

Les pays à aider

Projet de apprentissage non supervisé
Classification
Université de Rennes II : Master Mathématiques Appliquées, Statistiques

Margaux Bailleul
Oriane Duclos

12 April, 2023

Contents

1	Chargement des librairies	1
2	Compréhension et pré-traitement des données	1
2.1	Statistiques descriptives	2
2.2	Pre-traitement	2
2.3	Matrice de corrélation	3
3	Classification des pays en utilisant les différents algorithmes abordés en cours	4
3.1	CAH	4
3.2	Algorithme des Kmeans	20
3.3	Interprétation des groupes	22
4	Traitement du groupe des pays les moins favorisés	26
5	4 - Conclusion vis à vis des choix effectués	39

1 Chargement des librairies

2 Compréhension et pré-traitement des données

```
donnee <- read.csv("Pays_donnees.csv", sep = ',', row.names = 1)
head(donnee,3)
```

```
##          enfant_mort exports sant. imports revenu inflation esper_vie fert
## Afghanistan      90.2    10.0  7.58    44.9    1610        9.44    56.2 5.82
## Albania           16.6    28.0  6.55    48.6    9930        4.49    76.3 1.65
## Algeria           27.3    38.4  4.17    31.4   12900       16.10    76.5 2.89
##          pib_h
## Afghanistan    553
```

```
## Albania      4090
## Algeria      4460
```

```
str(donnee)
```

Nous observons que toutes les colonnes ont des données qui sont en cohérence avec leur type.

```
dim(donnee)
```

Nous avons 167 individus et 9 variables

2.1 Statistiques descriptives

```
summary(donnee)
```

```
##   enfant_mort      exports      sant.      imports
##   Min.   :  2.60   Min.   :  0.109   Min.   :  1.810   Min.   :  0.0659
##   1st Qu.:  8.25   1st Qu.: 23.800   1st Qu.:  4.920   1st Qu.: 30.2000
##   Median : 19.30   Median : 35.000   Median :  6.320   Median : 43.3000
##   Mean   : 38.27   Mean   : 41.109   Mean   :  6.816   Mean   : 46.8902
##   3rd Qu.: 62.10   3rd Qu.: 51.350   3rd Qu.:  8.600   3rd Qu.: 58.7500
##   Max.   :208.00   Max.   :200.000   Max.   :17.900   Max.   :174.0000
##      revenu      inflation      esper_vie      fert
##   Min.   :   609   Min.   : -4.210   Min.   :32.10   Min.   :1.150
##   1st Qu.: 3355   1st Qu.:  1.810   1st Qu.:65.30   1st Qu.:1.795
##   Median : 9960   Median :  5.390   Median :73.10   Median :2.410
##   Mean   :17145   Mean   :  7.782   Mean   :70.56   Mean   :2.948
##   3rd Qu.:22800   3rd Qu.:10.750   3rd Qu.:76.80   3rd Qu.:3.880
##   Max.   :125000   Max.   :104.000   Max.   :82.80   Max.   :7.490
##      pib_h
##   Min.   :   231
##   1st Qu.: 1330
##   Median : 4660
##   Mean   :12964
##   3rd Qu.:14050
##   Max.   :105000
```

2.2 Pre-traitement

Donnees manquantes ? Outliers

```
table(is.na(donnee))
```

```
##
## FALSE
## 1503
```

Aucune donnée manquante.

Valeur aberrante

Exports max à 200 ? Bizarre

Ce sont à première vue des pays riche comme malte, luxembourg, singapour

Imports max à 174 ? Idem, finalement c'est logique Aucune valeur aberrante

Mais y a des valeurs "leviers", certains pays comme malte, singapour se dégagent des valeurs moyennes

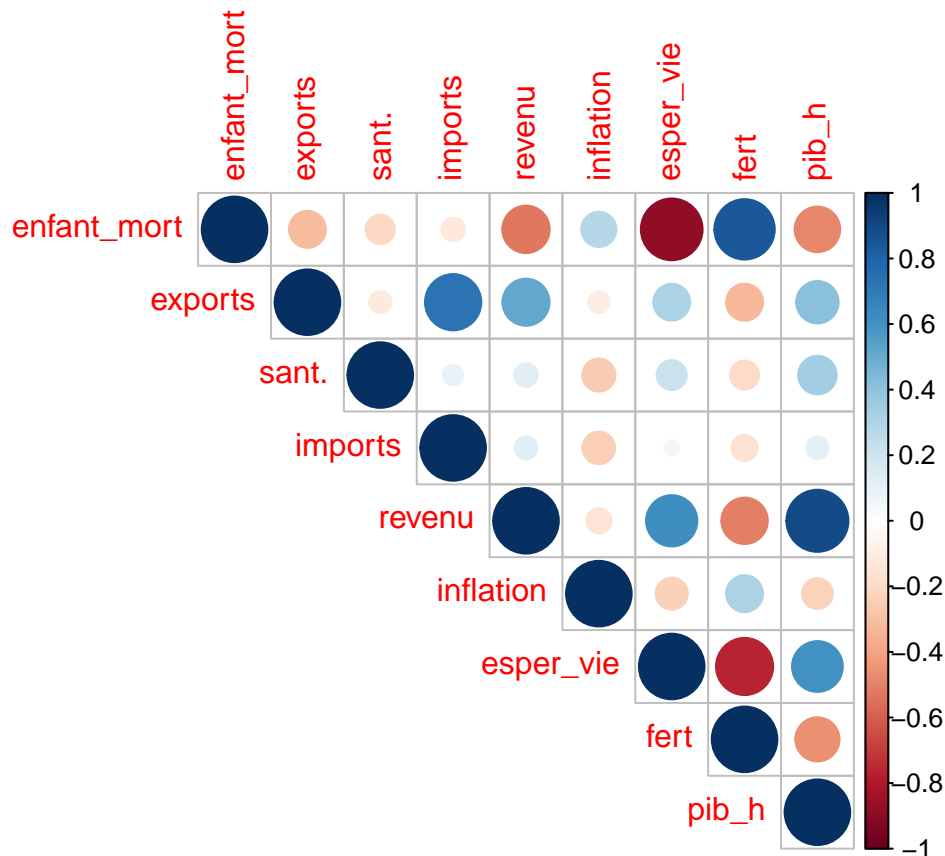
Standardisation ?

Lorsque l'on a des données avec des unités différentes (par exemple des pourcentages, des espérances de vie, des PIB par habitant), il est recommandé de centrer et de réduire ces données. Centrer les données signifie soustraire la moyenne de la variable de toutes les observations, ce qui permet d'avoir une moyenne égale à zéro. Réduire les données signifie diviser chaque observation par l'écart-type de la variable, ce qui met toutes les variables à la même échelle. Cela facilite la comparaison entre les différentes variables et permet des analyses statistiques plus fiables. Il est cependant important de garder à l'esprit que la signification des résultats dépend toujours du contexte et de la validité des données utilisées

```
donnee <- data.frame(scale(donnee))
```

2.3 Matrice de corrélation

```
var <- donnee[,1:9]  
corrplot(cor(var), type = "upper")
```



La matrice de corrélation nous aide à mieux comprendre les relations entre chaque variable et pourra nous aider à interpréter plus tard.

Nous allons alors classer les pays en fonction de NOMS DE COLONNES.

3 Classification des pays en utilisant les différents algorithmes abordés en cours

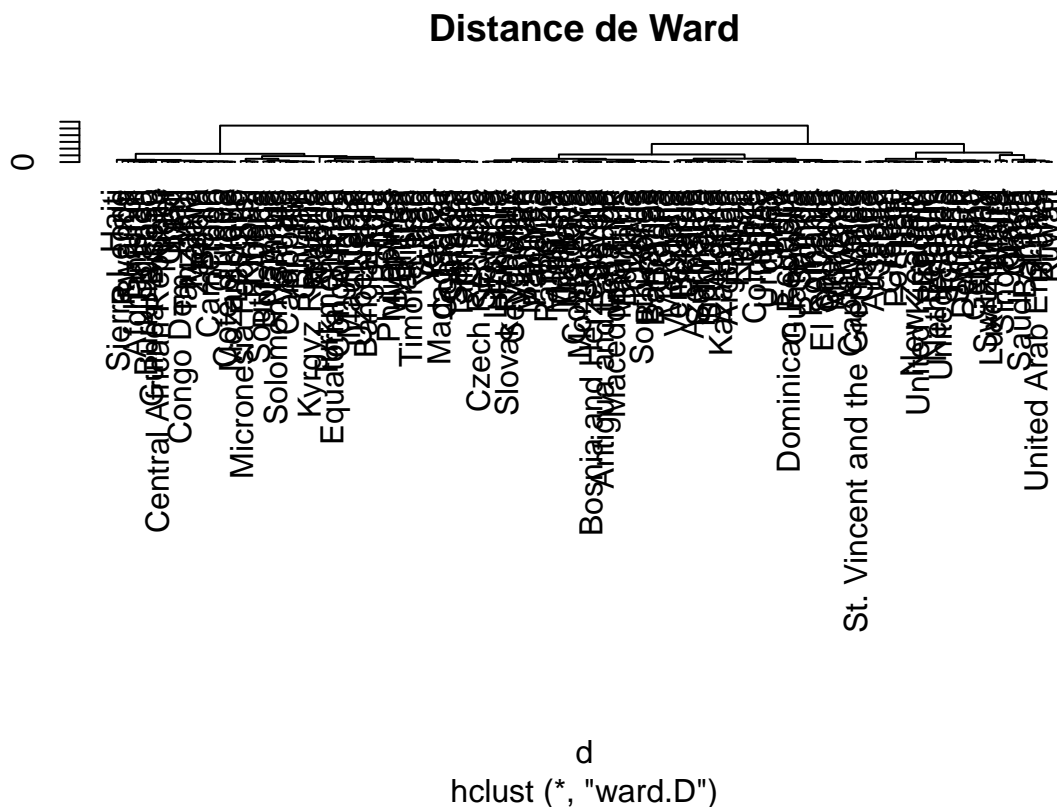
Utilisation des algorithmes de classification vus en cours . Reflexion sur les choix operer decider d'une classification finale . Nombre de groupes ?

3.1 CAH

```
set.seed(123)
d <- dist(donnee)
#d <- dist(e19, method = "manhattan")
#d <- dist(e19, method = "minkowski")
cah.ward <- hclust(d, method = "ward.D")
cah.min <- hclust(d, method = "single")
cah.max <- hclust(d, method = "complete")
```

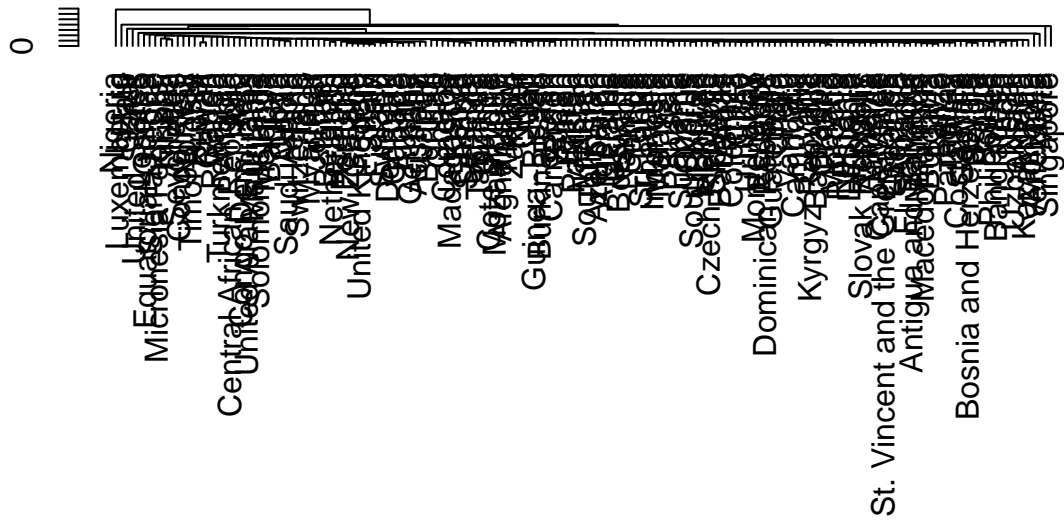
Dengrogrammes

```
plot(cah.ward, hang = -1, main = "Distance de Ward", ylab = " ")
```



```
plot(cah.min, hang = -1, main = "Distance du saut minimal", ylab = " ")
```

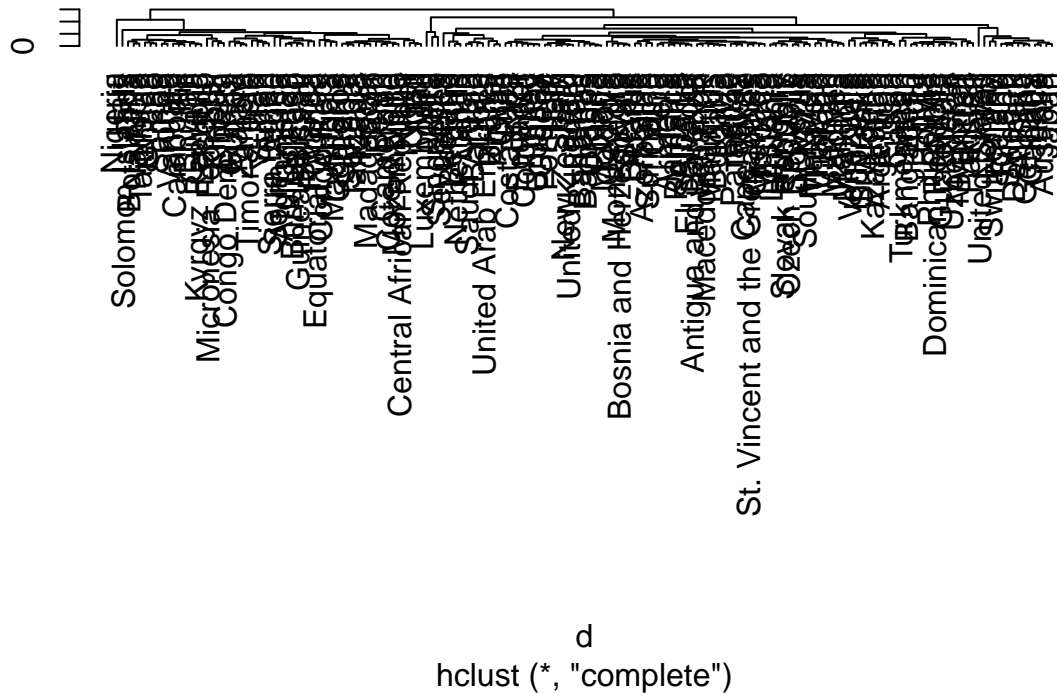
Distance du saut minimal



d
hclust (*, "single")

```
plot(cah.max, hang = -1, main = "Distance du saut maximal", ylab = " ")
```

Distance du saut maximal



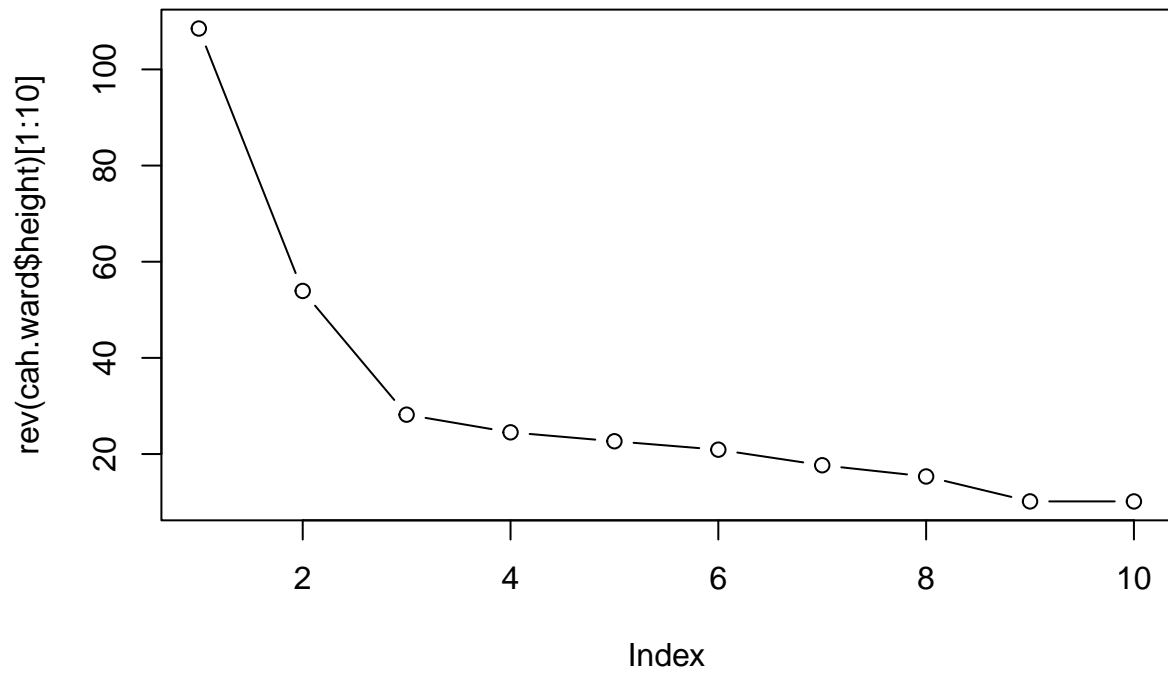
On s'aperçoit rapidement que c'est le critère de Ward qui correspond le mieux à nos données. On voit déjà qu'on peut partitionner nos données en 3 ou 4 groupes

Fonction de perte

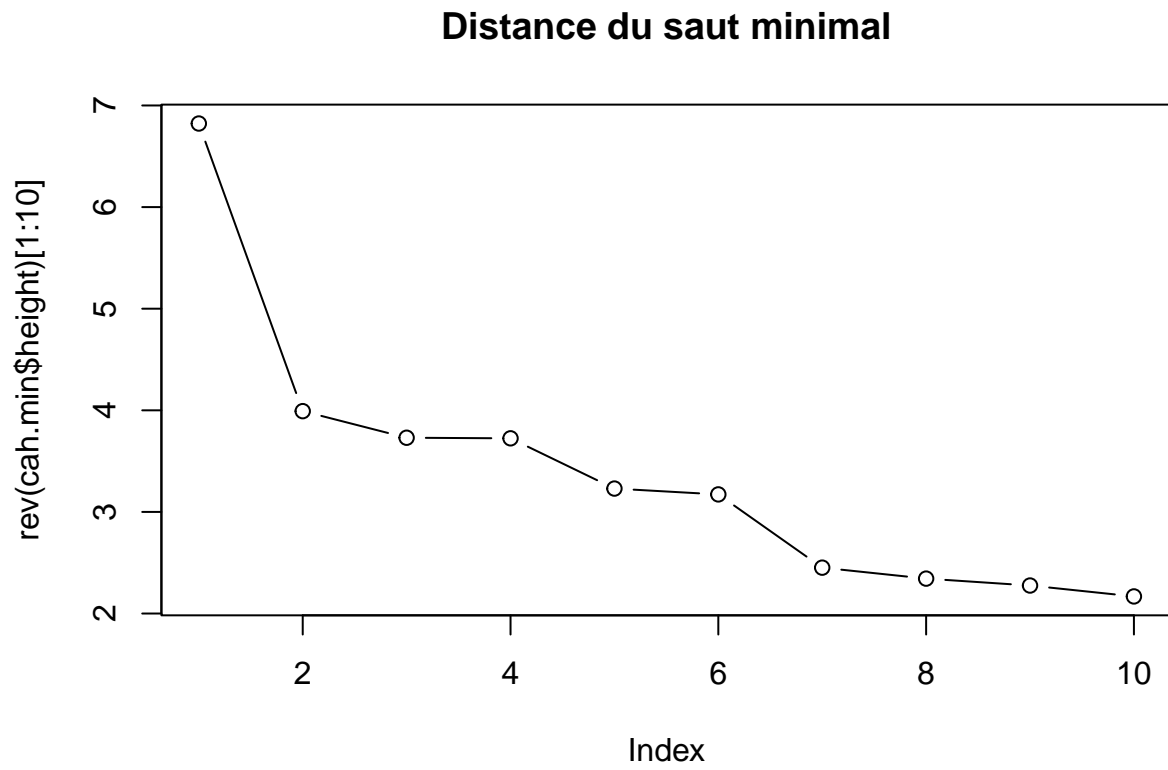
Pour rappel, on cherche à maximiser l'inertie inter-classe. En effet, nous avons pour objectif de créer des groupes d'individus se ressemblant fortement (inertie intra-classes faible) et tels que les groupes soient les plus distincts possible (inertie inter-classes élevée). L'inertie inter-classe est logiquement maximale (égale à l'inertie totale) lorsqu'il y a autant de classes que d'individus. Nous cherchons dans le graphique ci-dessous un "coude" qui correspond à une rupture dans la courbe (moment où l'inertie inter augmente beaucoup).

```
plot(rev(cah.ward$height)[1:10], type = "b", main = "Distance de Ward")
```

Distance de Ward

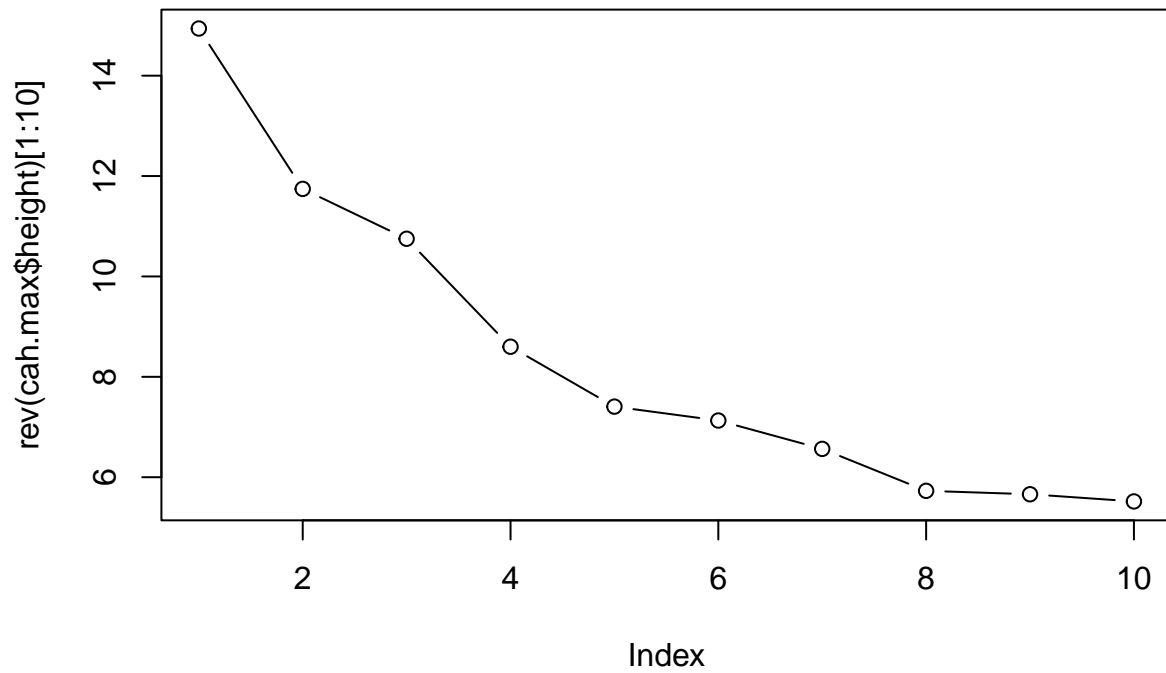


```
plot(rev(cah.min$height)[1:10], type = "b", main = "Distance du saut minimal")
```



```
plot(rev(cah.max$height)[1:10], type = "b", main = "Distance du saut maximal")
```

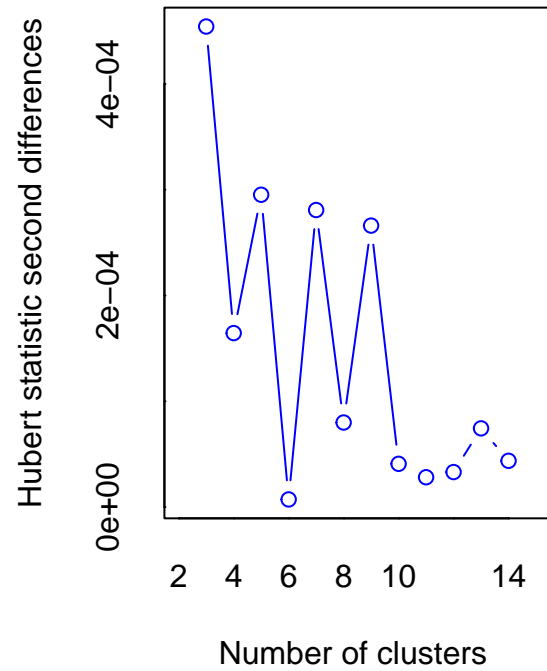
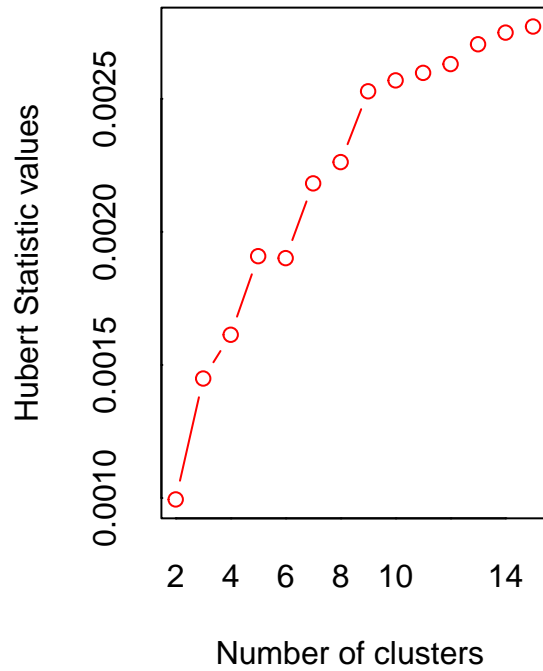

Distance du saut maximal



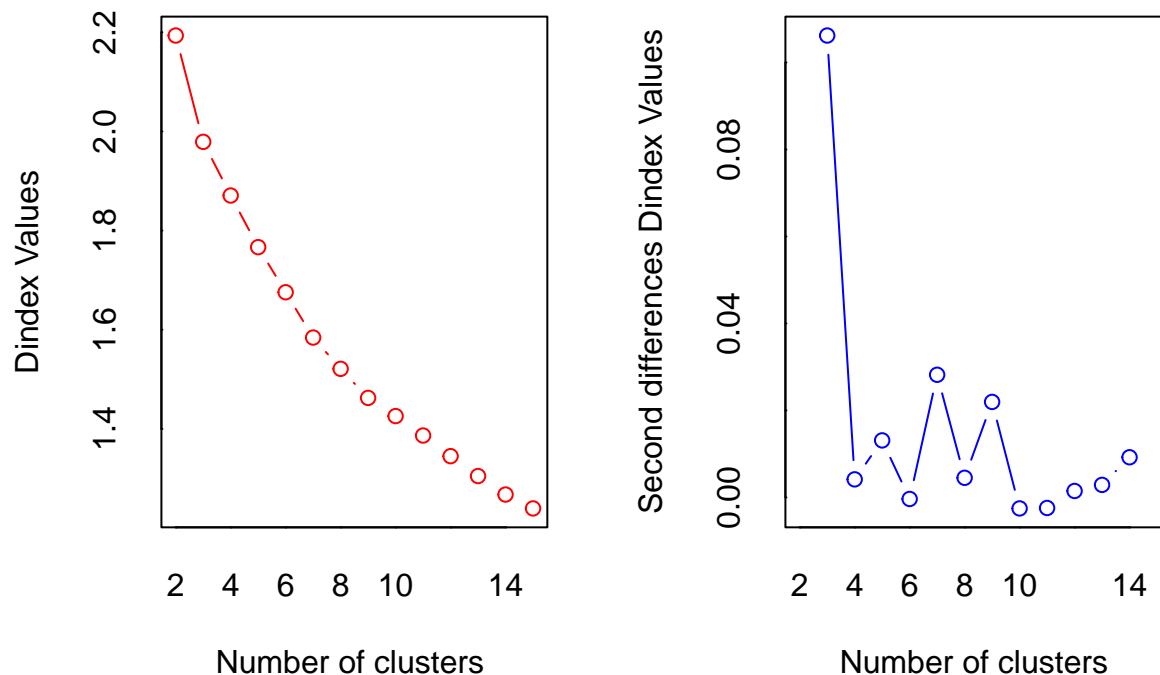
Avec le critère de Ward, la trace de la perte d'inertie nous incite à choisir des partitions en 3 groupes ("coude" très visible).

```
matrix <- as.matrix(donnee)
```

```
NbClust(matrix, min.nc = 2, max.nc = 15, method = "ward.D")
```



```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##       In the plot of Hubert index, we seek a significant knee that corresponds to a
##       significant increase of the value of the measure i.e the significant peak in Hubert
##       index second differences plot.
##
```



```
## *** : The D index is a graphical method of determining the number of clusters.
##           In the plot of D index, we seek a significant knee (the significant peak in Dindex
##           second differences plot) that corresponds to a significant increase of the value of
##           the measure.
##
## *****
## * Among all indices:
## * 5 proposed 2 as the best number of clusters
## * 4 proposed 3 as the best number of clusters
## * 5 proposed 4 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 4 proposed 9 as the best number of clusters
## * 1 proposed 12 as the best number of clusters
## * 1 proposed 14 as the best number of clusters
## * 1 proposed 15 as the best number of clusters
##
##           ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
## *****

## $All.index
##           KL           CH Hartigan           CCC           Scott           Marriot           TrCovW           TraceW
## 2  2.3386 68.6210 33.7214 -2.6434 225.4314 7.313743e+16 23244.901 1055.1703
## 3  1.8427 57.8307 20.9221 -2.4313 418.9626 5.164539e+16 17615.022 876.1166
```

```

## 4  1.5564 50.1387  14.8863 -2.6786  609.3518 2.936225e+16 13591.412  776.9927
## 5  1.1442 44.4853  12.5772 -3.1997  758.7814 1.875028e+16 12060.060  711.9705
## 6  0.2926 40.6143  28.1333 -3.4454  808.6017 2.003593e+16  9447.050  660.6776
## 7  2.0767 44.1722  16.2945  0.1436  962.9977 1.081897e+16  6363.185  562.4029
## 8  0.5419 43.7703  28.2797  1.4605 1070.1097 7.440776e+15  4995.833  510.4213
## 9  4.3721 48.3399   9.3875  5.1119 1220.3724 3.829616e+15  3394.298  433.3464
## 10 1.1997 46.2701   8.1650  5.2609 1293.2347 3.056237e+15  3031.433  409.0434
## 11 0.8340 44.3410   8.9717  5.3083 1376.7840 2.242316e+15  2823.262  388.8221
## 12 1.2044 43.1654   7.8186  5.6184 1447.0356 1.752186e+15  2505.380  367.6767
## 13 0.9742 41.9434   7.8612  5.7962 1506.8698 1.437148e+15  2361.741  350.0208
## 14 1.5395 41.0299   5.8597  6.0549 1560.7539 1.207096e+15  2068.108  333.0212
## 15 1.2204 39.7156   5.0920  6.0050 1614.6635 1.003397e+15  1975.666  320.7374
##      Friedman  Rubin Cindex      DB Silhouette      Duda Pseudot2      Beale Ratkowsky
## 2   16.5255  1.4159  0.2743  1.5019      0.2817  0.7340  36.2418  2.1548    0.3334
## 3   25.3844  1.7053  0.2347  1.5929      0.2289  0.6755  15.3735  2.7975    0.3307
## 4   36.3323  1.9228  0.2246  1.4508      0.2470  0.8298  12.9207  1.2123    0.3286
## 5   40.4147  2.0984  0.2063  1.7295      0.2079  0.7280  24.6545  2.2097    0.3070
## 6   42.7143  2.2613  0.2032  1.7717      0.1599  0.3414  17.3592 10.4241    0.2951
## 7   51.2007  2.6565  0.1987  1.5066      0.1827  0.7968  10.4568  1.4951    0.2928
## 8   54.2108  2.9270  0.1882  1.4706      0.2036  1.0665  -1.6830 -0.3609    0.2832
## 9   56.8841  3.4476  0.3108  1.2326      0.2160  0.7129  10.4715  2.3289    0.2800
## 10  59.8853  3.6524  0.2997  1.2020      0.2206  0.7251  12.1343  2.2081    0.2688
## 11  62.3662  3.8424  0.2914  1.2141      0.2056  0.4977  12.1126  5.5951    0.2587
## 12  64.6251  4.0634  0.2862  1.1860      0.2105  0.6908   9.4000  2.5657    0.2503
## 13  68.3495  4.2683  0.2798  1.2286      0.1915  0.7334  11.6328  2.1168    0.2424
## 14  70.8060  4.4862  0.2721  1.2700      0.1875  0.7291   7.4314  2.1250    0.2353
## 15  72.5606  4.6580  0.2654  1.3223      0.1767  0.5385  13.7138  4.8442    0.2286
##      Ball Ptbiserial      Frey McClain      Dunn Hubert SDindex Dindex      SDbw
## 2  527.5852      0.3422  0.2595  0.6490  0.0751  0.0010  2.7240  2.1934  1.0880
## 3  292.0389      0.4053 -0.1502  1.1448  0.0751  0.0014  2.8973  1.9790  0.9576
## 4  194.2482      0.4357  0.4909  1.1797  0.0757  0.0016  3.0800  1.8707  1.0002
## 5  142.3941      0.4312  5.5517  1.5461  0.0757  0.0019  3.1602  1.7666  0.9660
## 6  110.1129      0.3474 -0.1479  2.5747  0.0685  0.0019  3.1581  1.6756  0.7597
## 7   80.3433      0.3571  0.1256  2.5488  0.0685  0.0022  3.1016  1.5843  0.6454
## 8   63.8027      0.3659 -0.1291  2.8953  0.0717  0.0023  3.1233  1.5211  0.6410
## 9   48.1496      0.3835  0.1872  2.8069  0.1221  0.0025  2.9451  1.4625  0.4967
## 10  40.9043      0.3829  0.8481  2.9427  0.1221  0.0026  2.9247  1.4258  0.4672
## 11  35.3475      0.3608  0.0927  3.4443  0.1221  0.0026  3.1190  1.3866  0.4425
## 12  30.6397      0.3617  0.4867  3.4868  0.1221  0.0026  3.0365  1.3450  0.4064
## 13  26.9247      0.3521  1.0162  3.7714  0.1154  0.0027  2.9761  1.3048  0.3864
## 14  23.7872      0.3208  0.4286  4.6963  0.1154  0.0027  3.4028  1.2676  0.3676
## 15  21.3825      0.3095  1.4422  5.1586  0.1154  0.0028  3.4239  1.2396  0.3519
##
## $All.CriticalValues
##      CritValue_Duda CritValue_PseudoT2 Fvalue_Beale
## 2           0.7868           27.0994           0.0231
## 3           0.6825           14.8875           0.0037
## 4           0.7508           20.9124           0.2845
## 5           0.7548           21.4445           0.0200
## 6           0.4954            9.1671           0.0000
## 7           0.7098           16.7607           0.1478
## 8           0.6621           13.7821           1.0000
## 9           0.6573           13.5542           0.0158
## 10          0.6825           14.8875           0.0216
## 11          0.5447           10.0311           0.0000
## 12          0.6292           12.3746           0.0083

```

```

## 13      0.6825      14.8875      0.0282
## 14      0.6225      12.1297      0.0296
## 15      0.5901      11.1141      0.0000
##
## $Best.nc
##           KL      CH Hartigan      CCC      Scott      Marriot      TrCovW
## Number_clusters 9.0000 2.000 9.0000 14.0000 3.0000 5.000000e+00 3.000
## Value_Index    4.3721 68.621 18.8923 6.0549 193.5312 1.189761e+16 5629.879
##           TraceW Friedman  Rubin Cindex      DB Silhouette  Duda
## Number_clusters 3.00 4.000 9.0000 8.0000 12.000 2.0000 4.0000
## Value_Index    79.93 10.948 -0.3158 0.1882 1.186 0.2817 0.8298
##           PseudoT2 Beale Ratkowsky      Ball PtBiserial Frey McClain
## Number_clusters 4.0000 4.0000 2.0000 3.0000 4.0000 1 2.000
## Value_Index    12.9207 1.2123 0.3334 235.5463 0.4357 NA 0.649
##           Dunn Hubert SDindex Dindex      SDbw
## Number_clusters 9.0000 0 2.000 0 15.0000
## Value_Index    0.1221 0 2.724 0 0.3519
##
## $Best.partition
##           Afghanistan      Albania
##           1 2
##           Algeria      Angola
##           2 1
##           Antigua and Barbuda      Argentina
##           2 2
##           Armenia      Australia
##           2 2
##           Austria      Azerbaijan
##           2 2
##           Bahamas      Bahrain
##           2 2
##           Bangladesh      Barbados
##           1 2
##           Belarus      Belgium
##           2 2
##           Belize      Benin
##           2 1
##           Bhutan      Bolivia
##           1 1
##           Bosnia and Herzegovina      Botswana
##           2 1
##           Brazil      Brunei
##           2 2
##           Bulgaria      Burkina Faso
##           2 1
##           Burundi      Cambodia
##           1 1
##           Cameroon      Canada
##           1 2
##           Cape Verde      Central African Republic
##           2 1
##           Chad      Chile
##           1 2
##           China      Colombia
##           2 2
##           Comoros      Congo Dem. Rep.

```

##	1	1
##	Congo Rep.	Costa Rica
##	1	2
##	Cote d'Ivoire	Croatia
##	1	2
##	Cyprus	Czech Republic
##	2	2
##	Denmark	Dominican Republic
##	2	2
##	Ecuador	Egypt
##	2	1
##	El Salvador	Equatorial Guinea
##	2	1
##	Eritrea	Estonia
##	1	2
##	Fiji	Finland
##	1	2
##	France	Gabon
##	2	1
##	Gambia	Georgia
##	1	2
##	Germany	Ghana
##	2	1
##	Greece	Grenada
##	2	2
##	Guatemala	Guinea
##	2	1
##	Guinea-Bissau	Guyana
##	1	1
##	Haiti	Hungary
##	1	2
##	Iceland	India
##	2	1
##	Indonesia	Iran
##	2	2
##	Iraq	Ireland
##	1	2
##	Israel	Italy
##	2	2
##	Jamaica	Japan
##	2	2
##	Jordan	Kazakhstan
##	2	2
##	Kenya	Kiribati
##	1	1
##	Kuwait	Kyrgyz Republic
##	2	1
##	Lao	Latvia
##	1	2
##	Lebanon	Lesotho
##	2	1
##	Liberia	Libya
##	1	2
##	Lithuania	Luxembourg
##	2	2
##	Macedonia FYR	Madagascar

##	2	1
##	Malawi	Malaysia
##	1	2
##	Maldives	Mali
##	2	1
##	Malta	Mauritania
##	2	1
##	Mauritius	Micronesia Fed. Sts.
##	2	1
##	Moldova	Mongolia
##	2	2
##	Montenegro	Morocco
##	2	2
##	Mozambique	Myanmar
##	1	1
##	Namibia	Nepal
##	1	1
##	Netherlands	New Zealand
##	2	2
##	Niger	Nigeria
##	1	1
##	Norway	Oman
##	2	2
##	Pakistan	Panama
##	1	2
##	Paraguay	Peru
##	2	2
##	Philippines	Poland
##	1	2
##	Portugal	Qatar
##	2	2
##	Romania	Russia
##	2	2
##	Rwanda	Samoa
##	1	2
##	Saudi Arabia	Senegal
##	2	1
##	Serbia	Seychelles
##	2	2
##	Sierra Leone	Singapore
##	1	2
##	Slovak Republic	Slovenia
##	2	2
##	Solomon Islands	South Africa
##	1	1
##	South Korea	Spain
##	2	2
##	Sri Lanka St. Vincent and the Grenadines	
##	2	2
##	Sudan	Suriname
##	1	2
##	Sweden	Switzerland
##	2	2
##	Tajikistan	Tanzania
##	1	1
##	Thailand	Timor-Leste

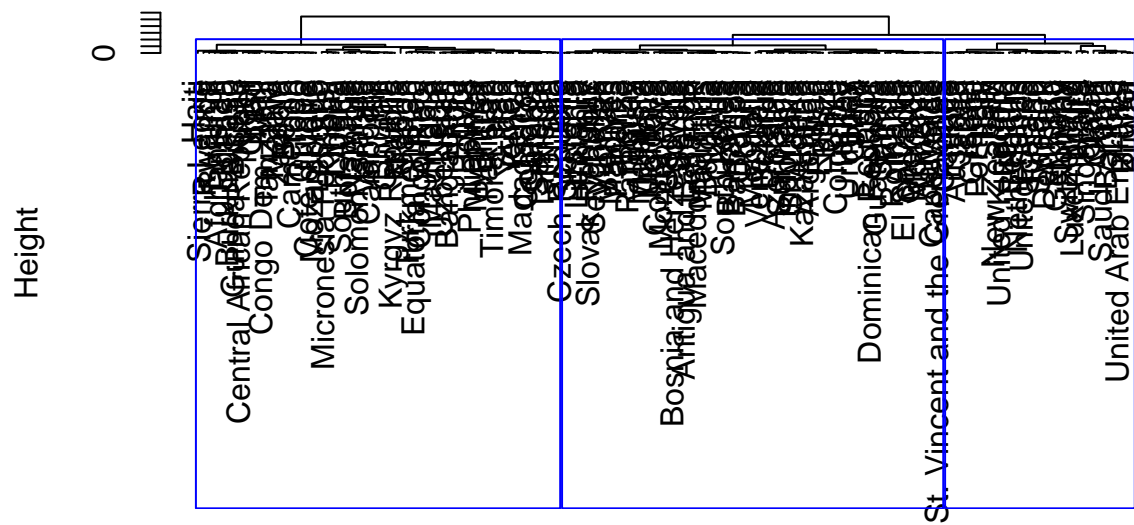
##	2	1
##	Togo	Tonga
##	1	2
##	Tunisia	Turkey
##	2	2
##	Turkmenistan	Uganda
##	1	1
##	Ukraine	United Arab Emirates
##	2	2
##	United Kingdom	United States
##	2	2
##	Uruguay	Uzbekistan
##	2	1
##	Vanuatu	Venezuela
##	1	2
##	Vietnam	Yemen
##	2	1
##	Zambia	
##	1	

Voir quand on knit en html pour l'interprétation

Cutree

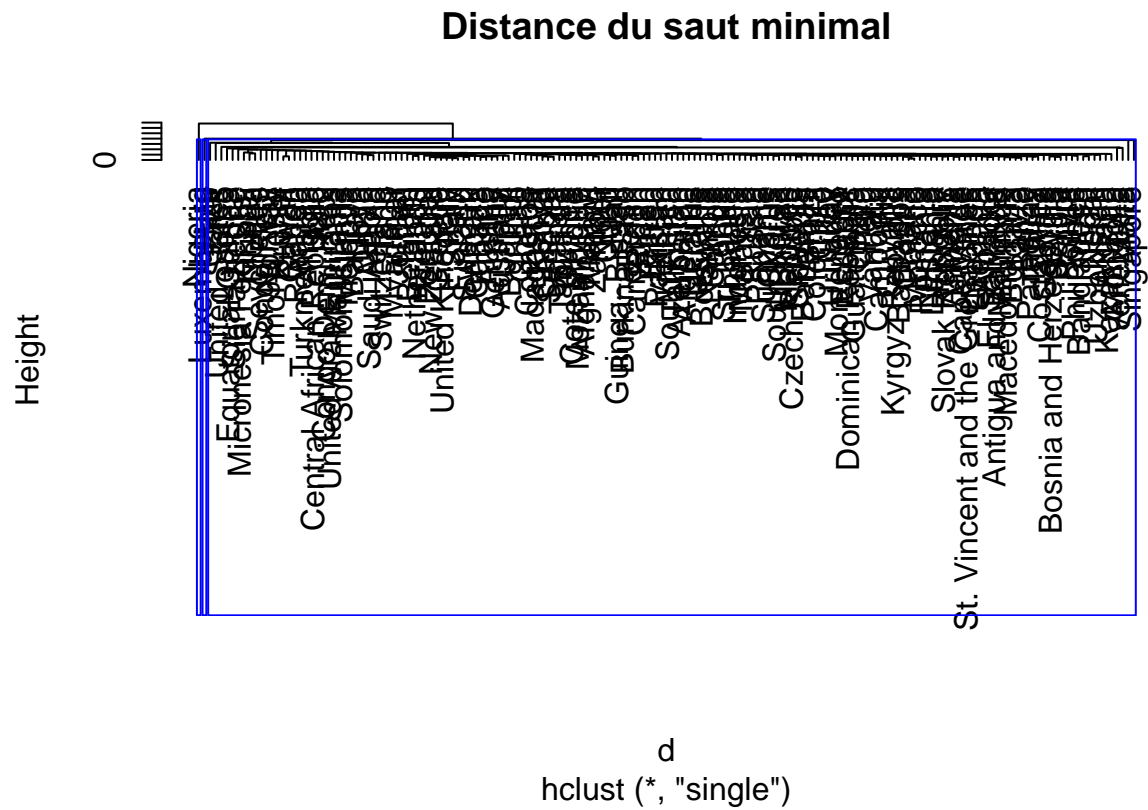
```
nbc <- 3
gpe.ward <- cutree(cah.ward, k = nbc) # Classe affectée pour chaque individu
gpe.min <- cutree(cah.min, k = nbc)
gpe.max <- cutree(cah.max, k = nbc)
plot(cah.ward, hang = -1, main = "Distance de Ward")
rect.hclust(cah.ward, nbc, border = "blue")
```


Distance de Ward



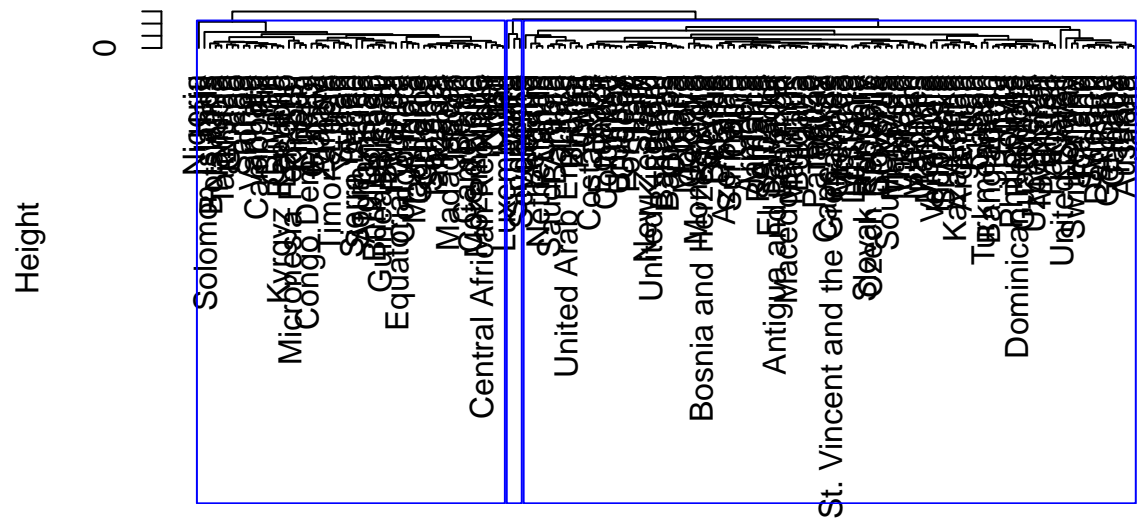
```
hclust (*, "ward.D")
```

```
plot(cah.min, hang = -1, main = "Distance du saut minimal")
rect.hclust(cah.min, nbc, border = "blue")
```



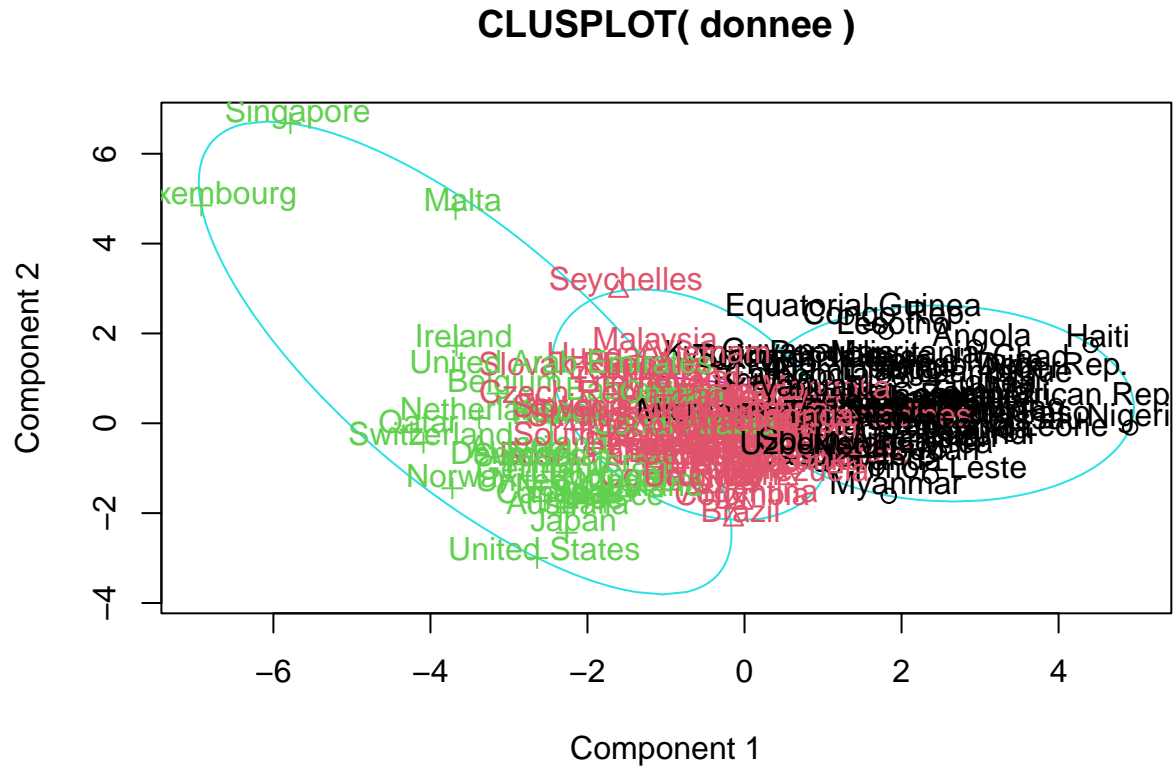
```
plot(cah.max, hang = -1, main = "Distance du saut maximal")
rect.hclust(cah.max, nbc, border = "blue")
```

Distance du saut maximal



d
hclust (*, "complete")

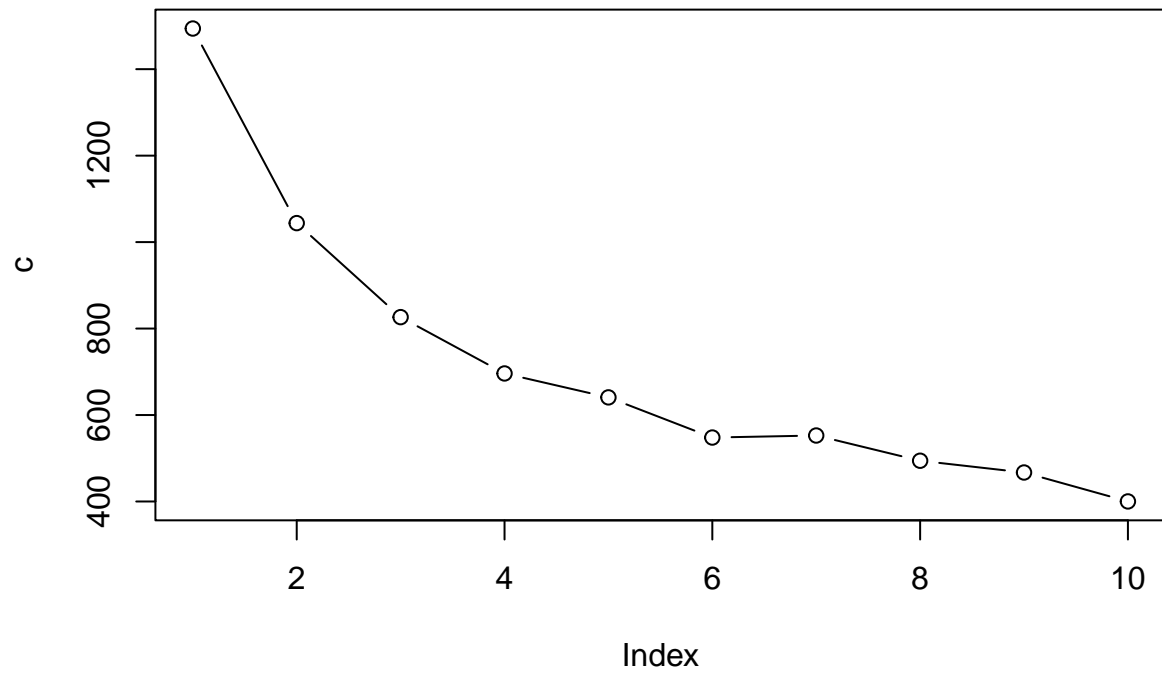
```
clusplot(donnee, gpe.ward, labels = nbc, col.p = as.numeric(gpe.ward))
```



These two components explain 63.13 % of the point variability.

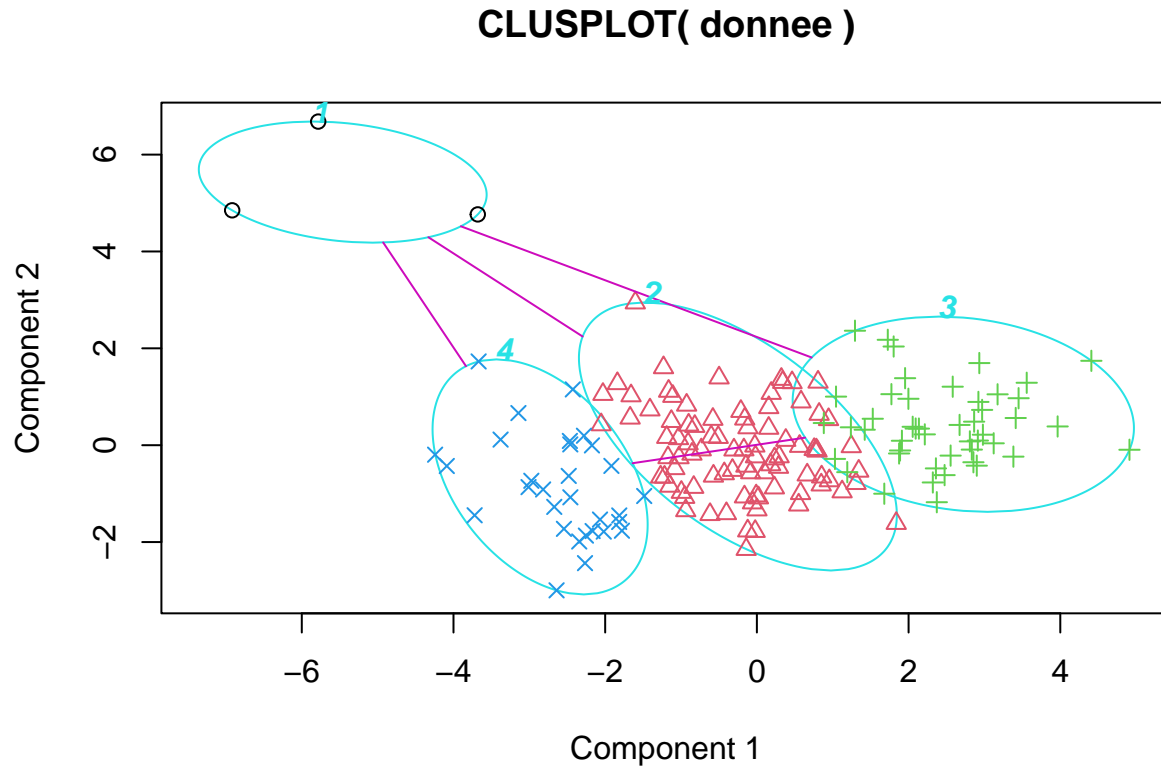
3.2 Algoritme des Kmeans

Tout d'abord nous allons utiliser l'algorithme des k-means pour avoir une première idée de notre classification finale. Si on ne sait pas a priori combien de groupes comporte le jeu de données, on peut appliquer l'algorithme pour plusieurs choix de K possibles et tracer la courbe d'évolution de l'inertie . On lance l'algorithme des kmeans et on observe l'évolution de la variance intra-groupes en fonction du nombre de groupes. On rajoute également l'option « nstart =50 » pour stabiliser les résultats.



A la vue de ce graphique, on aurait tendance à choisir $K=3,4$ ou 5 groupes en appliquant la méthode dite « du coude »

```
K=4
cl = kmeans(donnee,K,nstart=50)
gpe = cl$cluster
clusplot(donnee,gpe,labels=4,col.p=gpe)
```



These two components explain 63.13 % of the point variability.

La représentation en clusplot nous permet de voir qu'il y a 4 groupes qui se séparent plutôt bien sur les composante 1, 2, 3 et 4. (on le voit au travers des différents couleur sur le graphique).

Représentation des groupes sur le premier plan factoriel

3.3 Interprétation des groupes

Nous allons maintenant chercher à interpréter les groupes obtenus à l'aide de la fonction catdes.

```
gpe = cutree(cah.ward,k=3)
donnees$gpecah = as.factor(gpe)
interpcah = catdes(donnee,num.var = 10)
interpcah

##
## Link between the cluster variable and the quantitative variables
## =====
##               Eta2      P-value
## pib_h         0.72363077 1.592773e-46
## esper_vie     0.71468093 2.173330e-45
## revenu        0.69791718 2.346035e-43
## enfant_mort   0.65580744 1.041649e-38
## fert          0.62232782 2.105862e-35
## exports       0.13621101 6.101387e-06
## sant.         0.10399187 1.229080e-04
## inflation     0.05970019 6.424517e-03
##
## Description of each cluster by quantitative variables
```

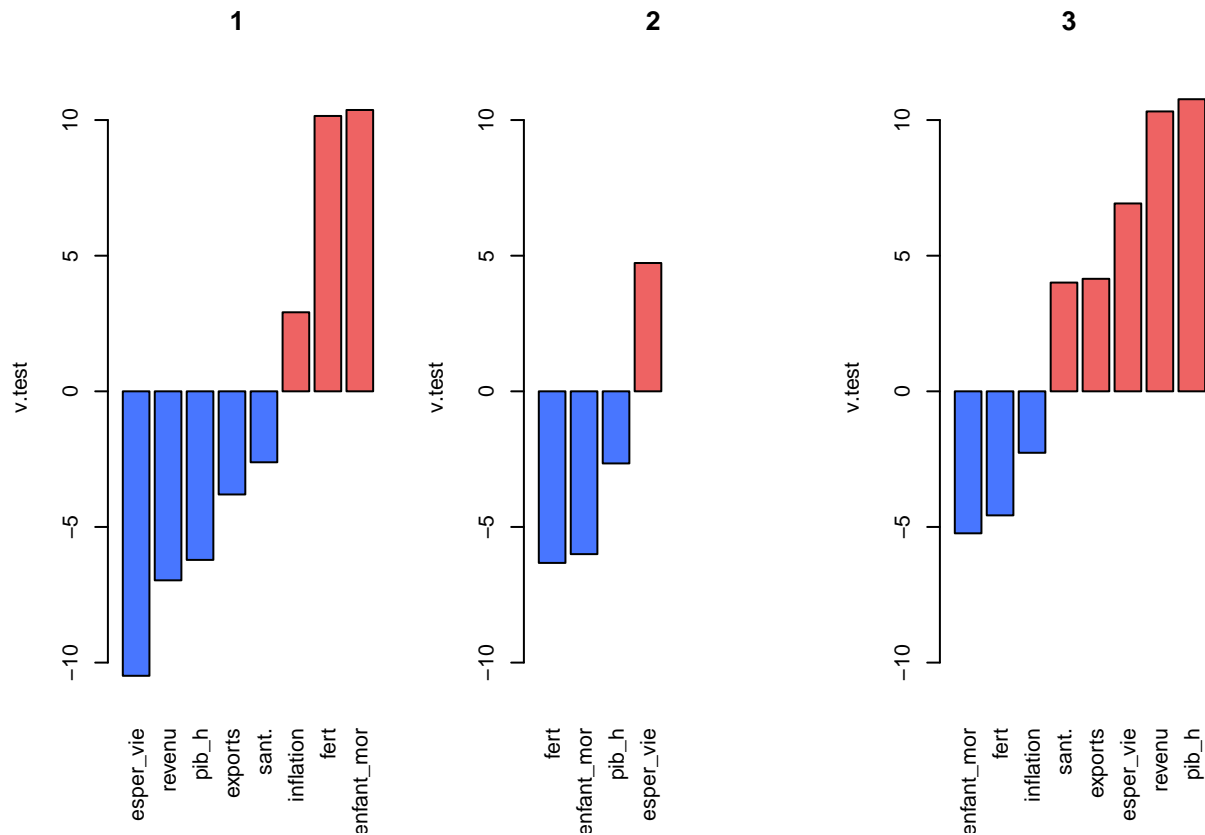
```
## =====
## $'1'
##          v.test Mean in category Overall mean sd in category Overall sd
## enfant_mort 10.370649      1.0052894 1.555642e-16      0.9111885 0.9970015
## fert        10.150674      0.9839659 1.728491e-17      0.8579969 0.9970015
## inflation    2.913863      0.2824583 1.329608e-17      1.2650602 0.9970015
## sant.        -2.615632     -0.2535490 -1.403069e-15      0.9651035 0.9970015
## exports      -3.802046     -0.3685552 -3.478588e-16      0.6775757 0.9970015
## pib_h        -6.214645     -0.6024230 2.393295e-17      0.1393288 0.9970015
## revenu       -6.968102     -0.6754601 -7.445807e-17      0.2556201 0.9970015
## esper_vie    -10.486756    -1.0165444 3.616535e-16      0.7596372 0.9970015
##          p.value
## enfant_mort 3.372265e-25
## fert        3.290824e-24
## inflation    3.569862e-03
## sant.        8.906240e-03
## exports      1.435063e-04
## pib_h        5.144073e-10
## revenu       3.212442e-12
## esper_vie    9.938179e-26
##
## $'2'
##          v.test Mean in category Overall mean sd in category Overall sd
## esper_vie    4.730693      0.4417023 3.616535e-16      0.3438267 0.9970015
## pib_h        -2.657982     -0.2481744 2.393295e-17      0.3297308 0.9970015
## enfant_mort  -6.001733     -0.5603786 1.555642e-16      0.2069384 0.9970015
## fert        -6.325848     -0.5906411 1.728491e-17      0.4173776 0.9970015
##          p.value
## esper_vie    2.237542e-06
## pib_h        7.861014e-03
## enfant_mort  1.952226e-09
## fert        2.518459e-10
##
## $'3'
##          v.test Mean in category Overall mean sd in category Overall sd
## pib_h        10.768111      1.6480397 2.393295e-17      1.04631502 0.9970015
## revenu       10.318192      1.5791804 -7.445807e-17      1.05150950 0.9970015
## esper_vie     6.925852      1.0599890 3.616535e-16      0.23111362 0.9970015
## exports       4.147405      0.6347527 -3.478588e-16      1.54724683 0.9970015
## sant.         4.009263      0.6136103 -1.403069e-15      1.27746390 0.9970015
## inflation    -2.266208     -0.3468390 1.329608e-17      0.50958288 0.9970015
## fert        -4.572607     -0.6998292 1.728491e-17      0.29691512 0.9970015
## enfant_mort  -5.234428     -0.8011195 1.555642e-16      0.08690297 0.9970015
##          p.value
## pib_h        4.868867e-27
## revenu       5.831062e-25
## esper_vie    4.333583e-12
## exports      3.362651e-05
## sant.        6.090858e-05
## inflation    2.343864e-02
## fert        4.816923e-06
## enfant_mort  1.654969e-07
```

```
head(donnee)
```

```
##          enfant_mort      exports      sant.      imports      revenu
```

## Afghanistan	1.2876597	-1.13486665	0.27825140	-0.08220771	-0.80582187
## Albania	-0.5373329	-0.47822017	-0.09672528	0.07062429	-0.37424335
## Algeria	-0.2720146	-0.09882442	-0.96317624	-0.63983800	-0.22018227
## Angola	2.0017872	0.77305618	-1.44372888	-0.16481961	-0.58328920
## Antigua and Barbuda	-0.6935483	0.16018613	-0.28603389	0.49607554	0.10142673
## Argentina	-0.5894047	-0.81019144	0.46756001	-1.27594958	0.08067776
##	inflation	esper_vie	fert	pib_h	gpecah
## Afghanistan	0.1568645	-1.6142372	1.89717646	-0.67714308	1
## Albania	-0.3114109	0.6459238	-0.85739418	-0.48416709	2
## Algeria	0.7869076	0.6684130	-0.03828924	-0.46398018	2
## Angola	1.3828944	-1.1756985	2.12176975	-0.51472026	1
## Antigua and Barbuda	-0.5999442	0.7021467	-0.54032130	-0.04169175	2
## Argentina	1.2409928	0.5897009	-0.38178486	-0.14535428	2

```
plot.catdes(interpcah,barplot=T)
```

Les 3 groupes sont donc caractérisés ainsi :

- Le premier groupe a une très faible espérance de vie, un faible revenu, un faible pib, et un fort taux de fertilité et de mortalité infantile.
- Le second groupe se démarque déjà très largement du premier. En effet, il a un faible taux de mort infantile et une haute espérance de vie. Il a cependant un pib par habitant plutôt faible, mais toujours moins que le premier groupe.
- Le troisième groupe se démarque également du deuxième groupe : il a un très fort pib par habitant, de forts revenus.

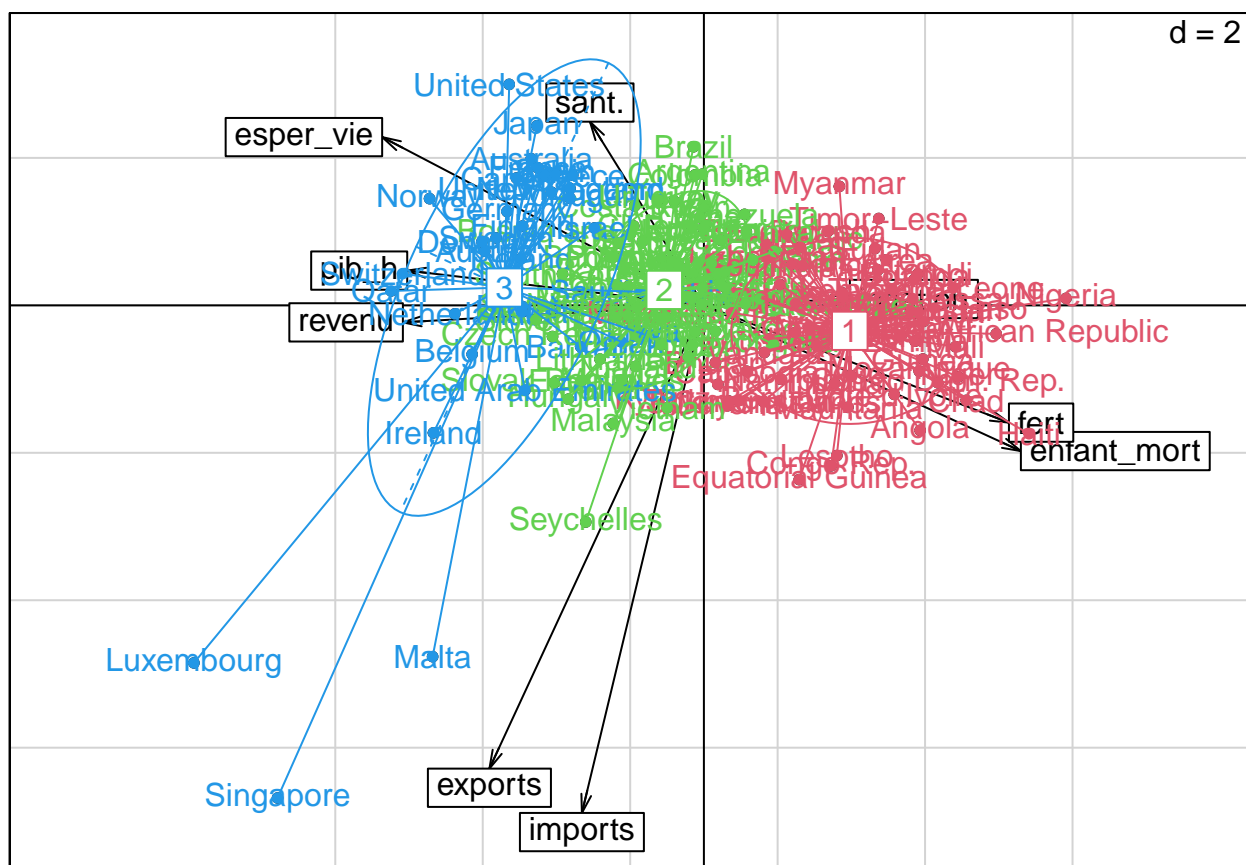
```
CCpca = dudi.pca(donnee[1:9],scannf=FALSE,nf=2)
cumsum(CCpca$eig)/sum(CCpca$eig) # 63% de variabilité expliquée sur les deux premiers axes

## [1] 0.4595174 0.6313337 0.7613762 0.8719079 0.9453100 0.9701523 0.9827566
## [8] 0.9925694 1.0000000

scatter(CCpca,posieig = "none",clab.row=0,pch=NA)

## NULL

text(CCpca$li[,1], CCpca$li[,2],labels = row.names(donnee),col=gpe+1,xpd=TRUE)
s.class(CCpca$li, factor(gpe), col = 2:4, add.plot = TRUE,clabel = 1)
```



Suite à l'analyse de nos différentes méthodes, nous nous rendons compte que 3 gros groupes se sont formés. Nous décidons de nous concentrer sur le groupe des pays les moins développés.

3.3.1 Partie 3 : Agrégation autour des centres mobiles

Classification mixte

Classification finale

4 Traitement du groupe des pays les moins favorisés

Caractérisation de la partition obtenue Représentation informative des résultats. Graphiques adaptés, représentations factorielles si adaptées Optionnel : Représentation spatiale des résultats sur la carte de Rennes Faire une ACP

Nous allons maintenant uniquement nous pencher sur les pays les moins développés (ceux appartenant au premier groupe).

4.0.1 Partie 1 : CAH sur les pays moins développés

On décide de réaliser une deuxième CAH sur le groupe 1, qui sont les pays moins développés :

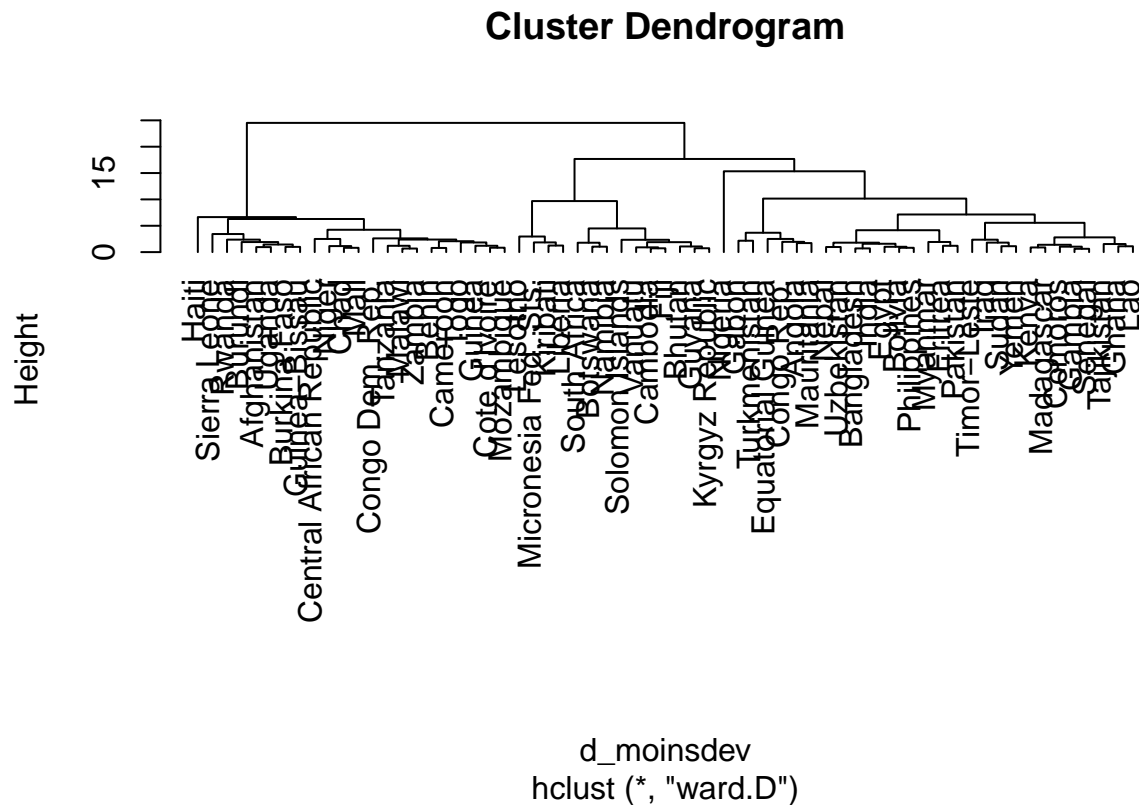
```
donnee_groupe <- donnee
donnee_groupe$gpecah <- as.factor(gpe.ward)
donnee_moinsdev <- donnee_groupe[donnee_groupe$gpecah ==1,]
donnee_moinsdev <-donnee_moinsdev[1:9]
```

On enlève la dernière colonne qui ne nous sert plus à rien.

On décide d'appliquer une CAH sur ces données avec la distance euclidienne et la stratégie d'aggrégation de ward (au vue du travail effectué plus haut c'est ce qui nous semble le plus pertinent)

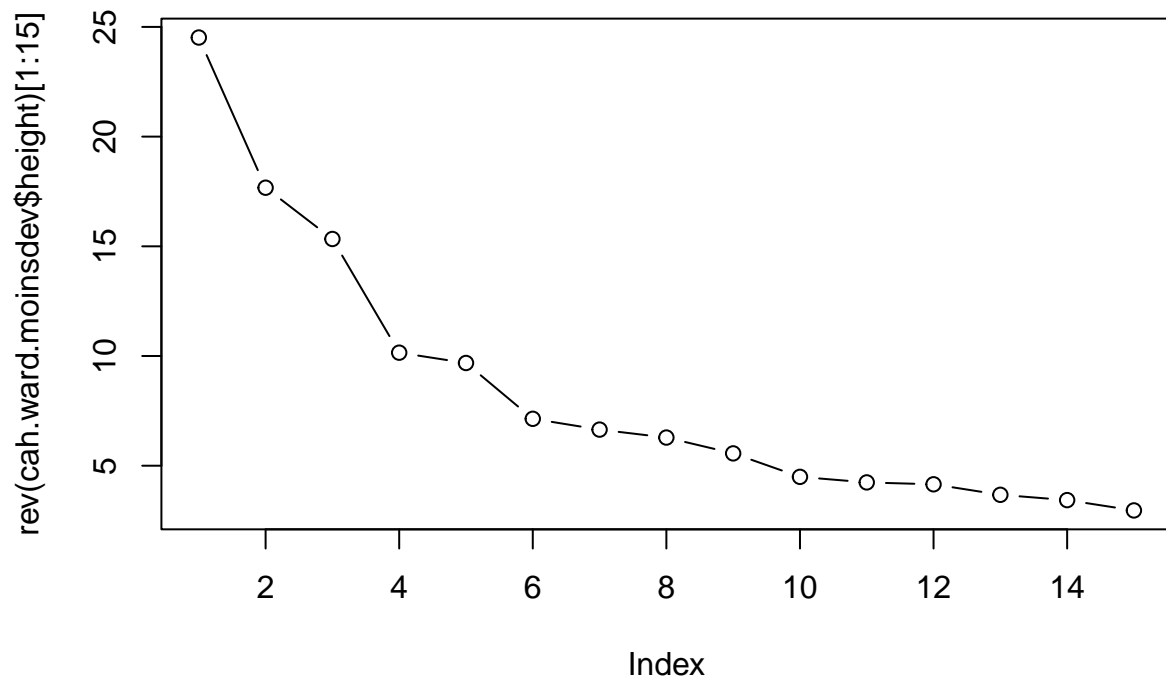
```
d_moinsdev = dist(donnee_moinsdev)
cah.ward.moinsdev = hclust(d_moinsdev,method="ward.D")

plot(cah.ward.moinsdev,hang=-1)
```



De la même façon que la seconde partie, on observe la présence d'une structure "naturelle" en un nombre de groupe modéré. Regardons la courbe de perte d'inertie (on se contente des 15 premières valeurs pour ne pas "noyer" l'information importante)

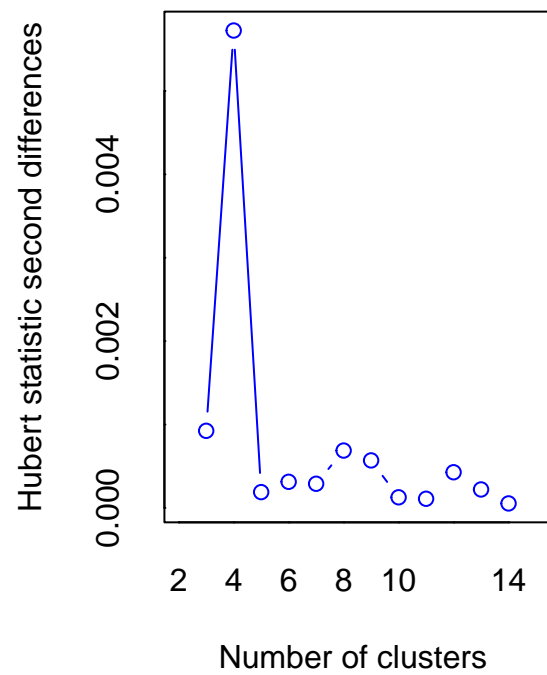
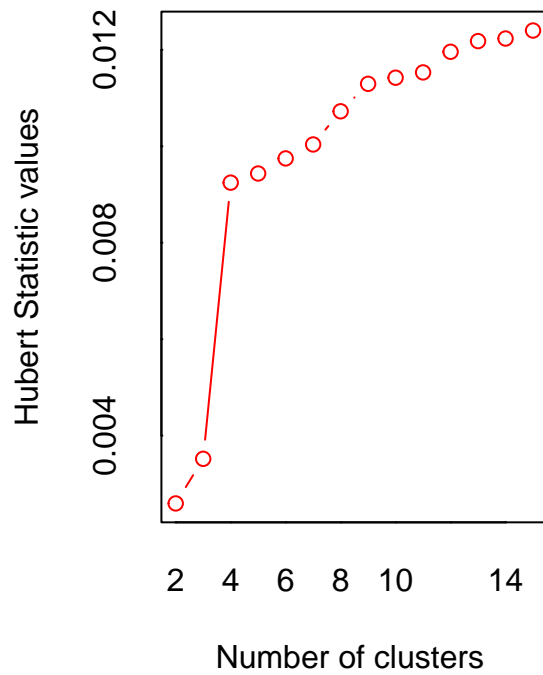
```
plot(rev(cah.ward.moinsdev$height)[1:15],type="b")
```



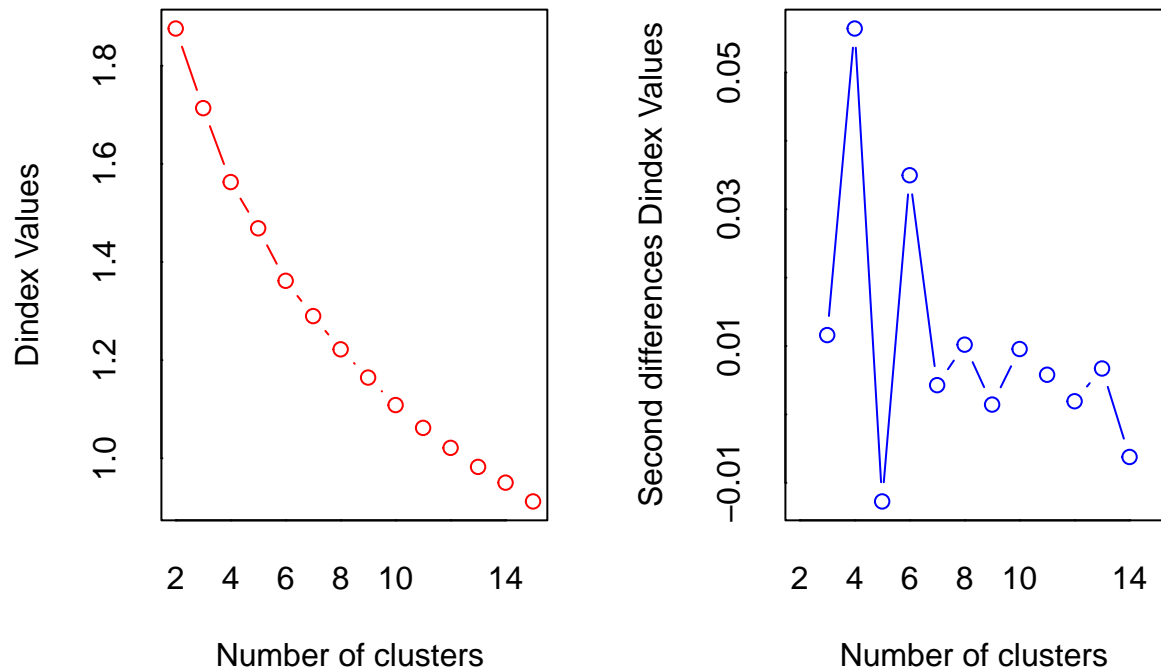
Le tracé de la perte d'inertie nous incite à choisir une partition en 4 groupes (lecture de gauche à droite : juste avant le coude ou changement de pente s'opérant au passage de 4 à 6 groupes)

On peut aussi s'aider de critères automatiques calculés dans le package NbClust

```
NbClust(donnee_moinsdev,min.nc = 2,max.nc = 15,method="ward.D",index="all")
```



```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##       In the plot of Hubert index, we seek a significant knee that corresponds to a
##       significant increase of the value of the measure i.e the significant peak in Hubert
##       index second differences plot.
##
```



```
## *** : The D index is a graphical method of determining the number of clusters.
##           In the plot of D index, we seek a significant knee (the significant peak in Dindex
##           second differences plot) that corresponds to a significant increase of the value of
##           the measure.
##
## *****
## * Among all indices:
## * 4 proposed 2 as the best number of clusters
## * 2 proposed 3 as the best number of clusters
## * 8 proposed 4 as the best number of clusters
## * 3 proposed 5 as the best number of clusters
## * 1 proposed 6 as the best number of clusters
## * 5 proposed 15 as the best number of clusters
##
##           ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is 4
##
## *****

## $All.index
##           KL      CH Hartigan      CCC      Scott      Marriot      TrCovW      TraceW
## 2  0.3883 12.9207 12.3551 2.5877 363.0264 13911997237 2553.4315 317.0411
## 3  0.3814 13.6842 25.4204 1.9072 440.1236 9559850713 1654.2922 265.0596
## 4  3.7956 20.9925 9.0571 5.4259 555.6019 2875846038 702.5026 187.9847
## 5  0.9972 20.0128 8.9011 6.0732 626.1356 1518160273 563.6929 163.6817
## 6  1.8912 19.8295 5.5642 6.7570 674.7096 1035465881 397.5363 142.5362
```

```

## 7 0.5562 18.6881 8.6302 6.8218 713.6353 774370946 351.8538 130.2524
## 8 1.3731 19.2962 6.8707 7.8378 773.6406 401800021 256.0183 113.3816
## 9 1.3054 19.4312 5.6523 8.4255 821.4176 243834537 187.1099 101.1850
## 10 1.3948 19.2928 4.3904 8.7362 867.9183 147203472 152.6398 91.9083
## 11 1.1455 18.8397 3.9477 8.7416 907.4463 96961404 129.8641 85.1141
## 12 0.9719 18.3910 4.0353 8.6853 935.3844 75077386 106.0332 79.3156
## 13 0.8223 18.1296 4.8131 8.7214 965.4181 55509159 88.7523 73.7040
## 14 1.1920 18.2955 4.2511 9.0531 991.5743 43049889 73.2914 67.4599
## 15 1.2670 18.3416 3.5485 9.2532 1036.6613 24697458 64.3842 62.2694
## Friedman Rubin Cindex DB Silhouette Duda Pseudot2 Beale Ratkowsky
## 2 474.1703 2.0505 0.2210 1.9433 0.1817 0.7968 10.4568 1.4951 0.2700
## 3 531.4995 2.4526 0.2011 1.8086 0.1960 1.0665 -1.6830 -0.3609 0.2980
## 4 544.3378 3.4582 0.3426 1.2290 0.2243 0.7129 10.4715 2.3289 0.2933
## 5 608.6435 3.9717 0.3089 1.1433 0.2397 0.4977 12.1126 5.5951 0.3139
## 6 619.1856 4.5609 0.2926 1.0923 0.2419 0.7291 7.4314 2.1250 0.3023
## 7 635.0339 4.9910 0.2841 1.2445 0.2118 1.3568 -5.2600 -1.5041 0.2869
## 8 646.2607 5.7337 0.3852 1.1287 0.2372 0.7251 7.2019 2.1624 0.2780
## 9 688.6921 6.4248 0.3854 1.1520 0.2286 0.5549 8.0228 4.3797 0.2673
## 10 707.5520 7.0733 0.3755 1.0662 0.2215 0.5265 7.1948 4.8005 0.2574
## 11 716.4159 7.6379 0.3635 1.0335 0.2291 0.7197 4.6729 2.1585 0.2521
## 12 724.0598 8.1963 0.3411 1.0730 0.2248 0.5408 6.7938 4.5329 0.2434
## 13 744.4823 8.8203 0.3242 1.0469 0.2356 0.5837 2.8526 3.4259 0.2362
## 14 756.5697 9.6367 0.3447 1.0386 0.2420 8.1930 -4.3897 -4.3934 0.2295
## 15 772.6338 10.4400 0.3486 0.9885 0.2570 3.1208 -1.3591 -2.7205 0.2230
## Ball Ptbiserial Frey McClain Dunn Hubert SDindex Dindex SDbw
## 2 158.5206 0.1661 -0.0183 0.7077 0.0987 0.0026 2.7495 1.8758 1.2264
## 3 88.3532 0.2527 -0.4439 1.4106 0.1033 0.0035 2.5346 1.7136 0.9173
## 4 46.9962 0.3661 0.2515 1.3011 0.1918 0.0092 2.0181 1.5629 0.3983
## 5 32.7363 0.3812 0.0828 1.6504 0.1918 0.0094 1.9970 1.4687 0.3693
## 6 23.7560 0.3932 2.1401 1.7577 0.1918 0.0097 2.0985 1.3617 0.3098
## 7 18.6075 0.3464 -0.1014 2.4649 0.1918 0.0100 2.5808 1.2898 0.2891
## 8 14.1727 0.3759 0.4820 2.4516 0.2717 0.0107 2.2849 1.2220 0.2419
## 9 11.2428 0.3489 0.2747 3.3008 0.2527 0.0113 2.3459 1.1645 0.2353
## 10 9.1908 0.3421 0.4000 3.6580 0.2597 0.0114 2.3922 1.1085 0.2164
## 11 7.7376 0.3339 0.4312 3.9643 0.2597 0.0115 2.4080 1.0620 0.1994
## 12 6.6096 0.3146 0.2876 4.7289 0.2597 0.0120 2.6006 1.0212 0.1949
## 13 5.6695 0.3051 0.0423 5.2169 0.2597 0.0122 2.6006 0.9825 0.1866
## 14 4.8186 0.3079 0.0437 5.3009 0.2867 0.0122 2.6004 0.9504 0.1820
## 15 4.1513 0.3099 0.0593 5.3619 0.3011 0.0124 2.5316 0.9121 0.1601
##
## $All.CriticalValues
## CritValue_Duda CritValue_PseudoT2 Fvalue_Beale
## 2 0.7098 16.7607 0.1478
## 3 0.6621 13.7821 1.0000
## 4 0.6573 13.5542 0.0158
## 5 0.5447 10.0311 0.0000
## 6 0.6225 12.1297 0.0296
## 7 0.6225 12.1297 1.0000
## 8 0.6153 11.8814 0.0269
## 9 0.5139 9.4601 0.0001
## 10 0.4742 8.8696 0.0001
## 11 0.5447 10.0311 0.0305
## 12 0.4742 8.8696 0.0001
## 13 0.3418 7.7024 0.0039
## 14 0.3854 7.9739 1.0000
## 15 0.2098 7.5336 1.0000

```

```

##
## $Best.nc
##           KL      CH Hartigan      CCC      Scott      Marriot      TrCovW
## Number_clusters 4.0000 4.0000 4.0000 15.0000 4.0000 4 4.0000
## Value_Index 3.7956 20.9925 16.3633 9.2532 115.4783 5326318910 951.7895
##           TraceW Friedman Rubin Cindex      DB Silhouette      Duda
## Number_clusters 4.0000 5.0000 4.0000 3.0000 15.0000 15.000 2.0000
## Value_Index 52.7719 64.3057 -0.4921 0.2011 0.9885 0.257 0.7968
##           PseudoT2 Beale Ratkowsky Ball PtBiserial Frey McClain
## Number_clusters 2.0000 2.0000 5.0000 3.0000 6.0000 1 2.0000
## Value_Index 10.4568 1.4951 0.3139 70.1674 0.3932 NA 0.7077
##           Dunn Hubert SDindex Dindex      SDbw
## Number_clusters 15.0000 0 5.000 0 15.0000
## Value_Index 0.3011 0 1.997 0 0.1601
##
## $Best.partition
##           Afghanistan      Angola      Bangladesh
##           1 2 2
##           Benin      Bhutan      Bolivia
##           1 3 2
##           Botswana      Burkina Faso      Burundi
##           3 1 1
##           Cambodia      Cameroon Central African Republic
##           3 1 1
##           Chad      Comoros      Congo Dem. Rep.
##           1 2 1
##           Congo Rep.      Cote d'Ivoire      Egypt
##           2 1 2
##           Equatorial Guinea      Eritrea      Fiji
##           2 2 3
##           Gabon      Gambia      Ghana
##           2 2 2
##           Guinea      Guinea-Bissau      Guyana
##           1 1 3
##           Haiti      India      Iraq
##           1 2 2
##           Kenya      Kiribati      Kyrgyz Republic
##           2 3 3
##           Lao      Lesotho      Liberia
##           2 3 3
##           Madagascar      Malawi      Mali
##           2 1 1
##           Mauritania      Micronesia Fed. Sts.      Mozambique
##           2 3 1
##           Myanmar      Namibia      Nepal
##           2 3 2
##           Niger      Nigeria      Pakistan
##           1 4 2
##           Philippines      Rwanda      Senegal
##           2 1 2
##           Sierra Leone      Solomon Islands      South Africa
##           1 3 3
##           Sudan      Tajikistan      Tanzania
##           2 2 1
##           Timor-Leste      Togo      Turkmenistan
##           2 1 2

```


##	Uganda	Uzbekistan	Vanuatu
##	1	2	3
##	Yemen	Zambia	
##	2	1	

C'est aussi une partition en 5 groupes qui obtient un vote majoritaire, nous confortant dans notre premier choix. Néanmoins, on peut déjà observé la variabilité des réponses apportées par les différents critères. Cela souligne l'importance de garder une inspection visuelle de la courbe d'inertie/dendrogramme.

- Partition en 5 groupes

K=6

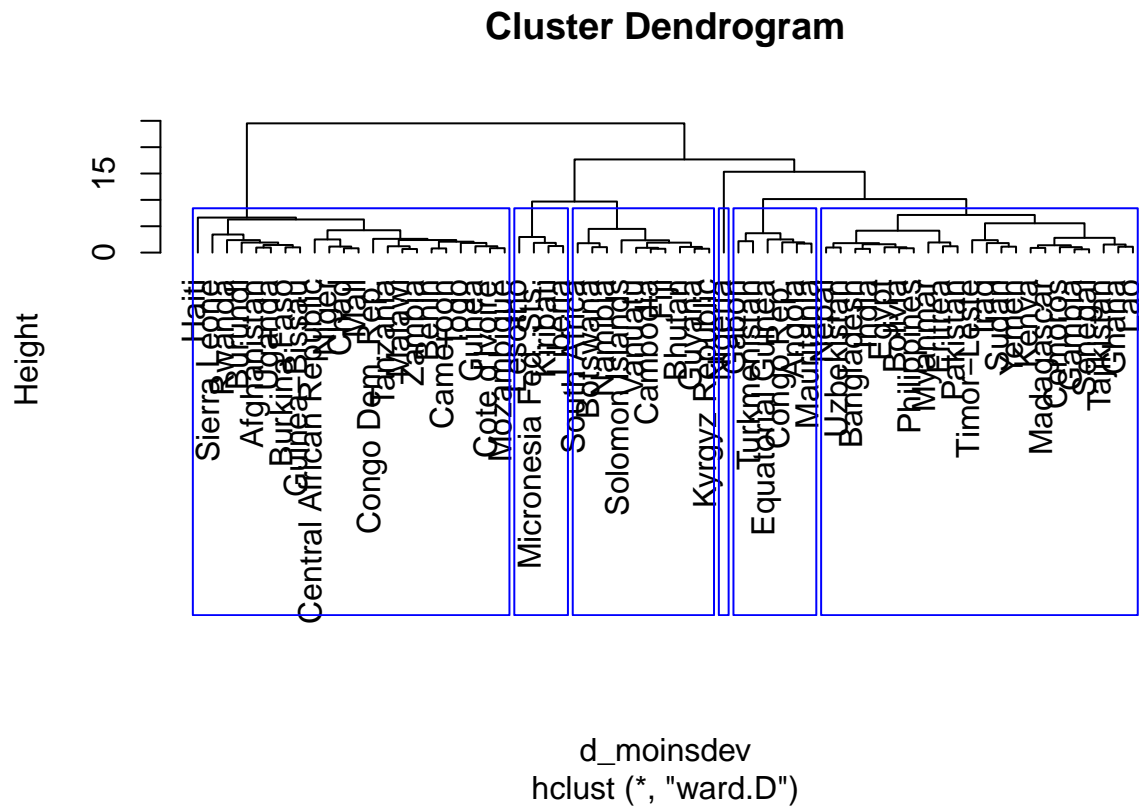
```
gpe.ward.moinsdev = cutree(cah.ward.moinsdev,k=K)
gpe.ward.moinsdev
```

##	Afghanistan	Angola	Bangladesh
##	1	2	3
##	Benin	Bhutan	Bolivia
##	1	4	3
##	Botswana	Burkina Faso	Burundi
##	4	1	1
##	Cambodia	Cameroon	Central African Republic
##	4	1	1
##	Chad	Comoros	Congo Dem. Rep.
##	1	3	1
##	Congo Rep.	Cote d'Ivoire	Egypt
##	2	1	3
##	Equatorial Guinea	Eritrea	Fiji
##	2	3	4
##	Gabon	Gambia	Ghana
##	2	3	3
##	Guinea	Guinea-Bissau	Guyana
##	1	1	4
##	Haiti	India	Iraq
##	1	3	3
##	Kenya	Kiribati	Kyrgyz Republic
##	3	5	4
##	Lao	Lesotho	Liberia
##	3	5	5
##	Madagascar	Malawi	Mali
##	3	1	1
##	Mauritania	Micronesia Fed. Sts.	Mozambique
##	2	5	1
##	Myanmar	Namibia	Nepal
##	3	4	3
##	Niger	Nigeria	Pakistan
##	1	6	3
##	Philippines	Rwanda	Senegal
##	3	1	3
##	Sierra Leone	Solomon Islands	South Africa
##	1	4	4
##	Sudan	Tajikistan	Tanzania
##	3	3	1
##	Timor-Leste	Togo	Turkmenistan
##	3	1	2
##	Uganda	Uzbekistan	Vanuatu

```
##          1          3          4
##      Yemen      Zambia
##          3          1
```

- Representation du dendrogramme avec les différents groupes obtenus

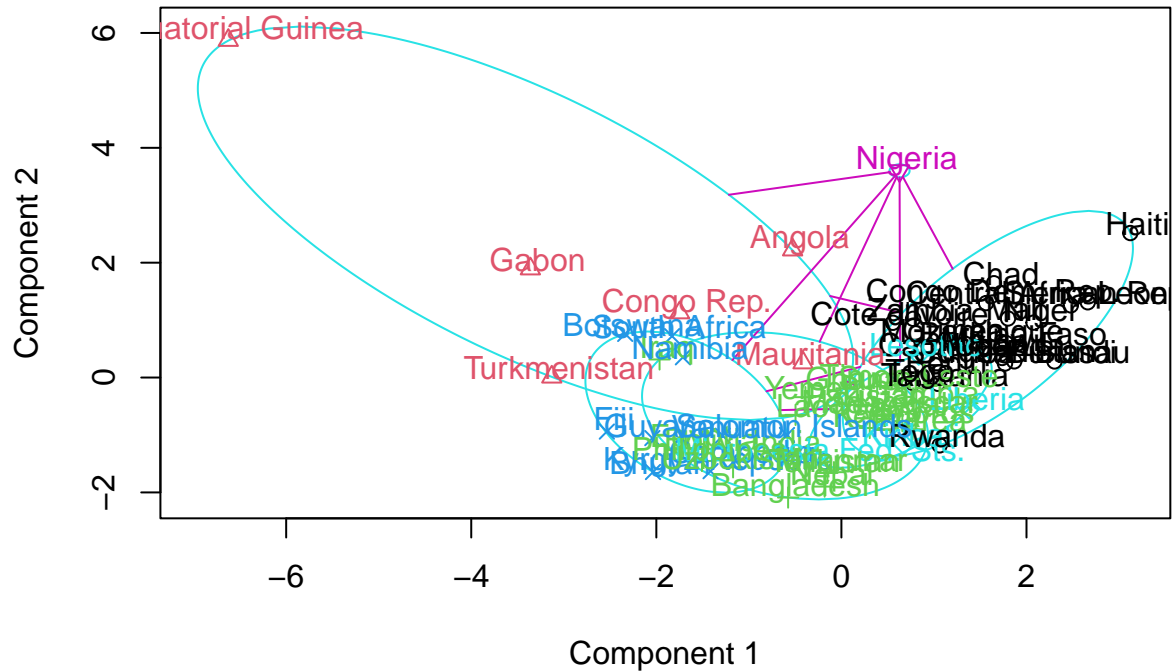
```
plot(cah.ward.moinsdev, hang=-1)
rect.hclust(cah.ward.moinsdev, K, border ="blue")
```



- Clusplot

```
clusplot(donnee_moinsdev, gpe.ward.moinsdev, labels = nbc, col.p = as.numeric(gpe.ward.moinsdev))
```

CLUSPLOT(donnee_moinsdev)



4.0.2 Avec les kmeans :

?????

Nous allons maintenant chercher à interpréter les groupes obtenus à l'aide de la fonction catdes.

```
gpe = cutree(cah.ward.moinsdev,k=6)
donnee_moinsdev$gpecah = as.factor(gpe)
interpcah_moinsdev = catdes(donnee_moinsdev,num.var = 10)
interpcah_moinsdev
```

```
##
## Link between the cluster variable and the quantitative variables
## =====
##              Eta2      P-value
## inflation    0.8277805 2.843357e-21
## exports      0.6632833 8.049063e-13
## imports      0.6101392 5.390651e-11
## enfant_mort  0.5882342 2.567210e-10
## sant.        0.5186260 2.154148e-08
## fert         0.5140868 2.806410e-08
## esper_vie    0.4686508 3.443915e-07
## pib_h        0.4190328 4.103777e-06
## revenu       0.3879273 1.718424e-05
##
## Description of each cluster by quantitative variables
## =====
## $'1'
```

```

##          v.test Mean in category Overall mean sd in category Overall sd
## enfant_mort  5.261392      1.8430922      1.0052894      0.79735402  0.9111885
## fert         4.925025      1.7224261      0.9839659      0.55035786  0.8579969
## exports      -2.186203     -0.6274248     -0.3685552      0.40605966  0.6775757
## pib_h        -2.849173     -0.6717963     -0.6024230      0.01693266  0.1393288
## revenu       -3.003598     -0.8096345     -0.6754601      0.03448857  0.2556201
## esper_vie    -4.875814     -1.6638156     -1.0165444      0.69122515  0.7596372
##          p.value
## enfant_mort  1.429687e-07
## fert         8.434950e-07
## exports      2.880075e-02
## pib_h        4.383308e-03
## revenu       2.668077e-03
## esper_vie    1.083606e-06
##
## $'2'
##          v.test Mean in category Overall mean sd in category Overall sd
## exports      5.307884      1.0411868     -0.3685552      0.4938734  0.6775757
## pib_h        4.366455     -0.3639549     -0.6024230      0.2925059  0.1393288
## revenu       4.202999     -0.2543316     -0.6754601      0.5379379  0.2556201
## sant.       -2.648928     -1.2556338     -0.2535490      0.3040798  0.9651035
##          p.value
## exports      1.109051e-07
## pib_h        1.262793e-05
## revenu       2.634019e-05
## sant.        8.074750e-03
##
## $'3'
##          v.test Mean in category Overall mean sd in category Overall sd
## esper_vie     4.310725     -0.4442895     -1.0165444      0.3459917  0.7596372
## fert         -2.316082      0.6366917      0.9839659      0.7296534  0.8579969
## sant.        -2.646476     -0.6998983     -0.2535490      0.6137348  0.9651035
## exports      -2.864561     -0.7077496     -0.3685552      0.4090737  0.6775757
## enfant_mort  -3.007297      0.5264196      1.0052894      0.4282767  0.9111885
## imports      -3.175471     -0.5612330     -0.1108258      0.5264829  0.8116414
##          p.value
## esper_vie     1.627198e-05
## fert          2.055378e-02
## sant.         8.133517e-03
## exports       4.175879e-03
## enfant_mort   2.635821e-03
## imports       1.495932e-03
##
## $'4'
##          v.test Mean in category Overall mean sd in category Overall sd
## imports       3.231242      0.65799486     -0.1108258      0.6567251  0.8116414
## exports       2.998008      0.22694519     -0.3685552      0.2785296  0.6775757
## enfant_mort  -3.622402      0.03768858      1.0052894      0.2773539  0.9111885
## fert         -3.646689      0.06674115      0.9839659      0.3585549  0.8579969
##          p.value
## imports       0.0012325344
## exports       0.0027175010
## enfant_mort   0.0002918801
## fert          0.0002656413
##
## $'5'

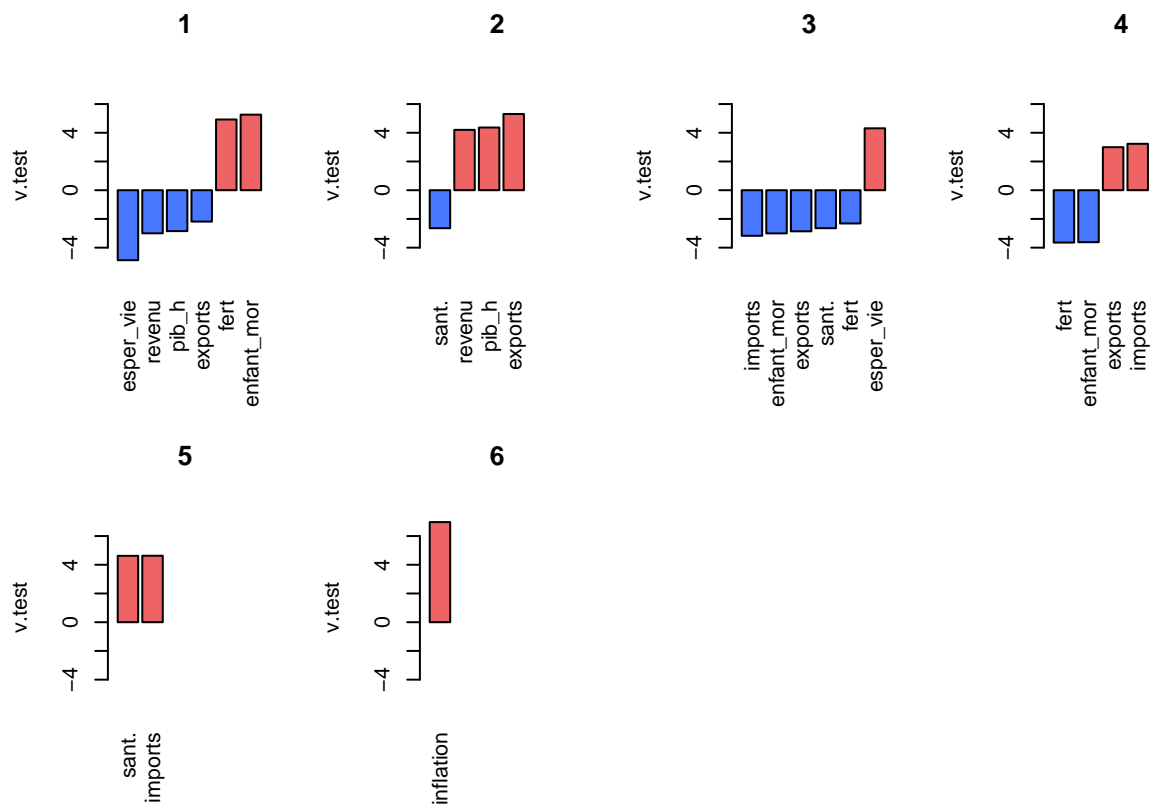
```

```
##          v.test Mean in category Overall mean sd in category Overall sd
## imports 4.630851          1.723895   -0.1108258      0.3596272  0.8116414
## sant.   4.621737          1.923780   -0.2535490      0.4510464  0.9651035
##          p.value
## imports 3.641654e-06
## sant.   3.805401e-06
##
## $'6'
##          v.test Mean in category Overall mean sd in category Overall sd
## inflation 6.971909          9.102343    0.2824583          0    1.26506
##          p.value
## inflation 3.126695e-12
```

```
head(donnee_moinsdev)
```

```
##          enfant_mort      exports      sant.      imports      revenu
## Afghanistan  1.2876597 -1.134866649  0.2782514 -0.08220771 -0.8058219
## Angola        2.0017872  0.773056185 -1.4437289 -0.16481961 -0.5832892
## Bangladesh    0.2759790 -0.915984488 -1.1998120 -1.03637509 -0.7627678
## Benin         1.8034185 -0.631437679 -0.9886601 -0.40026351 -0.7949287
## Bhutan        0.1098452  0.050745055 -0.5881996  0.98348572 -0.5563155
## Bolivia       0.2065500  0.003320587 -0.7192594 -0.52005075 -0.6087067
##          inflation  esper_vie      fert      pib_h  gpecah
## Afghanistan  0.15686445 -1.61423717  1.8971765 -0.6771431      1
## Angola        1.38289444 -1.17569847  2.1217698 -0.5147203      2
## Bangladesh    -0.06071803 -0.01750653 -0.4082076 -0.6659584      3
## Benin         -0.65244778 -0.98454058  1.5933150 -0.6659584      1
## Bhutan        -0.16950927  0.17365136 -0.3751792 -0.5883752      4
## Bolivia       0.09442774  0.11742845  0.1664870 -0.5992871      3
```

```
plot.catdes(interpcah_moinsdev,barplot=T)
```



```
CCpca_moinsdev = dudi.pca(donnee_moinsdev[1:9],scannf=FALSE,nf=2)
```

```

cumsum(CCpca_moinsdev$eig)/sum(CCpca_moinsdev$eig) # 52% de variabilité expliquée sur les deux premiers

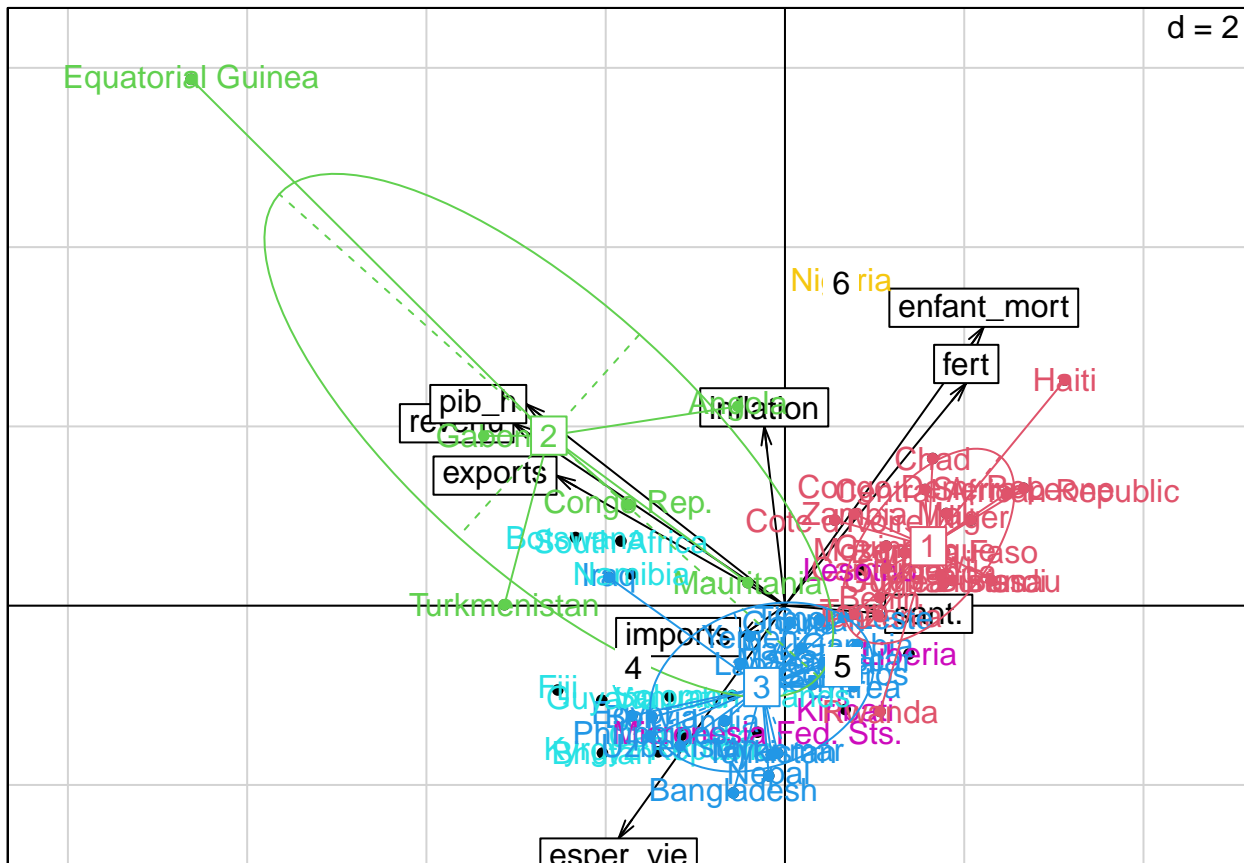
## [1] 0.3176225 0.5259903 0.7065002 0.8096545 0.8955336 0.9556743 0.9804839
## [8] 0.9966107 1.0000000

scatter(CCpca_moinsdev, posieig = "none", clab.row=0, pch=NA)

## NULL

text(CCpca_moinsdev$li[,1], CCpca_moinsdev$li[,2], labels = row.names(donnee_moinsdev), col=gpe+1, xpd=TRUE,
s.class(CCpca_moinsdev$li, factor(gpe), col = 2:4, add.plot = TRUE, clabel = 1)

```



5 4 - Conclusion vis à vis des choix effectués

Quels points peuvent être critiqués dans votre choix Quelles pistes pourraient être explorées pour aller plus loin et/ou mieux explorer ces données ?

Nous avons fait un premier gros choix suite à l'obtention de nos premiers résultats. En effet, nous n'avons sélectionné que le groupe dont les pays étaient en sous-développement. Ce choix peut être critiqué. Cependant, ayant déjà un grand nombre de pays dans ce groupe et n'ayant "que" 10 millions de dollars à partager, nous avons décidé de ne prioriser que ce groupe.

Nous avons ensuite retraité ce groupe de pays défavorisés afin de pouvoir observer les pays qui étaient le plus en difficulté. Là encore, nous avons dû faire un choix : donner une grosse somme d'argent aux pays dans le besoin puis une somme d'argent plus faible aux pays qui en ont moins besoin, mais une aide sera là quand même.