TP 2 : Analyse en composantes principales

Stefka Asenova, Johan Segers

LINGE1222 Analyse statistique multivari?f©e

Résumé de l'analyse en composantes principales

L'ACP est utilisée en analyse multivariée pour réduire le nombre de variables d'un jeu de données tout en essayant à retenir en même temps autant de la variance des données que possible.

Données originales

- X est la matrice des données originales avec p variables et n observations
- X_{cs} est la matrice des données centrées et réduites
- $R = \frac{1}{n} X_{cs}^T X_{cs}$ la matrice des corrélations, dimension $p \times p$

Composantes principales

- $\lambda_1, \ldots, \lambda_p$ les valeurs propres (`eig`)
- $A = [\mathbf{a}^1 \cdots \mathbf{a}^p]$ la matrice de vecteurs propres, dimensions $p \times p$ (`svd$V`)

Pour chaque $j = 1, \ldots, p$ on a:

$$R\mathbf{a}^j = \lambda_j \mathbf{a}^j$$

- $Y = X_{cs} A$ la matrice des composantes principales, dimensions $n \times p$ (`ind$coord`)

Metriques de l'analyse en composantes principales

- $cor(\mathbf{x}^k, \mathbf{y}^j)$ pour $k, j = 1, \ldots, p$ la matrice de correlations entre les variables et les composantes, de dimensions $p \times p$ (quand on dit variables on se refère aux colonnes de X_{cs} et quand on dit composantes on se refère aux colonnes de Y) (`var$cor` ou `var$coord`)

Soit la matrice des correlations au carré (`var$cos2` or `(var$cor)^2`):

$$\begin{bmatrix} cor^2(\mathbf{x}^1, \mathbf{y}^1) & \cdots & cor^2(\mathbf{x}^1, \mathbf{y}^p) \\ \vdots & \ddots & \vdots \\ cor^2(\mathbf{x}^p, \mathbf{y}^1) & \cdots & cor^2(\mathbf{x}^p, \mathbf{y}^p) \end{bmatrix}$$

La somme par colonne nous donne les valeurs propres, $\lambda_1, \ldots, \lambda_d$ et la somme par rangée donne 1.

Si on considère la somme par colonne on peut calculer la contribution d'une variable à la variance d'une composante.

Si on considère la somme par rangée on peut calculer la qualité de la representation d'une variable par une (ou plusieurs) composantes comme une partie de la représentation par toutes les composantes.

- contribution de la variable k à la composante j, matrice de dimensions $p \times p$ (`var$contrib` en pourcentage (!))

$$\frac{cor^2(\mathbf{x}^k, \mathbf{y}^j)}{cor^2(\mathbf{x}^1, \mathbf{y}^j) + \cdots + cor^2(\mathbf{x}^p, \mathbf{y}^j)}$$

- pour juger la qualité de la representation de la variable, soit $\mathbf{x}^k$, par les q premières composantes il faut additionner les premières q correlations au carré (regarder `var$cos2` or `(var$cor)^2` et additionner les q premières colonnes de la rangée correspondante à la variable d'interêt)

$$cor^2(\mathbf{x}^k, \mathbf{y}^1) + \cdots + cor^2(\mathbf{x}^k, \mathbf{y}^q), \quad q < p$$

Soit la matrice des composantes au carré :

$$\begin{bmatrix} y_{11}^2 & \cdots & y_{1p}^2 \\ \vdots & \ddots & \vdots \\ y_{n1}^2 & \cdots & y_{np}^2 \end{bmatrix}$$

Si on considère la somme par colonne on peut calculer la contribution d'un individu à la variance d' une composante.

Si on considère la somme par rangée on peut calculer la qualité de la representation d'un individu par une (ou plusieurs) composantes comme une partie de la représentation par toutes les composantes.

- contribution d'individu i à la variance de la composante j. Puisque il y a n individus et p composantes, c'est une matrice de dimensions $n \times p$ (`ind$contrib` en pourcentage (!))

$$\frac{y_{ij}^2}{y_{1j}^2 + \cdots + y_{nj}^2}$$

  notez que

$$\frac{1}{n}(y_{1j}^2 + \cdots + y_{nj}^2) = \lambda_j$$

- pour juger la qualité de la representation d'individu dans la composante j. Ce sera une matrice $n \times p$, parce que le nombre des individus est n et le nombre de composantes est p (`ind$cos2`):

$$\frac{y_{ij}^2}{y_{i1}^2 + \cdots + y_{ip}^2}$$

- pour juger la qualité de la representation d'individu dans les premières q composantes. (regarder `ind$cos2` et additionner les q premières colonnes de la rangée correspondante à l'individu d'interêt)

$$\frac{y_{i1}^2 + \cdots + y_{iq}^2}{y_{i1}^2 + \cdots + y_{ip}^2}$$

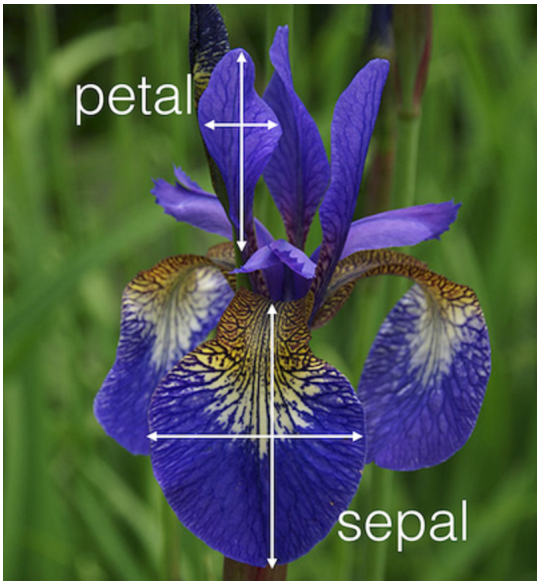
Jeu de données 'iris' - visualisation des données

Le jeu de données `iris` contient 150 fleurs et 5 variables. Le code suivant affiche les 6 premiers individus de cette base de données (fonction `head()`) :

```
head(iris)
```

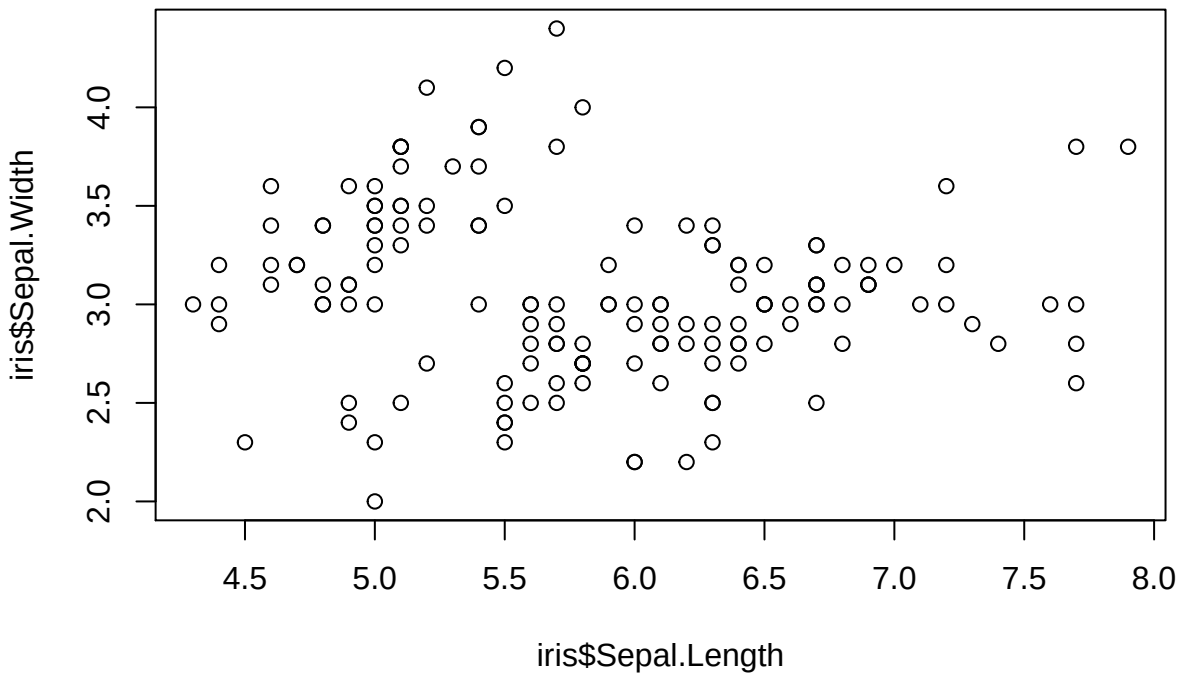
```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

Les 5 variables sont : longueur et largeur du sépale, longueur et largeur du pétale et espèce de fleur parmi 3 espèces possibles (setosa, versicolor et virginica). Les 4 premières variables sont quantitatives, la dernière variable est une variable qualitative.



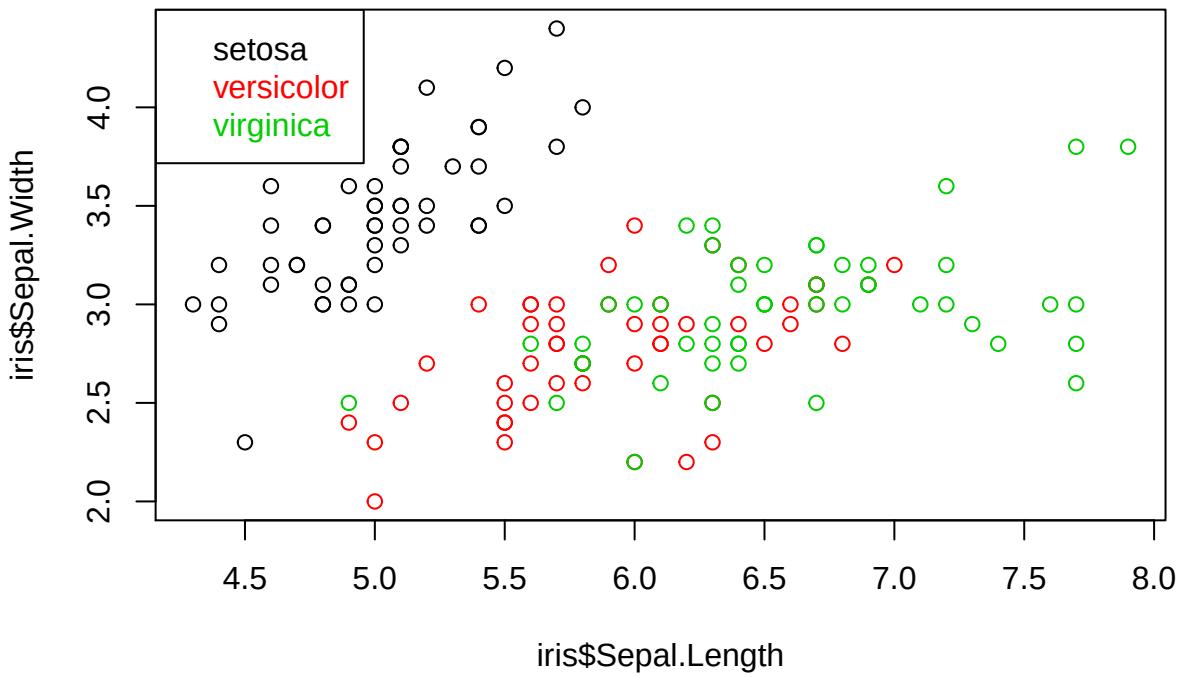
Supposons que l'on souhaite visualiser ce jeu de données deux variables quantitatives ? la fois. On utilise la fonction `plot()` :

```
plot(iris$Sepal.Length, iris$Sepal.Width)
```



Afin de colorier les points en fonction de l'espèce, faisant ainsi usage de la variable qualitative, on peut ajouter l'argument `col` dans la fonction `plot()` et utiliser la fonction `legend` pour ajouter un texte sur le graphique mentionnant quelle espèce correspond à quelle couleur :

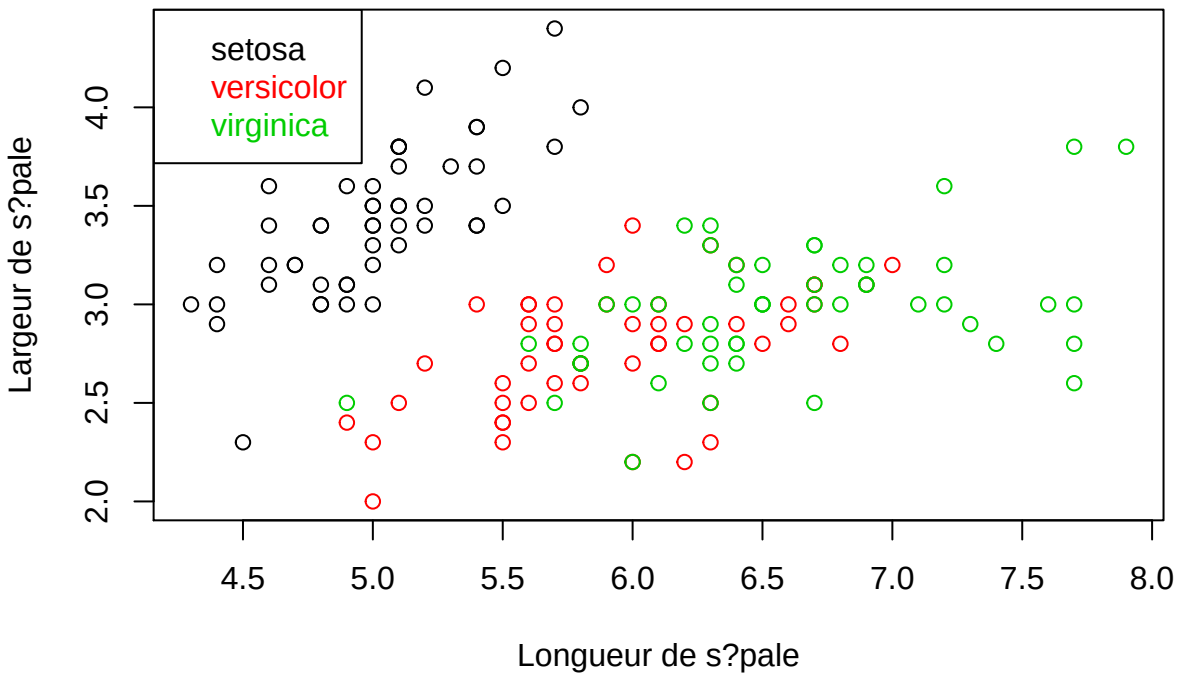
```
plot(iris$Sepal.Length, iris$Sepal.Width, col=iris$Species)
legend('topleft', legend = levels(iris$Species), text.col = 1:3)
```



Note : dans R, chaque couleur poss?de un numéro. Noir = 1, rouge = 2, vert = 3, bleu = 4, cyan = 5, jaune = 6, ...

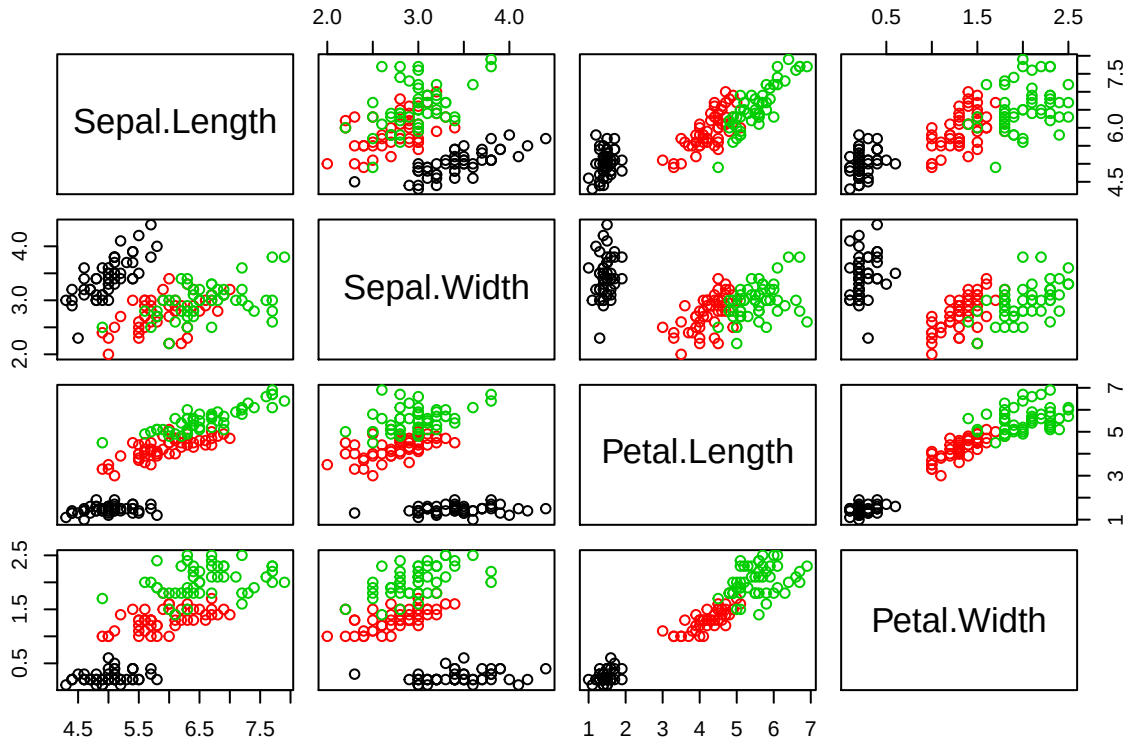
Pour changer le texte sur l'axe horizontal et vertical, on peut utiliser les arguments `xlab` et `ylab` :

```
plot(iris$Sepal.Length, iris$Sepal.Width, col=iris$Species,
     xlab="Longueur de s?pale", ylab="Largeur de s?pale")
legend('topleft', legend = levels(iris$Species), text.col = 1:3)
```



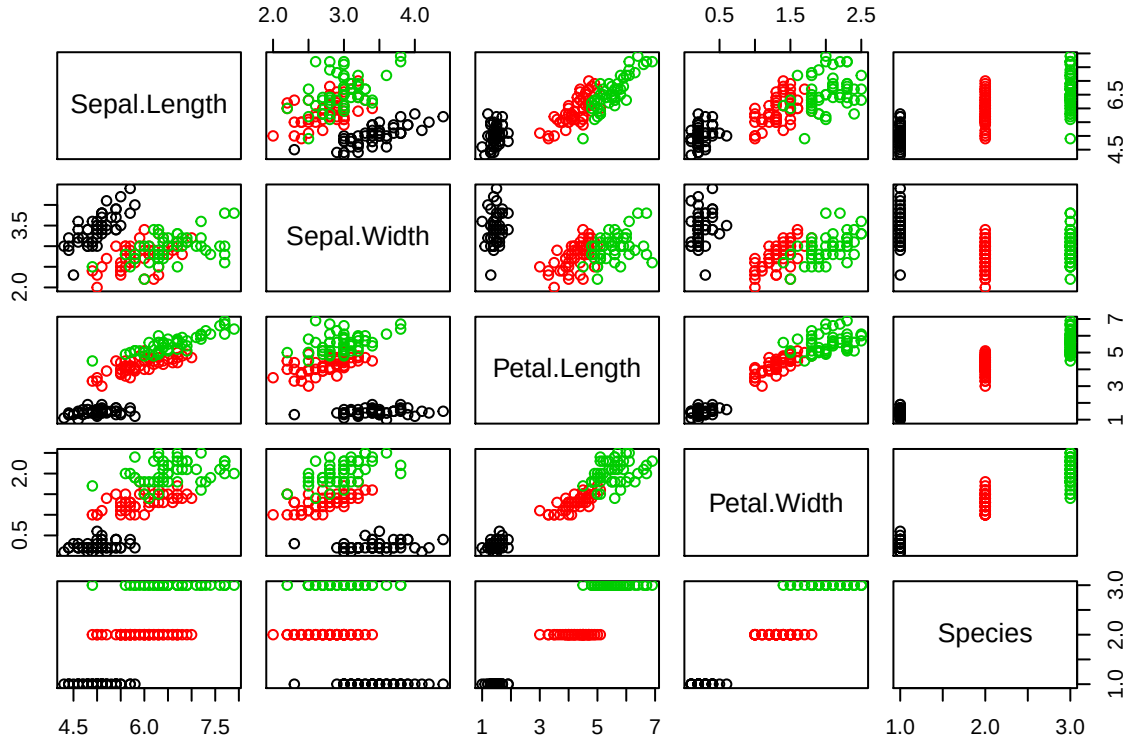
Il est possible de tracer tous les graphiques bivariés via la fonction `pairs()` :

```
pairs(iris[,-5], col=iris$Species)
```



La notation `[,-5]` permet d'ignorer la colonne 5 du jeu de données. Voici le résultat si on décide de ne pas ignorer cette colonne :

```
pairs(iris, col=iris$Species)
```



Il est aussi possible de tracer des graphiques trivariés à l'aide la fonction `plot3d()`. Cette fonction ne fait pas partie des fonctions de base de R. Pour l'utiliser, il faut installer le package `rgl` et ensuite charger le package pour avoir accès à `plot3d()`.

```
install.packages("rgl") # ? ne faire qu'une seule fois
library(rgl) # ? faire ? chaque fois que vous ouvrez R Studio
```

Note1 : il se peut que sur les ordinateurs de Louvain-la-Neuve, le package `rgl` soit déjà installé. Comment en être sûr ? Exécutez simplement `library(rgl)`. Si aucun message d'erreur n'apparaît, c'est bon, vous êtes prêt à utiliser la fonction `plot3d()`.

Note2 : pour les utilisateurs de macOS, simplement installer le package `rgl` et ensuite le charger via la fonction `library()` ne sera pas suffisant. Il faut en plus installer **XQuartz** depuis

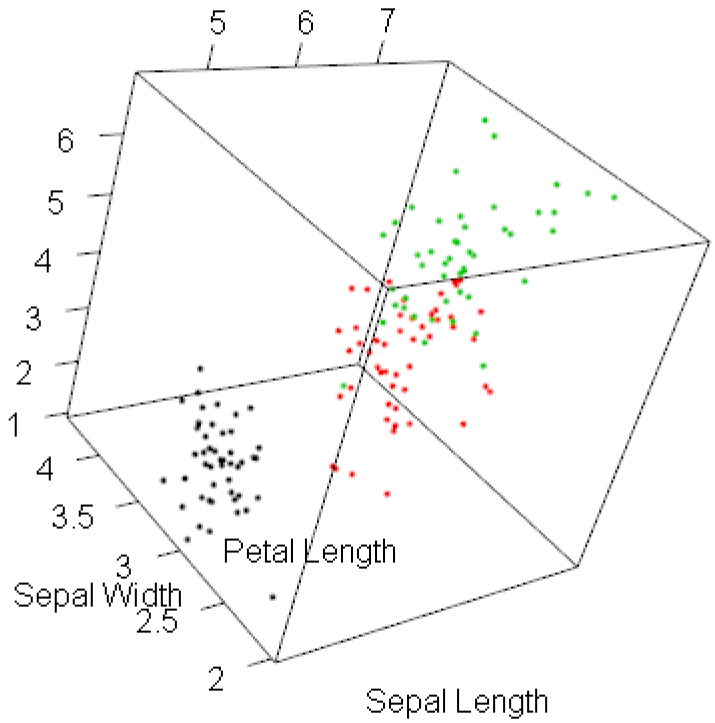
https://www.xquartz.org/

et redémarrer l'ordinateur ensuite, ce dernier point est très important.

Exemple de code utilisant la fonction `plot3d()` :

```
plot3d(iris$Sepal.Length, iris$Sepal.Width, iris$Petal.Length,
       col=as.numeric(iris$Species), xlab="Sepal Length",
       ylab="Sepal Width", zlab="Petal Length")
```

L'exécution de ce code ouvre une nouvelle fenêtre, externe à R Studio, dans laquelle vous pouvez manipulez le graphique en 3D à la souris.



À la différence de la fonction `plot()`, la fonction `plot3d()` éprouve quelques difficultés à identifier qu'il y a 3 espèces dans la colonne de données `iris$Species` et à assigner une couleur à chaque espèce via l'argument `col`. La fonction `as.numeric()` permet de donner un coup de pouce.

Pour visualiser notre jeu de données dans toute sa splendeur et sa richesse, ce que nous aimerions toutefois faire, ce n'est pas le visualiser 2 ou trois variables quantitatives à la fois. Ce qu'il nous faut, c'est un graphique quadridimensionnel, puisqu'il y a 4 variables quantitatives.

Problème : pratiquement parlant, aller au-delà de 3 dimensions spatiales est impossible (l'être humain est limité à trois dimensions spatiales).

Solution : réduire la dimensionnalité du jeu de données pour qu'il ne reste plus que 3, 2 ou 1 dimension (au choix).

Comment ? C'est ce que fait l'analyse en composantes principales (ACP).

Un premier exemple d'ACP

Pour réaliser une ACP (PCA en anglais) dans R, nous allons utiliser la fonction `PCA()`. Cette fonction n'est pas disponible par défaut : il faut d'abord installer puis charger le package `FactoMineR`.

```
install.packages("FactoMineR") # ? ne faire qu'une seule fois
```

```
library(FactoMineR) # ? ex?cuter chaque fois que vous ouvrez R Studio
```

Note : il se peut que sur les ordinateurs de Louvain-la-Neuve, le package `FactoMineR` soit déjà installé. Comment en être sûr ? Exécutez simplement `library(FactoMineR)`. Si aucun message d'erreur n'apparaît, c'est bon, vous êtes prêt à utiliser la fonction `PCA()`. Exemple :

```
res=PCA(iris, graph=FALSE, quali.sup=5)
```

L'argument `quali.sup=5` signale à la fonction `PCA()` que la 5ème colonne du jeu de données `iris` est une variable qualitative. Il faut travailler l'objet `res`, qui contient à présent tous les résultats de l'analyse en composantes principales des 4 variables quantitatives du jeu de données `iris`.

Par exemple, pour obtenir les valeurs propres de la matrice des corrélations depuis l'objet `res`, on fait :

```
round(res$eig,3)
```

```
       eigenvalue percentage of variance cumulative percentage of variance
comp 1      2.918                 72.962                             72.962
comp 2      0.914                 22.851                             95.813
comp 3      0.147                  3.669                             99.482
comp 4      0.021                  0.518                            100.000
```

Ces valeurs propres (ainsi que les vecteurs propres) peuvent aussi être obtenus en faisant

```
R=cor(iris[,-5])
round(eigen(R)$values,3) #les valeurs propres
```

```
[1] 2.918 0.914 0.147 0.021
```

```
round(eigen(R)$vectors,3) # les vecteurs propres
```

```
       [,1]   [,2]   [,3]   [,4]
[1,]  0.521 -0.377  0.720  0.261
[2,] -0.269 -0.923 -0.244 -0.124
[3,]  0.580 -0.024 -0.142 -0.801
[4,]  0.565 -0.067 -0.634  0.524
```

Les vecteurs propres suit à la exécution de la fonction `PCA()`:

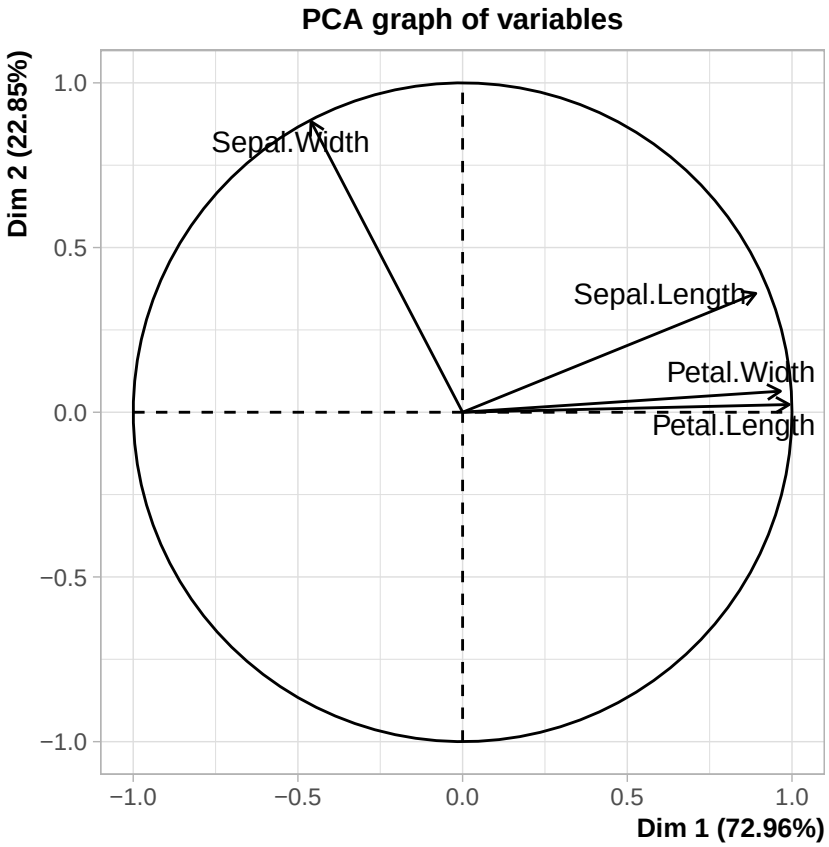
```
round(res$svd$V, 3)
```

```
       [,1]  [,2]   [,3]   [,4]
[1,]  0.521 0.377 -0.720 -0.261
[2,] -0.269 0.923  0.244  0.124
[3,]  0.580 0.024  0.142  0.801
[4,]  0.565 0.067  0.634 -0.524
```

Les vecteurs propres sont egaux à part des signes: les vecteurs propres ne sont déterminés qu'à la signe prés. Ici la fonction `PCA` a fait une different choix du signe.

Le cercle des corrélations pour la première composante et la seconde composante est obtenu via le code

```
plot.PCA(res, choix = "var", axes=c(1,2))
```



On peut faire quelques observations sur la base du cercle des corrélations:

- le premières deux composantes contiennent 95.81 de la variation totale, ce qui permettra à réduire les dimensions à 2.
- la longueur de toutes les fléches est proche à 1. Puisque la longueur est la somme des correlations de la variable avec les deux composantes, ici $\mathbf{y}^1$ et $\mathbf{y}^2$:

$$cor^2(\mathbf{x}^j, \mathbf{y}^1) + cor^2(\mathbf{x}^j, \mathbf{y}^2)$$

  on peut déduire que les variables sont fortement correlées avec les deux premières composantes, ce qui signifie que les composantes contiennent beaucoup de la variance de chaque variable.
- `Petal.Width` et `Petal.Length` sont fortement et positivement corrélées, puisque les flèches ont la même direction et elles sont très proches les unes des autres.

```
cor(iris$Petal.Width, iris$Petal.Length)
```

```
[1] 0.9628654
```

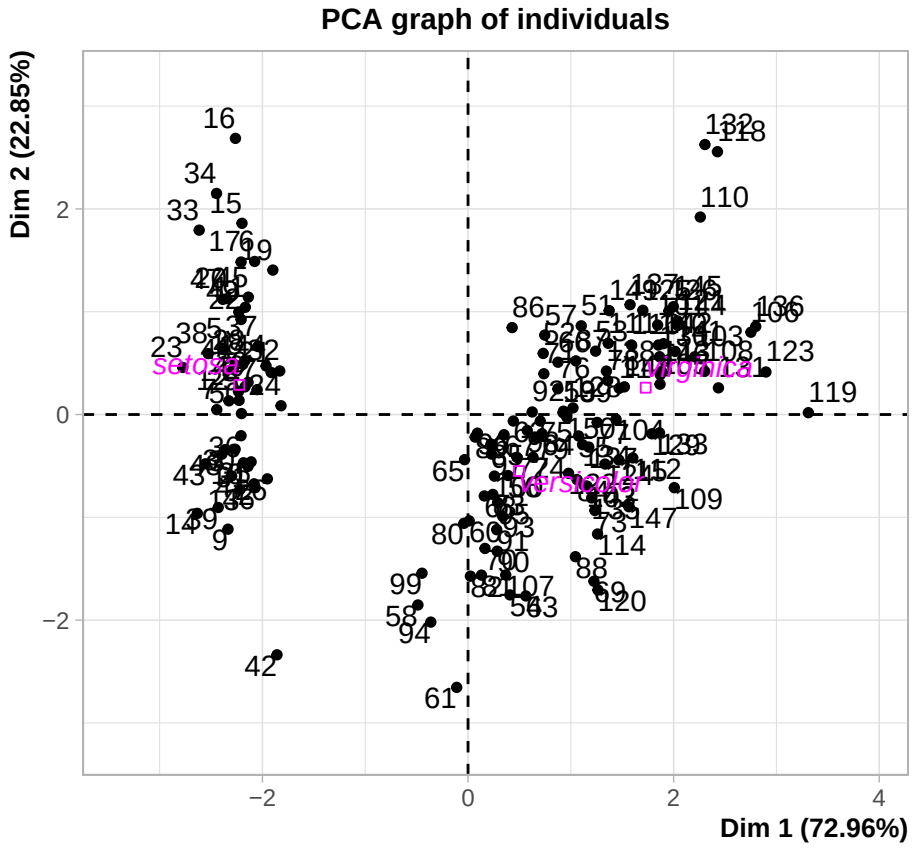
- `Sepal.Width` et `Sepal.Length` ont des fléches que sont presque orthogonales, ce qui signifie que les deux variables ont une corrélation faible.

```
cor(iris$Sepal.Width, iris$Sepal.Length)
```

```
[1] -0.1175698
```

La carte des individus pour les deux premières composantes s'obtient en faisant

```
plot.PCA(res, choix = "ind", axes=c(1,2))
```



Le jeu de données `iris` comporte des noms pour chaque colonne et des noms pour chaque ligne.

```
colnames(iris)
```

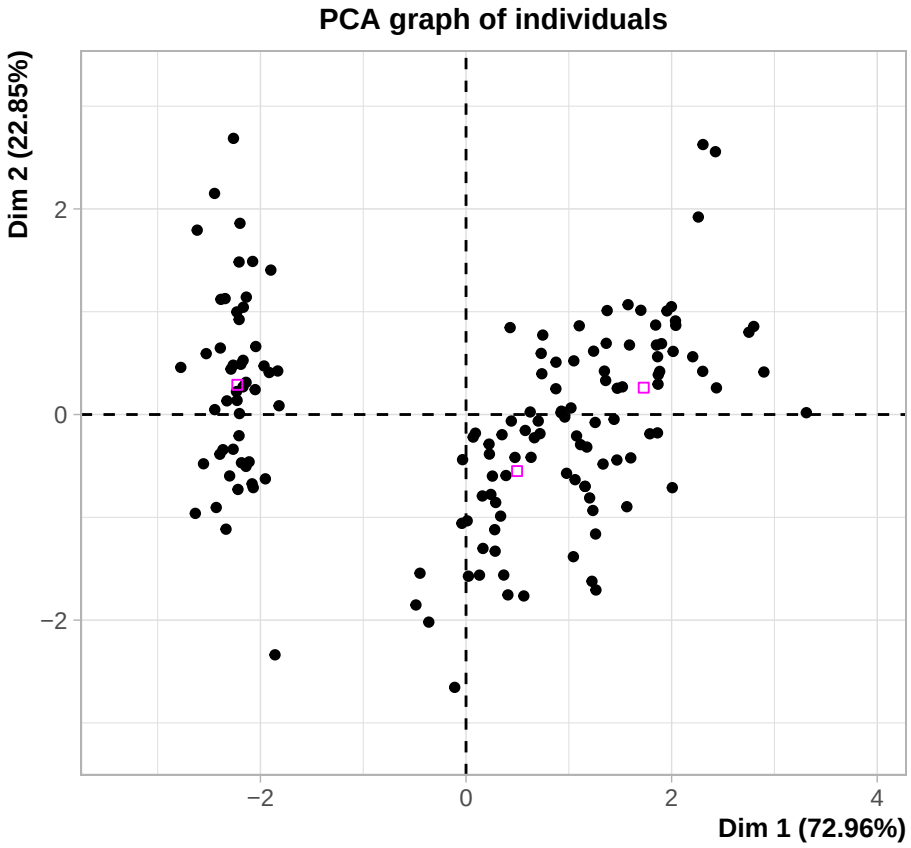
```
[1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```
rownames(iris)
```

```
  [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"  "12"
 [13] "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"  "23"  "24"
 [25] "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"  "34"  "35"  "36"
 [37] "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"  "45"  "46"  "47"  "48"
 [49] "49"  "50"  "51"  "52"  "53"  "54"  "55"  "56"  "57"  "58"  "59"  "60"
 [61] "61"  "62"  "63"  "64"  "65"  "66"  "67"  "68"  "69"  "70"  "71"  "72"
 [73] "73"  "74"  "75"  "76"  "77"  "78"  "79"  "80"  "81"  "82"  "83"  "84"
 [85] "85"  "86"  "87"  "88"  "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"
 [97] "97"  "98"  "99"  "100" "101" "102" "103" "104" "105" "106" "107" "108"
[109] "109" "110" "111" "112" "113" "114" "115" "116" "117" "118" "119" "120"
[121] "121" "122" "123" "124" "125" "126" "127" "128" "129" "130" "131" "132"
[133] "133" "134" "135" "136" "137" "138" "139" "140" "141" "142" "143" "144"
[145] "145" "146" "147" "148" "149" "150"
```

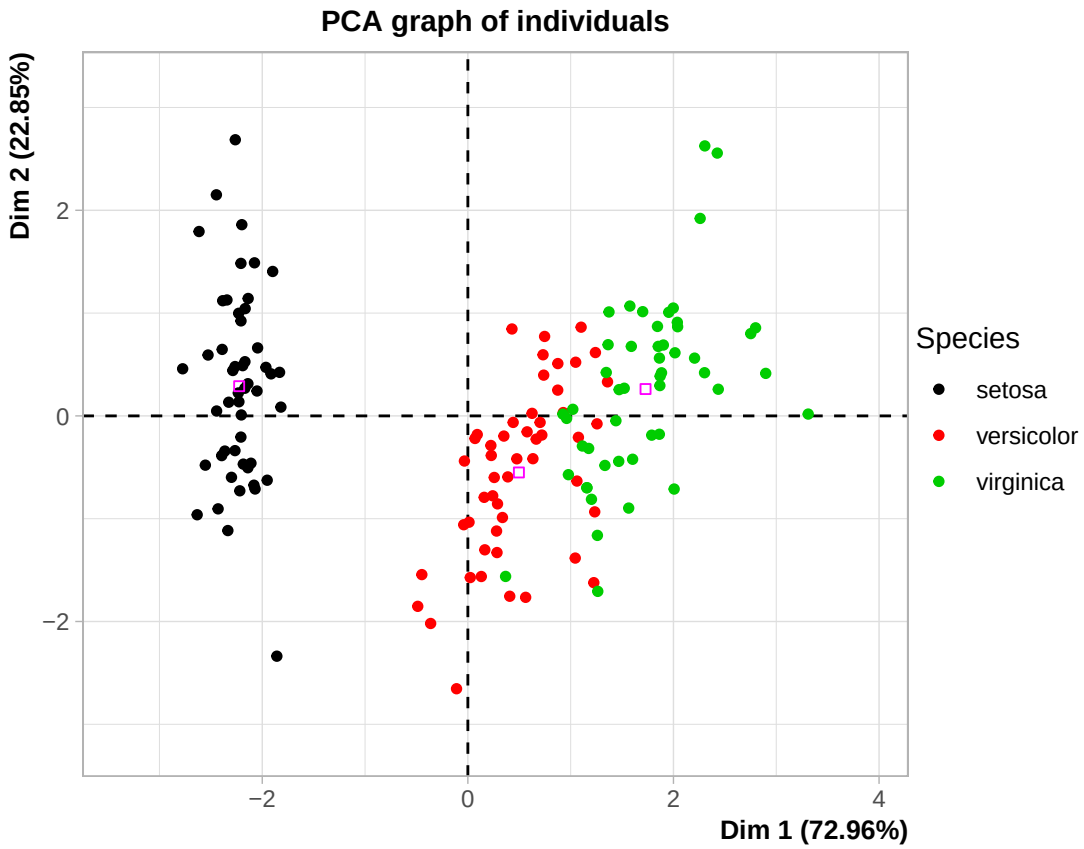
Pour éviter que les noms des lignes, juste des numéros dans le cas du jeu de données `iris`, ne soient utilisés sur la carte des individus, il faut utiliser l'argument `label` :

```
plot.PCA(res, choix = "ind", axes=c(1,2), label="none")
```



Si l'on souhaite colorier les individus par espèce, on utilise l'argument `habillage=5`, indiquant que la 5ème colonne du jeu de données qui a été fourni à la fonction `PCA` plus tôt contient la variable qualitative à utiliser :

```
plot.PCA(res, choix = "ind", axes=c(1,2), label="none", habillage=5)
```



Le graphique ci-dessus constitue le résultat de la réduction de dimensionnalité de notre jeu de données (de 4 ? 2). La première composante principale capture 72.96% de la variabilité du jeu de données original. La seconde composante en capture 22.85% supplémentaire. Les deux premières composantes principales capturent donc environ 96% de la variabilité originale. Le passage de 4 dimensions à 2 dimensions est dès lors tout à fait justifié (ce n'est pas toujours le cas, parfois les deux ou trois premières composantes capturent trop peu de variabilité et la réduction de dimensionnalité via l'ACP est considérée un échec).

Note : le graphique ci-dessus ne se suffit pas à lui-même : il ne peut s'interpréter qu'en regard du cercle des corrélations tracé plus tôt.

Ci-après, d'autres résultats utiles.

Les contributions par variable (lignes) pour chaque composante (colonnes), en % :

```
round(res$var$contrib, 2)
```

```
             Dim.1 Dim.2 Dim.3 Dim.4
Sepal.Length 27.15 14.24 51.78  6.83
Sepal.Width   7.25 85.25  5.97  1.53
Petal.Length 33.69  0.06  2.02 64.23
Petal.Width  31.91  0.45 40.23 27.42
```

Les contributions par individu (pour les 6 premiers individus uniquement, via la fonction `head()`) :

```
round(head(res$ind$contrib), 3)
```

```
  Dim.1 Dim.2 Dim.3 Dim.4
1 1.172 0.168 0.074 0.019
2 0.989 0.331 0.250 0.341
3 1.277 0.085 0.009 0.026
4 1.208 0.260 0.038 0.140
5 1.305 0.305 0.001 0.042
6 0.984 1.617 0.003 0.001
```

Les coordonnées pour les individus dans les composantes principales (pour les 6 premiers individus) :

```
round(head(res$ind$coord), 3)
```

```
   Dim.1  Dim.2  Dim.3  Dim.4
1 -2.265  0.480 -0.128 -0.024
2 -2.081 -0.674 -0.235 -0.103
3 -2.364 -0.342  0.044 -0.028
4 -2.299 -0.597  0.091  0.066
5 -2.390  0.647  0.016  0.036
6 -2.076  1.489  0.027 -0.007
```

(Les deux premières colonnes ont été utilisées par la fonction `plot.PCA()` pour tracer la carte des individus).

Les coordonnées du sommet des différentes fléches dans le cercle des corrélations :

```
round(res$var$cor, 3)
```

```
              Dim.1 Dim.2  Dim.3  Dim.4
Sepal.Length  0.890 0.361 -0.276 -0.038
Sepal.Width  -0.460 0.883  0.094  0.018
Petal.Length  0.992 0.023  0.054  0.115
Petal.Width   0.965 0.064  0.243 -0.075
```

(Le carré de ces valeurs donne les cos2 des variables).

Les cos2 des variables (nous dit à quel point une variable est bien représentée dans la composante correspondante, échelle de 0 à 1) :

```
round(res$var$cos2, 3)
```

```
             Dim.1 Dim.2 Dim.3 Dim.4
Sepal.Length 0.792 0.130 0.076 0.001
Sepal.Width  0.212 0.779 0.009 0.000
Petal.Length 0.983 0.001 0.003 0.013
Petal.Width  0.931 0.004 0.059 0.006
```

Les cos2 des individus (nous dit à quel point un individu est bien représenté dans la composante correspondante, échelle de 0 à 1) :

```
round(head(res$ind$cos2), 3)
```

```
  Dim.1 Dim.2 Dim.3 Dim.4
```

```
1 0.954 0.043 0.003 0.000
2 0.893 0.094 0.011 0.002
3 0.979 0.020 0.000 0.000
4 0.935 0.063 0.001 0.001
5 0.932 0.068 0.000 0.000
6 0.660 0.340 0.000 0.000
```

Une analyse en composante principale, pour rappel, n'est pas parfaite : réduire la dimensionnalité de vos données a un coût. Ce coût est en général faible, mais pas toujours. Si un individu ou une variable est mal représenté, il faut s'abstenir d'interpréter la variable ou l'individu. Si le pourcentage de variabilité cumulé sur les deux (ou trois) premières composantes est trop faible, l'ACP, dans un but de visualisation de vos données, est considérée un échec.

Calcul manuel

Nous savons que

$$\boldsymbol{Y} = \boldsymbol{X}_{cs}\boldsymbol{A}$$

où $\boldsymbol{Y}$ correspond à `res$ind$coord`, $\boldsymbol{X}_{cs}$ est la matrice des données centrées et réduites, $\boldsymbol{A}$ est une matrice contenant tous les vecteurs propres de la matrice des corrélations de $\boldsymbol{X}$.

Nous allons commencer par définir la matrice $\boldsymbol{X}$ dans R.

(Pour faire apparaître le jeu de données dans la fenêtre supérieure à droit vous pouvez créer un objet appelé 'iris_data' et lui assigner le jeu de données `iris`:

```
  iris_data<- iris
```

```
X=as.matrix(iris[,1:4])
```

Pour centrer les données, il s'agit de soustraire à chaque colonne sa propre moyenne. L'objet $\bar{\boldsymbol{x}}$ est un vecteur de dimension (4×1) contenant la moyenne de chaque colonne de $\boldsymbol{X}$:

```
xbar=colMeans(X)
```

Le vecteur $\mathbf{1}_{150}$ est un vecteur de dimension (150×1), contenant 150 fois la valeur 1 :

```
rep(1, 150)
```

```
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[149] 1 1
```

Nous pouvons à présent centrer les données par calcul matriciel :

$$\boldsymbol{X}_c = \boldsymbol{X} - \mathbf{1}_{150}\bar{\boldsymbol{x}}'$$

```
    Xc=X-rep(1, 150)%*%t(xbar)
```

Pour la réduction des données, il nous faut diviser chaque colonne de $\boldsymbol{X}_c$ par son propre écart-type. Attention : en ACP, l'écart-type se calcule via l'estimateur biaisé de l'ecart-type (division par n au lieu de $n-1$).

Définissons la matrice $\boldsymbol{S}_\backslash$ comme une matrice diagonale contenant les écarts-types biaisés de chaque colonne.

$$\boldsymbol{S}_\backslash = \begin{pmatrix} s_{11} & 0 & 0 & 0 \\ 0 & s_{22} & 0 & 0 \\ 0 & 0 & s_{33} & 0 \\ 0 & 0 & 0 & s_{44} \end{pmatrix}$$

```
    et=sqrt(apply(X, 2, var)) # les écarts-types non biaisés
    et=et*sqrt(149/150) # les écarts-types biaisés
    Sdiag=diag(et)
```

La fonction `apply()` est une fonction un rien plus complexe à comprendre. Le chiffre `2` dans cette fonction signifie que l'on travaille sur les **colonnes** de l'objet `X`. Que faire avec ces colonnes ? On en calcule à chaque fois la variance via la fonction `var()`.

Les données centrées et réduites s'obtiennent via

$$\boldsymbol{X}_{cs} = \boldsymbol{X}_c \boldsymbol{S}_\backslash^{-1}$$

```
  Xcs=Xc %*% solve(Sdiag)
```

La matrice $\boldsymbol{A}$ s'obtient via le code

```
  A=eigen(cor(X))$vectors
```

Et finalement les composantes principales $\boldsymbol{Y}$ s'obtiennent via le code

```
    Y=Xcs%*%A
```

En affichant les 6 premières lignes de `res$ind$coord` et de `Y`, on peut constater obtenir (presque) la même chose :

```
    round(head(res$ind$coord), 3)
```

```
   Dim.1  Dim.2  Dim.3  Dim.4
1 -2.265  0.480 -0.128 -0.024
2 -2.081 -0.674 -0.235 -0.103
3 -2.364 -0.342  0.044 -0.028
4 -2.299 -0.597  0.091  0.066
5 -2.390  0.647  0.016  0.036
6 -2.076  1.489  0.027 -0.007
```

```
    round(Y[1:6,], 3)
```

```
       [,1]   [,2]   [,3]   [,4]
[1,] -2.265 -0.480  0.128  0.024
[2,] -2.081  0.674  0.235  0.103
[3,] -2.364  0.342 -0.044  0.028
[4,] -2.299  0.597 -0.091 -0.066
[5,] -2.390 -0.647 -0.016 -0.036
[6,] -2.076 -1.489 -0.027  0.007
```

Les différences observées sont liées aux vecteurs propres : pour une matrice de corrélation donnée, les vecteurs propres normés ne sont définis qu'au signe prés. Le choix de ce signe est arbitraire et la fonction `PCA()` n'a pas fait le même choix que la fonction `eigen()`.

Pour calculer les corrélations entre variables et composantes, c'est à dire reconstituer `res$var$cor`, il nous faut utiliser la formule

$$\text{cor}(\boldsymbol{x}^k, \boldsymbol{y}^j) = \sqrt{\lambda_j} a_{kj}$$

où $\boldsymbol{x}^k$ est la variable k du jeu de données original, $\boldsymbol{y}^j$ est la composante principale j, λ_j est la valeur propre j, a_{kj} est l'élément k du vecteur propre j.

Les valeurs propres s'obtiennent via la commande

```
    lambda=eigen(cor(X))$values
    round(lambda, 3)
```

```
[1] 2.918 0.914 0.147 0.021
```

Le calcul est

```
    round(sqrt(rep(1, 4)%*%t(lambda))*A,3)
```

```
       [,1]   [,2]   [,3]   [,4]
[1,]  0.890 -0.361  0.276  0.038
[2,] -0.460 -0.883 -0.094 -0.018
[3,]  0.992 -0.023 -0.054 -0.115
[4,]  0.965 -0.064 -0.243  0.075
```

où `rep(1, 4)` est le vecteur $\mathbf{1}_4$, de dimension (4×1), contenant 4 fois la valeur 1.

On peut constater la correspondance avec `res$var$cor` :

```
    round(res$var$cor, 3)
```

```
              Dim.1 Dim.2  Dim.3  Dim.4
Sepal.Length  0.890 0.361 -0.276 -0.038
Sepal.Width  -0.460 0.883  0.094  0.018
Petal.Length  0.992 0.023  0.054  0.115
Petal.Width   0.965 0.064  0.243 -0.075
```

À nouveau, les différences proviennent du fait que les vecteurs propres normés ne sont définis qu'au signe prés et que la fonction `ACP()` n'a pas fait le même choix que la fonction `eigen()`.

Second exemple d'ACP

Pour ce second exemple, nous allons importer le jeu de données **Eurojobs.csv**, disponible sur Moodle.

Code pour importer le jeu de données :

```
    Eurojobs=read.csv(file="Eurojobs.csv", sep=",", dec=".", header=TRUE)
    Eurojobs
```

```
           Country  Agr Min  Man  PS  Con   SI  Fin  SPS  TC
1          Belgium  3.3 0.9 27.6 0.9  8.2 19.1  6.2 26.6 7.2
2          Denmark  9.2 0.1 21.8 0.6  8.3 14.6  6.5 32.2 7.1
3           France 10.8 0.8 27.5 0.9  8.9 16.8  6.0 22.6 5.7
4       W. Germany  6.7 1.3 35.8 0.9  7.3 14.4  5.0 22.3 6.1
5          Ireland 23.2 1.0 20.7 1.3  7.5 16.8  2.8 20.8 6.1
6            Italy 15.9 0.6 27.6 0.5 10.0 18.1  1.6 20.1 5.7
7       Luxembourg  7.7 3.1 30.8 0.8  9.2 18.5  4.6 19.2 6.2
8      Netherlands  6.3 0.1 22.5 1.0  9.9 18.0  6.8 28.5 6.8
9  United Kingdom  2.7 1.4 30.2 1.4  6.9 16.9  5.7 28.3 6.4
10         Austria 12.7 1.1 30.2 1.4  9.0 16.8  4.9 16.8 7.0
11         Finland 13.0 0.4 25.9 1.3  7.4 14.7  5.5 24.3 7.6
12          Greece 41.4 0.6 17.6 0.6  8.1 11.5  2.4 11.0 6.7
13          Norway  9.0 0.5 22.4 0.8  8.6 16.9  4.7 27.6 9.4
14        Portugal 27.8 0.3 24.5 0.6  8.4 13.3  2.7 16.7 5.7
15           Spain 22.9 0.8 28.5 0.7 11.5  9.7  8.5 11.8 5.5
16          Sweden  6.1 0.4 25.9 0.8  7.2 14.4  6.0 32.4 6.8
17     Switzerland  7.7 0.2 37.8 0.8  9.5 17.5  5.3 15.4 5.7
18          Turkey 66.8 0.7  7.9 0.1  2.8  5.2  1.1 11.9 3.2
19        Bulgaria 23.6 1.9 32.3 0.6  7.9  8.0  0.7 18.2 6.7
20 Czechoslovakia 16.5 2.9 35.5 1.2  8.7  9.2  0.9 17.9 7.0
21      E. Germany  4.2 2.9 41.2 1.3  7.6 11.2  1.2 22.1 8.4
22         Hungary 21.7 3.1 29.6 1.9  8.2  9.4  0.9 17.2 8.0
23          Poland 31.1 2.5 25.7 0.9  8.4  7.5  0.9 16.1 6.9
24         Rumania 34.7 2.1 30.1 0.6  8.7  5.9  1.3 11.7 5.0
25            USSR 23.7 1.4 25.8 0.6  9.2  6.1  0.5 23.6 9.3
26      Yugoslavia 48.7 1.5 16.8 1.1  4.9  6.4 11.3  5.3 4.0
```

Cette base de données contient la répartition (en %) des travailleurs dans 9 secteurs d'activité pour 26 pays européens avant la chute du mur de Berlin (on distingue l'Allemagne de l'Ouest et l'Allemagne de l'Est dans ce jeu de données).

Source : Euromonitor (1979), European Marketing Data and Statistics.

Les variables dans ce jeu de données sont :

- Country - the name of the country
- Agr - % of workforce employed in agriculture
- Min - % in mining
- Man - % in manufacturing
- PS - % in power supplies
- Con - % in construction
- SI - % in service industries
- Fin - % in finance
- SPS - % in social and personal services
- TC - % in transportation and communication

Observons les noms des colonnes et des lignes de ce jeu de données :

```
    colnames(Eurojobs)
```

```
 [1] "Country" "Agr"     "Min"     "Man"     "PS"      "Con"     "SI"
 [8] "Fin"     "SPS"     "TC"
```

```
    rownames(Eurojobs)
```

```
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14" "15"
[16] "16" "17" "18" "19" "20" "21" "22" "23" "24" "25" "26"
```

Une première chose à faire est de remplacer les noms des lignes par les noms des pays.

```
    rownames(Eurojobs)=Eurojobs$Country
    head(Eurojobs)
```

```
             Country  Agr Min  Man  PS  Con   SI Fin  SPS  TC
Belgium      Belgium  3.3 0.9 27.6 0.9  8.2 19.1 6.2 26.6 7.2
Denmark      Denmark  9.2 0.1 21.8 0.6  8.3 14.6 6.5 32.2 7.1
France        France 10.8 0.8 27.5 0.9  8.9 16.8 6.0 22.6 5.7
W. Germany W. Germany  6.7 1.3 35.8 0.9  7.3 14.4 5.0 22.3 6.1
Ireland      Ireland 23.2 1.0 20.7 1.3  7.5 16.8 2.8 20.8 6.1
Italy          Italy 15.9 0.6 27.6 0.5 10.0 18.1 1.6 20.1 5.7
```

L'intérêt de cette manipulation est que nous savons de l'exemple précédent (jeu de données `iris`) que les noms des lignes sont utilisés par la fonction `PCA()` pour tracer la carte des individus.

Cette manipulation réalisée, nous n'avons plus besoin de la variable `Country` et allons donc la supprimer du jeu de données :

```
    Eurojobs=Eurojobs[,-1]
    head(Eurojobs)
```

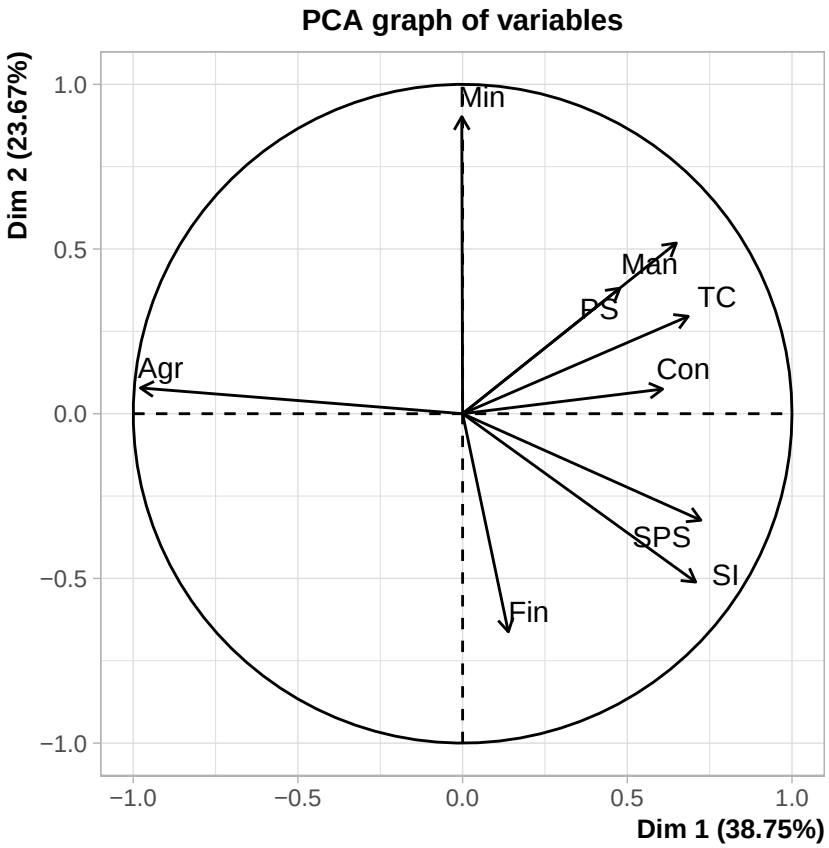
```
            Agr Min  Man  PS  Con   SI Fin  SPS  TC
Belgium     3.3 0.9 27.6 0.9  8.2 19.1 6.2 26.6 7.2
Denmark     9.2 0.1 21.8 0.6  8.3 14.6 6.5 32.2 7.1
France     10.8 0.8 27.5 0.9  8.9 16.8 6.0 22.6 5.7
W. Germany  6.7 1.3 35.8 0.9  7.3 14.4 5.0 22.3 6.1
Ireland    23.2 1.0 20.7 1.3  7.5 16.8 2.8 20.8 6.1
Italy      15.9 0.6 27.6 0.5 10.0 18.1 1.6 20.1 5.7
```

Code pour réaliser l'analyse en composantes principales :

```
res=PCA(Eurojobs, graph=FALSE)
```

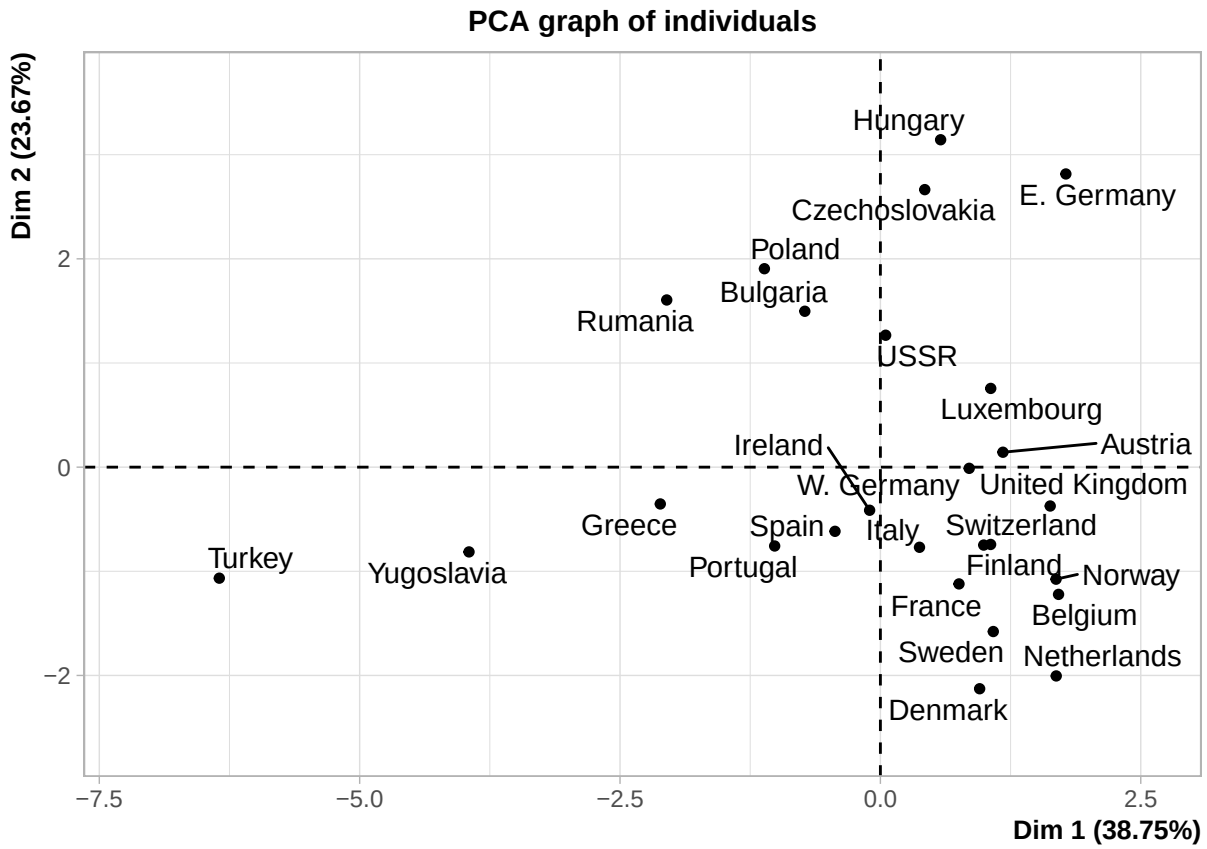
Cercle des corrélations :

```
plot.PCA(res, choix = "var", axes=c(1,2))
```



Carte des individus :

```
plot.PCA(res, choix = "ind", axes=c(1,2))
```



On note qu'avec 38.75%+23.67%=62.42%, le pourcentage de variabilité capturé par les deux premières composantes n'est pas aussi brillant que pour le jeu de données `iris`.

Interprétation : on constate que les pays d'Europe de l'Ouest s'agglomèrent dans le coin inférieur droit. En cause : l'économie dans ces pays était avant tout tertiaire. Les pays d'Europe de l'Est se situent plutôt vers le haut du graphique, caractérisés par une économie secondaire. Enfin sur la gauche, les pays dont l'économie est surtout basée sur l'agriculture (garder en tête que le monde a *considérablement* changé depuis, ces données ne sont plus du tout d'actualité).

Quelles variables sont mal représentées dans les deux premières composantes ?

```
round(res$var$cos2[,1:2], 3)
```

```
    Dim.1 Dim.2
Agr 0.957 0.006
Min 0.000 0.813
Man 0.421 0.269
PS  0.228 0.145
Con 0.369 0.006
SI  0.501 0.261
Fin 0.019 0.438
SPS 0.523 0.105
TC  0.469 0.087
```

Réponse : les variables `PS` et `Con`. Peut-être faudrait-il s'abstenir d'interpréter ces variables sur le cercle des corrélations...

Quels individus sont mal représentés dans les deux premières composantes ?

```
round(res$ind$cos2[,1:2],3)
```

```
               Dim.1 Dim.2
Belgium        0.593 0.303
Denmark        0.119 0.595
France         0.225 0.496
W. Germany     0.238 0.000
Ireland        0.003 0.051
Italy          0.026 0.108
Luxembourg     0.165 0.084
Netherlands    0.367 0.518
United Kingdom 0.410 0.021
Austria        0.379 0.006
Finland        0.259 0.147
Greece         0.600 0.017
Norway         0.348 0.142
Portugal       0.339 0.189
Spain          0.019 0.037
Sweden         0.193 0.409
Switzerland    0.157 0.077
Turkey         0.890 0.025
Bulgaria       0.114 0.486
Czechoslovakia 0.024 0.950
E. Germany     0.252 0.629
Hungary        0.023 0.687
Poland         0.224 0.654
Rumania        0.462 0.283
USSR           0.000 0.168
Yugoslavia     0.559 0.024
```

Réponse : W. Germany, Ireland, Italy, Luxembourg, Spain, Switzerland, USSR. Peut-être faudrait-il s'abstenir d'interpréter ces pays sur la carte des individus...

Pour finir, le lecteur attentif aura remarqué que la sortie qu'on obtient, par exemple, via `res$var$cos2` ne comporte que 5 composantes alors qu'il y en a 9 au total pour le jeu de données Eurojobs. Comment visualiser les valeurs pour les 4 composantes masquées ? Il faut relancer l'ACP avec l'argument `ncp=9` :

```
res=PCA(Eurojobs, graph=FALSE, ncp=9)
round(res$var$cos2, 3)
```

```
    Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6 Dim.7 Dim.8 Dim.9
Agr 0.957 0.006 0.003 0.001 0.025 0.009 0.000 0.000     0
Min 0.000 0.813 0.044 0.004 0.015 0.004 0.119 0.001     0
Man 0.421 0.269 0.025 0.119 0.081 0.032 0.052 0.002     0
PS  0.228 0.145 0.346 0.154 0.047 0.049 0.015 0.016     0
Con 0.369 0.006 0.026 0.444 0.121 0.007 0.011 0.017     0
```

```
SI  0.501 0.261 0.015 0.003 0.044 0.145 0.012 0.021     0
Fin 0.019 0.438 0.379 0.003 0.042 0.106 0.008 0.004     0
SPS 0.523 0.105 0.107 0.169 0.026 0.027 0.008 0.035     0
TC  0.469 0.087 0.155 0.098 0.143 0.006 0.001 0.041     0
```

En général toutefois, il n'est pas nécessaire de visualiser ce qui se passe au-delé des deux premières composantes.

Exercices

rangs.csv

La base de données **rangs.csv**, disponible sur la page Moodle du cours, contient plusieurs variables élaborées afin de mesurer la qualité scientifique de 50 universités. Les variables quantitatives d'intérêt sont

- **Alumni** : score basé sur le nombre d'alumnis ayant obtenu un prix Nobel ou une médaille Field ;
- **HiCi** : score basé sur le nombre de membres du personnel académique repris dans la liste des *highly cited researchers Sciences* entre 2002 et 2006 ;
- **Award** : score basé sur le nombre de membres du personnel académique ayant obtenu un prix Nobel ou une médaille Field ;
- **NS** : score basé sur le nombre d'articles publiés dans *Nature* et *Sciences* entre 2002 et 2006 ;
- **SCI** : score basé sur le nombre d'articles indexés dans *Science Citation Index-expanded* et dans *Social Science Citation Index 2006* ;
- **Size** : moyenne pondérée des cinq mesures précédentes divisées par le nombre d'équivalents temps-plein du personnel académique de l'institution.

Code pour l'importation des données :

```
rangs=read.csv("rangs.csv", sep=";", dec=",", header=TRUE)
```

Il est demandé de réaliser une analyse en composantes principales à partir des six variables quantitatives ci-dessus et de répondre briévement aux questions suivantes :

- Combien de composantes principales doit-on conserver dans cette analyse ?
- Dans cette analyse, quelle proportion de la variabilité totale des données est expliquée par les trois premières composantes ?
- Dans la première composante, quelle variable est la moins bien représentée ?
- L'analyse a permis de passer d'un espace $\mathbb{R}^6$? un espace $\mathbb{R}^2$. Quelle proportion de l'information (de la variabilité originale) a été perdue à la suite de cette procédure?
- Quelle proportion de la variance initiale de la variable **Award** est expliquée par les deux premières composantes?
- À partir du cercle des corrélations, déterminer quelle variable est la moins bien représentée par les deux premières composantes.

Solution :

- Deux composantes seront conservées puisque qu'elles sont les seules à avoir des valeurs propres supérieures à 1 (tous les statisticiens ne sont pas d'accord sur cette régle mais beaucoup l'appliquent).

```
res=PCA(rangs[,7:12], graph=FALSE)
# les variables d'int?r?t sont dans les colonnes 7 ? 12
res$eig
```

```
       eigenvalue percentage of variance cumulative percentage of variance
comp 1  3.9385062              65.641770                           65.64177
comp 2  1.0880378              18.133963                           83.77573
comp 3  0.4715535               7.859225                           91.63496
comp 4  0.2568757               4.281262                           95.91622
comp 5  0.1277077               2.128462                           98.04468
comp 6  0.1173191               1.955319                          100.00000
```

- 91.63% (voir ci-dessus)
- La variable **SCI** : cette dernière est surtout capturée par la seconde composante (0.677).

```
res$var$cos2
```

```
                   Dim.1       Dim.2      Dim.3       Dim.4        Dim.5
Score.on.Alumni 0.6882257 0.007576957 0.27455321 0.003245044 2.293253e-03
Score.on.Award  0.7018519 0.194291704 0.01660582 0.029299164 1.767712e-05
Score.on.HiCi   0.7452637 0.081697555 0.06626527 0.062399118 3.748269e-02
Score.on.N.S    0.8768566 0.003874564 0.02546639 0.004911843 8.221555e-02
Score.on.SCI    0.2637713 0.677020105 0.01151112 0.025947645 3.718915e-05
Score.on.Size   0.6625370 0.123576898 0.07715168 0.131072927 5.661342e-03
```

- On a $100\ \% - 83.77\ \% \approx 16\ \%$.
- Il faut faire

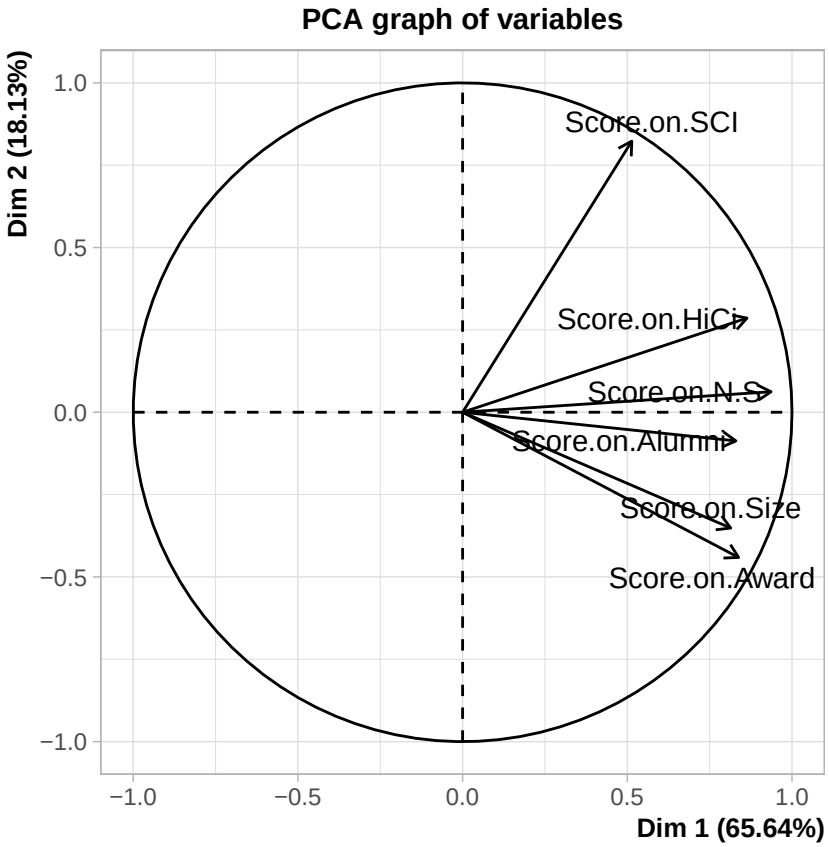
```
sum(res$var$cos2[2,c(1,2)])
```

```
[1] 0.8961436
```

C'est à dire 0.7018519+0.194291704.

- La variable **Alumni** car il s'agit de la variable dont la fléche est la plus courte.

```
plot.PCA(res, choix = "var", axes=c(1,2))
```



ACPcars.csv

La base de données **ACPcars.csv**, disponible sur Moodle, contient les résultats d'une étude faite pour différentes marques de voitures. Les variables quantitatives considérées sont * Economy : la consommation du véhicule * Service * Value : la non dépréciation de la valeur de la voiture * Price : le prix * Design * Sport : le caractère "sportif" du véhicule * Safety * Easy Handling : la facilité de prise en main pour le conducteur

Pour chacune des 24 marques considérées, un score est attribué à chacun des ces critères **(1 étant le meilleur et 6 étant le moins bon)**. Dans le cas du prix, un prix élevé est considéré moins bon et se voit attribuer un score élevé.

Réalisez une ACP sur cette base de données et interpréter la carte des individus et le cercle des corrélations.

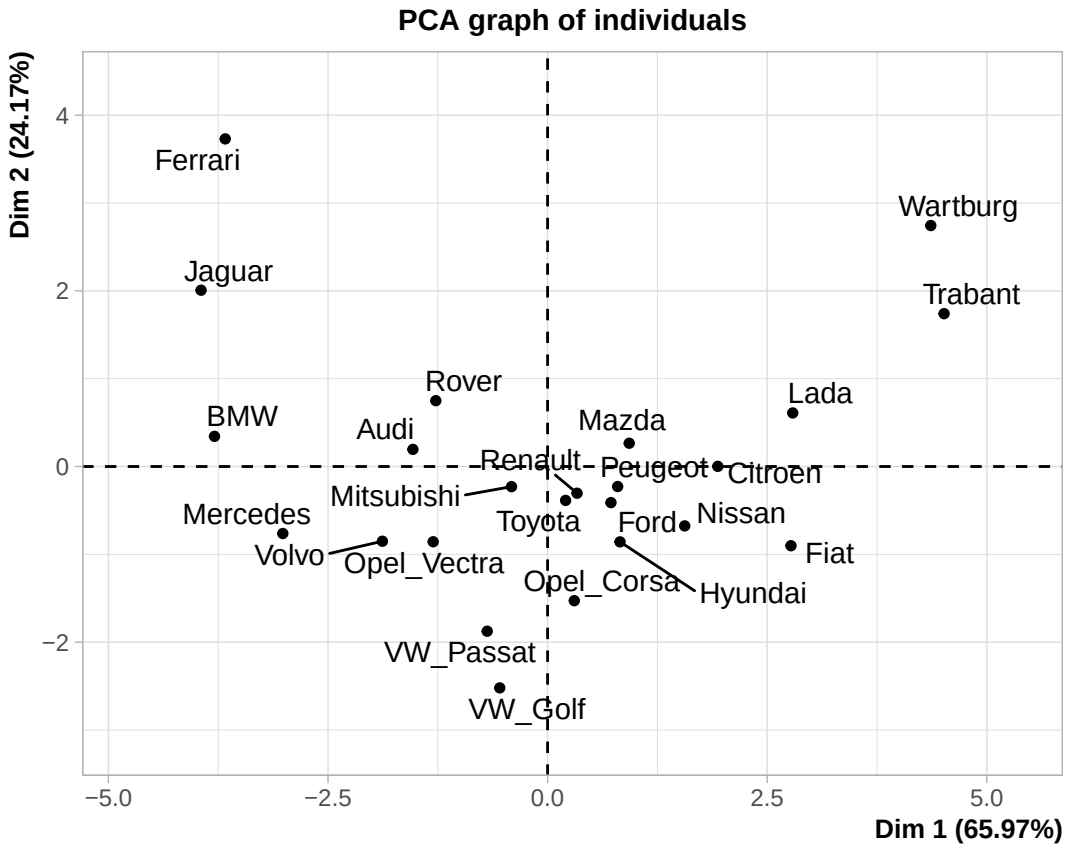
Solution :

On importe le jeu de données via le code

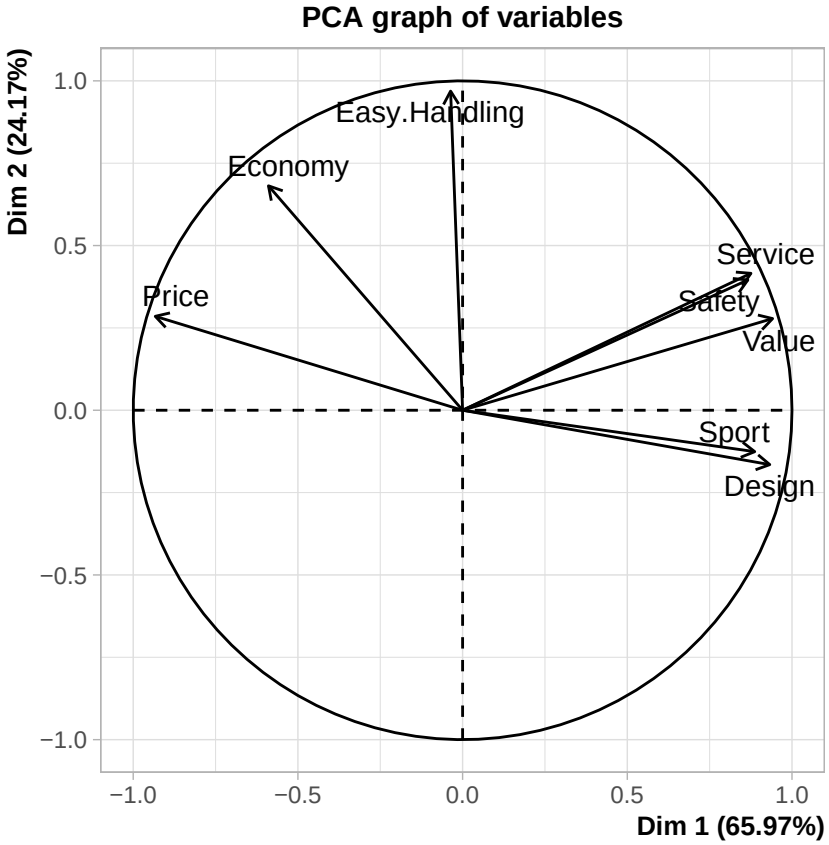
```
ACPcars=read.csv("ACPcars.csv", sep=";", dec=".", header=TRUE)
rownames(ACPcars)=ACPcars$Type
ACPcars=ACPcars[,-1]
```

Et on réalise l'ACP via les codes habituels :

```
res=PCA(ACPcars, graph=FALSE)
plot.PCA(res, choix = "ind", axes=c(1,2))
```




```
plot.PCA(res, choix = "var", axes=c(1,2))
```



On remarque que les deux premières composantes permettent d'expliquer une proportion égale à $0.6597 + 0.2417 = 0.9014$ de la variabilité initiale des données, ce qui est un plutôt bon résultat.

Toutes les variables sont bien expliquées par les deux premières composantes principales car les fléches touchent toutes le cercle des corrélations (ou presque).

La première composante permet de séparer les véhicules ayant un aspect peu sportif et un design peu audacieux (véhicules projetés sur la droite) et les véhicules avec un prix élevé (projetés sur la gauche).

Par exemple, BMW, Mercedes et Ferrari se retrouvent dans la moitié gauche et Lada et Fiat dans la moitié droite.

La seconde composante permet de séparer les véhicules peu gourmands en essence et simples d'utilisation (bas) et ceux plus gourmands et plus complexes d'utilisation (haut). Par exemple, Volkswagen et Opel se retrouvent dans la moitié inférieure et Ferrari, Wartburg et Jaguar se retrouvent dans la moitié supérieure.

europe.csv

Le jeu de données **europe.csv**, disponible sur la page Moodle du cours, contient des données économiques concernant des pays d'Europe de l'Est et certains pays d'Europe de l'Ouest pour l'année 2000.

Les variables quantitatives considérées sont

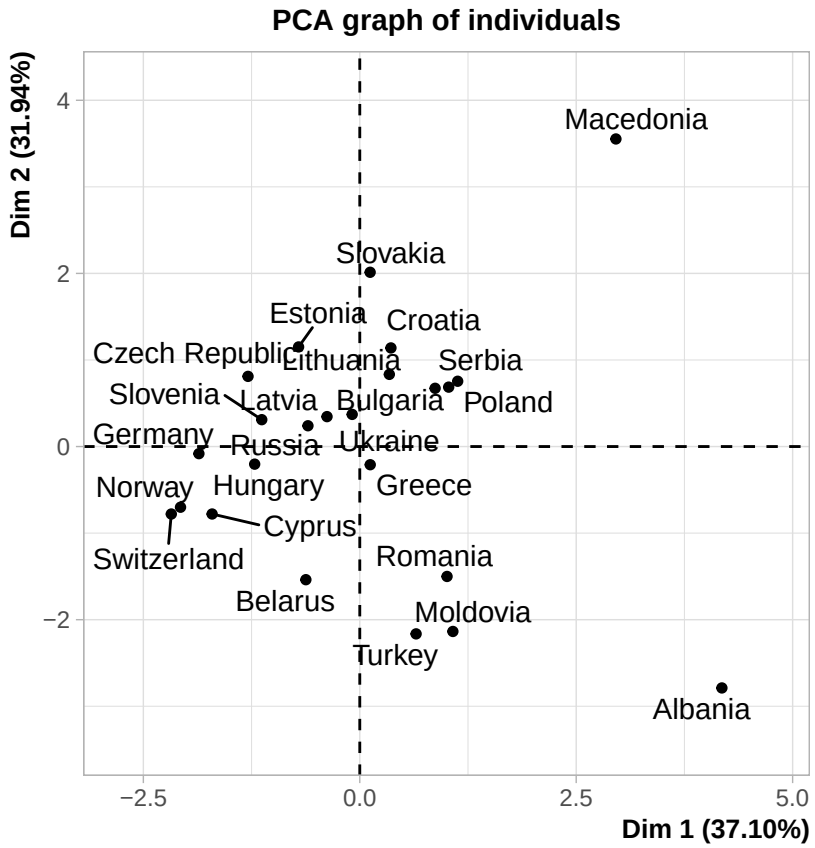
- **AGR** : la proportion des emplois en agriculture
- **IND** : la proportion des emplois en industrie
- **SER** : la proportion des emplois dans le domaine des services
- **WOMEN** : la proportion des emplois occupés par des femmes;
- **UNP_rate** : la proportion de gens sans emploi parmi la population totale (15 ans et plus)
- **yUNP_rate** : la proportion de gens sans emploi parmi les jeunes (15 à 24 ans).

Réalisez une ACP de ces données.

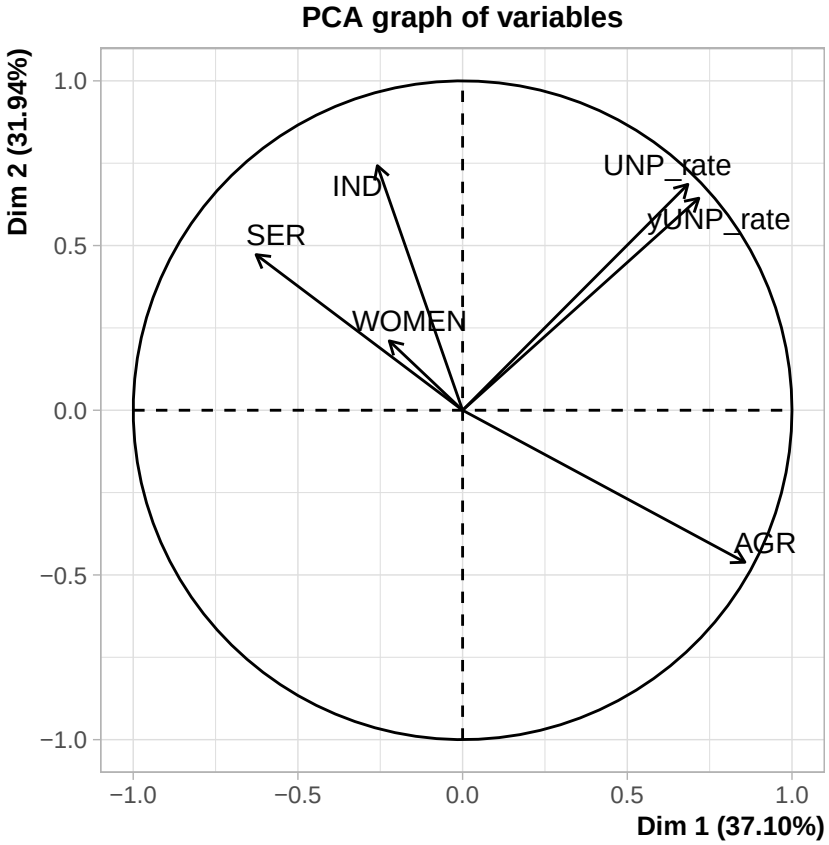
Solution :

```
europe=read.csv("europe.csv", sep=";", dec=".", header=TRUE)
rownames(europe)=europe$PAYS
europe=europe[,-1]
```

```
res=PCA(europe, graph=FALSE)
plot.PCA(res, choix = "ind", axes=c(1,2))
```




```
plot.PCA(res, choix = "var", axes=c(1,2))
```



À partir du cercle de corrélation, on remarque que la variable **AGR** est très fortement négativement corrélée avec les variables **IND** et **SER** puisque l'angle entre ces deux groupes de variables est pratiquement plat (la variable **WOMEN** étant mal représentée par les deux premières composantes, il vaut mieux s'abstenir de l'interpréter).

On remarque également une très forte corrélation positive entre les variables **UNP** et **yUNP**. Enfin, un angle presque droit entre d'un côté les variables **UNP** et **yUNP** et de l'autre les variables **AGR**, **IND** et **SER** indique une corrélation pratiquement nulle entre ces variables.

Sur la carte des individus, il est intéressant de remarquer que pour l'année 2000, la structure économique de Chypre, de la Hongrie ou encore de la Slovénie se rapproche le plus de la structure économique des pays de l'Europe de l'Ouest (Germany, Norway, Switzerland).

On note que Macedonia doit avoir très élévé taux de non-occupation générale et non-occupation parmi les jeunes aussi. C'est le cas aussi pour Sebia, Poland, Bulgaria, Croata et Lithuania. Albania, Romania, Moldovia, Turkey ont un taux d'occupation dans l'agriculture plus élévé par rapport aux autres pays.

On remarque également la proximité de la plupart des pays ex-communiste comme Slovenia, Estonia, Slovakia, Croatia, Lithuania, Latvia, Bulgaria, Serbia, Poland, dont l'économie est davantage tournée vers l'industrie.

Vecteurs propres et valeurs propres

Une analyse en composantes principales est réalisée à partir d'une base de données contenant 8 variables. La matrice des corrélations est

$$\boldsymbol{R} = \begin{pmatrix} 1.00 & -0.20 & -0.60 & -0.88 & 0.71 & 0.74 & 0.88 & 0.81 \\ -0.20 & 1.00 & 0.61 & 0.29 & 0.33 & 0.36 & -0.20 & 0.25 \\ -0.60 & 0.61 & 1.00 & 0.76 & -0.27 & -0.27 & -0.45 & -0.38 \\ -0.88 & 0.29 & 0.76 & 1.00 & -0.70 & -0.68 & -0.84 & -0.81 \\ 0.71 & 0.33 & -0.27 & -0.70 & 1.00 & 0.92 & 0.66 & 0.90 \\ 0.74 & 0.36 & -0.27 & -0.68 & 0.92 & 1.00 & 0.68 & 0.93 \\ 0.88 & -0.20 & -0.45 & -0.84 & 0.66 & 0.68 & 1.00 & 0.80 \\ 0.81 & 0.25 & -0.38 & -0.81 & 0.90 & 0.93 & 0.80 & 1.00 \end{pmatrix}$$

Une décomposition spectrale de cette matrice permet d'obtenir le vecteur de valeurs propres $\boldsymbol{\lambda} = (1.93, 0.13, 0.07, 0.02, 5.28, 0.41, 0.12, K)'$, où K est une valeur inconnue.

- Déterminer la valeur de K.
- Quelle proportion de la variabilité des données initiales sera expliquée par les trois premières composantes principales ?
- Voici deux vecteurs propres issus de la matrice des corrélations :

$$\boldsymbol{a}^i = (-0.117, 0.697, 0.491, 0.204, 0.284, 0.298, -0.091, 0.200)'$$

$$\boldsymbol{a}^j = (0.406, -0.016, -0.256, -0.406, 0.377, 0.381, 0.386, 0.410)'$$

Déterminer à quelles valeurs propres ces vecteurs correspondent.

- Définir explicitement l'équation de la première composante principale.

Solution :

- De façon générale, en travaillant avec la matrice des corrélations, on a $\sum_{j=1}^{p} \lambda_j = p$. Ainsi, $p = 8 = 1.93 + 0.13 + 0.07 + 0.02 + 5.28 + 0.41 + 0.12 + K$, et donc $K = 0.04$.
- La variabilité expliquée par les trois premières composantes principales est donnée respectivement par λ_1, λ_2 et λ_3, o? $\lambda_1 \geq \lambda_2 \geq \lambda_3$. La proportion de la variabilité initiale expliquée par les trois premières composantes est donc donnée par

$$\begin{aligned} \frac{\lambda_1 + \lambda_2 + \lambda_3}{\sum_{j=1}^{8} \lambda_j} &= \frac{5.28 + 1.93 + 0.41}{8} \\ &= 0.9525 \end{aligned}$$

- La valeur propre λ_i et le vecteur propre $\boldsymbol{a}^i$ sont tels que $\boldsymbol{R}\boldsymbol{a}^i = \lambda_i \boldsymbol{a}^i$.

Construisons la matrice $\boldsymbol{R}$ à l'aide de la fonction `matrix()`, ainsi que le vecteur $\boldsymbol{a}^i$ et le vecteur $\boldsymbol{\lambda}$:

```
R=matrix(c(
  1.00,-0.20,-0.60,-0.88, 0.71, 0.74, 0.88, 0.81,
  -0.20, 1.00, 0.61, 0.29, 0.33, 0.36,-0.20, 0.25,
  -0.60, 0.61, 1.00, 0.76,-0.27,-0.27,-0.45,-0.38,
  -0.88, 0.29, 0.76, 1.00,-0.70,-0.68,-0.84,-0.81,
  0.71, 0.33,-0.27,-0.70, 1.00, 0.92, 0.66, 0.90,
  0.74, 0.36,-0.27,-0.68, 0.92, 1.00, 0.68, 0.93,
  0.88,-0.20,-0.45,-0.84, 0.66, 0.68, 1.00, 0.80,
  0.81, 0.25,-0.38,-0.81, 0.90, 0.93, 0.80, 1.00
)
, nrow=8, ncol=8, byrow=TRUE)
ai=c(-0.117,0.697,0.491,0.204,0.284,0.298,-0.091,0.200)
lambda=eigen(R)$values
lambda
```

```
[1] 5.27689407 1.93444723 0.41273923 0.12628795 0.11846955 0.07118832 0.03703384
[8] 0.02293981
```

On calcule à présent

```
lambda[1]*ai
```

```
[1] -0.6173966  3.6779952  2.5909550  1.0764864  1.4986379  1.5725144 -0.4801974
[8]  1.0553788
```

```
lambda[2]*ai
```

```
[1] -0.2263303  1.3483097  0.9498136  0.3946272  0.5493830  0.5764653 -0.1760347
[8]  0.3868894
```

```
lambda[3]*ai
```

```
[1] -0.04829049  0.28767924  0.20265496  0.08419880  0.11721794  0.12299629
[7] -0.03755927  0.08254785
```

```
# etc
```

Et on identifie le résultat égal à `R%*%ai` :

```
R%*%ai
```

```
         [,1]
[1,] -0.22644
[2,]  1.34827
[3,]  0.94922
[4,]  0.39525
[5,]  0.54967
[6,]  0.57645
[7,] -0.17559
[8,]  0.38760
```

Ici, c'est `lambda[2]` le gagnant (moyennant les erreurs d'arrondi). `ai` est donc le second vecteur

propre et la valeur propre liée est

```
lambda[2]
```

```
[1] 1.934447
```

- $\boldsymbol{y^1} = \boldsymbol{X}_{cs}\boldsymbol{a}^1$ o? $\boldsymbol{a}^1$ est le vecteur propre associé à la plus grande valeur propre.

ACP sur une base de données bivariée

On souhaite réaliser une analyse en composantes principales à partir d'une base de données contenant 10 individus et deux variables. Le code ci-après crée cette base de données :

```
x1=c(-1.49,-0.07,0.57,1.22,2.44,0.24,0.25,1.12,-0.35,0.94)
x2=c(-0.61,-1.13,-0.73,0.91,1.04,1.24,0.16,0.61,0.19,-0.13)
X=data.frame(x1=x1, x2=x2)
X
```

```
      x1    x2
1  -1.49 -0.61
2  -0.07 -1.13
3   0.57 -0.73
4   1.22  0.91
5   2.44  1.04
6   0.24  1.24
7   0.25  0.16
8   1.12  0.61
9  -0.35  0.19
10  0.94 -0.13
```

La matrice des variances et covariances est donnée par

```
var(X)*9/10
```

```
         x1       x2
x1 1.000081 0.429695
x2 0.429695 0.583965
```

Note : si on fait `*9/10` ci-dessus, c'est parce que la fonction `var()` dans R calcule les variances via la formule $\frac{1}{n-1}\sum_{i=1}^{n}(x_{ij}-\bar{x}_j)^2$. Mais en ACP, on calcule toujours les variances via $\frac{1}{n}\sum_{i=1}^{n}(x_{ij}-\bar{x}_j)^2$. On peut v?rifier que $\left(\frac{1}{n-1}\sum_{i=1}^{n}(x_{ij}-\bar{x}_j)^2\right) \times (n-1)/n = \frac{1}{n}\sum_{i=1}^{n}(x_{ij}-\bar{x}_j)^2$.

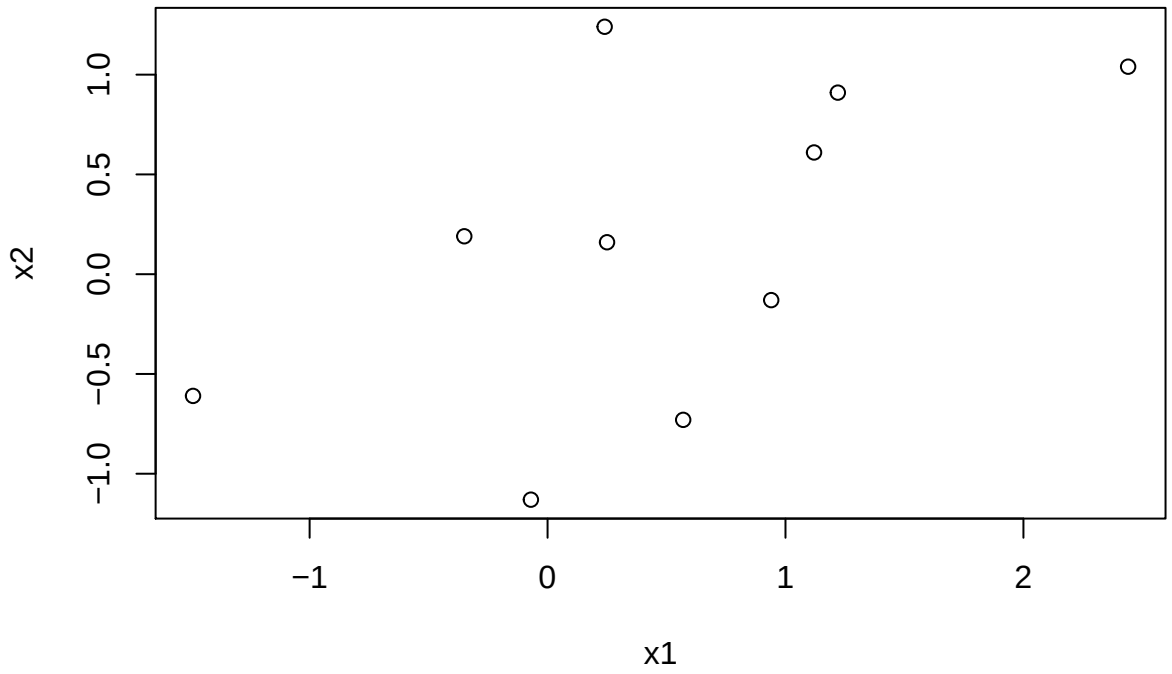
- Tracer la carte initiale des individus.
- Construire la matrice des corrélations à partir de la matrice des variances et covariances.
- Quelle proportion de la variabilité initiale des données est expliquée par la première composante principale ?
  - Si on considère qu'une seule composante sera conservée, quelle sera la coordonnée de l'individu 7 sur la nouvelle carte des individus?

– Si on considère que deux composantes sont conservées, quelle est la corrélation entre la variable $\boldsymbol{x}^1$ et la seconde composante principale?

Solution :

– Le code est

```
plot(X)
```



- Les éléments de la matrice des corrélations sont donnés par $r_{ij} = \frac{s_{ij}}{\sqrt{s_{ii}s_{jj}}}$. Dès lors, les éléments de cette matrice, calculés en utilisant la matrice des variances et des covariances, sont :

```
var(X)[1,1]/sqrt(var(X)[1,1]*var(X)[1,1])
```

```
[1] 1
```

```
var(X)[2,1]/sqrt(var(X)[2,2]*var(X)[1,1])
```

```
[1] 0.5622757
```

```
var(X)[1,2]/sqrt(var(X)[1,1]*var(X)[2,2])
```

```
[1] 0.5622757
```

```
var(X)[2,2]/sqrt(var(X)[2,2]*var(X)[2,2])
```

```
[1] 1
```

Ceci est confirmé en utilisant la fonction `cor()` sur `X` directement :

```
cor(X)
```

```
          x1        x2
x1 1.0000000 0.5622757
x2 0.5622757 1.0000000
```

- On a

$$\begin{aligned}\frac{\lambda_1}{\lambda_1+\lambda_2} &= \frac{1.56}{2}\\ &= 0.78\end{aligned}$$

Les valeurs propres étant :

```
eigen(cor(X))$values
```

```
[1] 1.5622757 0.4377243
```

- Les valeurs initiales des variables pour l'individu 7 sont 0.25 et 0.16. Il faut d'abord centrer et réduire ces valeurs. Pour ce faire, on calcule $\bar{x}_1 = 0.487$, $\bar{x}_2 = 0.155$, $s_{11} = \sqrt{1.000081}$ et $s_{22} = \sqrt{0.583965}$. On centre et réduit ensuite l'individu :

$$\frac{0.25 - 0.487}{\sqrt{1.000081}} = -0.237$$

et

$$\frac{0.16 - 0.155}{\sqrt{0.583965}} = 0.0065$$

La coordonnée de l'individu 7 dans la composante 1 est alors

$$(0.7071)(-0.237) + (0.7071)(0.0065) = -0.16$$

o? $(0.7071, 0.7071)'$ est le premier vecteur propre :

```
eigen(cor(X))$vectors[,1]
```

```
[1] 0.7071068 0.7071068
```

- On a

$$\text{cor}(\boldsymbol{x}^1, \boldsymbol{y}^2) = \sqrt{\lambda_2} a_{12}$$

```r
lambda2=eigen(cor(X))$values[2]
a12=eigen(cor(X))$vectors[1,2]
sqrt(lambda2)*a12
```

```
[1] -0.4678271
```