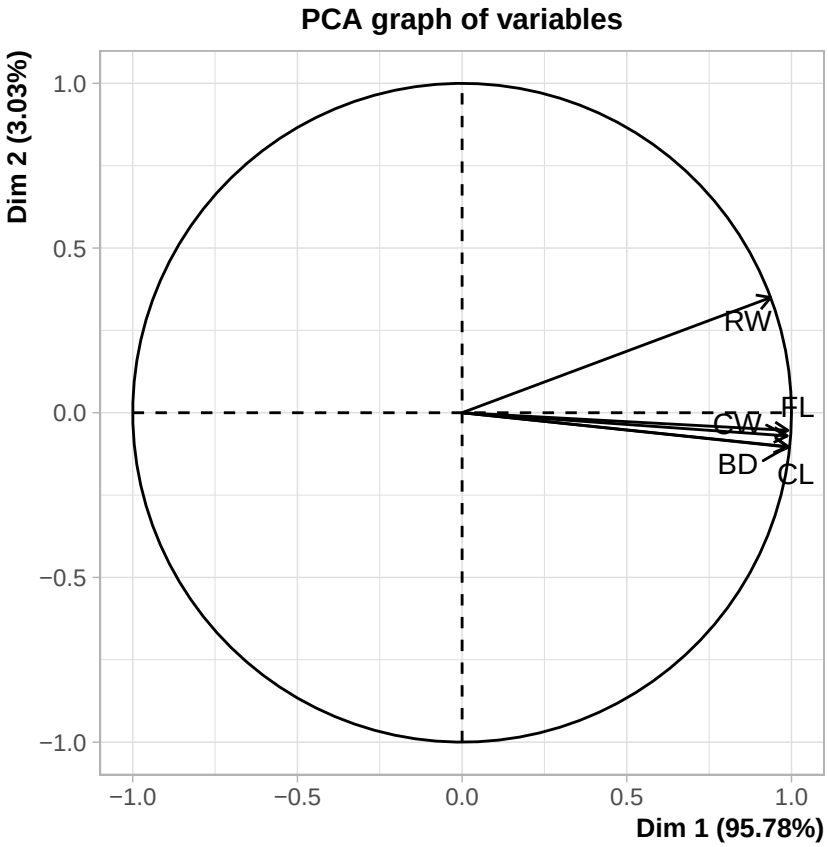



```
# 2) We use the library FactorMineR to improve the quality of PCA
library(FactoMineR)
sex.sp <- crabs$sex:crabs$sp # related sex vs species
X <- data.frame(crabsquant, sex.sp) # add the column sex.sp to the dataframe with crabsquant data
dim(X)
```

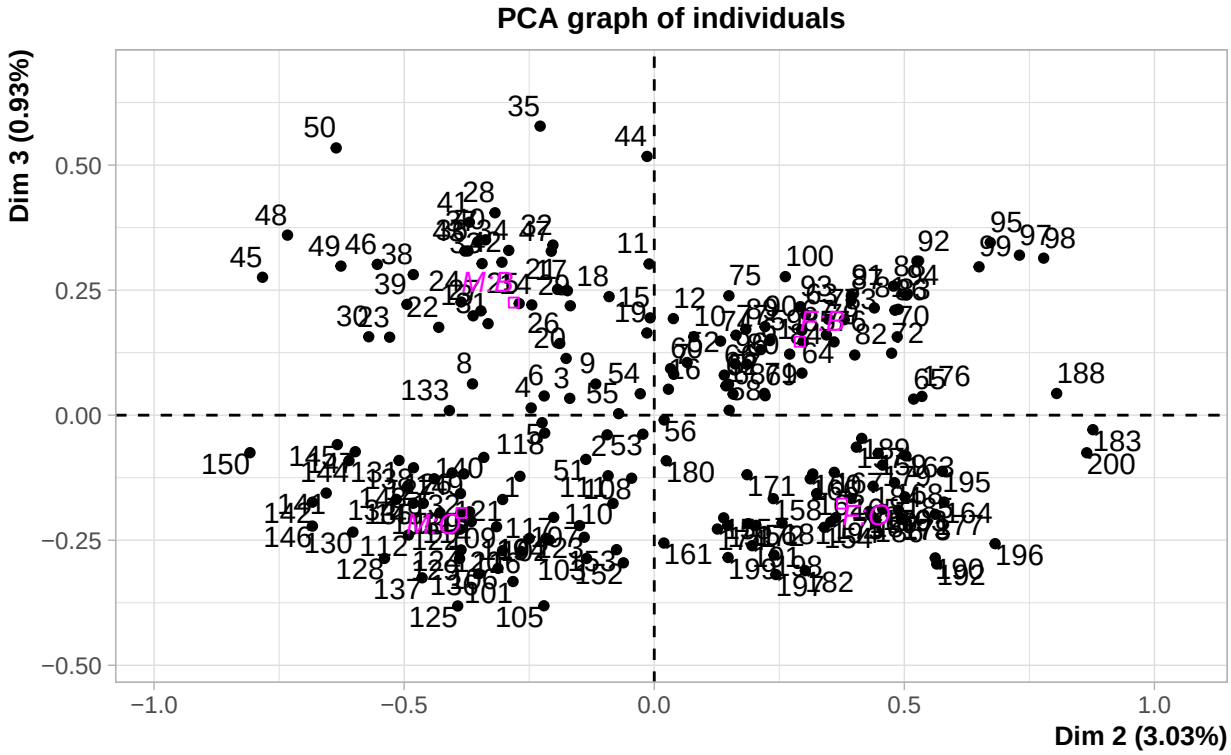
```
## [1] 200   6
```

```
res <- PCA(X, quali.sup = 6) # the 6th vector indicates the indexes of the categorical supplementary va
```

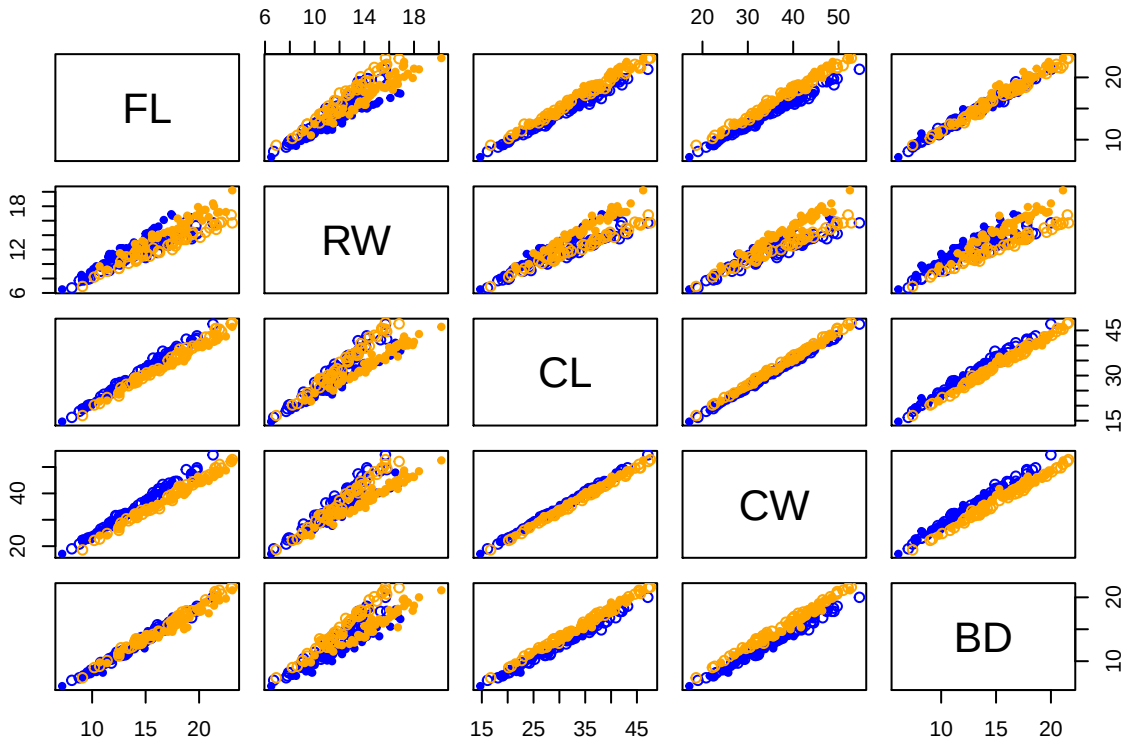




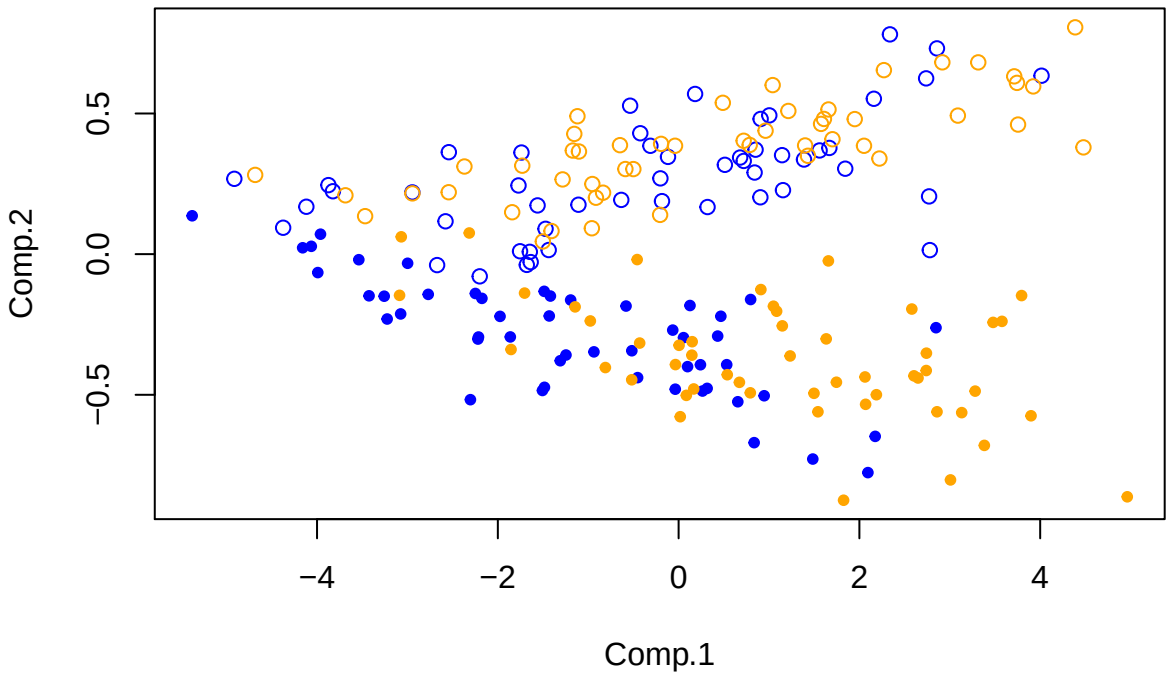

```
plot.PCA(res, axes = c(2, 3),
         col.ind.sup = c("blue1", "orange1", "blue4", "orange4")[sex.sp])
```



```
pairs(crabsquant,col=c("blue","orange")[crabs$sp],pch=c(20,21)[crabs$sex])
```



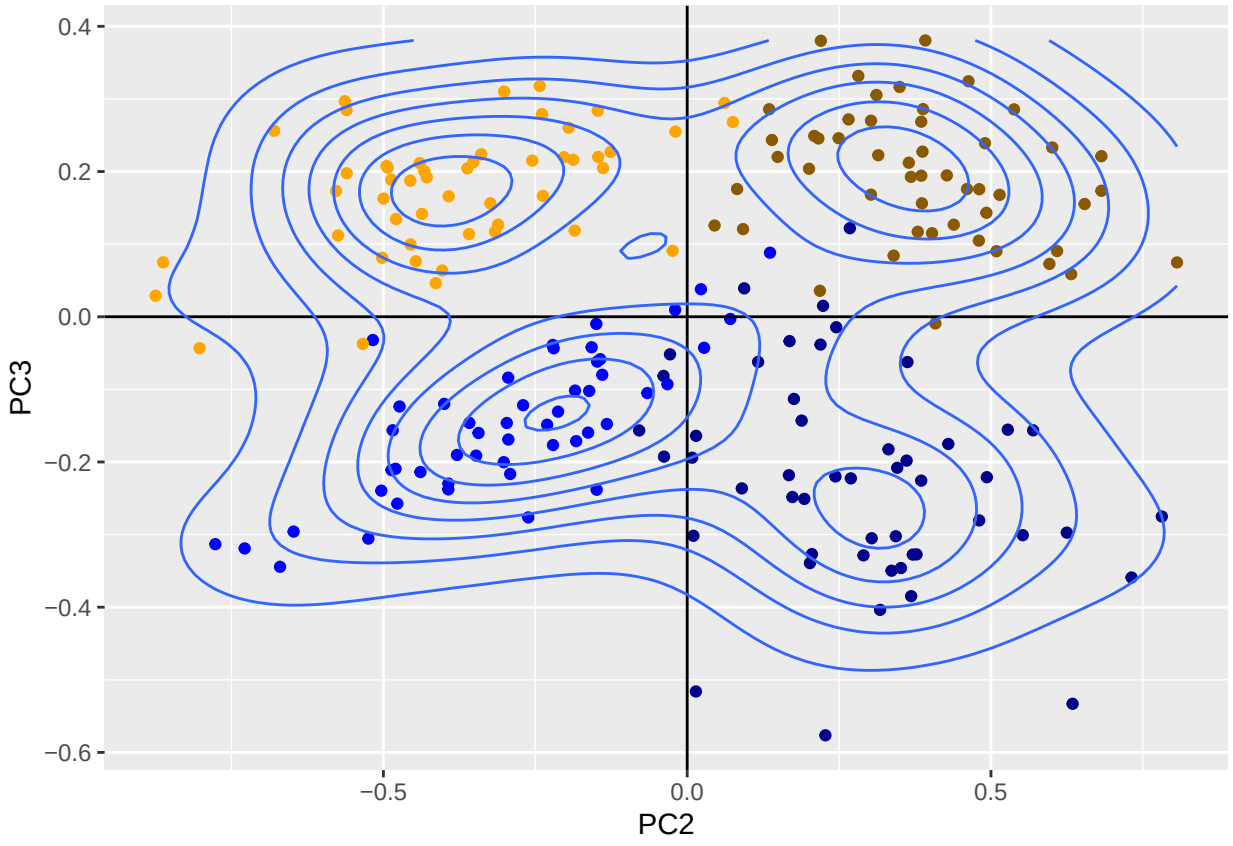

```
res<-princomp(scale(crabsquant))
plot(res$scores[,1:2],col=c("blue","orange")[crabs$sp],pch=c(20,21)[crabs$sex])
```



```
library(ggplot2)
respca <- prcomp(X[,1:5], scale. = TRUE)
respca
```

```
## Standard deviations (1, .., p=5):
## [1] 2.18834065 0.38946785 0.21594669 0.10552420 0.04137243
##
## Rotation (n x k) = (5 x 5):
##          PC1        PC2         PC3          PC4         PC5
## FL 0.4520437  0.1375813  0.53076841  0.696923372  0.09649156
## RW 0.4280774 -0.8981307 -0.01197915 -0.083703203 -0.05441759
## CL 0.4531910  0.2682381 -0.30968155 -0.001444633 -0.79168267
## CW 0.4511127  0.1805959 -0.65256956  0.089187816  0.57452672
## BD 0.4511336  0.2643219  0.44316103 -0.706636423  0.17574331
```

```
p <- ggplot(data = data.frame(respca$x), aes(x=PC2, y=PC3)) +
  geom_point(colour = c("blue1", "orange1", "blue4", "orange4")[sex.sp]) +
  geom_vline(xintercept = 0) + # vertical line
  geom_hline(yintercept = 0) + # horizontal line
  geom_density2d() # add the density in 2d (contour line)
p
```




```
ggsave(filename = "pca.pdf", p) # save file with name "pca.pdf", open in terminal by tapping p
```

```
## Saving 6.5 x 4.5 in image
```

Exercice 3 : Phylogeny of Globins

Let's us check that it's a dissimilarity dataset if it follows the conditions below:

1. The matrix is symmetric
2. The diagonal of the matrix is equal to zeros

```
# 1) Load the data
Delta <- read.table(file = "neighbor_globin.txt", header = FALSE, row.names = 1)
Delta <- as.matrix(Delta) # Delta here is (Dij) as a matrix
dim(Delta) # Delta is a squared matrix with n=21
```

```
## [1] 21 21
```

```
head(Delta)
```

```
##               V2     V3     V4     V5     V6     V7     V8     V9    V10    V11
## MYG_PHYCA 0.0000 0.1806 0.2434 0.3964 0.5656 0.4987 1.9654 2.1040 2.1278 2.0965
## MYG_HUMAN 0.1806 0.0000 0.1929 0.2997 0.4852 0.4271 1.9675 2.0689 2.2427 2.1483
## MYG_MOUSE 0.2434 0.1929 0.0000 0.3432 0.5312 0.4635 1.8727 2.1478 2.1478 2.1092
## MYG_CHICK 0.3964 0.2997 0.3432 0.0000 0.3657 0.3196 1.8520 2.0577 2.0649 1.8216
## MYG_ALLMI 0.5656 0.4852 0.5312 0.3657 0.0000 0.2970 1.8912 2.0551 2.0572 1.7896
## MYG_CHEMY 0.4987 0.4271 0.4635 0.3196 0.2970 0.0000 1.7142 1.9036 1.9751 1.6927
##              V12    V13    V14    V15    V16    V17    V18    V19    V20    V21
## MYG_PHYCA 2.2725 2.0807 1.9645 1.9928 1.9195 2.0944 1.9867 1.9486 1.8515 1.9880
## MYG_HUMAN 2.2753 2.0387 2.0941 2.1273 1.9495 2.0628 2.1114 1.9951 1.9200 2.0044
## MYG_MOUSE 2.2318 1.9386 2.0581 2.0567 1.9920 2.1235 2.1776 2.0310 1.9519 2.0735
## MYG_CHICK 1.9345 2.0096 1.9935 2.0463 1.8520 1.9878 2.1320 1.9407 1.8823 2.0378
## MYG_ALLMI 1.9478 1.9237 1.7647 1.9622 1.9429 1.9423 2.0500 1.9352 1.9823 2.0511
## MYG_CHEMY 1.8907 1.8523 1.8770 1.8414 1.7849 1.8503 1.9604 1.9075 1.8643 1.7584
##              V22
## MYG_PHYCA 2.6100
## MYG_HUMAN 2.5663
## MYG_MOUSE 2.6225
## MYG_CHICK 2.5424
## MYG_ALLMI 2.3154
## MYG_CHEMY 2.4536
```

```
# 2) Check the dissimilarities properties (check that these scores correspond well to dissimilarities)
diag(Delta) # is egal to 0 (which is good)
```

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

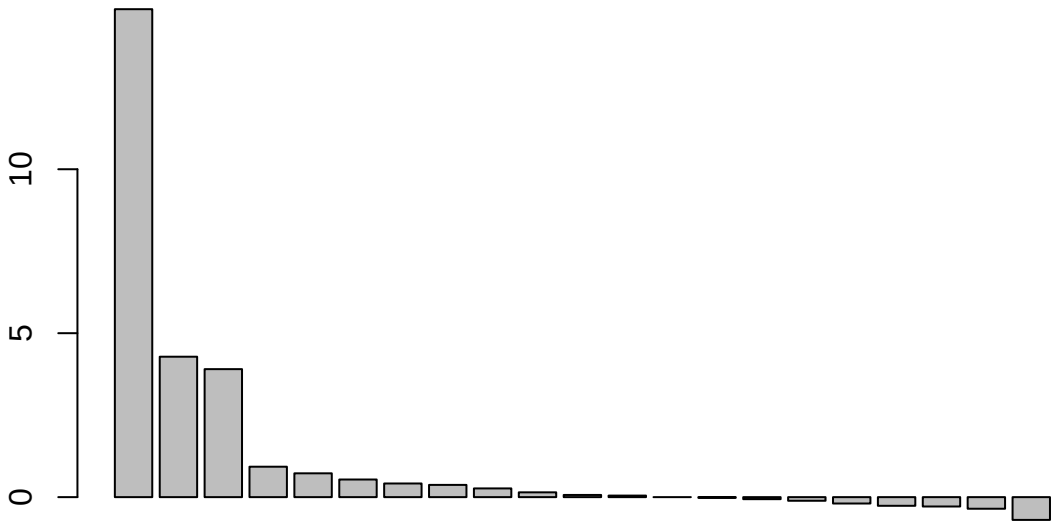
```
sum(Delta - t(Delta)) # implies that Delta = t(Delta) <=> sum(Delta - t(Delta)) = 0
```

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

```
                      # => this matrix is symmetric
```

```r
# 3) Compute the matrix Delta of squared dissimilarities
Delta3 <- Delta%*%Delta
Delta2 <- Delta^2
```

```r
# 4) Compute the centering matrix J : J = I - (1/n)1(n,n)
n <- ncol(Delta) # = nrow(Delta)
J <- diag(rep(1,n)) - (1/n)*matrix(1,n,n)
#J

# 5) Compute B = -1/2*J*Delta*J
B <- -(1/2)*J%*%Delta2%*%J

# B could be interpreted as a "pseudo" scal product
# 6 + 7) Perform the spectral decomposition of B
EigenB <- eigen(B)
Lambda <- EigenB$values
U <- EigenB$vectors # eigen vectors of B
# U%*%diag(Lambda)%*%t(U) = B, check
barplot(Lambda)
```




```r
# We keep the first 3 largest eigenvalues which are positive.
print(paste("The eigen values which are negative: "))
```

```
## [1] "The eigen values which are negative: "
```

```r
which(Lambda<0)
```

```
## [1] 13 14 15 16 17 18 19 20 21
```

```r
#print(paste("The eigen values which are negative: ", which(Lambda<0)))
```

The 6 first are myoglobines. The 7 next are Hemoglobines beta. The 7 next are Hemoglobines alpha The last one is a Globine 3.

```
# 8 + 9)
C13 <- U[,1:3] # we consider only the first 3 columns
Lambda13 <- Lambda[1:3]
X <- C13%*%diag(Lambda13)
X
```

```
##            [,1]        [,2]        [,3]
##  [1,] -5.276621  0.28071344 -0.25823651
##  [2,] -5.487849  0.23008442 -0.07782307
##  [3,] -5.445457 -0.09589149 -0.17885550
##  [4,] -5.055670 -0.25537567 -0.13507955
##  [5,] -4.833841 -0.26835562  0.40520087
##  [6,] -4.521246 -0.20279045 -0.14967938
##  [7,]  1.760164 -1.10955307 -0.03572142
##  [8,]  2.372571 -1.21231275  0.14725085
##  [9,]  2.604147 -1.16686912 -0.25369313
## [10,]  1.968873 -1.29156699 -0.08037707
## [11,]  2.630732 -1.15512949 -0.03790172
## [12,]  2.092185 -0.99181637  0.12099230
## [13,]  1.801248 -1.10014691  0.18096727
## [14,]  2.164523  1.01342495 -0.65430466
## [15,]  1.756286  0.98218221 -0.54749710
## [16,]  2.064014  1.02604838 -0.88438899
## [17,]  2.449972  1.22905080 -0.06411919
## [18,]  2.041120  1.13724030 -0.15555672
## [19,]  1.832813  1.06800316 -0.30258986
## [20,]  1.885433  0.93428258 -0.61385487
## [21,]  1.196604  0.94877768  3.57526744
```

```
rownames(Delta)
```

```
##  [1] "MYG_PHYCA"  "MYG_HUMAN"  "MYG_MOUSE"  "MYG_CHICK"  "MYG_ALLMI"
##  [6] "MYG_CHEMY"  "HBB_CHICK"  "HBB_CHRPI"  "HBB1_IGUIG" "HBB_PHYCA"
## [11] "HBB_HUMAN"  "HBB1_MOUSE" "HBB_ALLMI"  "HBA_CHICK"  "HBA_CHRPI"
## [16] "HBA_ALLMI"  "HBA_PHYCA"  "HBA_HUMAN"  "HBA_MOUSE"  "HBA1_IGUIG"
## [21] "GLB3_MYXGL"
```

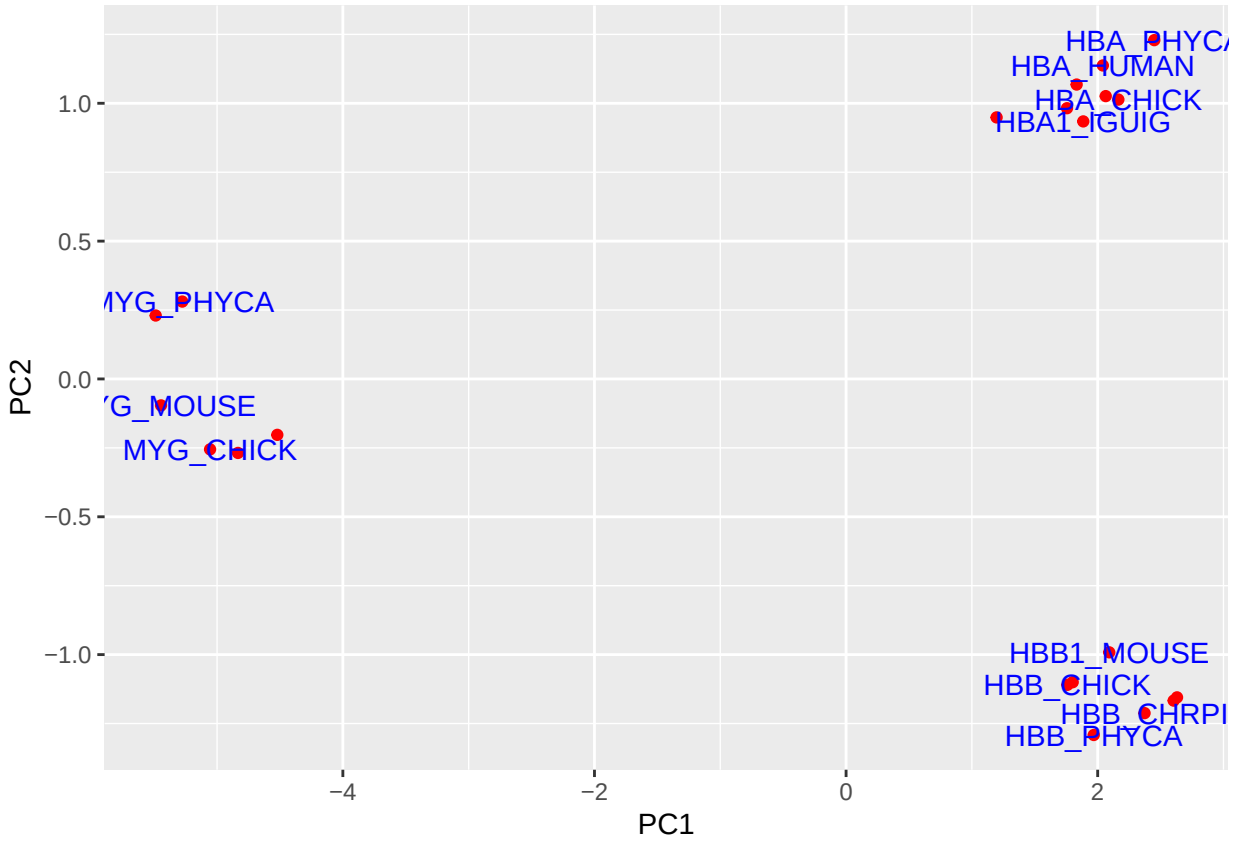
```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.1.0     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
```

```
X <- as_tibble(X)
```

```
## Warning: The ‘x‘ argument of ‘as_tibble.matrix()‘ must have unique column names if ‘.name_repair‘ is
## Using compatibility ‘.name_repair‘.
```

```
names(X) <- paste("PC", 1:3, sep = "") # name is required, otherwise error
ggplot(data = X, aes(x = PC1, y = PC2, label = row.names(Delta))) +
  geom_point(col = "red") + geom_text(check_overlap = TRUE, col = "blue")
```




```
# 10) Use cmdscale function
library(kernlab)
```

```
##
## Attaching package: 'kernlab'
```

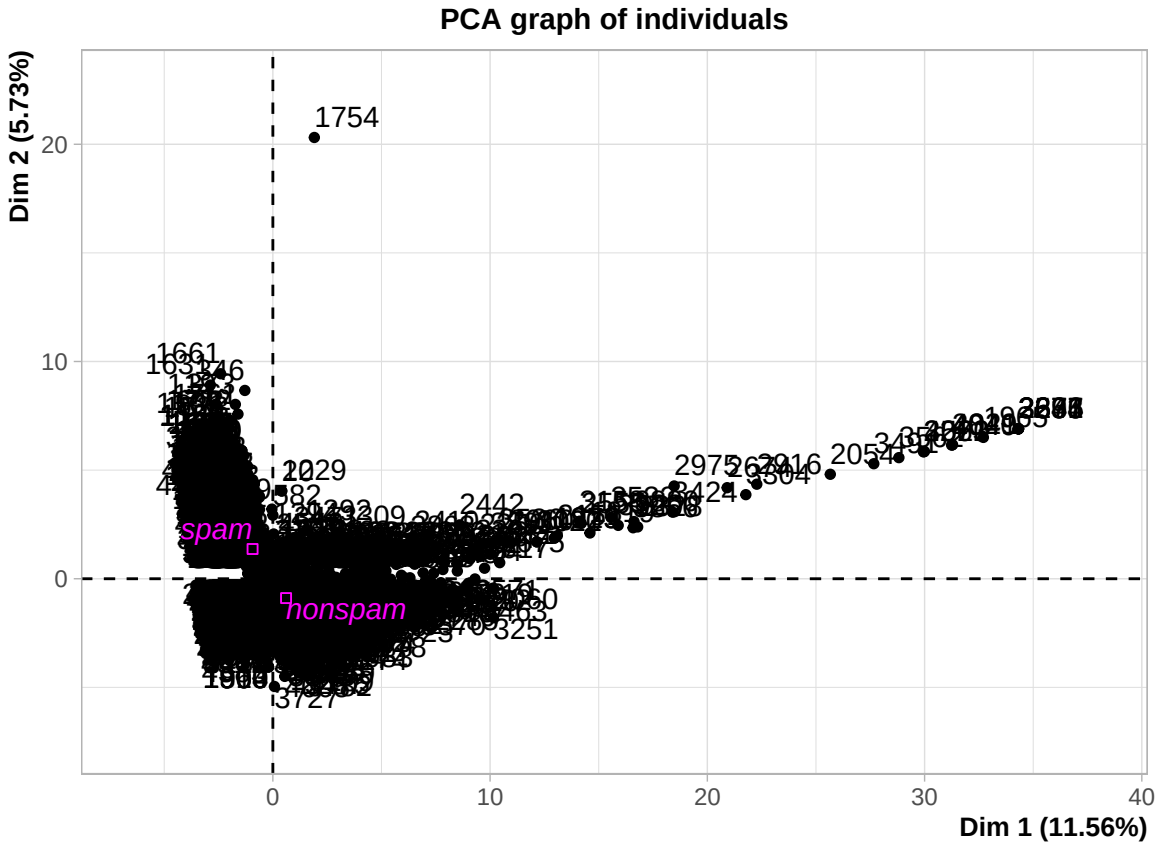
```
## The following object is masked from 'package:purrr':
##
##     cross
```

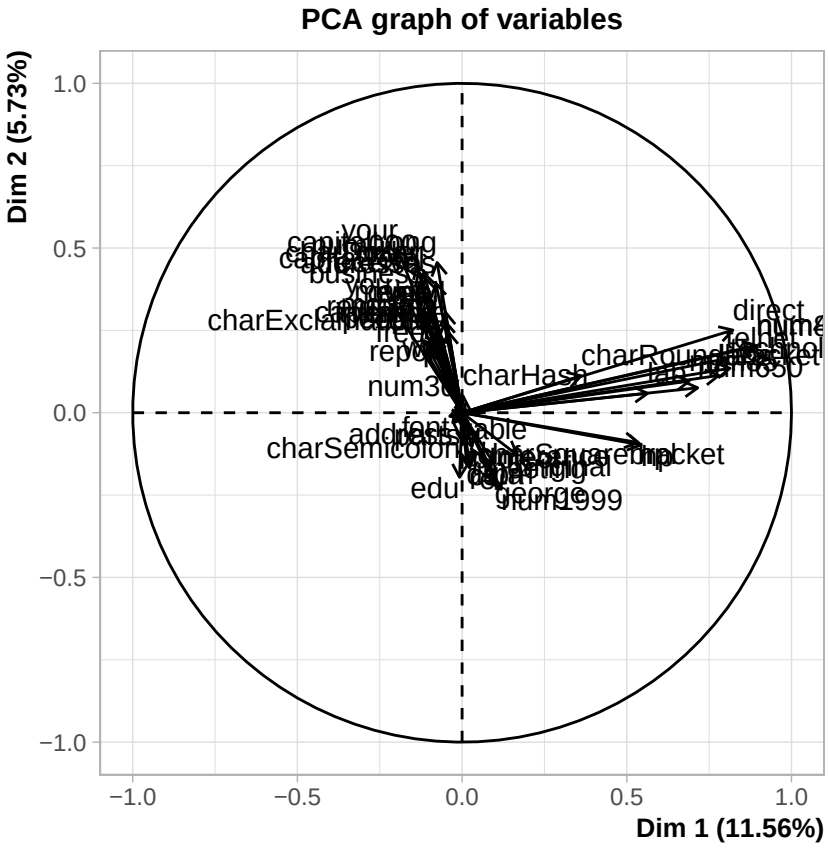
```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

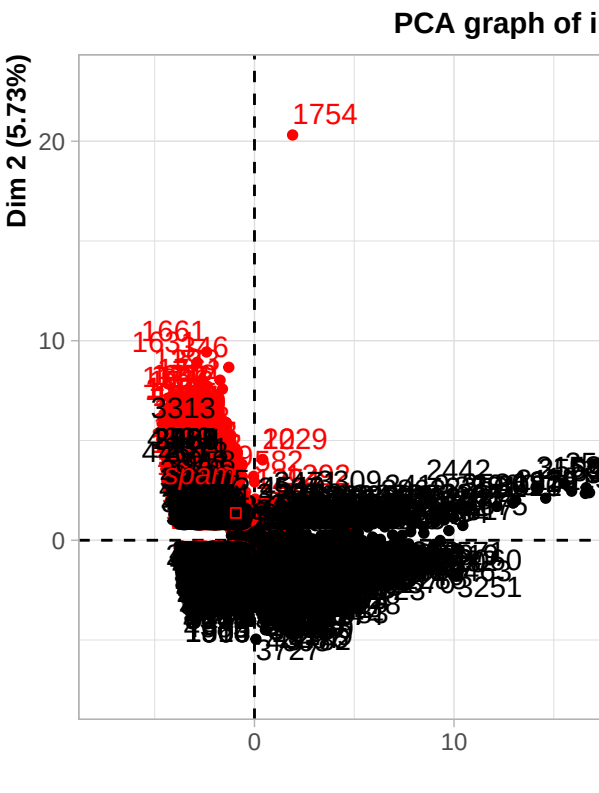
```
library(FactoMineR)
library(tidyverse)

data("spam")
dim(spam)
```

```
## [1] 4601   58
```

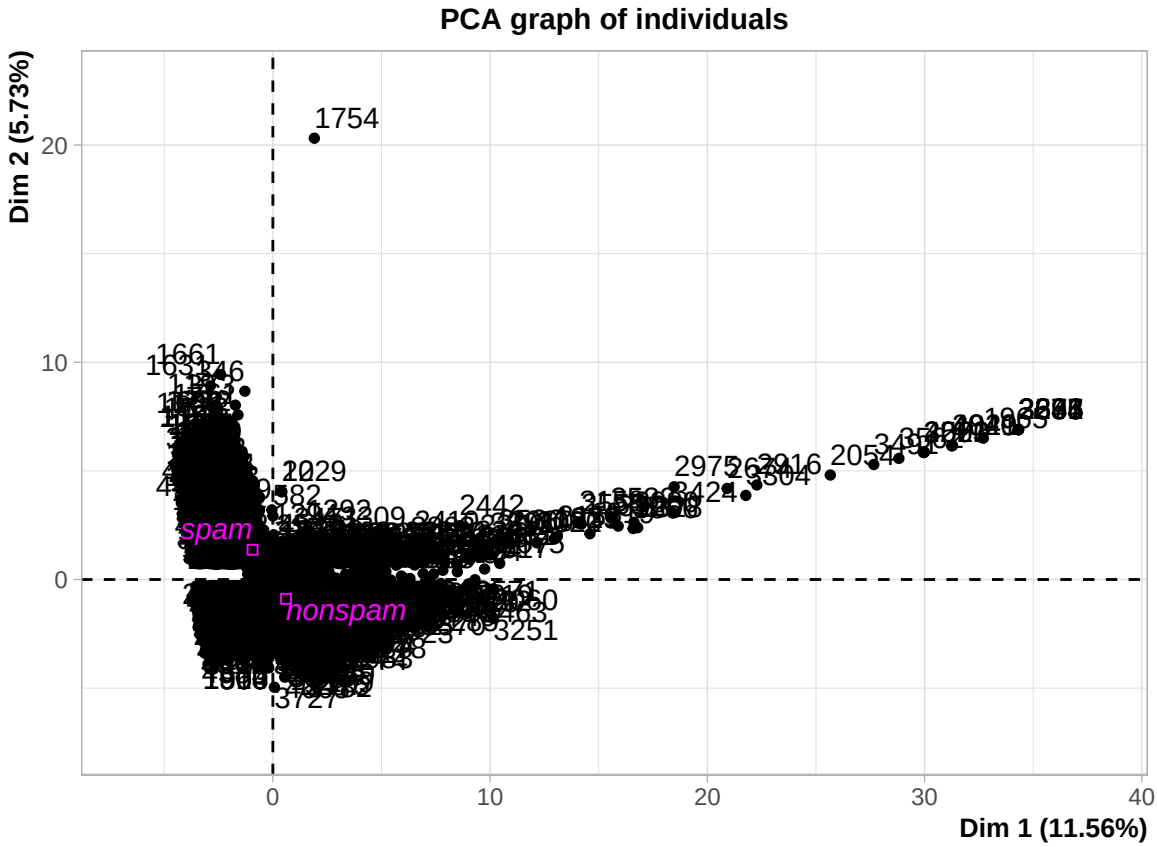
```
spam %>% PCA(., quali.sup = ncol(spam)) %>%

plot(habillage = ncol(spam), choix = "ind")
```

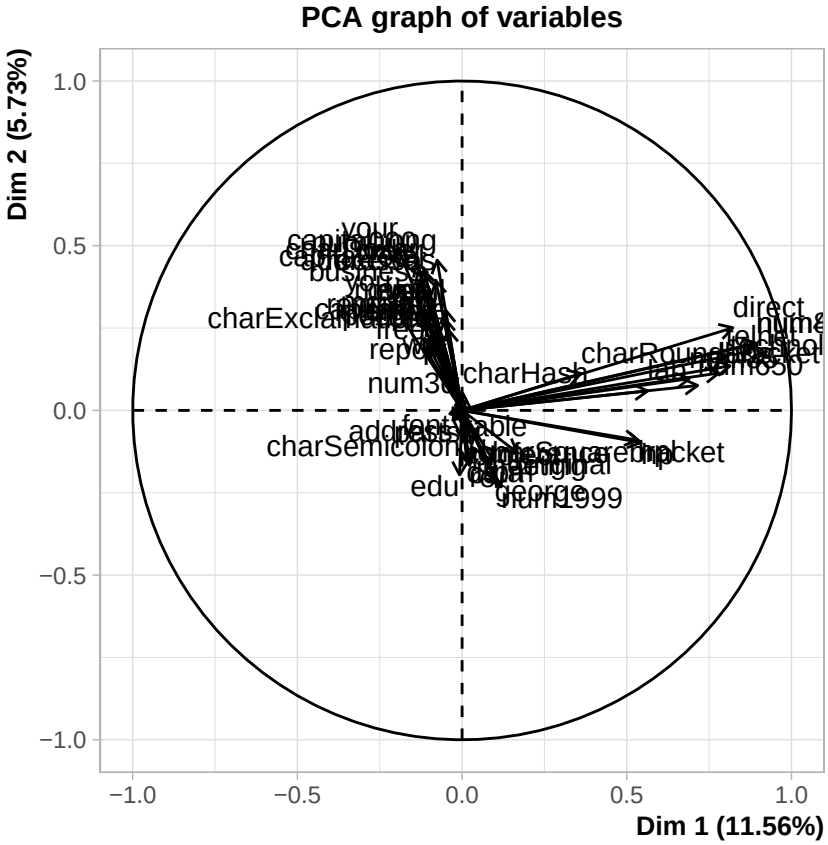




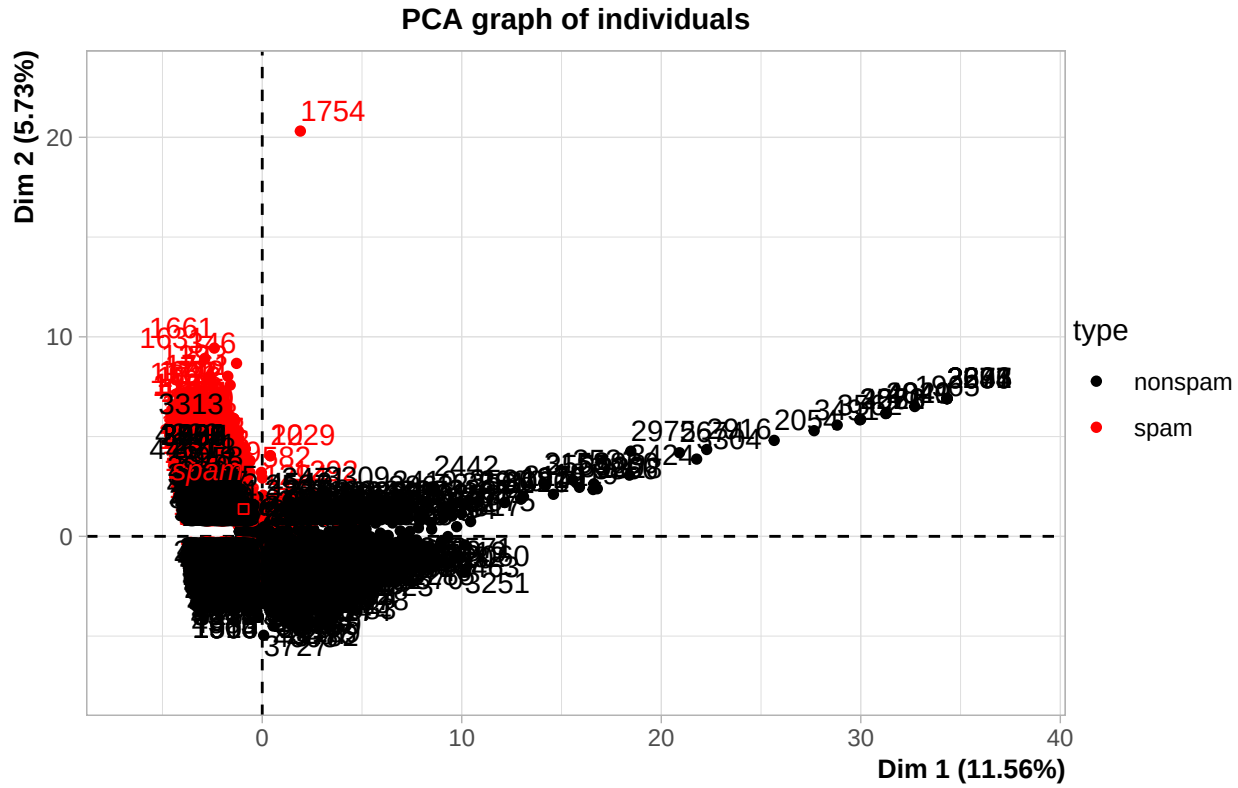



```
res.pca = PCA(spam, quali.sup = ncol(spam))
```


PCA graph of individuals
Dim 2 (5.73%)
Dim 1 (11.56%)
1754
spam
honspam
20
10
0
30
40


PCA graph of variables
Dim 2 (5.73%)
Dim 1 (11.56%)
your
charExclam
charHash
direct
charSemicolon
edu
george
num1999
1.0
0.5
0.0
−0.5
−1.0

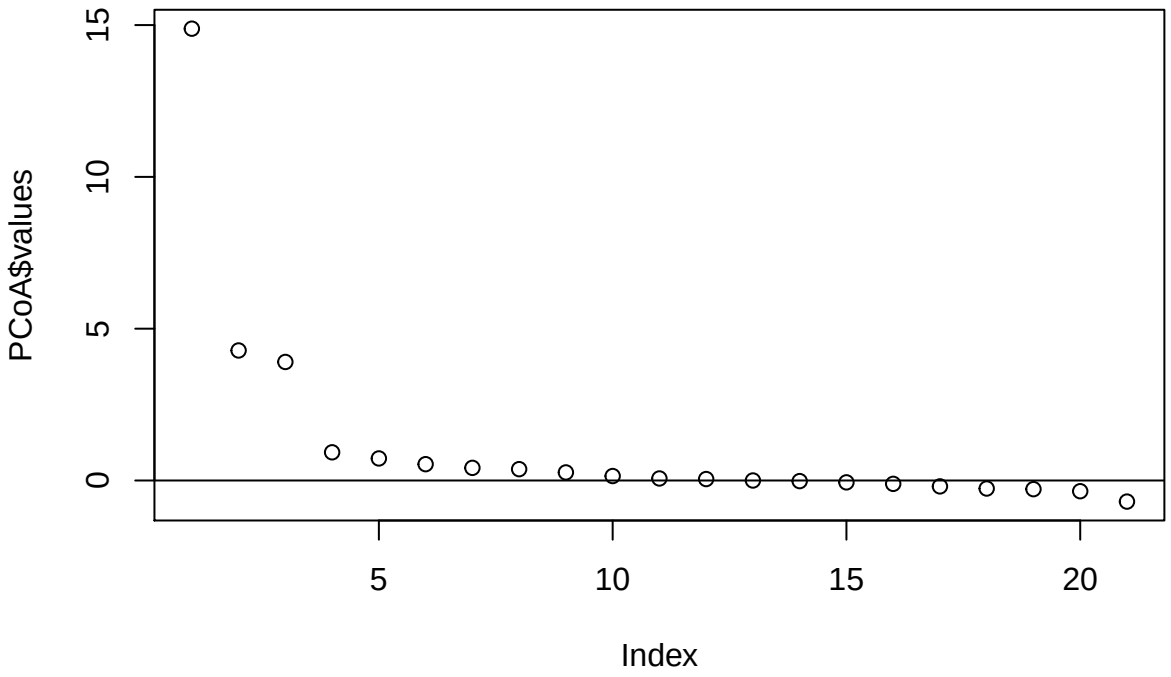
```
plot(res.pca, habillage = ncol(spam), choix = "ind")
```




```
computeJ<-function(n){
  diag(rep(1,n))  - 1/n * matrix(1,n,n)->J
}
print(computeJ(2))
```

```
##      [,1] [,2]
## [1,]  0.5 -0.5
## [2,] -0.5  0.5
```

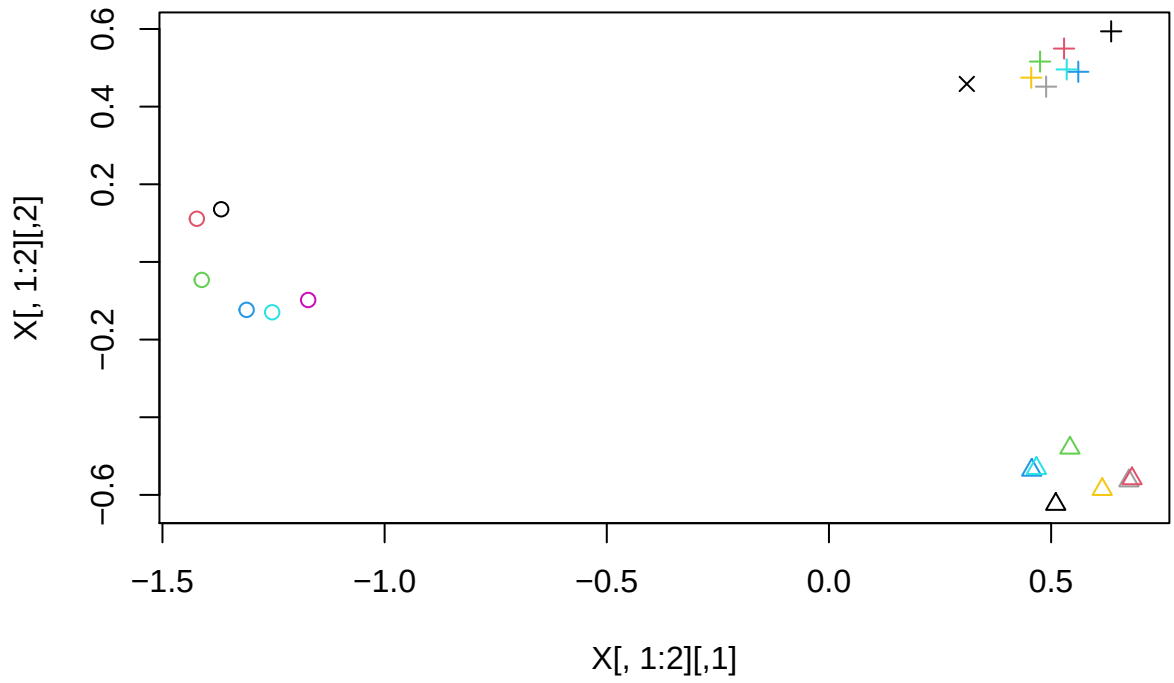
```
PCoA<-eigen(B)
plot(PCoA$values)
abline(h=0)
```




```
eigenval<-diag(PCoA$values)
eigenvect<-PCoA$vectors
q<-3
X<-eigenvect[,1:q]%*% sqrt(eigenval[1:q,1:q])
rownames(X)<-rownames(D)
pdf("PCoA.pdf")
plot(X[,1:2])
text(x=X[,1],y=X[,2],labels=rownames(X))
dev.off()
```

```
## pdf
##   2
```

```
# Type of proteins
pch.type <- c(rep(1,6),rep(2,7), rep(3,7),4)
# Different colors for the species
colors <- c(1:6,4,7,8,1:3,5,4,7,5,1:3,8,9)
plot(X[,1:2],pch=pch.type,col=colors)
```




```
#text(x=X[,1],y=X[,2],labels=rownames(X),cex=0.5)
```

```
afp <- cmdscale(as.matrix(Delta), k=4, eig=TRUE)
pch.type <- c(rep(1,6),rep(2,7), rep(3,7),4)
colors <- c(1:6,4,7,8,1:3,5,4,7,5,1:3,8,9)
plot(afp$points,pch=pch.type,col = rainbow(9)[colors])
legend("topleft",legend=c("Myoglobin","Hemoglobin Beta","Hemoglobin Alpha","Globin-3"),pch=1:4)
```