Fiche TD avec le logiciel R : tdr641

Analyse de co-inertie sur données simulées et sur données protéomiques

J.R. Lobry

Pratique de l'analyse de co-inertie sur un jeu de données simulées et sur un vrai jeu de données concernant la composition du protéome de *Escherichia coli* et les propriétés physico-chimiques des 20 acides aminés.

Table des matières

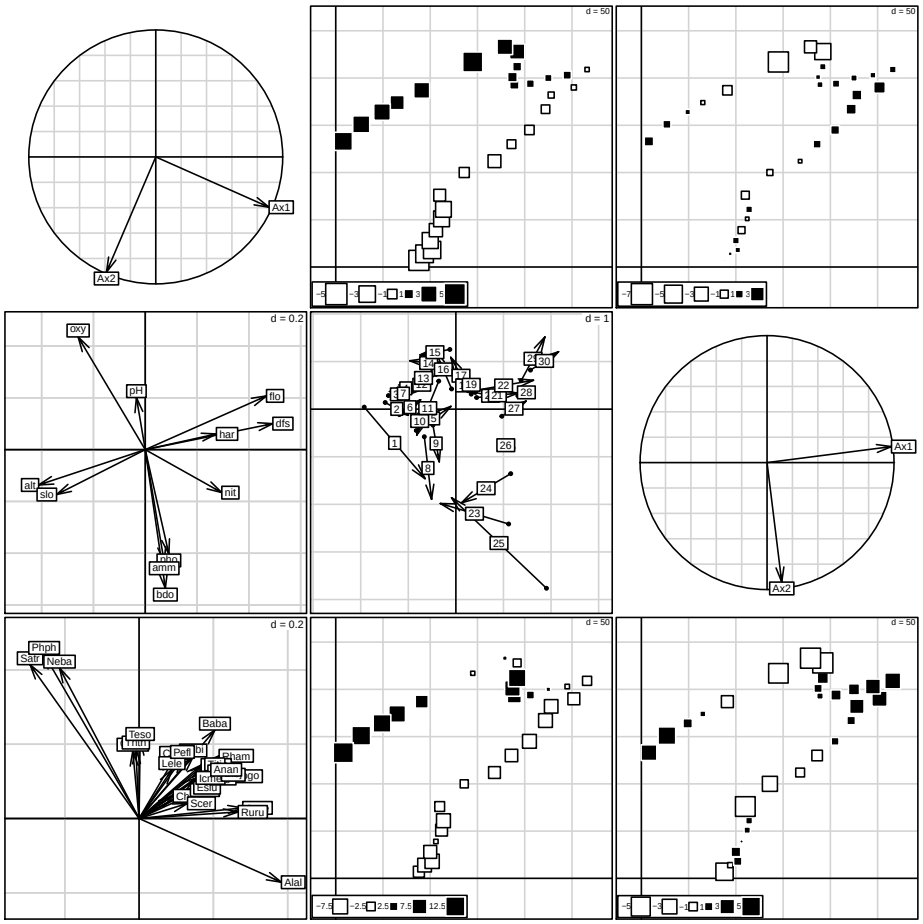
1 Introduction

L'analyse de co-inertie est une méthode de couplage entre deux tableaux [2, 11, 3, 4]. Cette méthode est disponible dans le paquet `ade4` [1] sous le nom de `coinertia()`. Commencez par vérifier que vous êtes capables de reproduire l'exemple de la fonction (`?coinertia`) basé sur le jeu de données `doubs` [12].

```
library(ade4)
data(doubs)
pca1 <- dudi.pca(doubs$env, scan = FALSE)
pca2 <- dudi.pca(doubs$fish, scale = FALSE, scan = FALSE)
coiner1 <- coinertia(pca1, pca2, scan = FALSE)
```

Faire les représentations graphiques en utilisant uniquement les fonctions du paquet `ade4` :

```
par(mfrow = c(3, 3))
s.corcircle(coiner1$aX)
s.value(doubs$xy, coiner1$lX[, 1])
s.value(doubs$xy, coiner1$lX[, 2])
s.arrow(coiner1$c1)
s.match(coiner1$mX, coiner1$mY)
s.corcircle(coiner1$aY)
s.arrow(coiner1$l1)
s.value(doubs$xy, coiner1$lY[, 1])
s.value(doubs$xy, coiner1$lY[, 2])
```



Si vous êtes arrivés à reproduire ces exemples c'est que vous avez tous les outils nécessaires pour faire une analyse de co-inertie dans R.

2 Données simulées

Il nous faut deux tableaux. Le premier est repris de la fiche « L'effet arc-en-ciel » [1], le deuxième est spécifique de cette fiche. On commence par faire l'analyse séparée de chaque tableau.

2.1 Premier tableau

Le premier tableau est une simulation des abondances d'espèces dans plusieurs stations réparties régulièrement le long d'un gradient de température. On suppose que les abondances des espèces sont données par le modèle dit des températures cardinales [8].

```
CTMI <- function(T, param)
{
  Tmin <- param[1]
  Topt <- param[2]
  Tmax <- param[3]
  Muopt <- param[4]
  if( T <= Tmin || T >= Tmax )
  {
    return(0)
  }
  else
  {
    Num <- (T-Tmax)*(T-Tmin)^2
    Den <- (Topt-Tmin)*((Topt-Tmin)*(T-Topt)-(Topt-Tmax)*(Topt+Tmin-2*T))
    return(Muopt*Num/Den)
  }
}
```

On suppose qu'il y a 16 stations réparties régulièrement sur le gradient de température :

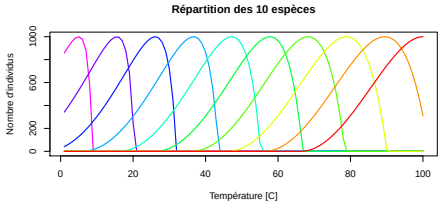
```
Tstations <- seq(from = 5, to = 80, by = 5)
Tstations
```

```
[1]  5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80
```

On suppose qu'il y a en tout 10 espèces observées :

```
nsp <- 10
topt <- seq(from = 5, to = 100, length = nsp)
tmin <- 0.953*topt - 28.913
tmax <- 1.101*topt + 3.203
data <- data.frame(matrix(nrow = nsp, ncol = length(Tstations)))
names(data) <- paste("st",1:length(Tstations), sep = "")
row.names(data) <- paste("sp", round(topt, 0), sep = "")
for( i in 1:nsp )
  for( j in 1:length(Tstations))
    data[i, j] <- round(CTMI(T = Tstations[j], param = c(tmin[i], topt[i],
      tmax[i], 1000)), 0)
Taxis <- 1:100
cols <- rev(rainbow(nsp, end = 5/6))
plot(x = Taxis, y = sapply(Taxis, CTMI, param = c(tmin[1], topt[1], tmax[1], 1000)),
  type = "l", col = cols[1], main = paste("Répartition des", nsp, "espèces"),
  xlab = "Température [C]", ylab = "Nombre d'individus", lwd = 2)
for( i in 2:nsp )
  lines(x = Taxis, y = sapply(Taxis, CTMI, param = c(tmin[i], topt[i], tmax[i], 1000)),
  col = cols[i], lwd = 2)
```

1. http://pbil.univ-lyon1.fr/R/fichestd/tdr622.pdf

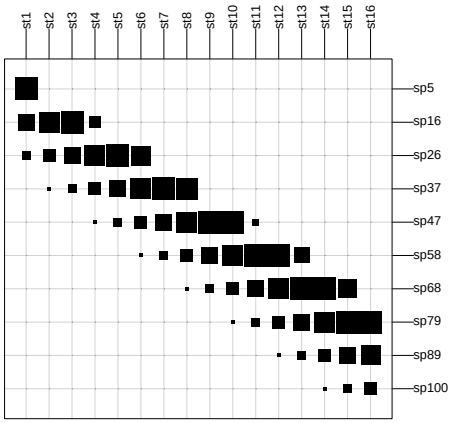


La structure des données est donc très simple : on a une succession des espèces le long du gradient :

```
head(data[ , 1:10])
```

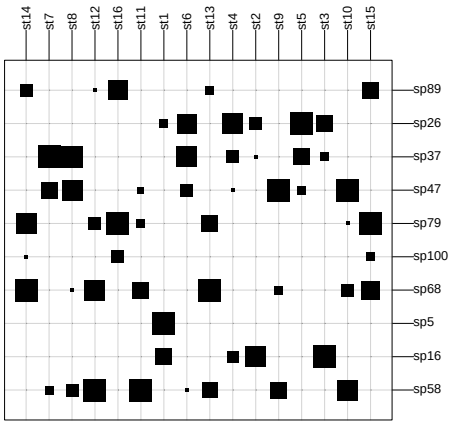
```
      st1 st2 st3 st4 st5 st6 st7 st8 st9 st10
sp5  1000   0   0   0   0   0   0   0   0    0
sp16  533 799 996 248   0   0   0   0   0    0
sp26  129 303 534 793 989 719   0   0   0    0
sp37    0  26 130 305 535 786 980 865   0    0
sp47    0   0   0  26 131 307 535 780 971  931
sp58    0   0   0   0   0  26 133 309 534  773
```

```
table.value(data, clegend = 0)
```



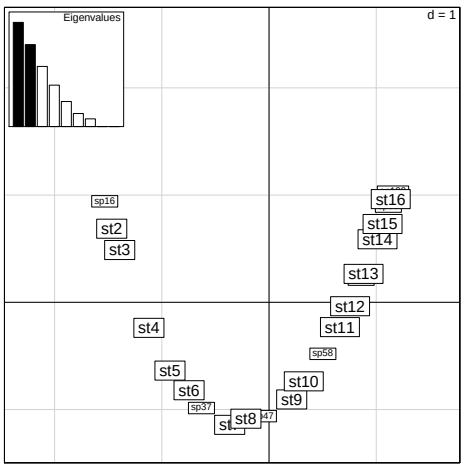
Mais dans la vraie vie nous ne connaissons pas encore l'ordre des espèces et des stations qui met en évidence la structure des données. Nos données ressembleraient plutôt à ceci :

```
data.vv <- data[sample(nrow(data)), sample(ncol(data))]
table.value(data.vv, clegend = 0)
```



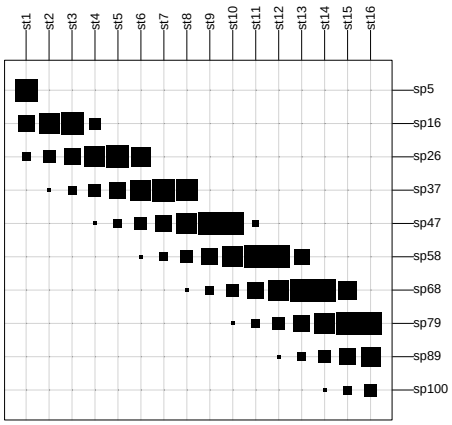
C'est nettement moins parlant. Heureusement, le premier plan de l'AFC donne un effet Guttman :

```
afc <- dudi.coa(data.vv, scann = FALSE)
scatter(afc)
```



Ceci nous permet de ré-ordonner le tableau pour mettre en évidence le gradient :

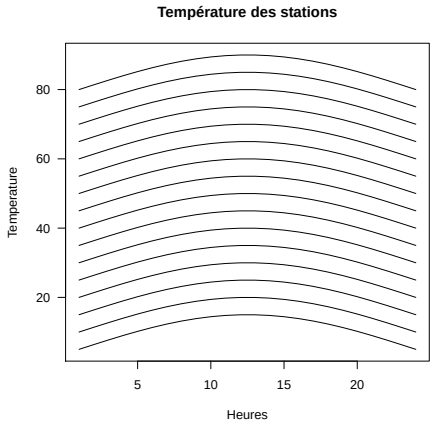
```
data.vv.ord <- data.vv[order(afc$li[, 1]), order(afc$co[, 1])]
table.value(data.vv.ord, clegend = 0)
```



2.2 Deuxième tableau

On suppose que l'on a mesuré la température des stations toutes les heures pendant une journée :

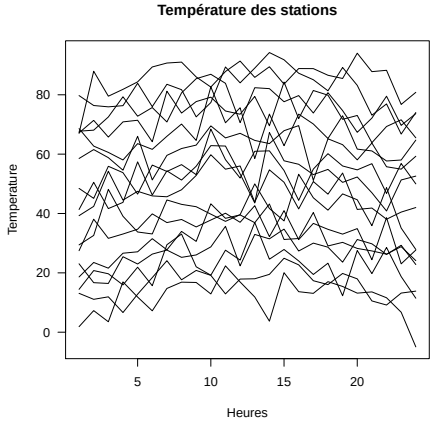
```
mesureT <- sapply(Tstations, function(x){10*sin(seq(0, pi, le = 24)) + x})
colnames(mesureT) <- paste("st",1:16,sep="")
rownames(mesureT)<-paste("h",1:24,sep="")
plot(0, type = "n", xlim = c(1,24), ylim = range(mesureT), xlab = "Heures", ylab = "Temperature",
las = 1, main = "Température des stations")
 apply(mesureT,2,lines)
```



On ajoute un bruit gaussien pour plus de réalisme.

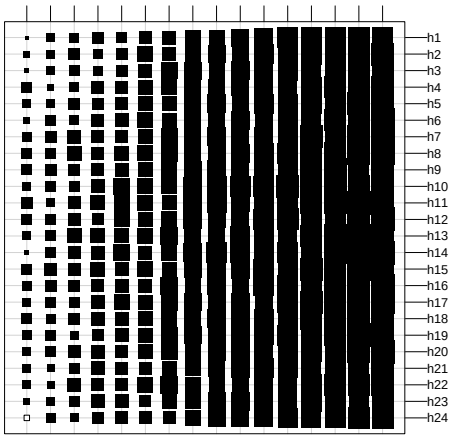
```
set.seed(1)
mesureT <- sapply(Tstations, function(x){10*sin(seq(0, pi,le = 24)) + x +
  rnorm(n = 24, sd = 5)})
colnames(mesureT) <- paste("st", 1:16, sep = "")
rownames(mesureT) <-paste("h", 1:24, sep = "")
plot(0, type = "n", xlim = c(1,24), ylim = range(mesureT), xlab = "Heures",
  ylab = "Temperature", las = 1, main = "Température des stations")
apply(mesureT,2,lines)
```

```
NULL
```



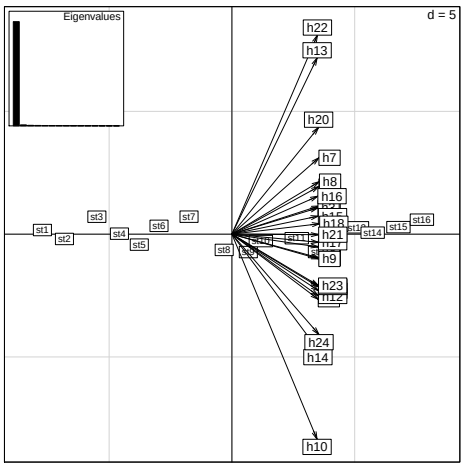
La structure du jeu de données est donc très simple : on a un gradient des stations froides aux stations chaude avec une petite composante circadienne difficile à voir directement vu l'intensité de bruit introduite ici :

```
table.value(mesureT, clegend = 0)
```



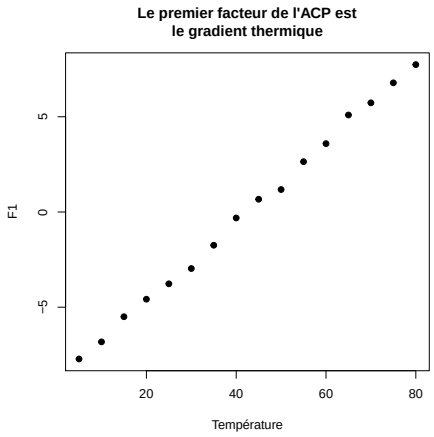
Faire l'ACP du tableau :

```
acp <- dudi.pca(t(mesureT), scan=F)
scatter(acp)
```



Le premier facteur extrait 96.31 % de la variabilité, on a simplement récupéré le gradient thermique entre les stations :

```
plot(Tstations, acp$li[,1], ylab = "F1", xlab = "Température",
main = "Le premier facteur de l'ACP est\nle gradient thermique",
pch = 19)
```



2.3 Couplage des deux tableaux

Pour pouvoir coupler les deux tableaux, il faut qu'ils aient quelque chose en commun. Ici ce sont les 16 stations qui sont communes aux deux tableaux. Par convention, dans la fonction `coinertia()`, le couplage se fait en supposant que les individus en communs sont en ligne dans les deux tableaux. Est-ce le cas ici ?

```
 dim(data)
[1] 10 16
 dim(mesureT)
[1] 24 16
```

Non, ce n'est pas le cas : les stations sont en colonne. La première chose à faire est donc de transposer les deux tableaux.

```
tab1 <- as.data.frame(t(data))
tab2 <- as.data.frame(t(mesureT))
```

Mais encore faut il que l'ordre des individus soit le même dans les deux tableaux. C'est un point qu'il est prudent de vérifier pour ne pas avoir de mauvaises surprises par la suite.

```
rownames(tab1)
```

```
 [1] "st1"  "st2"  "st3"  "st4"  "st5"  "st6"  "st7"  "st8"  "st9"  "st10" "st11"
[12] "st12" "st13" "st14" "st15" "st16"
```

```
rownames(tab2)
```

```
 [1] "st1"  "st2"  "st3"  "st4"  "st5"  "st6"  "st7"  "st8"  "st9"  "st10" "st11"
[12] "st12" "st13" "st14" "st15" "st16"
```

```
all.equal(rownames(tab1),rownames(tab2))
```

```
[1] TRUE
```

Pour que le couplage ait un sens if faut que la pondération des individus soit la même dans les deux analyses. On décide d'utiliser ici la pondération de l'AFC :

```
coa <- dudi.coa(df = tab1, scannf = FALSE, nf = 2)
pca <- dudi.pca(df = tab2, row.w = coa$lw, scannf = FALSE, nf = 2)
```

Vérifier que la même pondération a bien été utilisée dans les deux analyses :

```
all.equal(pca$lw,coa$lw)
```

```
[1] TRUE
```

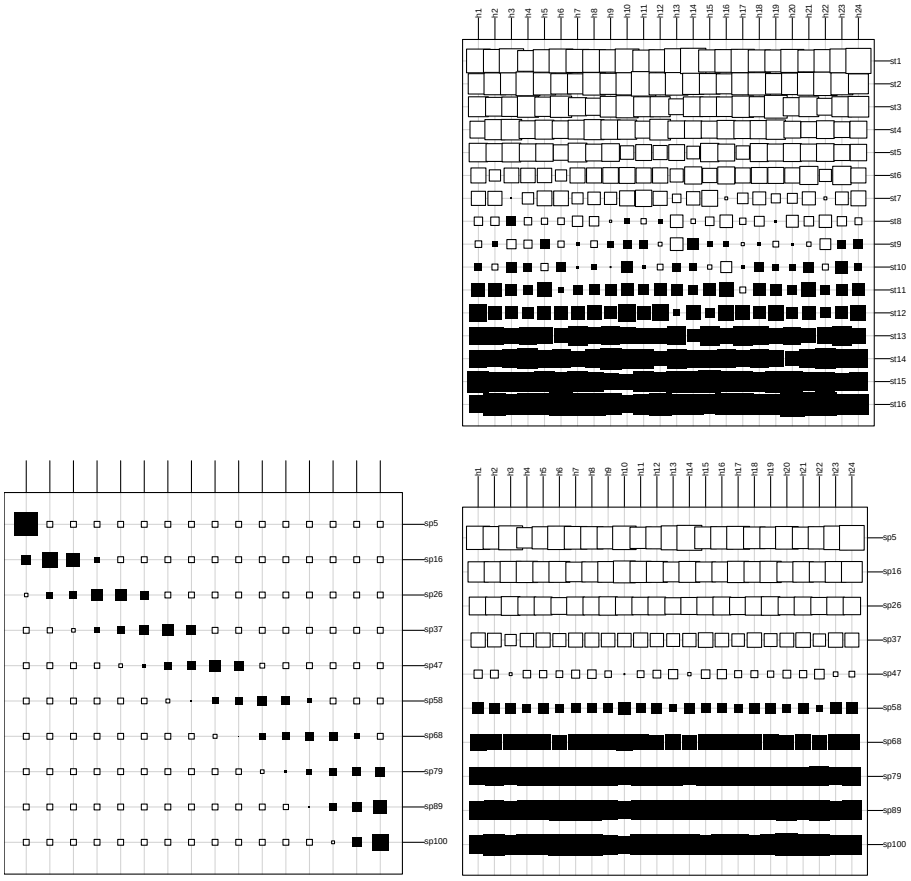
On peut alors utiliser l'analyse de co-inertie pour coupler les deux tableaux :

```
cia <- coinertia(dudiX = pca, dudiY = coa, scannf = FALSE, nf = 2)
cia$eig[1]/sum(cia$eig)
```

```
[1] 0.993294
```

Le premier facteur extrait 99.33 % de la variabilité, nous avons redécouvert ici le gradient thermique commun aux deux tableaux. Représenter graphiquement les trois tableaux analysés ici :

```
plot.new()
par(mfrow = c(2, 2))
par(mfg = c(1, 2))
table.value(pca$tab, clegend = 0, clabel.row = 0.7, clabel.col = 0.7)
par(mfg = c(2, 1))
table.value(t(coa$tab), clegend = 0, clabel.row = 0.7, clabel.col = 0.5)
par(mfg = c(2, 2))
table.value(cia$tab, clegend = 0, clabel.row = 0.7, clabel.col = 0.7)
```



3 Données protéomiques

Il s'agit ici de reproduire les résultats de [9], une version PDF de l'article est disponible à `http://pbil.univ-lyon1.fr/JTHome/Biblio/Cabios95.pdf`.

3.1 Premier tableau

Le premier tableau contient les fréquences absolues des acides aminés dans un échantillon des protéines de *Escherichia coli*. C'est ce jeu de données qui avait été utilisé dans l'article [6] disponible à `http://nar.oxfordjournals.org/cgi/content/abstract/22/15/3174`. Importer les données sous R :

```
ProtComp <- read.table("http://pbil.univ-lyon1.fr/members/lobry/repro/cabios95/ProtComp",
  sep = "\t", header = FALSE)
dim(ProtComp)
```

```
[1] 999  20
```

Le nom des acides-aminés est donné dans le fichier `AANames`, le nom des protéines dans le fichier `ProtNames`, les utiliser pour donner un nom aux lignes et aux colonnes de l'objet `ProtComp` :

```
AANames <- readLines("http://pbil.univ-lyon1.fr/members/lobry/repro/cabios95/AANames")
ProtNames <- readLines("http://pbil.univ-lyon1.fr/members/lobry/repro/cabios95/ProtNames")
names(ProtComp) <- AANames
rownames(ProtComp) <- ProtNames
head(ProtComp)
```

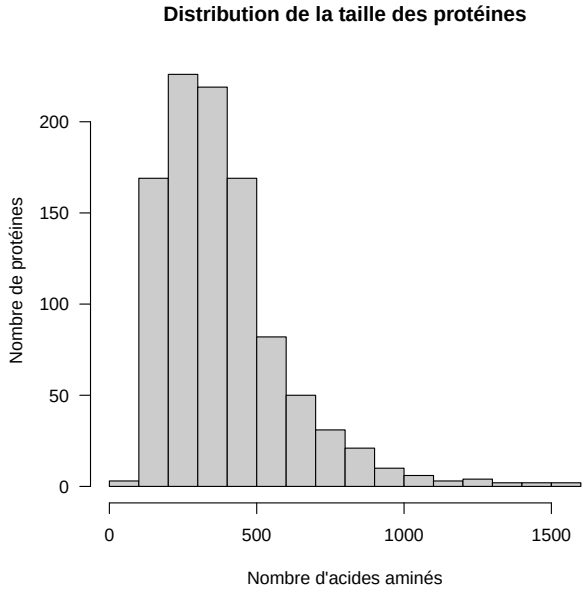
```
               ala arg asn asp cys gln glu gly his ile leu lys met phe pro ser thr
ECFOLE.FOLE     19  13   9  12   2   9  13   9  10  15  23  11   8   7   8  14  19
ECMSBAG.MSBA    48  38  21  30   2  25  28  38   9  47  60  23  26  23  13  50  36
ECNARZYW-C.NARV 12  15   4   5   3   9   3  20   5  18  29   3  12  18   8  15  13
ECNARZYW-C.NARW 21  13   6  21   4  14  20  11   3   6  38   9   4   6  10  13   7
ECNARZYW-C.NARY 32  34  20  29  18  15  43  41  11  31  33  27  19  17  31  26  20
ECNARZYW-C.NARZ 89  71  55  72  17  50  70 102  36  62 104  56  36  36  75  87  67
               trp tyr val
ECFOLE.FOLE      0   4  18
ECMSBAG.MSBA     5  13  47
ECNARZYW-C.NARV  8  11  15
ECNARZYW-C.NARW  4   8  13
ECNARZYW-C.NARY 11  23  33
ECNARZYW-C.NARZ 40  50  71
```

Transposer le tableau de façon à avoir les acides aminés en ligne :

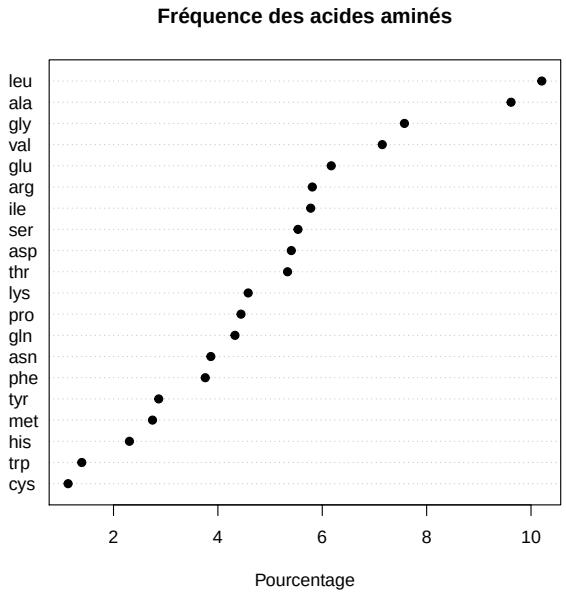
```
ProtComp <- as.data.frame(t(ProtComp))
ProtComp[1:5,1:5]
```

```
    ECFOLE.FOLE ECMSBAG.MSBA ECNARZYW-C.NARV ECNARZYW-C.NARW ECNARZYW-C.NARY
ala          19           48              12              21              32
arg          13           38              15              13              34
asn           9           21               4               6              20
asp          12           30               5              21              29
cys           2            2               3               4              18
```

Représenter graphiquement la distribution de la taille des protéines :

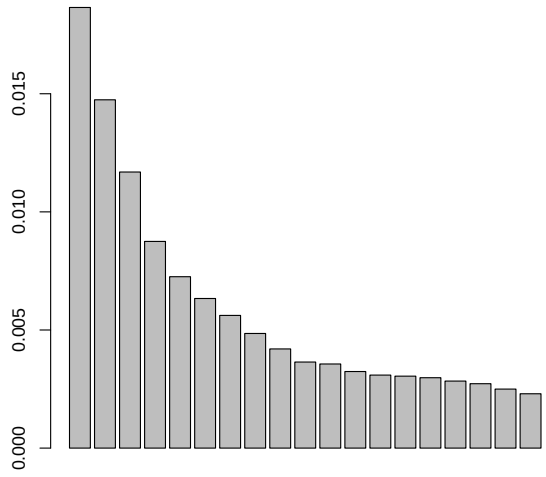


Représenter graphiquement les fréquences des acide-aminés dans les protéines :

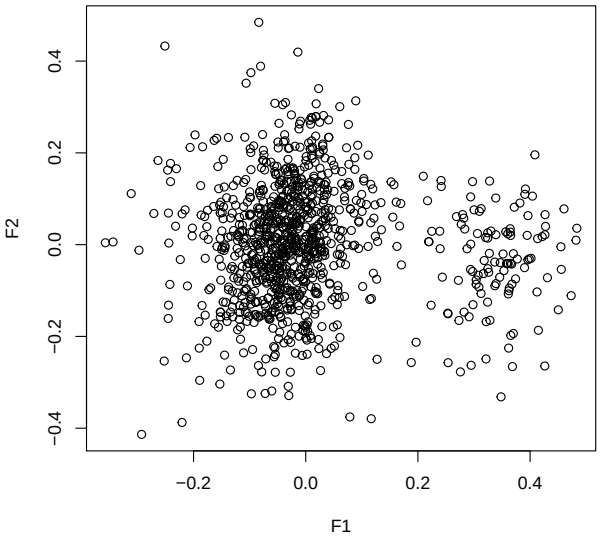


Faire l'AFC du tableau et représenter le graphe des valeurs propres :

```
coa <- dudi.coa(df = ProtComp, scannf = FALSE, nf = 2)
barplot(coa$eig)
```



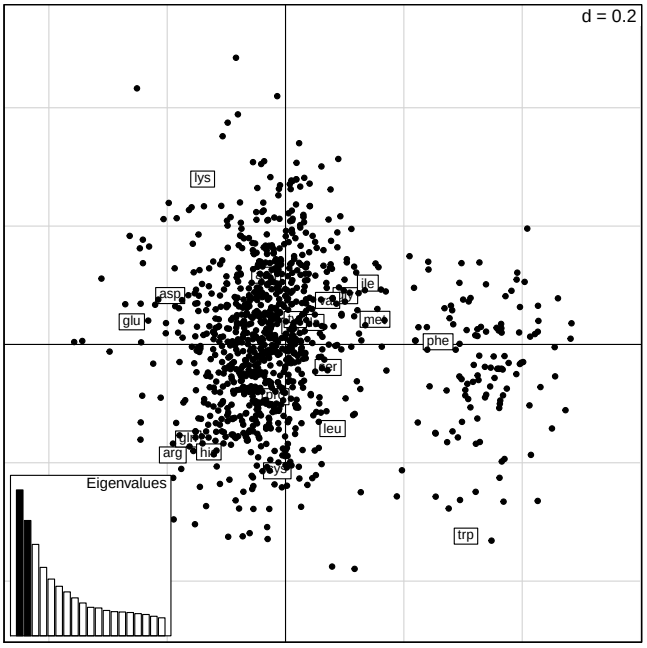
Représenter les protéines sur le premier plan factoriel :



Quelle structure dans les données le premier facteur de l'AFC a-t-il mis en évidence ?

Représentez maintenant simultanément les protéines et les acides aminés sur le premier plan factoriel :

```
scatter(coa, clab.col=0, posieig = "bottomleft")
```



Qu'est ce qui caractérise les deux groupes de protéines en terme de composition en acides aminés ?

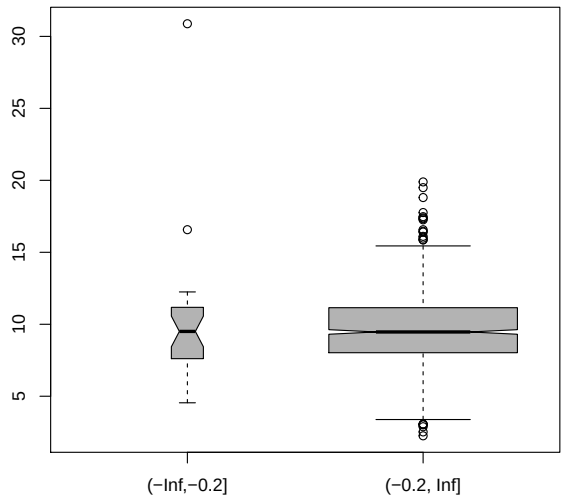
Utilisez la fonction `cut()` pour transformer les valeurs sur le premier facteur en une variable qualitative :

```
gprot <- cut(x = coa$co[,1], breaks = c(-Inf,-0.2,+Inf))
summary(gprot)
```

```
(-Inf,-0.2] (-0.2, Inf]
         28         971
```

Revenir aux données de départ. Etudier comment les deux groupes de caractérisent pour leur fréquences en acides aminés. Par exemple pour l'alanine :

```
ProtFreq <- 100*apply(ProtComp, 2, function(x) x/sum(x))
ala <- ProtFreq["ala",]
boxplot(ProtFreq["ala",]~gprot, notch = TRUE, varwidth = TRUE, col = grey(0.7))
```



Faire la même chose pour tous les acides-aminés en examinant à chaque fois le premier plan factoriel de l'AFC pour voir si les résultats sont cohérents.

3.2 Deuxième tableau

Le deuxième tableau contient 402 propriétés physico-chimiques pour les 20 acides-aminés. Ces données sont extraites de `aaindex` [5, 10, 7]. Importer les données sous R :

```
IndiceVals <- read.table("http://pbil.univ-lyon1.fr/members/lobry/repro/cabios95/IndiceVals",
  sep = "\t", header = FALSE)
dim(IndiceVals)
```

```
[1] 402  20
```

Comme pour le premier fichier, le nom des acides-aminés est donné dans le fichier `AANames` que nous avons déjà importé sous R, le nom des propriétés physico-chimiques est dans le fichier `IndiceNames`, les utiliser pour donner un nom aux lignes et aux colonnes de l'objet `IndiceVals` :

```
names(IndiceVals) <- AANames
IndiceNames <- readLines("http://pbil.univ-lyon1.fr/members/lobry/repro/cabios95/IndiceNames")
rownames(IndiceVals) <- IndiceNames
head(IndiceVals)
```

```
            ala  arg  asn  asp  cys  gln  glu  gly  his  ile  leu  lys  met  phe
ANDN920101 4.35 4.38 4.75 4.76 4.65 4.37 4.29 3.97 4.63 3.95 4.17 4.36 4.52 4.66
ARGP820101 0.61 0.60 0.06 0.46 1.07 0.00 0.47 0.07 0.61 2.22 1.53 1.15 1.18 2.02
ARGP820102 1.18 0.20 0.23 0.05 1.89 0.72 0.11 0.49 0.31 1.45 3.23 0.06 2.67 1.96
ARGP820103 1.56 0.45 0.27 0.14 1.23 0.51 0.23 0.62 0.29 1.67 2.93 0.15 2.96 2.03
BEGF750101 1.00 0.52 0.35 0.44 0.06 0.44 0.73 0.35 0.60 0.73 1.00 0.60 1.00 0.60
BEGF750102 0.77 0.72 0.55 0.65 0.65 0.72 0.55 0.65 0.83 0.98 0.83 0.55 0.98 0.98
            pro  ser  thr  trp  tyr  val
ANDN920101 4.44 4.50 4.35 4.70 4.60 3.95
ARGP820101 1.95 0.05 0.05 2.65 1.88 1.32
ARGP820102 0.76 0.97 0.84 0.77 0.39 1.08
ARGP820103 0.76 0.81 0.91 1.08 0.68 1.14
BEGF750101 0.06 0.35 0.44 0.73 0.44 0.82
BEGF750102 0.55 0.55 0.83 0.77 0.83 0.98
```

Transposer le tableau pour avoir les acides aminés en ligne comme précédemment :

```
IndiceVals <- as.data.frame(t(IndiceVals))
IndiceVals[1:5,1:5]
```

```
    ANDN920101 ARGP820101 ARGP820102 ARGP820103 BEGF750101
ala       4.35       0.61       1.18       1.56       1.00
arg       4.38       0.60       0.20       0.45       0.52
asn       4.75       0.06       0.23       0.27       0.35
asp       4.76       0.46       0.05       0.14       0.44
cys       4.65       1.07       1.89       1.23       0.06
```

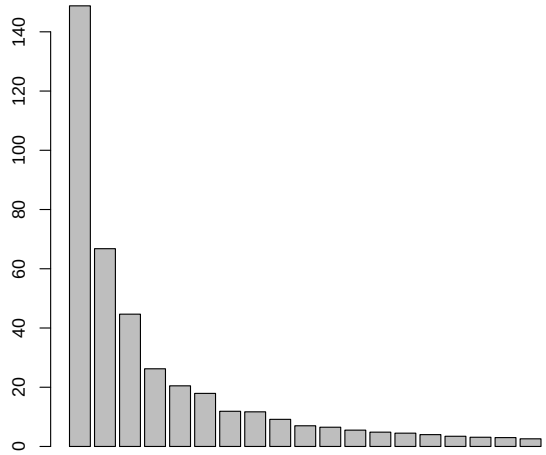
La signification des indices physico-chimiques est accessible sous R via le jeu de données `aaindex` du package `seqinr`, par exemple pour connaître la signification de l'indice `ANDN920101` :

```
library(seqinr)
data(aaindex)
aaindex[["ANDN920101"]]
```

```
$H
[1] "ANDN920101"
$D
[1] "alpha-CH chemical shifts (Andersen et al., 1992)"

$R
[1] "LIT:1810048b PMID:1575719"

$A
[1] "Andersen, N.H., Cao, B. and Chen, C."

$T
[1] "Peptide/protein structure analysis using the chemical shift index method: upfield alpha-CH values reveal d

$J
[1] "Biochem. and Biophys. Res. Comm. 184, 1008-1014 (1992)"

$C
[1] "BUNA790102    0.949"

$I
 Ala  Arg  Asn  Asp  Cys  Gln  Glu  Gly  His  Ile  Leu  Lys  Met  Phe  Pro  Ser  Thr
4.35 4.38 4.75 4.76 4.65 4.37 4.29 3.97 4.63 3.95 4.17 4.36 4.52 4.66 4.44 4.50 4.35
 Trp  Tyr  Val
4.70 4.60 3.95
```

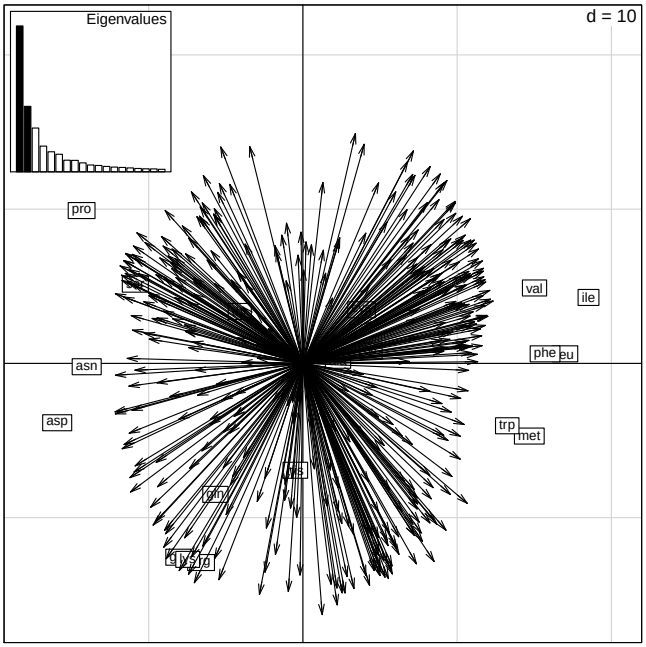
Faire l'ACP du tableau en utilisant la même pondération pour les lignes, c'est à dire pour les acides aminés, que pour le premier tableau. Tracer le graphe des valeurs propres :

```
pca <- dudi.pca(df = IndiceVals, row.w = coa$lw, scannf = FALSE, nf = 2)
barplot(pca$eig)
```



Représenter le premier plan factoriel :

```
scatter(pca, clab.col=0)
```



3.3 Couplage des deux tableaux

Avant de vous lancer dans l'analyse de co-inertie, vérifiez que les conditions d'utilisation sont bien remplies. Donner les le code R permettant de répondre aux questions suivantes :

1. Les individus communs aux deux tableaux sont-ils bien en ligne dans les deux tableaux ?

```
[1]  20 402
[1]  20 999
[1] TRUE
```

2. Les individus communs aux deux tableaux sont-ils bien dans le même ordre dans les deux tableaux ?

```
 [1] "ala" "arg" "asn" "asp" "cys" "gln" "glu" "gly" "his" "ile" "leu" "lys" "met"
[14] "phe" "pro" "ser" "thr" "trp" "tyr" "val"
 [1] "ala" "arg" "asn" "asp" "cys" "gln" "glu" "gly" "his" "ile" "leu" "lys" "met"
[14] "phe" "pro" "ser" "thr" "trp" "tyr" "val"
[1] TRUE
```

3. La même pondération a-t-elle bien été utilisée pour les deux analyses ?

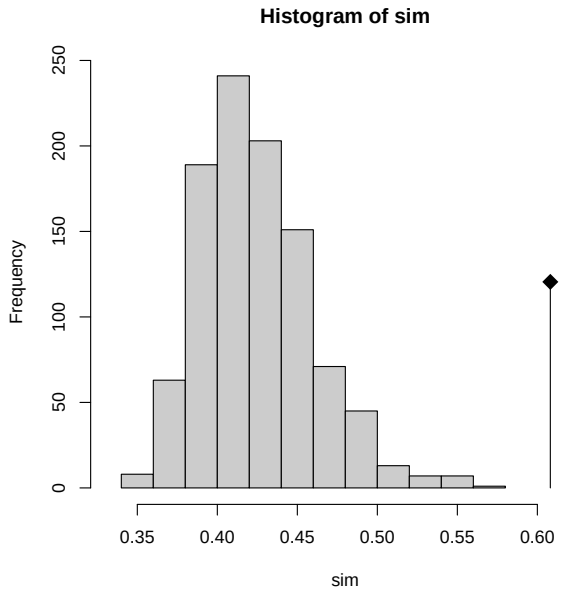
```
      ala       arg       asn       asp       cys       gln       glu
0.09617700 0.05809748 0.03864776 0.05407313 0.01129983 0.04327407 0.06172224
      gly       his       ile       leu       lys       met       phe
0.07574649 0.02304854 0.05778612 0.10206173 0.04580907 0.02746988 0.03758134
      pro       ser       thr       trp       tyr       val
0.04443389 0.05534971 0.05334662 0.01391267 0.02866343 0.07149900
      ala       arg       asn       asp       cys       gln       glu
0.09617700 0.05809748 0.03864776 0.05407313 0.01129983 0.04327407 0.06172224
      gly       his       ile       leu       lys       met       phe
0.07574649 0.02304854 0.05778612 0.10206173 0.04580907 0.02746988 0.03758134
      pro       ser       thr       trp       tyr       val
0.04443389 0.05534971 0.05334662 0.01391267 0.02866343 0.07149900
[1] TRUE
```

Faire l'analyse de co-inertie des deux tableaux :

```
cia <- coinertia(dudiX = pca, dudiY = coa, scannf = FALSE, nf = 2)
```

Tester la significativité de la co-structure entre les deux tables :

```
ciatest <- randtest(cia, nrepet =  999, fixed = 2)
Warning: non uniform weight. The results from permutations
are valid only if the row weights come from the fixed table.
The fixed table is table Y : ProtComp

plot(ciatest)
```



Le test est basé sur la comparaison entre la valeur de la co-inertie totale, le coefficient RV entre les deux table :

```
cia$RV
```

```
[1] 0.6081122
```

et la distribution empirique du coefficient RV quand on détruit la co-structure entre les deux tables en permutant les lignes de l'une (c'est l'histogramme ci-dessus). Le nombre de permutations est contrôlé par le paramètre `nrepet`, la table dont on ne permute pas les lignes par le paramètre `fixed`. Il doit s'agit de la table qui a fourni les pondérations des lignes, ici la table des fréquences des acides aminés `ProtComp`.

Il y a donc ici une co-structure très significative entre les deux tables. Donner la liste des indices qui contribuent le plus à la définition du premier facteur :

```
cia$co[order(cia$co[,1]), 1, drop = FALSE][1:10, , drop = FALSE]
```

```
               Comp1
GRAR740102 -0.1185902
PRAM900101 -0.1164040
OOBM770101 -0.1163041
MEIH800102 -0.1156226
ROSM880102 -0.1156093
WOEC730101 -0.1153977
ROSM880101 -0.1149005
HOPT810101 -0.1142365
VHEG790101 -0.1132352
WOLS870101 -0.1120542
```

```
cia$co[rev(order(cia$co[,1])), 1, drop = FALSE][1:10, , drop = FALSE]
```

```
              Comp1
DESM900102 0.1212092
NAKH900110 0.1196475
MEIH800103 0.1181477
BIOV880102 0.1170724
EISD860103 0.1169063
JANJ780102 0.1162511
FAUJ830101 0.1153082
KYTJ820101 0.1152559
BIOV880101 0.1138223
ROSG850102 0.1135967
```

```
aaindex[["DESM900102"]]$D
```

```
[1] "Average membrane preference: AMP07 (Degli Esposti et al., 1990)"
```

En vous aidant de la signification des indices physico-chimiques, donner une interprétation biologique du premier facteur de l'analyse de co-inertie.

Références

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

[2] D. Chessel and P. Mercier. Couplage de triplets statistiques et liaisons espèces-environnement. In J.D. Lebreton and B. Asselain, editors, *Biométrie et Environnement*, pages 15–44. Masson, Paris, 1993.

[3] S. Dolédec and D. Chessel. Co-inertia analysis : an alternative method for studying species-environment relationships. *Freshwater Biology*, 31 :277–294, 1994.

[4] S. Dray, D. Chessel, and J. Thioulouse. Co-inertia analysis and the linking of ecological tables. *Ecology*, 84(11) :3078–3089, 2003.

[5] S. Kawashima and M. Kanehisa. AAindex : amino acid index database. *Nucleic Acids Res.*, 28 :374–374, 2000.

[6] J.R. Lobry and C. Gautier. Hydrophobicity, expressivity and aromaticity are the major trends of amino-acid usage in 999 *Escherichia coli* chromosome-encoded genes. *Nucleic Acids Research*, 22 :3174–3180, 1994.

[7] K. Nakai, A. Kidera, and M. Kanehisa. Cluster analysis of amino acid indices for prediction of protein structure and function. *Protein Eng.*, 2 :93–100, 1988.

[8] L. Rosso, J.R. Lobry, and J.-P. Flandrois. An unexpected correlation between cardinal temperatures of microbial growth highlighted by a new model. *Journal of Theoretical Biology*, 162(4) :447–463, 1993.

[9] J. Thioulouse and J.R. Lobry. Co-inertia analysis of amino-acid physicochemical properties and protein composition with the ADE package. *Computer Applications in the Biosciences*, 11 :321–329, 1995.

[10] K. Tomii and M. Kanehisa. Analysis of amino acid indices and mutation matrices for sequence comparison and structure prediction of proteins. *Protein Eng.*, 9 :27–36, 1996.

[11] F. Torre and D. Chessel. Co-structure de deux tableaux totalement appariés. *Revue de Statistique Appliquée*, 43 :109–121, 1994.

[12] J. Verneaux. Cours d'eau de franche-comté (massif du jura). recherches écologiques sur le réseau hydrographique du doubs. essai de biotypologie. Thèse d'état, besançon, 1973.