Fiche TD avec le logiciel : tdr601b

Initiation à l'analyse en composantes principales

A.B. Dufour & J.R. Lobry

Une première approche très intuitive et interactive de l'ACP. Centrage et réduction des données.

Table des matières

1 Introduction

Les méthodes d'analyse multivariées sont une branche à part entière des statistiques. Cette fiche contient une initiation à la plus ancienne de ces méthodes : l'analyse en composantes principales de données centrées réduites appelée ACP normée.

2 Les données

On étudie un jeu de données très simple extrait de `survey` de la librairie `MASS` [3]. Il s'agit d'une enquête sur la latéralité, réalisée auprès d'étudiants de l'université d'Adélaïde (Australie).

```
library(MASS)
data(survey)
names(survey)
```

```
 [1] "Sex"    "Wr.Hnd" "NW.Hnd" "W.Hnd"  "Fold"   "Pulse"  "Clap"   "Exer"   "Smoke"
[10] "Height" "M.I"    "Age"
```

```
head(survey)
```

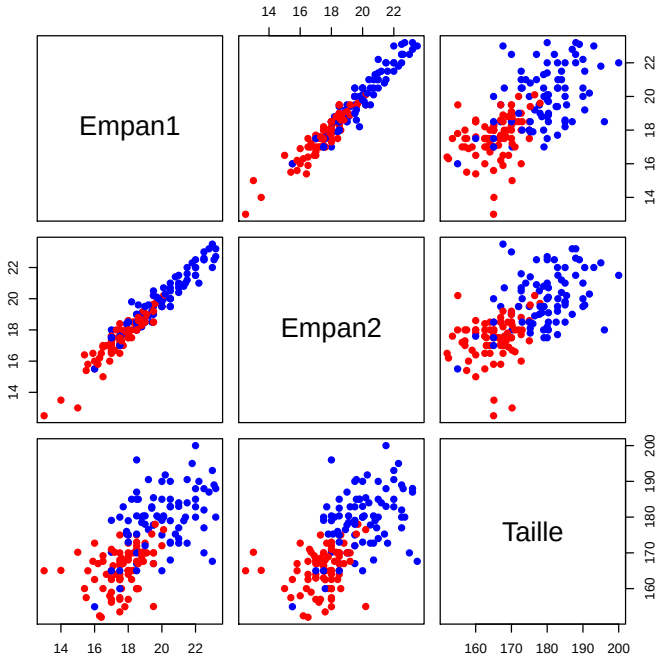
```
     Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke Height      M.I   Age
1 Female   18.5   18.0 Right  R on L    92    Left Some Never  173.0   Metric 18.25
2   Male   19.5   20.5  Left  R on L   104    Left None Regul  177.8 Imperial 17.58
3   Male   18.0   13.3 Right  L on R    87 Neither None Occas     NA     <NA> 16.92
4   Male   18.8   18.9 Right  R on L    NA Neither None Never  160.0   Metric 20.33
5   Male   20.0   20.0 Right Neither    35   Right Some Never  165.0   Metric 23.67
6 Female   18.0   17.7 Right  L on R    64   Right Some Never  172.7 Imperial 21.00
```

Le data frame `survey` contient des données manquantes. On choisit de ne conserver que les individus entièrement documentés :

```
survey.cc <- survey[complete.cases(survey), ]
```

On extrait trois variables quantitatives à analyser : l'empan de la main d'écriture dite main dominante, l'empan de la main non dominante, la taille des sujets. Toutes sont exprimées en centimètres. On conserve également l'information 'sexe' qui servira de variable illustrative dans les représentations graphiques.

```
mesures <- survey.cc[,c("Wr.Hnd", "NW.Hnd", "Height")]
names(mesures) <- c("Empan1", "Empan2", "Taille")
sexe <- survey.cc$Sex
plot(mesures, col = c("red","blue")[sexe], pch=19)
```

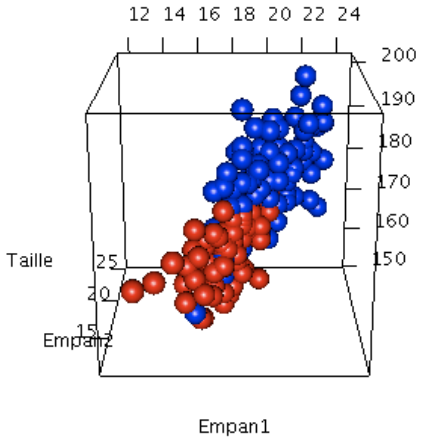


Noter que les variables morphométriques des garçons (en bleu) sont globalement plus grandes que celles des filles (en rouge). Ceci permettra de s'orienter plus facilement dans les représentations graphiques suivantes.

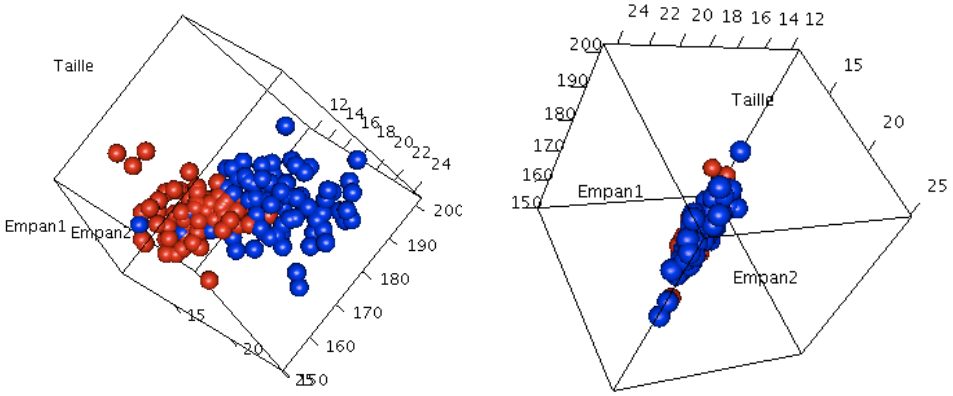
3 Visualisation des données en trois dimensions

Chaque individu est caractérisé par trois variables, soit par un point dans $\mathbb{R}^3$. La fonction `plot3d()` de la bibliothèque `rgl` [1] permet d'explorer facilement un nuage de points en 3 dimensions.

```
library(rgl)
plot3d(mesures, type = "s", col = c("red","blue")[sexe])
```



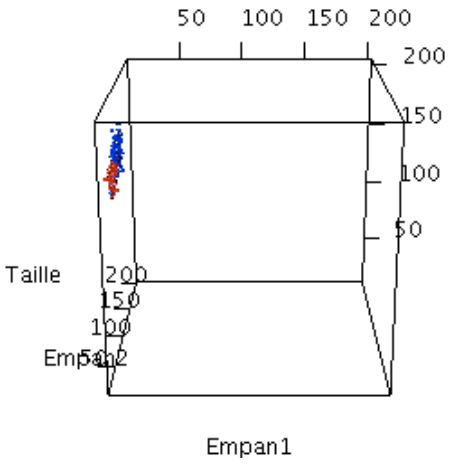
Faire tourner avec le curseur de la souris cette représentation pour en avoir différents points de vue :



4 Centrage et réduction

Les représentations graphiques précédentes sont trompeuses parce qu'on n'a pas utilisé la même échelle en x, y et z. On choisit d'imposer une échelle commune pour visualiser les données : le minimum et le maximum de l'ensemble des variables quantitatives.

```
lims <- c(min(mesures),max(mesures))
plot3d(mesures, type = "s", col = c("red","blue")[sexe], xlim = lims, ylim = lims,
zlim = lims)
```

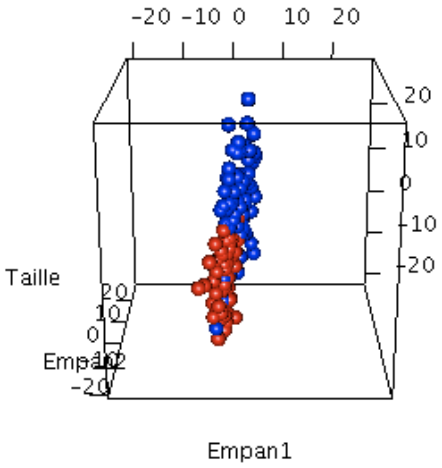


Voici donc à quoi ressemblent réellement les données. On voit tout de suite le problème, comme les tailles sont en moyenne beaucoup plus grandes que les empans, les points se trouvent complètement collés sur le plan des empans.

4.1 Centrage

L'opération de centrage consiste à enlever la moyenne à chaque variable. La fonction `scale()` permet d'effectuer directement cette opération :

```
mesures.c <- scale(mesures, center = TRUE, scale = FALSE)
lims <- c(min(mesures.c),max(mesures.c))
plot3d(mesures.c, type = "s", col = c("red","blue")[sexe], xlim = lims, ylim = lims,
       zlim = lims)
```



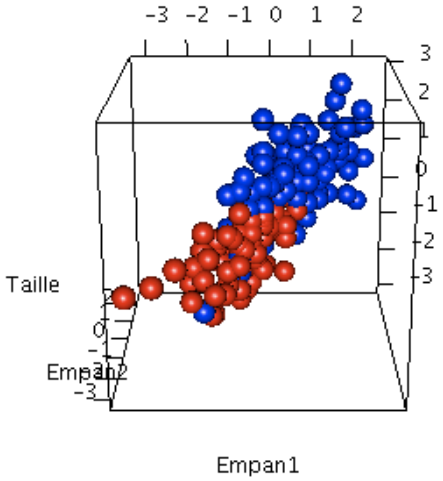
C'est mieux. Le nuage est maintenant centré autour de l'origine. Mais comme la variabilité est beaucoup plus forte pour la taille que pour les empans, notre nuage de points a l'aspect d'une galette complètement aplatie.

4.2 Centrage et réduction

Cette opération consiste à centrer les données puis diviser les valeurs par l'écart-type. La fonction `scale()` permet d'effectuer directement cette opération :

```
mesures.cr <- scale(mesures)


lims <- c(min(mesures.cr),max(mesures.cr))
plot3d(mesures.cr, type = "s", col = c("red","blue")[sexe], xlim = lims, ylim = lims,
       zlim = lims)
```

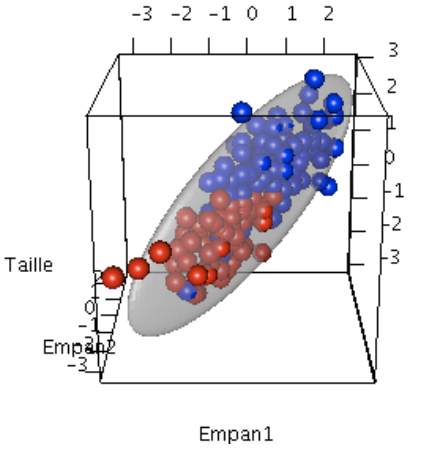


Quand on fait une ACP normée, on travaille avec les données après centrage et réduction. Il est donc important de bien comprendre à quoi correspondent ces opérations.

5 La forme générale du nuage

Dans l'exemple qu'on a choisi d'étudier, la forme générale du nuage de points est celui d'une dragée (le terme technique est un ellipsoïde).

```
plot3d(mesures.cr, type = "s", col = c("red","blue")[sexe], xlim = lims, ylim = lims,
       zlim = lims)
plot3d( ellipse3d(cor(mesures.cr)), col="grey", alpha=0.5, add = TRUE)
```

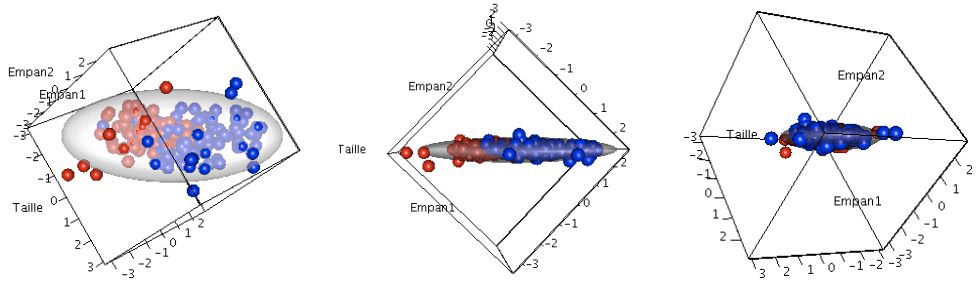


Une dragée est définie par ses trois axes :

1. le premier axe correspond au plus grand diamètre de l'ellipsoïde, la longueur de la dragée,
2. le deuxième axe correspond au diamètre moyen de l'ellipsoïde, la largeur de la dragée,

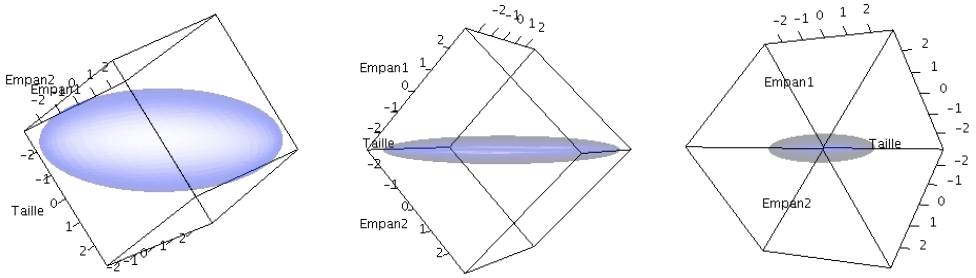
3. le troisième axe correspond au plus petit diamètre de l'ellipsoïde, l'épaisseur de la dragée.

Faire tourner le graphique précédent pour représenter le nuage de points dans le plan des axes (1,2), puis (1,3) puis (2,3) :



Faire le même exercice en ne conservant que la dragée :

```
plot3d( ellipse3d(cor(mesures.cr)), col="blue", alpha=0.25,
xlab = "Empan1", ylab = "Empan2", zlab = "Taille")
```



Noter que pour dessiner la dragée, on a eu besoin que de la matrice de variances-covariances :

```
cor(mesures.cr)
```

```
       Empan1 Empan2 Taille
Empan1 1.0000 0.9651 0.6510
Empan2 0.9651 1.0000 0.6296
Taille 0.6510 0.6296 1.0000
```

Si on avait à choisir entre les trois plans envisagés ci-dessus, on n'hésiterait pas à prendre le plan défini par les deux premiers axes de la dragée parce que c'est dans cette représentation que l'on a le moins de perte d'information par rapport au nuage de points dans $\mathbb{R}^3$: c'est dans cette projection que les points sont les plus étalés dans le plan. On dit aussi que l'on a conservé le maximum possible de l'inertie initiale du nuage de points. Ce faisant, on a en fait réalisé une ACP à la main.

6 ACP centrée-réduite dans ade4

6.1 Calculs

Utiliser la fonction `dudi.pca()` de la librairie `ade4` [2] pour exécuter une ACP centrée réduite :

```
library(ade4)
library(adegraphics)
acp <- dudi.pca(mesures, center=TRUE, scale=TRUE, scannf = FALSE, nf = 3)
names(acp)
```

```
 [1] "tab"  "cw"   "lw"   "eig"  "rank" "nf"   "c1"   "li"   "co"   "l1"   "call"
[12] "cent" "norm"
```

Remarque 1. On a fait apparaître dans les arguments de la fonction les termes `center=TRUE` et `scale=TRUE`, arguments naturellement par défaut, mais bien significatifs du centrage et de la réduction.

Remarque 2. On a utilisé les options `scann = FALSE` pour conserver automatiquement `nf = 3` facteurs. En général on ne procède pas ainsi : on commence par examiner le graphe des valeurs propres qui exprime quelle part de la variance totale est prise en compte par les axes successifs. Essayer avec :

```
acp <- dudi.pca(mesures)
```

Répondre 3 à la question `"Select the number of axes:"`. Dans la pratique, on ne conserve qu'un nombre réduit d'axes. Ici, on les a tous conservés pour des raisons pédagogiques.
L'objet de type `list` renvoyé par la fonction `dudi.pca()` est très riche. On va examiner toutes ses composantes une à une.

6.1.1 `tab`

Le data frame `tab` contient les données du tableau initial après centrage et réduction. Par rapport au résultat obtenu avec la fonction `scale()`, on note de petites différences :

```
head(acp$tab)
```

```
   Empan1  Empan2   Taille
1 -0.1580 -0.3711  0.05274
2  0.3646  0.8972  0.53613
5  0.6259  0.6435 -0.75291
6 -0.4193 -0.5233  0.02454
7 -0.5761 -0.5233  1.04772
8 -0.9419 -0.7263 -1.55856
```

```
head(mesures.cr)
```

```
   Empan1  Empan2   Taille
1 -0.1576 -0.3700  0.05258
2  0.3635  0.8945  0.53453
5  0.6240  0.6416 -0.75067
6 -0.4181 -0.5218  0.02447
7 -0.5744 -0.5218  1.04460
8 -0.9391 -0.7241 -1.55392
```

Cette petite différence est due à l'utilisation d'une variance en $\frac{1}{n}$ dans `dudi.pca()` contre une variance en $\frac{1}{n-1}$ dans `scale()`. Pour retrouver exactement le tableau utilisé dans `dudi.pca()` faire :

```
var.n <- function(x) sum((x-mean(x))^2)/length(x)
scale.n <- function(x) (x - mean(x))/sqrt(var.n(x))
head(apply(mesures, 2, scale.n))
```

```
   Empan1  Empan2   Taille
1 -0.1580 -0.3711  0.05274
2  0.3646  0.8972  0.53613
5  0.6259  0.6435 -0.75291
6 -0.4193 -0.5233  0.02454
7 -0.5761 -0.5233  1.04772
8 -0.9419 -0.7263 -1.55856
```

6.1.2 `cw`

Le vecteur `cw` donne le poids des colonnes (*column weight*), c'est-à-dire le poids des variables. Par défaut, chaque variable a un poids de 1.

```
 acp$cw
[1] 1 1 1
```

6.1.3 `lw`

Le vecteur `lw` donne le poids des lignes (*line weight*), c'est-à-dire le poids des individus. Par défaut, chaque individu a un poids de $\frac{1}{n}$.

```
 head(acp$lw)
[1] 0.005952 0.005952 0.005952 0.005952 0.005952 0.005952
 head(acp$lw)*nrow(mesures)
[1] 1 1 1 1 1 1
```

6.1.4 `eig`

Le vecteur `eig` donne les valeurs propres (*eigen values*) dans le plus petit des deux espaces diagonalisés.

```
 acp$eig
[1] 2.50872 0.45683 0.03445
 sum(acp$eig)
[1] 3
```

Les valeurs propres renseignent sur la part de l'inertie totale prise en compte par chaque axe :

```
 (pve <- 100*acp$eig/sum(acp$eig))
[1] 83.624 15.228  1.148
 cumsum(pve)
[1]  83.62  98.85 100.00
```

Dans l'exemple, le premier axe factoriel extrait 83.6 % de l'inertie totale, le deuxième axe factoriel 15.2 % de l'inertie totale. Le premier plan factoriel représente donc 98.9 % de l'inertie initiale. Ceci signifie que lorsqu'on projette le nuage de points initial dans $\mathbb{R}^3$ sur le plan défini par les deux premiers axes factoriels, on a perdu que très peu d'information.

6.1.5 rank

Cet entier donne le rang (*rank*) de la matrice diagonalisée, ici le nombre de variables indépendantes.

```
 acp$rank
```

```
[1] 3
```

```
 bismesures <- cbind(mesures,mesures)
 head(bismesures)
```

```
  Empan1 Empan2 Taille Empan1 Empan2 Taille
1   18.5   18.0  173.0   18.5   18.0  173.0
2   19.5   20.5  177.8   19.5   20.5  177.8
5   20.0   20.0  165.0   20.0   20.0  165.0
6   18.0   17.7  172.7   18.0   17.7  172.7
7   17.7   17.7  182.9   17.7   17.7  182.9
8   17.0   17.3  157.0   17.0   17.3  157.0
```

```
 colnames(bismesures) <- c("Empan11","Empan21","Taille1",
                           "Empan12","Empan22","Taille2")
 dudi.pca(bismesures,scann=F,n=3)$rank
```

```
[1] 3
```

6.1.6 nf

Cet entier donne le nombre de facteurs conservés dans l'analyse :

```
 acp$nf
```

```
[1] 3
```

6.1.7 c1

`c1` donne les coordonnées des variables (colonnes). Les vecteurs sont de norme unité :

```
 acp$c1
```

```
          CS1     CS2      CS3
Empan1 0.6085 -0.3421  0.71604
Empan2 0.6040 -0.3855 -0.69750
Taille 0.5147  0.8569 -0.02795
```

```
 sum(acp$cw * acp$c1$CS1^2)
```

```
[1] 1
```

6.1.8 l1

`l1` donne les coordonnées des individus (lignes). Les vecteurs sont de norme unité :

```
 head(acp$l1)
```

```
       RS1      RS2     RS3
1 -0.18511  0.35854  0.7772
2  0.65643 -0.01655 -2.0459
5  0.24122 -1.63843  0.1096
6 -0.35271  0.54186  0.3453
7 -0.08047  1.91846 -0.4136
8 -1.14527 -1.08502 -0.6699
```

```
 sum(acp$lw * acp$l1$RS1^2)
```

```
[1] 1
```

6.1.9 co

co donne les coordonnées des variables (colonnes). Les vecteurs sont normés à la racine carrée de la valeur propre correspondante :

```
 acp$co
```

```
        Comp1   Comp2     Comp3
Empan1 0.9638 -0.2312  0.132897
Empan2 0.9567 -0.2606 -0.129457
Taille 0.8152  0.5792 -0.005187
```

```
 sum(acp$cw * acp$co$Comp1^2)
```

```
[1] 2.509
```

Le lien entre les c1 et les co s'obtient par :

```
 acp$c1$CS1 * sqrt(acp$eig[1])
```

```
[1] 0.9638 0.9567 0.8152
```

```
 t(t(acp$c1) * sqrt(acp$eig))
```

```
          CS1     CS2       CS3
Empan1 0.9638 -0.2312  0.132897
Empan2 0.9567 -0.2606 -0.129457
Taille 0.8152  0.5792 -0.005187
```

6.1.10 li

li donne les coordonnées des individus (lignes). Les vecteurs sont normés à la racine carrée de la valeur propre correspondante :

```
 head(acp$li)
```

```
    Axis1    Axis2    Axis3
1 -0.2932  0.24234  0.14424
2  1.0397 -0.01118 -0.37973
5  0.3821 -1.10741  0.02033
6 -0.5586  0.36624  0.06409
7 -0.1275  1.29668 -0.07677
8 -1.8140 -0.73336 -0.12433
```

```
 sum(acp$lw * acp$li$Axis1^2)
```

```
[1] 2.509
```

```
 head(acp$l1$RS1 * sqrt(acp$eig[1]))
```

```
[1] -0.2932  1.0397  0.3821 -0.5586 -0.1275 -1.8140
```

```
 head(t(t(acp$l1) * sqrt(acp$eig)))
```

```
      RS1      RS2      RS3
1 -0.2932  0.24234  0.14424
2  1.0397 -0.01118 -0.37973
5  0.3821 -1.10741  0.02033
6 -0.5586  0.36624  0.06409
7 -0.1275  1.29668 -0.07677
8 -1.8140 -0.73336 -0.12433
```

6.1.11 call

Cet objet garde une trace de la façon dont ont été conduits les calculs lors de l'appel de la fonction dudi.pca() :

```
 acp$call
```

```
dudi.pca(df = mesures, center = TRUE, scale = TRUE, scannf = FALSE,
    nf = 3)
```

La fonction `eval()` permet de refaire les mêmes calculs :

```
 eval(acp$call)
Duality diagramm
class: pca dudi
$call: dudi.pca(df = mesures, center = TRUE, scale = TRUE, scannf = FALSE,
    nf = 3)
$nf: 3 axis-components saved
$rank: 3
eigen values: 2.509 0.4568 0.03445
  vector length mode    content
1 $cw    3      numeric column weights
2 $lw    168    numeric row weights
3 $eig   3      numeric eigen values

  data.frame nrow ncol content
1 $tab       168  3    modified array
2 $li        168  3    row coordinates
3 $l1        168  3    row normed scores
4 $co        3    3    column coordinates
5 $c1        3    3    column normed scores
other elements: cent norm
 identical(eval(acp$call), acp)
[1] TRUE
```

6.1.12 `cent`

Ce vecteur donne les moyennes (cent pour centrage) des variables analysées :

```
 acp$cent
Empan1 Empan2 Taille
 18.80  18.73 172.48
 colMeans(mesures)
Empan1 Empan2 Taille
 18.80  18.73 172.48
```

6.1.13 `norm`

Ce vecteur donne les écarts-types (sur $\sqrt{n}$) des variables analysées :

```
 acp$norm
Empan1 Empan2 Taille
 1.913  1.971  9.930
 sd.n <- function(x) sqrt(var.n(x))
 apply(mesures, 2, sd.n)
Empan1 Empan2 Taille
 1.913  1.971  9.930
```

6.2 Dé-réduction et dé-centrage

Si on comprend bien en quoi consiste l'opération de centrage et de réduction, on doit pouvoir être capable de faire l'opération inverse. À partir des objets `acp$tab`, `acp$cent` et `acp$norm` reconstituer les données de départ, placer le résultat dans l'objet `recon` :

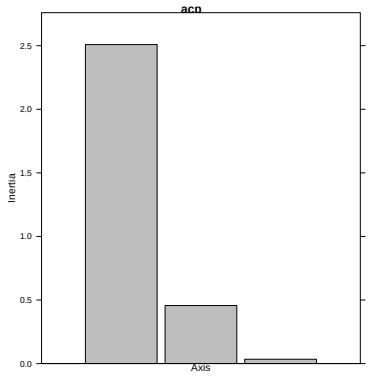
```
 head(recon)
  Empan1 Empan2 Taille
1   18.5   18.0  173.0
2   19.5   20.5  177.8
5   20.0   20.0  165.0
6   18.0   17.7  172.7
7   17.7   17.7  182.9
8   17.0   17.3  157.0
```

6.3 Représentations graphiques dans ade4

6.3.1 Représentation des valeurs propres

La fonction `screeplot` permet de visualiser les valeurs propres associées à l'analyse en composantes principales `acp` c'est-à-dire les inerties sur chaque axe.

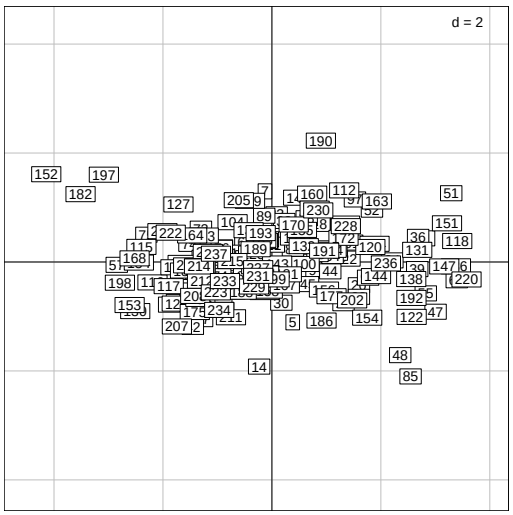
```
screeplot(acp)
```



6.3.2 Représentation des individus

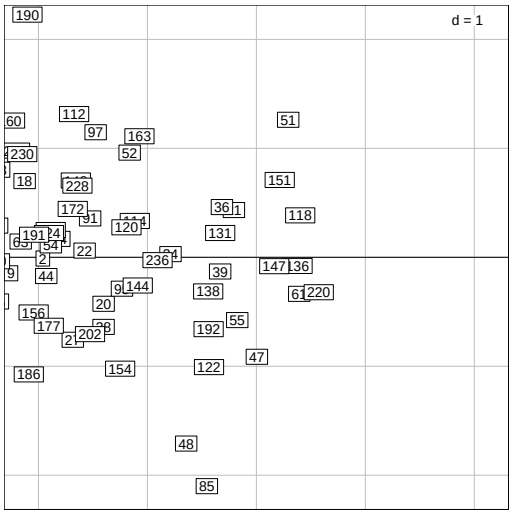
La fonction `s.label()` permet de représenter les individus sur les différents plans factoriels, par exemple sur le premier plan factoriel et peut être sauvegardé dans un objet de R.

```
gli <- s.label(acp$li, xax = 1, yax = 2)
```



Si on veut affiner la lecture de la carte factorielle, on peut utiliser la fonction `zoom`. On souhaite par exemple, mieux visualiser les individus se situant entre 2 et 4 sur l'axe horizontal.

```
zoom(gli,zoom=2,center=c(3,0))
```

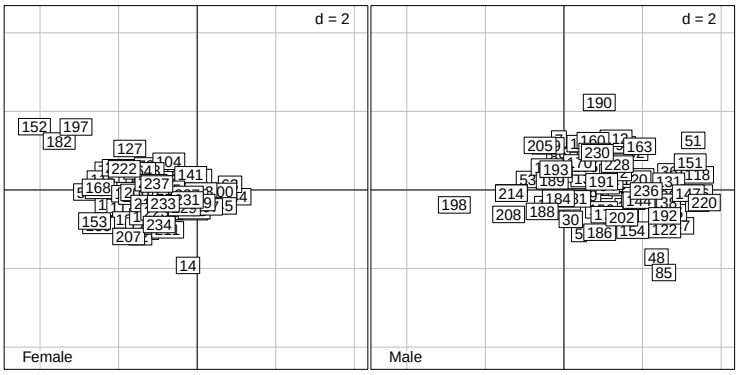


Exercice. Faire la représentation dans le plan (2,3). Commenter.

L'introduction d'une information supplémentaire comme la variable sexe peut se réaliser de plusieurs façons.

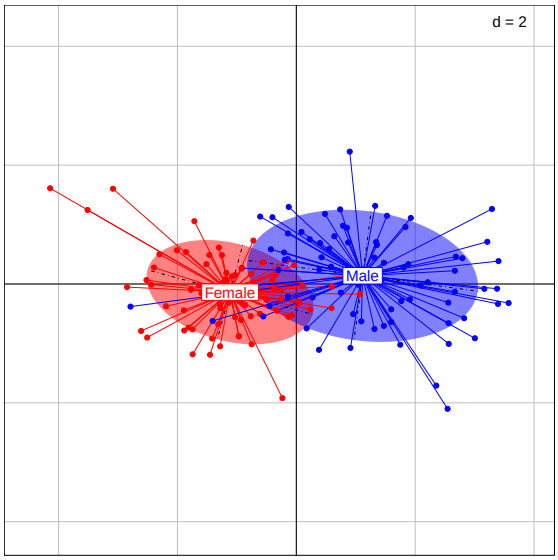
1. L'information est entrée via l'argument `facets` de la fonction `s.label` :

```
s.label(acp$li, xax = 1, yax = 2, facets=sexe)
```



2. L'information est entrée via l'argument `fac` de la fonction `s.class` :

```
colsexe <- c("red", "blue")
s.class(dfxy = acp$li, fac = sexe, col = colsexe, xax = 1, yax = 2)
gli2 <- s.class(dfxy = acp$li, fac = sexe, col = colsexe, xax = 1, yax = 2)
```

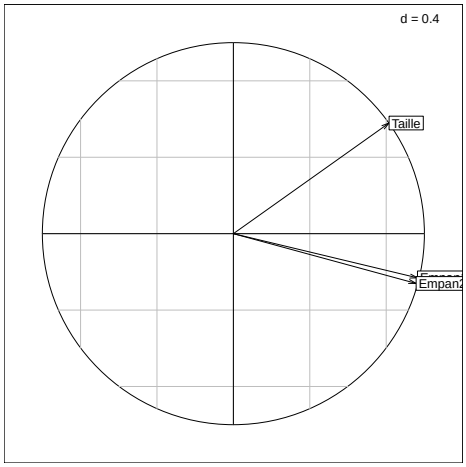


Exercice. Faire la représentation dans le plan factoriel (2,3). Commenter.

6.3.3 Représentation des variables

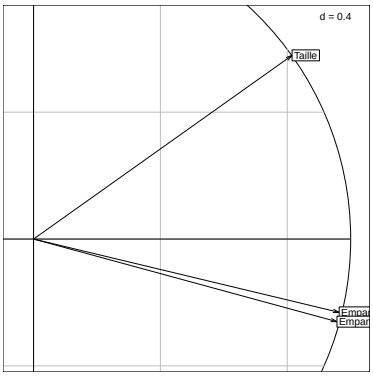
La fonction `s.corcircle()` représente les variables initiales dans le nouvel espace. Cette représentation est appelée cercle des corrélations :

```
s.corcircle(acp$co, xax = 1, yax = 2)
```



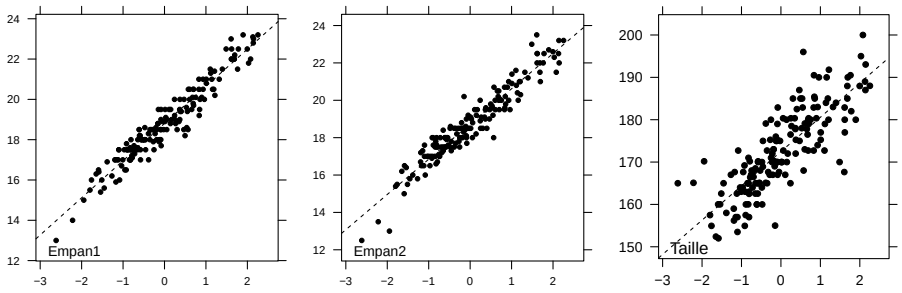
Comme toute l'information est concentrée à droite du cercle, on peut choisir de ne pas le représenter en entier et de le sauvegarder dans l'objet `gco` :

```
gco <- s.corcircle(acp$co, xax = 1, yax = 2, fullcircle=FALSE)
```



Le premier facteur de l'ACP est corrélé positivement aux trois variables de départ, on dit que c'est un effet "taille". L'ACP joue ici son rôle de recherche de *variable latente*. La première composante principale prédit les trois variables. C'est une *explicative cachée*. C'est la variable cachée qui prédit au mieux les autres, c'est aussi la variable cachée qui est la mieux prédite par toutes les autres.

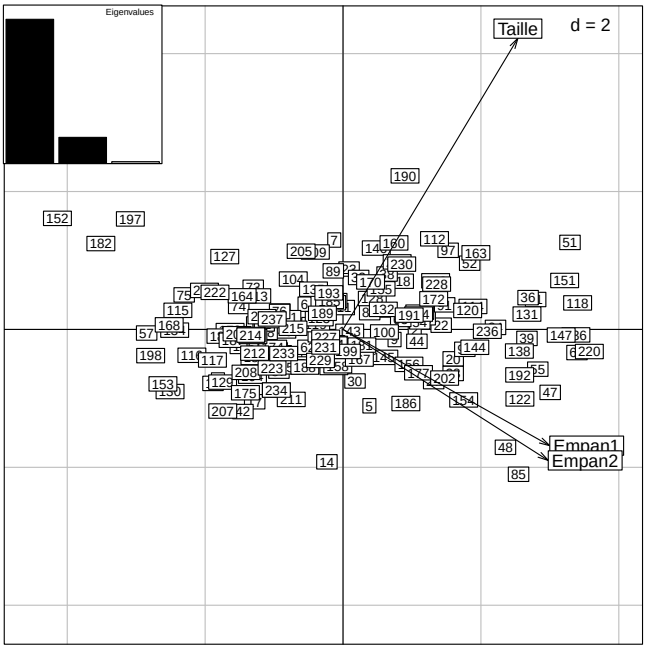
```
score(acp)
```



6.3.4 Représentation simultanée des individus et des variables

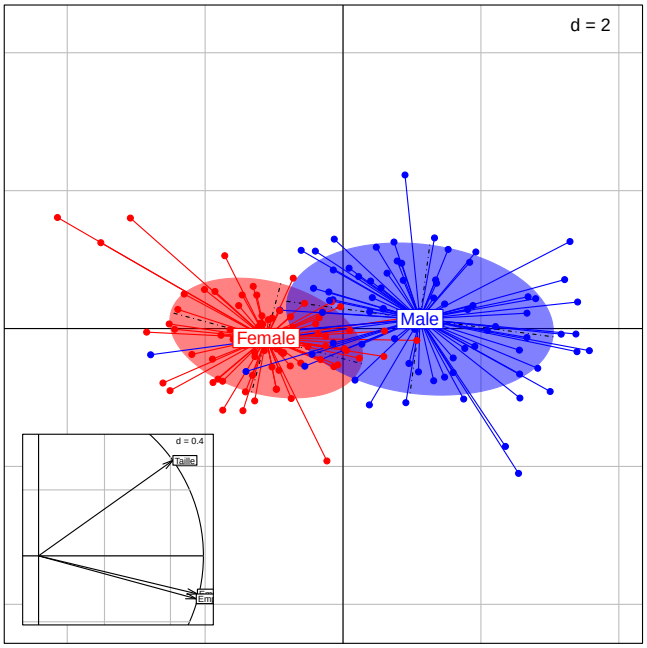
La fonction `scatter()` permet de représenter simultanément les individus et les variables.

```
scatter(acp)
```

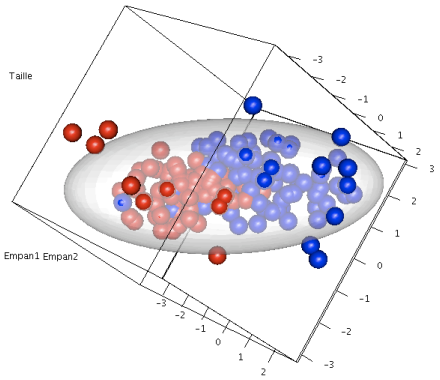


En conclusion, une fois les graphiques retenues, on peut choisir de superposer la partie informative du cercle des corrélation placé dans l'objet `gco` avec la représentation du nuage des individus placée dans l'objet `gli2`.

```
insert(gco,gli2,posi=c(0.03,0.03,0.33,0.33))
```



Faire le lien avec ce qui avait été obtenu à la main avec la fonction `plot3d()`, noter en particulier comment les axes de la base initiale se projettent sur le premier plan factoriel :



Références

[1] Daniel Adler and Duncan Murdoch. *rgl : 3D visualization device system (OpenGL)*, 2006. R package version 0.68.

[2] D. Chessel, A.-.B. Dufour, and J. Thioulouse. The ade4 package-I- One-table methods. *R News*, 4 :5–10, 2004.

[3] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002.