CH 3 : Classification

A- Généralités

B- Mesure d'éloignement

C- Critère d'homogénéité

D- Choix d'une méthode

E- Mesures de la qualité

F- Interprétation

G- ACP/Classification

H- Exemple

Terminologie : de nombreux synonymes

- **Classification**, ou classification automatique, terme généralement employé par les auteurs français
  - attention : il est employé dans un autre sens par les anglo-saxons (qui disent « classification » pour désigner la technique prédictive que les français appellent « classement »)

- **Segmentation** : terme employé en marketing (les « segments de clientèle ») et assez explicite
- **Typologie**, ou **analyse typologique**
- **Clustering** : terme anglo-saxon le plus courant
- **Taxinomie** ou **taxonomie** (biologie, zoologie)
- **Nosologie** (médecine)
- **Reconnaissance de forme non supervisée**

A- Généralités

- ✓ Données= tableau n*p individus*variables
- ✓ Objectif = recherche d'une typologie ou segmentation, c'est à dire d'une partition ou répartition des n individus dans des classes, sur la base de l'observation de p descripteurs
- ✓ Moyen= chaque classe doit être la plus homogène possible et, entre elles, les plus distinctes possibles, au sens d'un critère à définir.

A- Généralités

Qu'est-ce que la classification ?

- Regrouper des objets en groupes, ou classes, ou familles, ou segments, ou clusters, de sorte que :
  - 2 objets d'un même groupe se ressemblent le + possible
  - 2 objets de groupes distincts diffèrent le + possible
  - le nombre des groupes est parfois fixé

A- Généralités

Mise en œuvre d'une classification :

- Choix de la mesure d'éloignement (dissimilarité, distance) entre individus (généralement distance euclidienne)
- Choix du critère d'homogénéité des classes à optimiser (généralement inertie).
- Choix de la méthode utilisée : la Classification Ascendante Hiérarchique (CAH) ou celle par réallocation dynamique sont les plus utilisées
- Mesure de la qualité de la classification
- Choix du nombre de classes et leur interprétation

B- Mesure d'éloignement

✓ Ensemble des individus $E = (e_1, \ldots e_i, \ldots e_n)$

✓ On définit d /

(1) $d(e_i,e_i)=0$

(2) $d(e_i,e_j)=d(e_j,e_i)$

(3) $d(e_i,e_j)=0 \Rightarrow e_i=e_j$

(4) $d(e_i,e_j) \leq d(e_i,e_k)+d(e_k,e_j)$

(5) $d(e_i,e_j) \leq \max(d(e_i,e_k),d(e_k,e_j))$

- Indice de dissimilarité : (1) (2)
- Indice de distance : (1) (2) (3) ; Distance : (1) (2) (3) (4)
- Distance ultramétrique : (1) (2) (3) (4) (5)

B- Mesure d'éloignement

- ✓ Le choix d'une mesure dépend de la nature des descripteurs. Le plus souvent, le choix se porte sur une distance euclidienne (i.e. engendrée par un produit scalaire) appropriée.

- ✓ Variables quantitatives : M de produit scalaire sur l'espace des variables:

$$(1)\ M = I_p$$

$$(2)\ M = D_{\sigma}^{-1}$$

$$(3)\ M = V^{-1}$$

$$d(e_i, e_j) = \left\| e_i - e_j \right\|_M^2 = e_i' M e_j$$

(1) variables homogènes, (2) hétérogènes, (3) corrélées

C- Critère d'homogénéité

✓ Si l'ensemble des points est regroupé en K classes $N_k \sim (G_k, P_k)$

- Inertie intra-classes : $$I_W = \sum_{k=1}^{K} I_k = \sum_{k=1}^{K} \sum_{e_i \in N_k} p_i \left\| X_i - G_k \right\|^2$$
- Inertie inter-classes : $$I_B = \sum_{k=1}^{K} P_k \left\| G_k - G \right\|^2$$

$$I = I_w + I_B$$

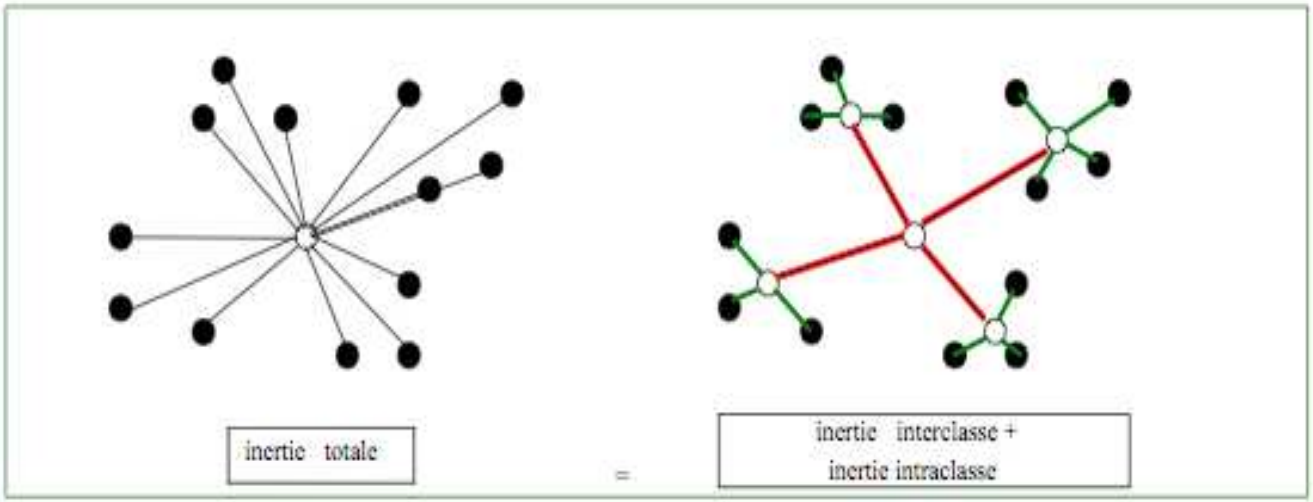
✓ **Critère le plus utilisé** (variables quantitatives)=

$$\boxed{\min_{N_k} I_w}$$

K fixé

C- Critère d'homogénéité

- Une classe est homogène ⇔ son inertie est faible
- Deux critères de bonne classification : grande I_R, petite I_A
- Ces deux critères sont équivalents d'après la *formule de Huygens :* $I_{TOT} = I_A + I_R$



D- Les méthodes

- Méthodes hiérarchiques
  - ascendantes (agglomératives)
    - basées sur une notion de distance ou de densité
  - descendantes (divisives)
- Méthodes de partitionnement
  - centres mobiles, k-means et nuées dynamiques
  - k-modes, k-prototypes, k-représentants (k-medoids)
  - réseaux de Kohonen
  - méthodes basées sur une notion de densité
  - méthode « de Condorcet » (analyse relationnelle)
- Méthodes mixtes
- Analyse floue (fuzzy clustering)

D- 1 Algorithmes de partitionnement

Principe : on part d'une partition arbitraire en K classes (K choisi) que l'on améliore itérativement jusqu'à la convergence du critère choisi.

- Méthode des centres mobiles
- Méthode des k-means
- Méthode des « nuées dynamiques »

D- 1 Algorithmes de partitionnement

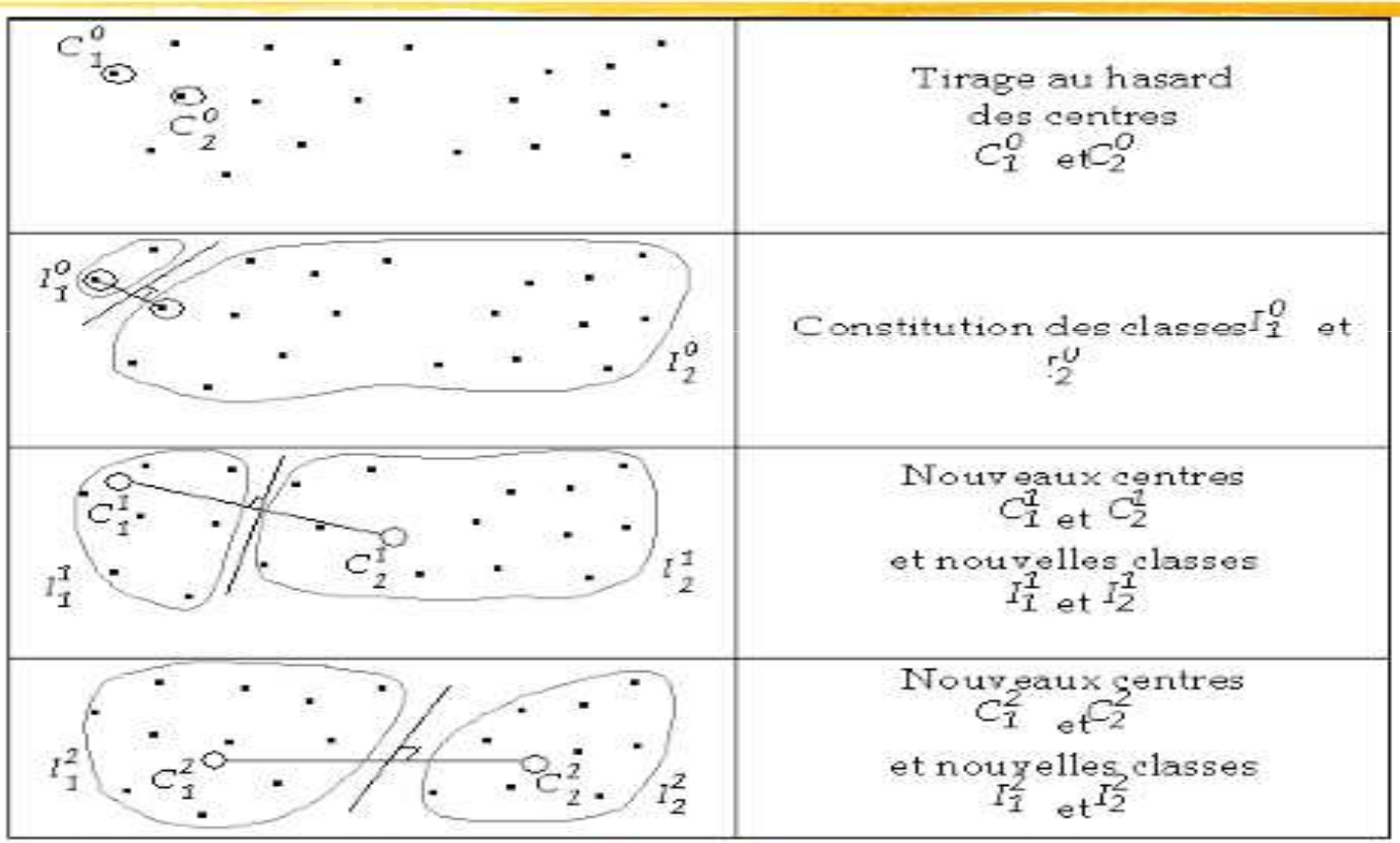
Centres mobiles (Forgy):

- ✓ Initialisation : Choix aléatoire de k points de l'espace (centres des classes)
- ✓ Itérer les deux étapes suivantes jusqu'à ce que le critère à minimiser (inertie intraclasses) ne décroisse plus de manière significative (minimum local), ou bien jusqu'à atteindre un nombre d'itérations fixées:

  - ➢ Tous les individus sont affectés à la classe dont le centre est le plus proche au sens de la distance choisie. On construit ainsi k classes d'individus

  - ➢ On calcule les barycentres des classes créées qui deviennent les k nouveaux centres

D- 1 Algorithmes de partitionnement



Source : Ludovic Lebart – *Analyse des données appliquée* - 2002

D- 1 Algorithmes de partitionnement

- ✓ « k-means » (Mc Queen): les barycentres des classes ne sont pas recalculés à la fin des affectations , mais à la fin de chaque allocation d'un individu à une classe. L'algorithme est ainsi plus rapide, mais l'ordre d'apparition des individus dans le fichier n'est pas neutre.

- ✓ « nuées dynamiques » (Diday): ce n'est plus un seul point qui représente une classe mais un noyau de points constitués d'éléments représentatifs de la classe. Cela permet de corriger l'influence d'éventuelles valeurs extrêmes sur le calcul du barycentre.

D- 1 Algorithmes de partitionnement

Remarques sur la méthode :

- ✓ On peut montrer que l'algorithme fait diminuer l'inertie inter-classes à chaque itération donc l'algorithme converge
- ✓ L'expérience montre que le nombre d'itérations nécessaires est très faible
- ✓ Inconvénients des algorithmes de partitionnement:
  - ➢ Instabilité : Le minimum obtenu est un minimum local: la répartition en classes dépend du choix initial des centres (faire tourner l'algorithme plusieurs fois pour identifier des formes fortes)
  - ➢ Le nombre de classes est fixé par avance (on peut s'aider d'une ACP pour le déterminer)

D- 1 Algorithmes de partitionnement

Programmation sous R :

>km=kmeans(x, centers=, iter.max=, algorithm=)

centers=soit un vecteur indiquant les centres de départ, soit le nombre de classes choisies

iter.max= nombre d'itérations maximale; jusqu'à convergence par défaut

algorithm=« Forgy » pour centres mobiles, « MacQueen » pour k-means, nuées dynamiques par défaut

D- 1 Algorithmes de partitionnement

```
>names(km)
[1] "cluster"  "centers"  "withinss" "size"
```

cluster: vecteur d'entier indiquant le numero de classe de chaque individu.

centers: matrice des coordonnées des centres de gravité des classes (valeurs moyennes des variables pour chaque classe).

withinss: vecteur somme des carrés des écarts intra-classes pour chaque classe (=Ik*n lorsque les poids sont égaux).

size: vecteur de l'effectifs dans chaque classe.

D-2 Classification ascendante hiérarchique (CAH)

Principe: Fournir un ensemble de partitions de moins en moins fines obtenus par regroupement successifs de parties.

Une fois cet algorithme terminé, on ne récupère donc pas directement une partition, mais une hiérarchie de partitions en n,...1 classes, avec diminution de l'inertie inter-classes à chaque agrégation

D-2 Classification ascendante hiérarchique (CAH)

Algorithme :

- ✓ Initialisation : les classes initiales sont les n singletons-individus. Calculer la matrice de leurs distances deux à deux
- ✓ Itérer les deux étapes suivantes jusqu'à l'agrégation en une seule classe :
  - ➢ Regrouper les deux éléments (classes) les plus proches au sens de la distance entre groupes choisie
  - ➢ Mettre à jour le tableau de distances en remplaçant les deux classes regroupées par la nouvelle et en calculant sa distance avec chacune des autres classes

D-2 Classification ascendante hiérarchique (CAH)

- ✓ Nécessité de définir une distance entre groupes d'individus (appelé stratégie d'agrégation).

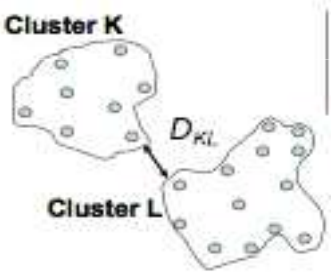
- ✓ Nécessite de choisir le nombre de classes à retenir

D-2 Classification ascendante hiérarchique (CAH)

Stratégies d'agrégation :

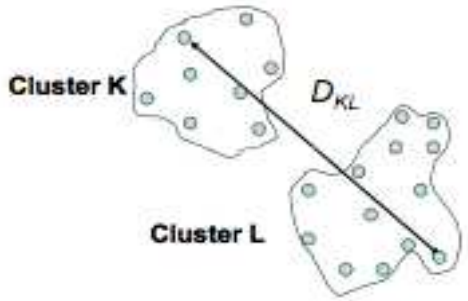
- ✓ stratégie du saut minimum ou single linkage (la distance entre parties est la plus petite distance entre éléments des deux parties):

$$\Delta(A,B) = \min_{i \in A,\, j \in B} d(i,j)$$



- ✓ stratégie du saut maximum ou du diamètre ou complete linkage (la distance entre parties est la plus grande distance entre éléments des deux parties):

$$\Delta(A,B) = \max_{i \in A,\, j \in B} d(i,j)$$



D-2 Classification ascendante hiérarchique (CAH)

- ✓ Méthode du saut Ward (en espace euclidien, option par défaut de SAS)

$$\Delta(A,B)=\frac{p_A p_B}{p_A+p_B}d^2(G_A,G_B)$$

A chaque itération, on agrège de manière à avoir une gain minimum d'inertie intra-classe $\Delta(A,B)=$ perte d'inertie inter-classe due à cette agrégation

D-2 Classification ascendante hiérarchique (CAH)

Remarques sur les méthode :

- ✓ Avec la méthode de Ward, on agrège a chaque itération les classes dont l'agrégation fait perdre le moins d'inertie interclasse il s'agit donc d'une optimisation pas-a-pas, qui ne dépend pas d'un choix initial arbitraire

- ✓ Inconvénients

  - ➢ Tourne lentement
  - ➢ Vraisemblablement assez éloigné d'un optimum si le nombre de pas n –p est trop élevé
  - ➢ La CAH est peu robuste: il suffit de modifier une distance pour que le saut change (sensibilité aux valeurs extrêmes).

D-2 Classification ascendante hiérarchique (CAH)

- ✓ La méthode du single linkage occasionne des problème de chaînage : Très souvent, en effet, on se retrouve avec un groupe démesurément gros et plusieurs petits groupes satellites.

- ✓ Le complete linkage ne présente pas ce problème. Il tend, au contraire à former des groupes de taille égale. Cependant, la méthode est très sensible aux points aberrants et est peu utilisée en pratique.

D-2 Classification ascendante hiérarchique (CAH)

Choix du nombre de classes:

On décidera de retenir la partition qui semble la meilleure, généralement

- ✓ celle qui précède une valeur de la distance inter brutalement plus faible (à chaque agrégation, on perd de la distance entre les groupes)
- ✓ celle telle que les partitions ont un sens « naturel ».

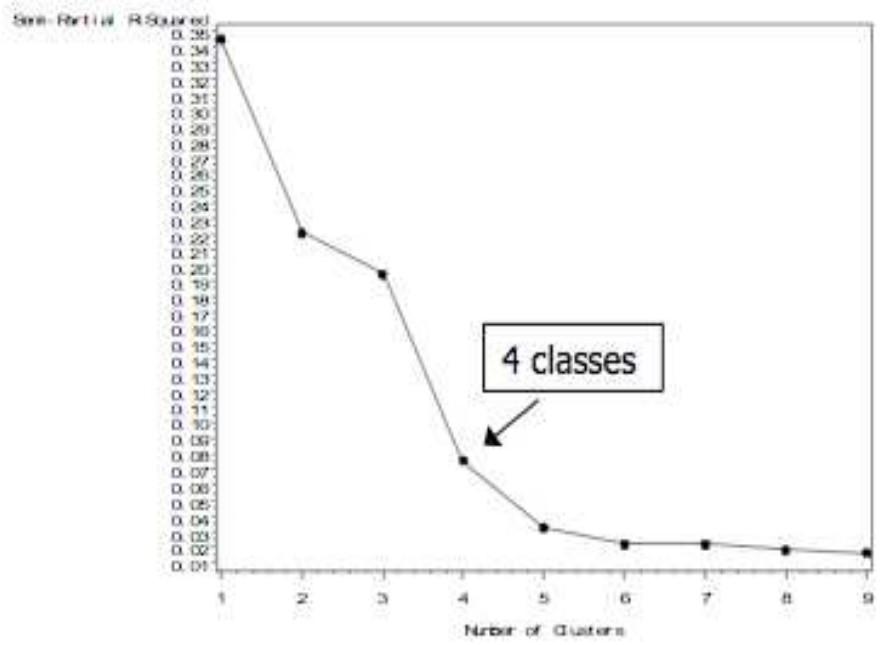
D-2 Classification ascendante hiérarchique (CAH)

Graphiques utiles au choix du nombre de classes:

- ✓ Graphique représentant la perte de distance (resp. gain) en fonction du nombre de d'iterations (resp. de classes). La i° itération correspond au passage de n-i+1 à n-i classes. On coupe l'arbre avant une perte trop importante de de distance (inertie inter pour ward). ΔI_B
- ✓ De facon équivalente, on peut aussi utiliser le graphe du R-square semi-partiel :

D-2 Classification ascendante hiérarchique (CAH)

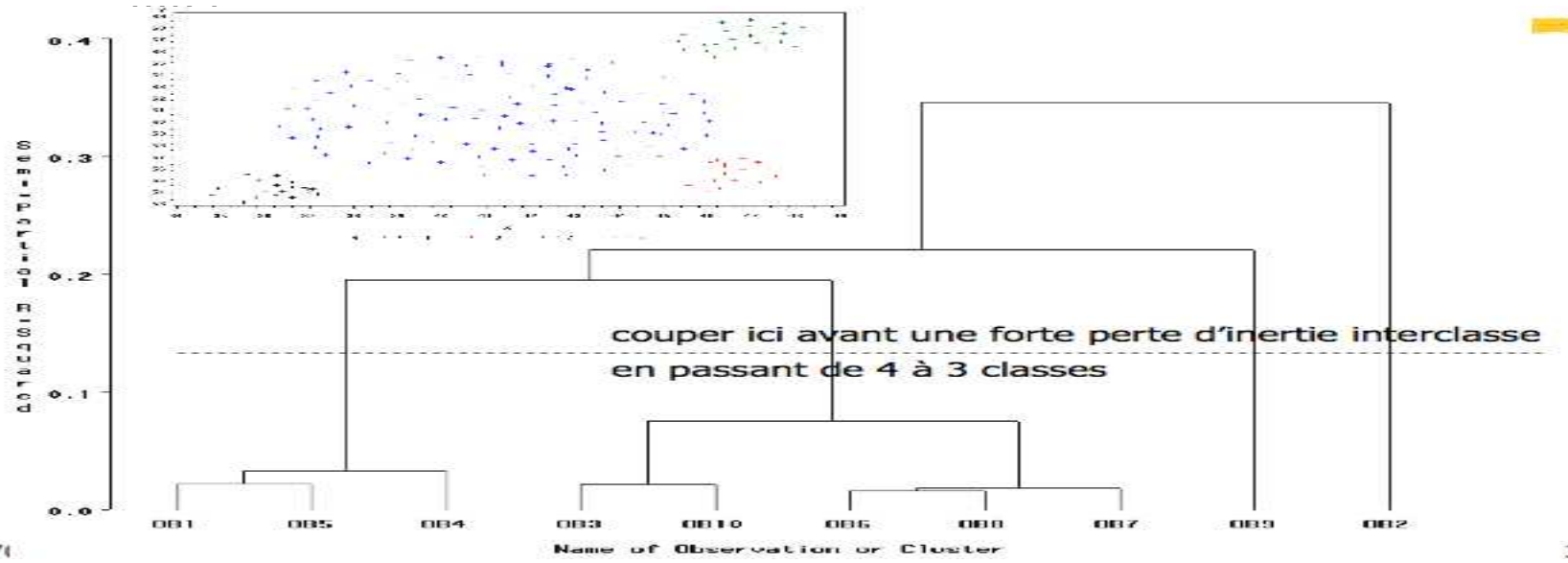
- Semi-partial R-squared (SPRSQ)= mesure la perte d'inertie interclasse (ou de distance) provoquée en regroupant 2 classes. Le but étant d'avoir une inertie interclasse maximum, on recherche un faible SPRSQ suivi d'un fort SPRSQ à l'agrégation suivante : un pic pour k classes et un creux pour k+1 classes indique une bonne classification en k+1 classes



$$SPRSQ = \frac{\Delta I_B}{I}$$

D-2 Classification ascendante hiérarchique (CAH)

- ✓ Dendogramme = représentation graphique sous forme d'arbre binaire, d'agrégations successives jusqu'à réunion en une seule classe de tous les individus. La hauteur d'une branche est proportionnelle à la distance entre les deux objets regroupés. Dans le cas du saut de Ward, à la perte d'inertie interclasses. On coupe avant une forte perte d'inertie



D-2 Classification ascendante hiérarchique (CAH)

Programmation sous R:

>hc=hclust(d, method=)

d= un tableau de distances comme produit par dist()

method= méthode d'agrégation: "ward", 'single",
"complete« (méthode par défaut)

fonction dist(x, method=):

x= tableau sur lequel calculer la distance

method='"euclidean"', '"maximum"', '"manhattan"', '"canberra"',
'"binary"' or '"minkowski"'. Par défaut, euclidienne

D-2 Classification ascendante hiérarchique (CAH)

```
>names (hc)
[1] "merge"      "height"     "order"      "labels"     "method"
[6] "call"       "dist.method«
```

>plot(hc, hang=, cex=) # dessine le dendogramme, hang=-1 pour mettre tous le noms des individus au même niveau, cex pour baisser la casse

>cutree(hc,k=) #attribue la classe de chaque individu lorsqu'on coupe à k classes

>rect.hclust(hc, k=, border=« red ») #dessine les classes sur le dendogramme

D-2 Classification ascendante hiérarchique (CAH)

- ✓ merge: an n-1 by 2 matrix. Row i of 'merge' describes the merging of clusters at step i of the clustering.
- ✓ height: a set of n-1 non-decreasing real values. The clustering _height_: that is, the value of the criterion associated with the clustering 'method' for the particular agglomeration.
- ✓ order: a vector giving the permutation of the original observations suitable for plotting, in the sense that a cluster plot using this ordering and matrix 'merge' will not have crossings of the branches.
- ✓ labels: labels for each of the objects being clustered.
- ✓ call: the call which produced the result.
- ✓ method: the cluster method that has been used.
- ✓ dist.method: the distance that has been used to create 'd'

D-2 Classification ascendante hiérarchique (CAH)

Ce que font ces fonctions en pratique :

```
>m=matrix(1:16,4,4)
> m
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
>dist(m)^2   #il vaut mieux utiliser le carré, plus proche de la perte d'inertie à
       # chaque agrégation, qui est proportionnelle à une distance²
   1  2  3
2  4*
3 16  4
4 36 16  4
```

*= carré de la distance euclidienne entre lignes 1 et 2 de la matrice=$(2-1)^2+(6-5)^2+(10-9)^2+(14-13)^2=4$

D-2 Classification ascendante hiérarchique (CAH)

```
> hc=hclust(dist(m)^2,method="ward")
```

Donne les agrégations successives, faites de manière à perdre le moins possible d'inertie inter-classe (i.e. les éléments les plus proches à chaque agrégation au sens de la distance de Ward :

$$\Delta(A,B) = \frac{p_A p_B}{p_A + p_B} \left\| G_B - G_A \right\|^2$$

```
> hc$merge
     [,1] [,2]
[1,]   -1   -2
[2,]   -3   -4
[3,]    1    2
```

$merge donne les agrégations faites à chaque pas, que l'on va détailler:

D-2 Classification ascendante hiérarchique (CAH)

1° agrégation : on part de la matrice des distances de Ward , à ce stade $=d^2(ei,ej)/8$

```
>de1=dist(m)^2/8
>de1
  1  2  3
2  0.5
3  2  0.5
4 4.5  2  0.5
```

On agrège 1 et 2 en cl1: delta(1,2)=0.5, min. (on aurait pu aussi regrouper 2 et 3 ou 3 et 4)

2° agrégation : nouveaux éléments et calcul de la nouvelle matrice de distance de ward

```
>cl1=apply(m[1:2,],2,mean)
>m2=rbind(cl1,m[3:4,])
>rownames(m2)=c("cl1",3,4)
> m2
    [,1] [,2] [,3] [,4]
cl1  1.5  5.5  9.5 13.5
3    3.0  7.0 11.0 15.0
4    4.0  8.0 12.0 16.0
```

D-2 Classification ascendante hiérarchique (CAH)

```
> d2=dist(m2,diag=T,upper=T)^2
   cl1  3  4
cl1   0  9 25
3     9  0  4
4    25  4  0

> de2=matrix(c(0, (2/12)*9, (2/12)*25,(2/12)*9,0, (1/8)*4,(2/12)*25,(1/8)*4,0),3,3,byrow=T)
> rownames(de2)=c("cl1",3,4)
> colnames(de2)=c("cl1",3,4)
> de2
         cl1  3  4
cl1 0 1.5 4.166667
3     1.5 0 0.5
4   4.166667 0.5 0
```

D-2 Classification ascendante hiérarchique (CAH)

On agrège 3 et 4 en cl2: delta(3,4)=0.5, minimal

3° agrégation: nouveaux éléments et nouvelle matrice de Ward

```
>cl2=apply(m2[2:3,],2,mean)
 >m3=rbind(cl1,cl2)
> m3
    [,1] [,2] [,3] [,4]
cl1   1.5  5.5  9.5 13.5
cl2  3.5  7.5 11.5 15.5
> de3=dist(m3)^2/4
[1] 4
```

On retrouve bien les résultats de $merge. Par ailleurs, le vecteur des distances de Ward (perte d'inertie inter à chaque agrégation est : dib=[0.5,0.5,4](on vérifie que la somme des composantes=inertie totale du tableau). La fonction $height donne le vecteur 2*n*dib

```
>hc$height
[1]  4  4 32
```

D- 3 Méthode mixte

- ✓ Combinent efficacement les avantages des deux types

de méthodes vues précédemment et permettent d'en annuler les inconvénients

- ✓ CAH, suivie mais aussi parfois précédée d'une méthode de type centres mobiles

- ✓ Si le nombre d'individus n est élevé (plusieurs milliers), lancer plusieurs fois un algorithme de type centres mobiles, en faisant varier le choix des K points de départ : le produit des partitions obtenues permet d'obtenir les groupes d'individus toujours classés ensemble : les formes fortes.

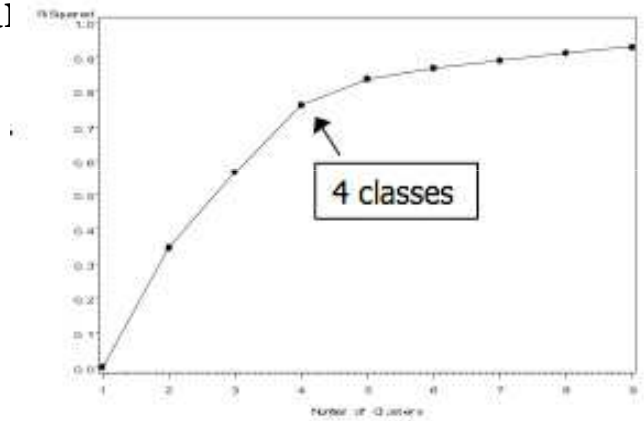
D- 3 Méthode mixte

- ✓ On lance une CAH sur les centres de ces formes fortes, avec la méthode de Ward : on chemine ainsi vers une partition en K classes, en minimisant a chaque pas la perte d'inertie interclasse.

- ✓ On lance les centres mobiles sur la partition issue de la CAH, avec pour points de départ les barycentres des K classes : on est ainsi assure d'aboutir a un optimum local,en autorisant quelques reaffectations individuelles

E- Mesure de la qualité (toute classification)

- R^2 = proportion de la variance expliquée par les classes $R^2 = \frac{I_B}{I}$
  - doit être le plus proche possible de 1 sans avoir trop de classes
  - s'arrêter après le dernier saut importa



- Pseudo F = mesure la séparation entre toutes les classes
  - n=observations; k=classes
  - doit être grand

$$F = \frac{R^2/k-1}{(1-R^2)/n-k}$$

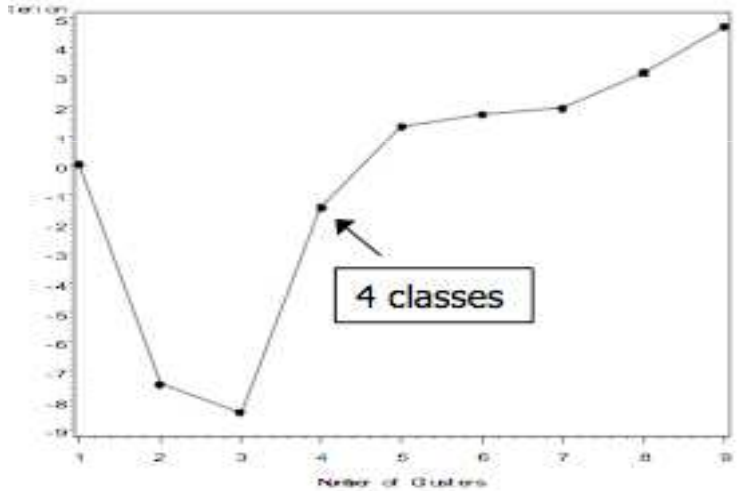
E- Mesure de la qualité (toute classification)

- Cubic clustering criterion CCC (valable sous SAS) permet de tester

H0 = Les données sont issues d'une distribution uniforme (pas de classes)

$$CCC = \ln\left[\frac{1 - E(R^2)}{1 - R^2}\right] \times K$$ où K est une constante (voir Sarle (1983))

- CCC > 2 : bonne classification
- 0< CCC<2 : classification peut être OK mais à vérifier
- CCC< 0 : présence d'outliers gênants (surtout si CCC < -30)



On trace CCC versus le nombre de classes. Un creux pour k classes suivi d'un pic pour k+1 classes indique une bonne classification en k+1 classes (surtout si on a une croissance ou décroissance douce à partir de k+2 classes)

Rq : ne pas utiliser CCC et F en single linkage

F- Caractérisation des classes

Chaque classe est décrite grâce aux variables actives - celles sur lesquelles on a souhaité différencier les classes ; mais aussi avec toute autre variable supplémentaire (quel qu'en soit le type) dont on connaît les valeurs sur notre population de n individus.

- ✓ Généralement, on commence par calculer la distance entre le barycentre de chaque classe et le barycentre global, afin de connaître l'excentricité globale de chaque groupe.

- ✓ Ensuite, on peut comparer, pour chaque variable, sa distribution sur chaque classe et sur l'ensemble de la population. Lorsque les variables sont numériques, on compare généralement leurs moyennes sur les différents groupes(en tenant compte des effectifs et des variances respectifs), à l'aide de tests statistiques.

F- Caractérisation des classes

- ✓ On peut aussi quantifier la force discriminante de chaque variable sur la classification dans son ensemble, afin de déterminer quelles sont les caractéristiques expliquant le plus les différenciations construites.

- ✓ En fin de course, il est fréquent de « nommer » chacune des classes obtenues par un qualificatif résumant la caractérisation.

- ✓ Pour illustrer ce qui caractérise une classe, on peut aussi rechercher l'individu le plus typique (ou central) de la classe, ou bien encore un noyau d'individus la représentant bien.

F- Caractérisation des classes

- Les tests utilisés sont ceux utilisés en analyse de variance à un facteur (test de Fisher, test de Student, test de Scheffe, test de Bartlett…)

- La force discriminante est la statistique de fisher : (variance inter/k-1)/(variance intra/n-k)

G- Complémentarité ACP/Classification

ACP = outil puissant pour analyser les corrélations entre plusieurs variables . Permet d'obtenir de nouvelles variables non corrélées à forte variance (i.e. à fort pouvoir informatif).

Limites de l'ACP:

- l'ACP permet de visualiser la structure des corrélations, mais ces visualisations ne sont réalisables que sur des plans, ce qui limite la perception précise des phénomènes si les variables sont nombreuses et leurs relations complexes

- la contrainte d'orthogonalité des axes rend souvent l'interprétation délicate au-delà du premier plan factoriel, car les informations délivrées sont alors résiduelles, sachant les proximités déjà observées sur les axes précédents.

G- Complémentarité ACP/Classification

En comparaison, les méthodes de classification prennent en compte l'ensemble des dimensions du nuage, et corrigent donc les distorsions des projections factorielles, en appréhendant les individus tels qu'ils sont en réalité, et non tels qu'ils apparaissent en projection.

Une solution : faire figurer sur les projections factorielles du nuage des individus la partition obtenue par classification : on fait figurer ces classes par leur barycentre, et/ou par leur enveloppe, voire en remplaçant chaque point projeté par le code de sa classe d'appartenance, si le nombre de points est assez restreint.

G- Complémentarité ACP/Classification

Exemples de programmes R pour la représentation des classes sur les axes factoriels

- ✓ Programme 1 :

```
k=      #nombre de classes choisies
p =princomp(don)    #don=tableau de données
u = p$loadings
x =(t(u) %*% t(don))[1:2,]
x = t(x)
plot(x, col="red")
c =hc$merge
y =NULL
n = NULL
```

G- Complémentarité ACP/Classification

```
for (i in (1:(length(don[,1])-1))) {
if( c[i,1]>0 ){
a =y[ c[i,1], ]
a.n = n[ c[i,1] ] }
else {
a =x[ -c[i,1], ]
a.n = 1 }
if( c[i,2]>0 ){
b = y[ c[i,2], ]
b.n = n[ c[i,2] ] }
else {
b =x[ -c[i,2], ]
b.n =1 }
n = append(n, a.n+b.n)
m = ( a.n * a + b.n * b )/( a.n + b.n )
```

G- Complémentarité ACP/Classification

```
y =rbind(y, m)
segments( m[1], m[2], a[1], a[2], col= "red" )
segments( m[1], m[2], b[1], b[2], col= "red" )
if( i> length(voit[,1])-1-k ){
op=par(ps=20)
text( m[1], m[2], paste(length(don[,1])-i), col= " blue" )
par(op) } }
text( x[,1], x[,2], attr(x, "dimnames")[[1]] , col=cutree(cl,k),cex=0.7)
```

G- Complémentarité ACP/Classification

- ✓ Programme 2 (plus simple):

```
don=
k=
p =princomp(don,corr=T)
u =p$loadings
x =(t(u) %*% t(don))[1:2,]
x = t(x)
par(pch ="*" )
plot(x, col=cutree(cl,k), pch="", lwd=3 )
text( x[,1], x[,2], attr(x, "dimnames")[[1]],cex=0.7,col=cutree(cl,k) )
```

G- Complémentarité ACP/Classification

✓ Programme 3 (avec les kmeans):

```
k=
cl <- kmeans(don, k, 20)
p <- princomp(don)
u <- p$loadings
x <- (t(u) %*% t(don))[1:2,]
 x <- t(x)
plot(x, col=cl$cluster, pch=3, lwd=3)
c <- (t(u) %*% t(cl$center))[1:2,]
c <- t(c)
points(c, col = 1:k, pch=7, lwd=3)
for (i in 1:k) {
print(paste("Cluster", i))
for (j in (1:length(don[,1]))[cl$cluster==i]) {
segments( x[j,1], x[j,2], c[i,1], c[i,2], col=i ) } }
text( x[,1], x[,2], attr(x, "dimnames")[[1]],col=cl$cluster,cex=0.7 )
```

H- Exemple

```
>crimer=scale(crime)*sqrt(20/19)
> kmr=kmeans(crimer,4)
> kmr
K-means clustering with 4 clusters of sizes 4, 2, 6, 8

Cluster means:
     Meurtre        Rapt         vol    attaque       viol     larcin
1  1.2885435 -0.09942123  0.1963340  0.3947703 -0.5107076 -0.7493501
2 -0.5262038 -0.28901522  0.1184812 -1.0311974  0.9375262  1.1600607
3  0.2852093  1.25856493  0.8854221  1.1317863  1.0281233  1.0429356
4 -0.7266277 -0.82195928 -0.7918539 -0.7884256 -0.7501202 -0.6975419
```

H- Exemple

```
Clustering vector:
   Alabama     Alaska    Arizona   Arkansas California   Colorado
         1          3          3          4          3          3
Connecticut   Delaware    Florida    Georgia     Hawaii      Idaho
         4          2          3          1          2          4
  Illinois    Indiana       Iowa     Kansas   Kentucky  Louisiana
         1          4          4          4          4          1
     Maine   Maryland
         4          3

Within cluster sum of squares by cluster:
[1]  5.459948  1.216823 18.691114 13.297253

Available components:
[1] "cluster"  "centers"  "withinss" "size"
```

H- Exemple

Remarques :

I=p=6

Ik=km$withinss/20

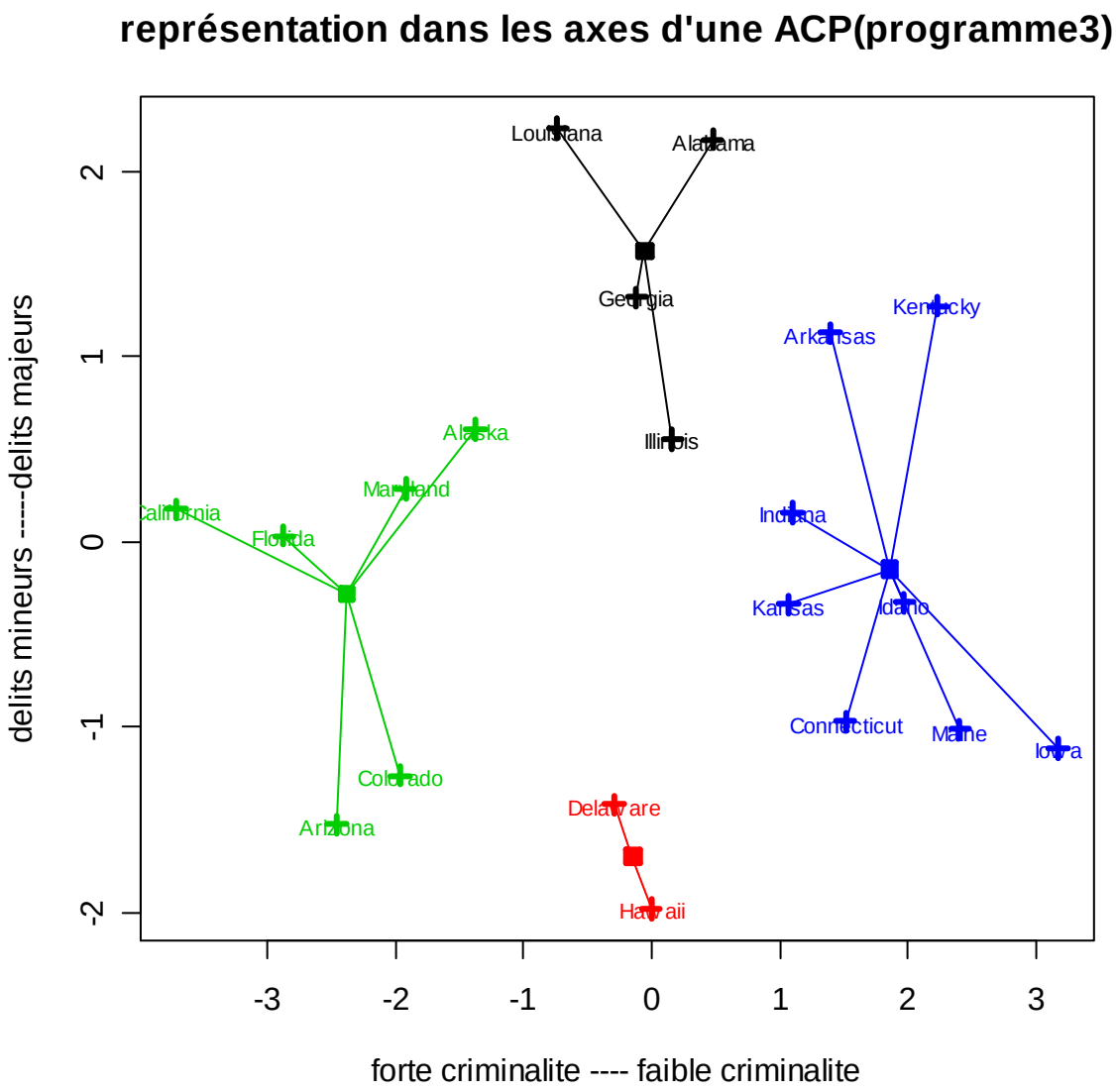
Iw=sum(Ik)

Ib=sum(km$size*(km$centers)^2)/20

Ib+Iw=I

Excentricité= apply(kmr$centers^2,1,sum)

```
      1        2        3        4
3.662522 5.874995 2.500805 3.931421
```


représentation dans les axes d'une ACP(programme3)
delits mineurs ----delits majeurs
forte criminalite ---- faible criminalite
-3
-2
-1
0
1
2
3
2
1
0
-1
-2
Louisiana
Alabama
Georgia
Illinois
Kentucky
Arkansas
Alaska
Maryland
California
Florida
Indiana
Kansas
Idaho
Connecticut
Maine
Iowa
Colorado
Arizona
Delaware
Hawaii

H- Exemple

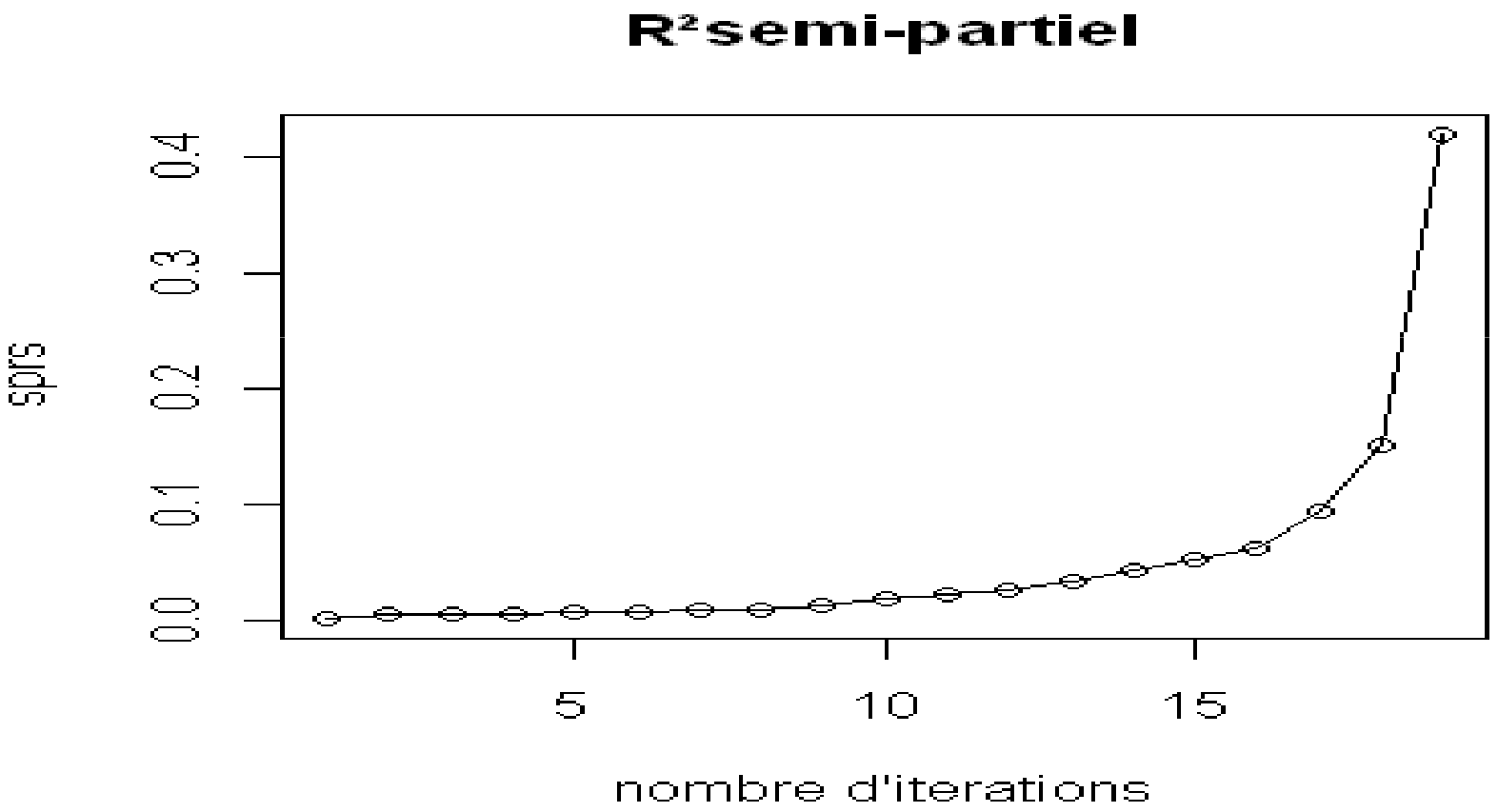
```
cl=hclust(dist(crimer)^2,method="ward")
dib=cl$height/(2*20); sprs=dib/6; sprs
 [1] 0.002904180 0.006039930 0.006443103 0.006999353 0.008278108 0.008817480
 [7] 0.010140189 0.010697026 0.013071394 0.018802508 0.022365056 0.026382177
[13] 0.034308073 0.043313249 0.054189955 0.062023921 0.094134200 0.151447604
[19] 0.419642494
```

```
plot(sprs type=« o", main="R² partiel",xlab="nombre d'iterations«
plot(dib,type="b", main=« perte d'inertie inter",xlab="nombre d'iterations")
```

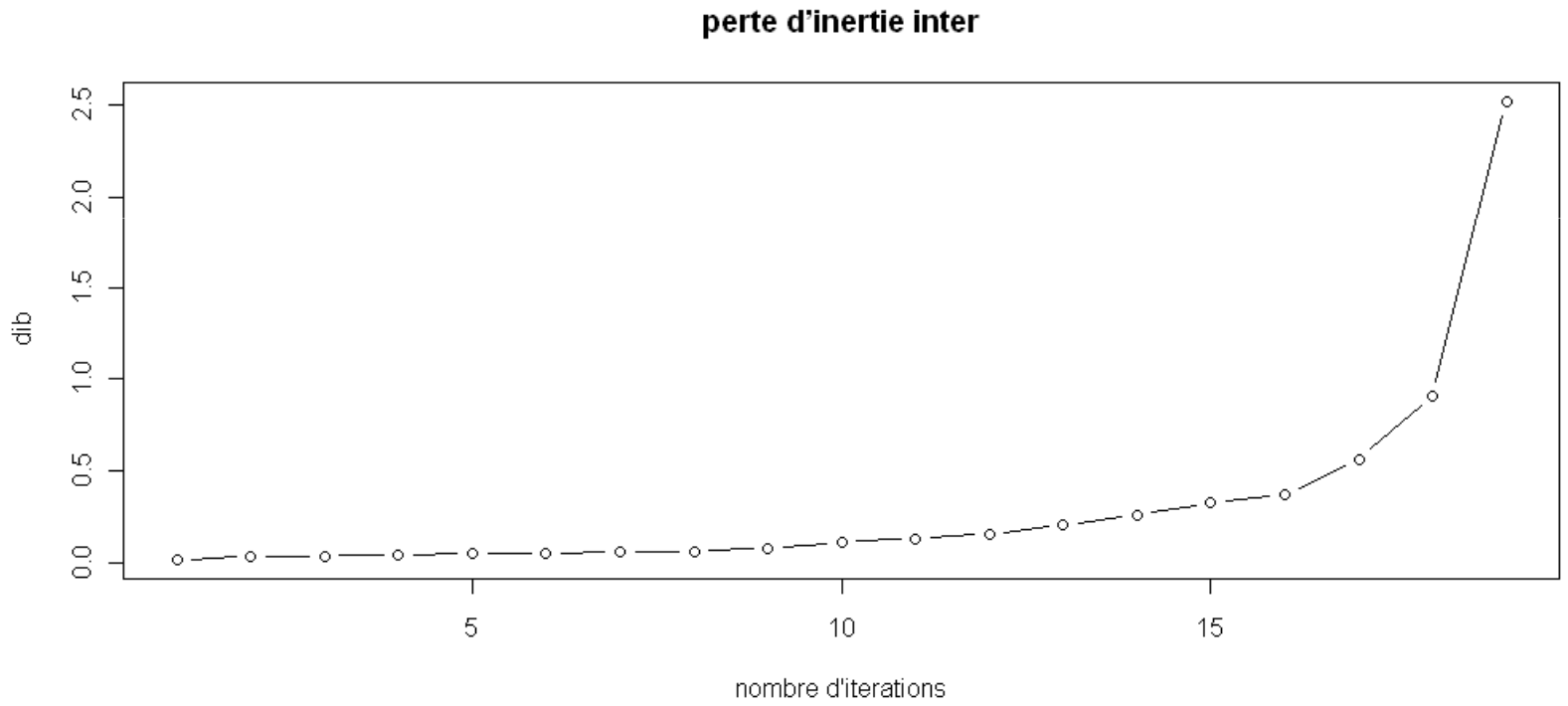
On perd au passage de 15 (pic ,5 classes) à 14 (creux, 6 classes) itérations. On garde 6 classes. Le R2 vaut alors 78% . Ou au passage de 17 à 16 (3 à 4 classes). On garde 4 classes. Le R2 vaut 66%.

```
r2= (6-cumsum(dib))/6; r2[14]; r2[16]
plot(r2, type="o", main="R²",xlab="nombre d'iterations")
```

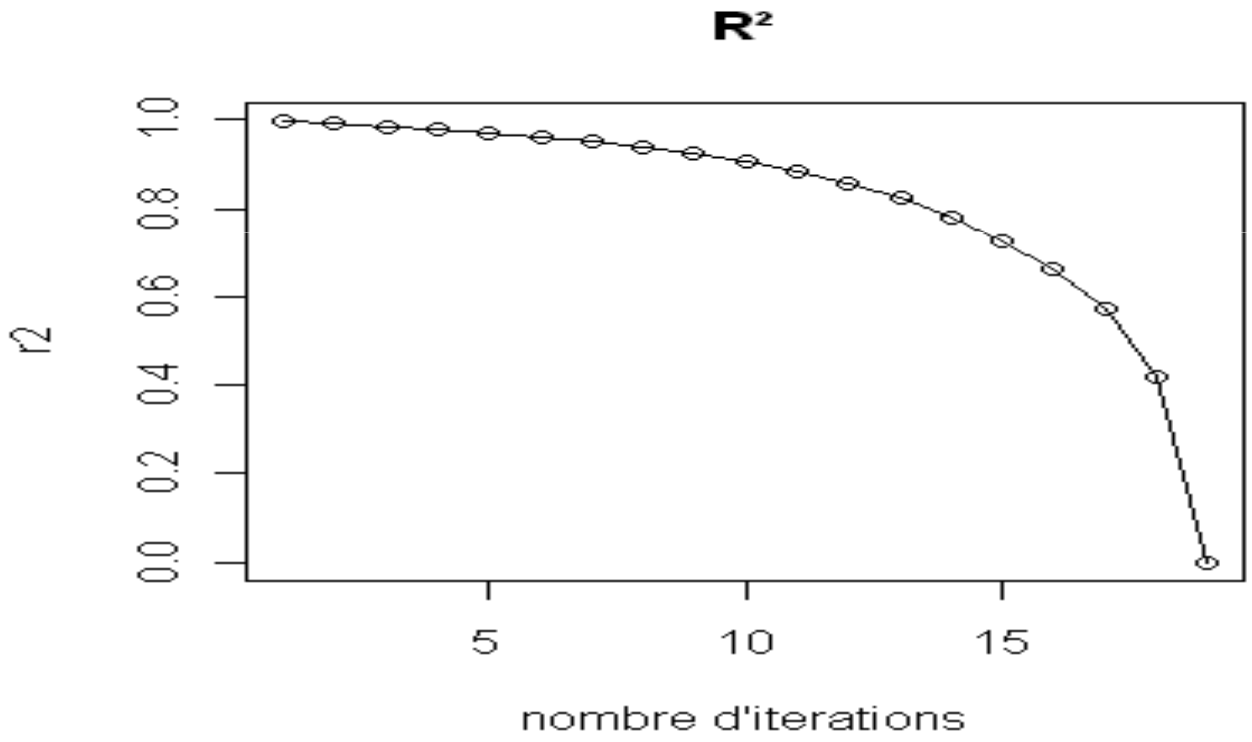
H- Exemple


R²semi-partiel
spris
0.0
0.1
0.2
0.3
0.4
5
10
15
nombre d'iterations

H- Exemple


perte d'inertie inter
dib
0.0
0.5
1.0
1.5
2.0
2.5
5
10
15
nombre d'iterations

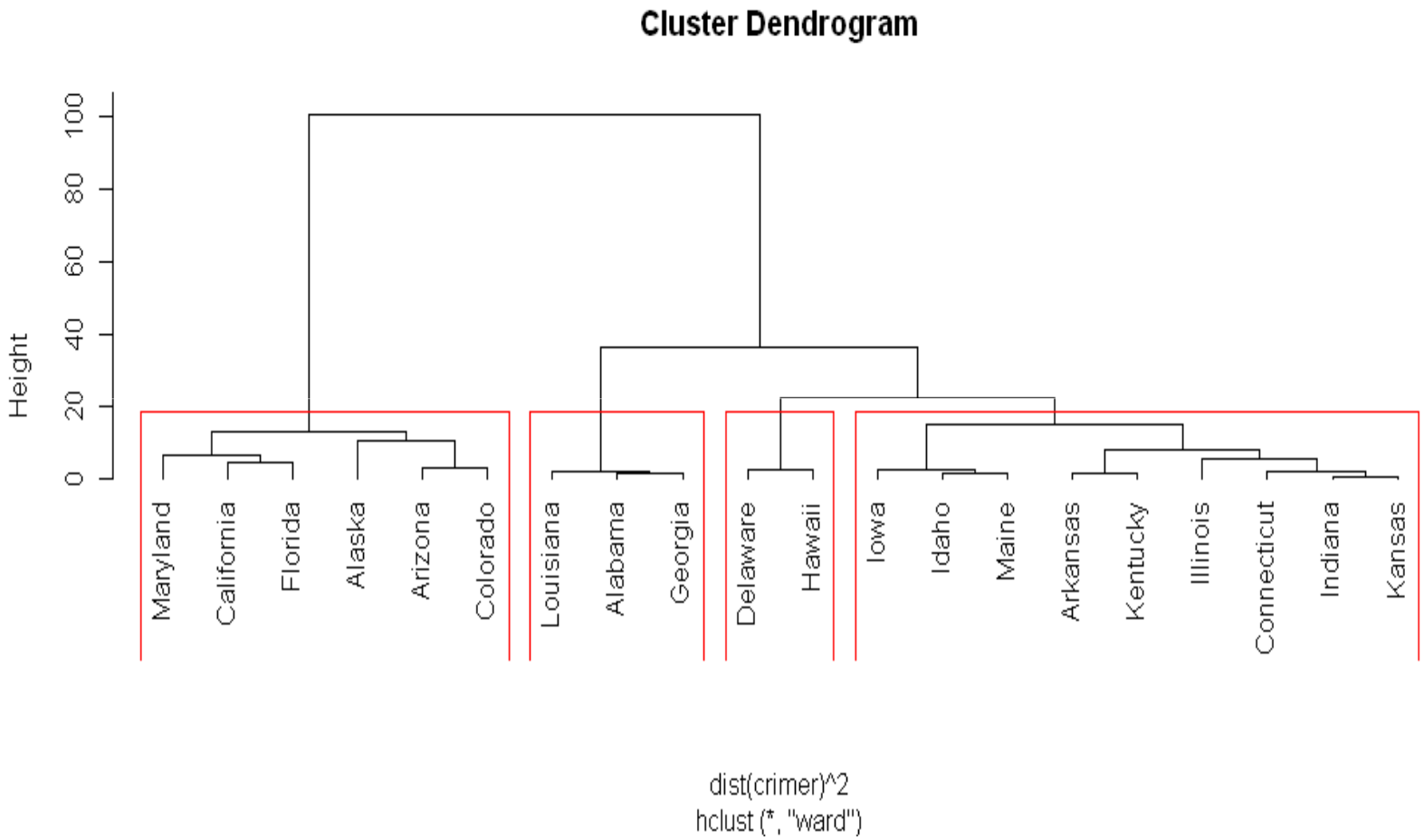
H- Exemple



H- Exemple

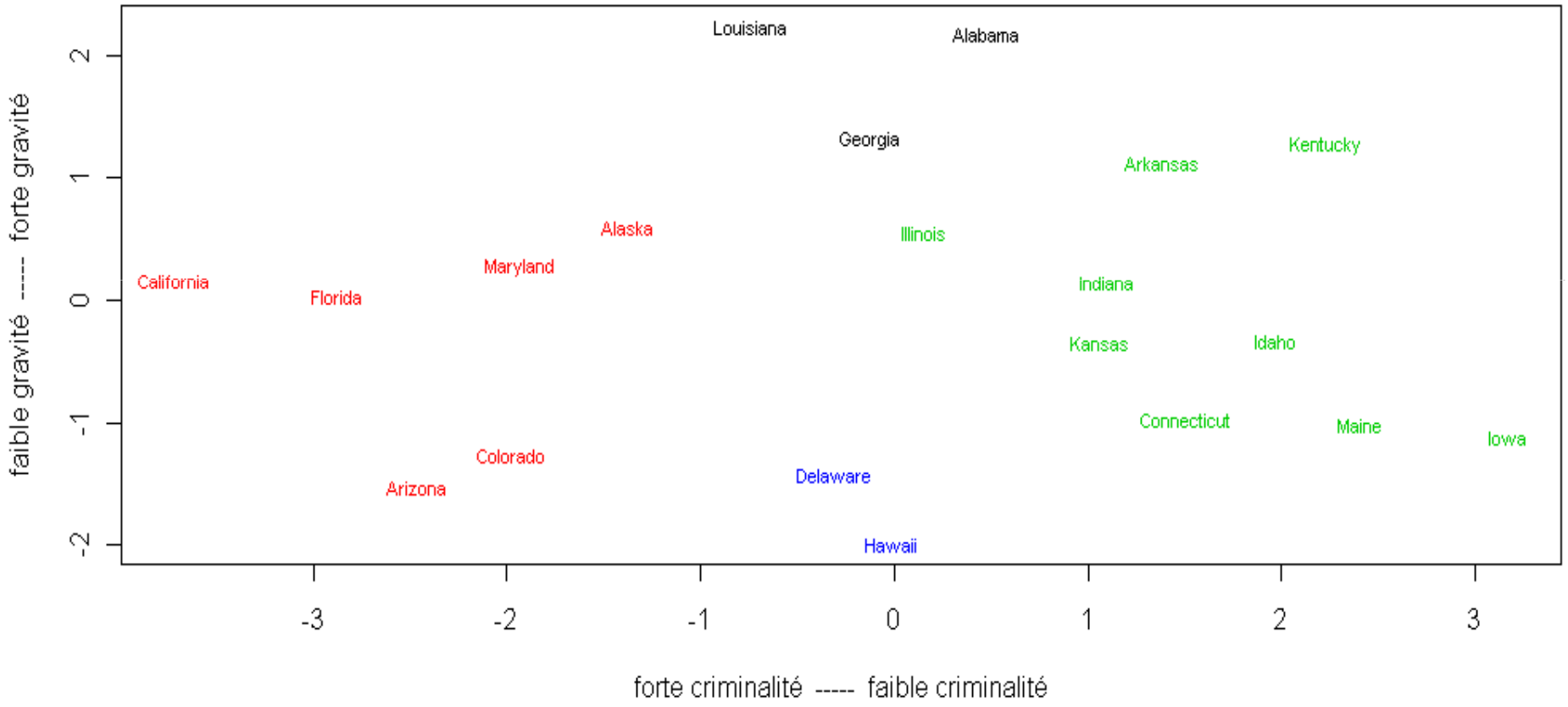
```
> plot(cl, hang=-1)
> rect.hclust(cl, k=4, border="red")
> class=cutree(cl, 4);class
```

```
   Alabama     Alaska    Arizona   Arkansas California   Colorado Connecticut
  Delaware    Florida    Georgia
         1          2          2          3          2          2          3          4          2          1
    Hawaii      Idaho   Illinois    Indiana       Iowa     Kansas   Kentucky  Louisiana
     Maine   Maryland
         4          3          3          3          3          3          3          1          3          2
```


Cluster Dendrogram
Height
0
20
40
60
80
100
Maryland
California
Florida
Alaska
Arizona
Colorado
Louisiana
Alabama
Georgia
Delaware
Hawaii
Iowa
Idaho
Maine
Arkansas
Kentucky
Illinois
Connecticut
Indiana
Kansas
dist(crimer)^2
hclust (*, "ward")

H- Exemple

```
k=4
p =princomp(crimer,corr=T)
u =p$loadings
x =(t(u) %*% t(crimer))[1:2,]
x = t(x)
par(pch ="*" )
plot(x, col=cutree(cl,k), pch="", lwd=3,xlab="forte criminalité  -----  faible
     criminalité", ylab="faible gravité  -----  forte gravité" )
text( x[,1], x[,2], attr(x, "dimnames")[[1]],cex=0.7,col=cutree(cl,k) )
```


Louisiana
Alabama
Georgia
Kentucky
Arkansas
Alaska
Illinois
Maryland
California
Indiana
Florida
Kansas
Idaho
Connecticut
Maine
Iowa
Colorado
Delaware
Arizona
Hawaii
faible gravité ----- forte gravité
2
1
0
-1
-2
-3
-2
-1
0
1
2
3
forte criminalité ----- faible criminalité

H- Exemple

```
cr=cbind(crimer, class)
cbind(apply(cr[class==1,],2,mean),apply(cr[class==2,],2,mean),apply(cr[class==3,]
    ,2,mean),apply(cr[class==4,],2,mean))
                [,1]       [,2]       [,3]       [,4]
Meutre    1.57278102 0.2852093 -0.5974657 -0.5262038
Rapt      0.06859294 1.2585649 -0.7976820 -0.2890152
Vol      -0.10988717 0.8854221 -0.5799815  0.1184812
Attaque   0.60029007 1.1317863 -0.7254659 -1.0311974
Viol     -0.43249865 1.0281233 -0.7495884  0.9375262
Larcin   -0.96308160 1.0429356 -0.6320545  1.1600607
class     1.00000000 2.0000000  3.0000000  4.0000000
```

- NOIR ROUGE VERT BLEU