Fiche TD avec le logiciel R : tdr1109

Introduction à la classification

A.B. Dufour & D. Clot

La fiche donne les principes généraux de la classification automatique. L'essentiel est consacré à la description des fonctions `hclust` et `kmeans` dans R.

Table des matières

1 Définitions

L'objectif principal des méthodes de classification automatique est de répartir les éléments d'un ensemble en groupes c'est-à-dire d'établir une partition de cet ensemble. Différentes contraintes sont bien sûr imposées, chaque groupe devant être le plus homogène possible, et les groupes devant être les plus différents possibles entre eux.

De plus, on ne se contente pas d'une partition, mais on cherche une hiérarchie de parties, qui constitue un arbre binaire appelé **dendrogramme**. Quelques définitions de base sont donc indispensables.

On considère ici des ensembles finis donc des collections d'objets au sens habituel. A est un **ensemble** :

$$A = \{a_1, a_2, ..., a_n\} \Leftrightarrow a_j \in A \text{ pour } 1 \leq j \leq n$$

Une **partie** de A est un sous-ensemble :

$$B = \{b_1, b_2, ..., b_p\} \subseteq A \Leftrightarrow b_k \in A \text{ pour } 1 \leq k \leq p$$

Si on compte la partie vide et l'ensemble tout entier, il y a 2^n parties dans A. L'**ensemble de toutes les parties** de A se note $\Phi(A)$. Si A est formé de $\{a, b, c, d\}$, $\Phi(A)$ compte 16 éléments qui sont :

$$\begin{gathered}
\varnothing \\
\{a\}, \{b\}, \{c\}, \{d\} \\
\{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{c, d\} \\
\{a, b, c\}, \{a, b, d\}, \{a, c, d\}, \{b, c, d\} \\
\{a, b, c, d\}
\end{gathered}$$

Deux parties d'un ensemble sont :
soit chevauchantes (non égales et d'intersection non nulle),
soit disjointes (sans élément commun, d'intersection nulle),
soit incluses l'une dans l'autre,
soit égales.

Une **partition** est un sous-ensemble de parties deux à deux disjointes dont la réunion fait l'ensemble tout entier.

$$\begin{gathered}
A = \{A_1, A_2, ..., A_K\} \text{ est une partition de } A \\
\Updownarrow \\
A_i \cap A_j = \varnothing \text{ pour } i \neq j \\
\cup_{k=1}^{K} A_k = A
\end{gathered}$$

Par exemple, $\{\{a, e, f, g\}, \{b\}, \{c, d\}\}$ est une partition de $\{a, b, c, d, e, f, g\}$. Une partition équivaut à une **variable qualitative**. Dans R, c'est un `factor` :

```
pop <- gl(4,5, labels = c("rouge", "vert", "bleu", "jaune")) # facteur couleur
w1 <- sample(pop) # réarrangement par permutation pour brasser les couleurs
w1
```

```
 [1] bleu  rouge jaune jaune rouge bleu  vert  rouge vert  vert  jaune rouge bleu
[14] rouge vert  bleu  jaune jaune vert  bleu
Levels: rouge vert bleu jaune
```

```
 split(1:20, w1)
```

```
$rouge
[1]  2  5  8 12 14
$vert
[1]  7  9 10 15 19

$bleu
[1]  1  6 13 16 20

$jaune
[1]  3  4 11 17 18
```

Les composantes de la liste sont les parties, les noms des composantes sont les niveaux du facteur. Les méthodes d'ordination (ACP, AFC, etc) fournissent, comme leur nom l'indique, une ordination des individus ; elles résument les données par un (ou plusieurs) score(s) numérique(s). Les méthodes de classification résument les données par une variable qualitative. Elles fournissent des partitions. Il n'y a pas de bonnes ou de mauvaises méthodes, mais des outils plus ou moins utiles pour parler des données. On peut les utiliser simultanément comme par exemple, en représentant les groupes d'individus obtenus par classification sur le plan factoriel issu d'une méthode d'ordination.

Un ensemble quelconque de parties est formé de parties chevauchantes, disjointes ou incluses. Un ensemble de parties formant une partition ne comporte que des parties disjointes recouvrant le tout. Entre ces deux classes, la première trop large pour être utile et la seconde trop étroite pour être nuancée, on trouve les hiérarchies de parties.

Une **hiérarchie** de parties de A est un ensemble de parties ayant quatre propriétés :

1. La partie vide en fait partie
2. Les parties réduites à un seul élément en font partie.
3. L'ensemble total A lui-même en fait partie.
4. Si X et Y en font partie, alors soit X et Y sont disjointes, soit X contient Y, soit Y contient X.

Par exemple, l'ensemble :

$$\{\{a\},\{b\},\{c\},\{d\},\{e\},\{a,b\},\{e,d\},\{a,b,c,d,e\}\}$$

est une hiérarchie de parties ou encore un n-arbre.

Un arbre est un graphe raciné :
- les feuilles sont les parties à un seul élément (qui sont toujours dans une hiérarchie),
- la racine est l'ensemble tout entier (qui est toujours dans la hiérarchie).

Chaque partie n'a qu'un ancêtre, à l'exclusion de la racine qui n'en a pas. Si l'arbre est binaire, chaque partie a deux descendants, à l'exclusion des feuilles qui n'en ont pas. On dit alors que la hiérarchie est **totalement résolue**.

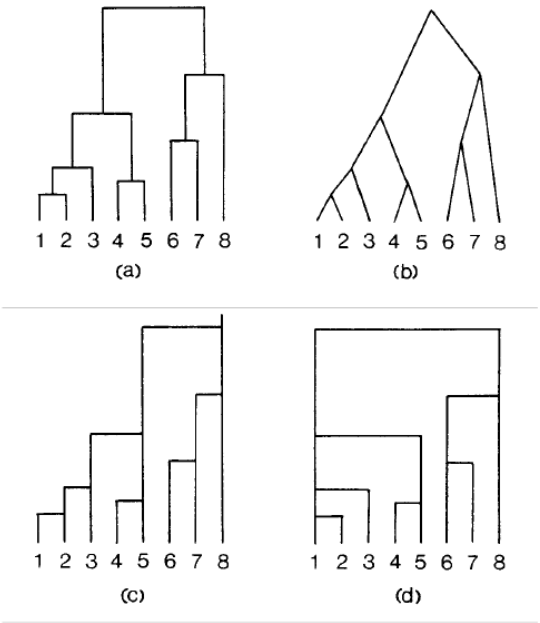
La hiérarchie est **valuée** si à chaque partie on peut associer une valeur numérique qui vérifie la définition :

$$X \subseteq Y \Leftrightarrow h(X) \leq h(Y)$$

où h est la fonction associant une valeur à la position d'un individu ou d'une classe d'individus dans la hiérarchie.

Si on prend par exemple le premier dendogramme (a) de la figure 1. On pose $X = \{1, 2\}$ et $Y = \{1, 2, 3\}$. X est contenu dans Y. La longueur $h(X)$ est plus petite que la longueur $h(Y)$ et ainsi de suite.

Cette valeur place les feuilles tout en bas et la racine tout en haut. La représentation graphique d'une hiérarchie valuée s'appelle un **dendrogramme**. Il est essentiel de comprendre d'entrée que cette représentation est très peu contrainte. On a ci-dessous 4 représentations (parmi un très grand nombre possible) d'une hiérarchie valuée.



Cette fiche introduit à la recherche d'une hiérarchie valuée pour décrire des données numériques puis à celle d'une partition pour les résumer.

2 Distance entre individus

La recherche d'une hiérarchie valuée s'appelle une classification hiérarchique (*hierarchical clustering*). Une telle recherche s'appuie sur une notion de distances entre individus qui induit une mesure de **l'hétérogénéité** d'une partie basée sur les distances entre individus qui sont dedans et une mesure de **dissimilarité** entre deux parties basée sur la distance entre un individu de l'un et un individu de l'autre.

Dans R, il existe un grand nombre de fonctions pour calculer des distances comme la fonction `dist` de la librairie `stats` ou les fonctions `dist.*` d'`ade4` (cf ci-dessous quelques exemples).

Fonctions	Données
`dist.binary`	variables dichotomiques (généralement présence / absence)
`dist.prop`	vecteurs de proportions dont la somme en ligne vaut 1
`dist.quant`	variables quantitatives

On étudie les estimations (GHE estimates) des différentes causes de mortalité dans 28 pays européens (extraction d'une base de données fournie par l'Organisation Mondiale de la Santé, 2015).
Récupérer le jeu de données `mortality_Europe.txt` :

```
me <- read.table("http://pbil.univ-lyon1.fr/R/donnees/mortality_Europe.txt", h=TRUE)
names(me)
 [1] "State" "Code"  "CS0"   "CS1"   "CS1A"  "CS1B"  "CS1C"  "CS1D"  "CS1E"  "CS2"
[11] "CS2A"  "CS2B"  "CS2C"  "CS2D"  "CS2E"  "CS2F"  "CS2H"  "CS2I"  "CS2J"  "CS2K"
[21] "CS2L"  "CS2M"  "CS2N"  "CS2P"  "CS3"   "CS3A"  "CS3B"
```

Les données sont codées comme suit :

Colonne	Signification
State	nom du pays européen
Code	trigramme associé au nom du pays
CS0	toutes les causes de mortalités
CS1	maladies transmissibles
CS1A	maladies infectieuses et parasitaires
CS1B	infections respiratoires
CS1C	conditions maternelles
CS1D	conditions post-natales
CS1E	déficiences nutritionnelles
CS2	maladies non transmissibles
CS2A	néoplasmes malins (e.g. cancer, leucémie, etc)
CS2B	autres néoplasmes
CS2C	diabète sucré
CS2D	troubles endocriniens, sanguins et immunitaires
CS2E	troubles mentaux et troubles liés à l'usage de drogue
CS2F	conditions neurologiques
CS2H	maladies cardiovasculaires
CS2I	maladies respiratoires chroniques
CS2J	maladies digestives
CS2K	maladies génitales
CS2L	maladies de peau

CS2M	maladies musculo-squelettiques
CS2N	anomalies congénitales
CS2P	mort subit du nourrisson
CS3	blessures
CS3A	blessures non volontaires (e.g. accidents de la route, chutes, etc)
CS3B	blessures volontaires (e.g. automutilation, crime, etc)

On sélectionne les maladies transmissibles et non transmissibles et on nomme les lignes du tableau par les codes des pays.

```
mame <- me[,c(5:9,11:24)]
rownames(mame) <- me$Code
head(mame)
```

```
    CS1A CS1B CS1C CS1D CS1E CS2A CS2B CS2C CS2D CS2E CS2F CS2H CS2I CS2J CS2K CS2L
AUT  0.5  0.9    0  0.1  0.0 21.0  0.7  3.1  1.1  0.6  4.1 32.7  3.7  3.2  2.0  0.1
BEL  1.6  6.2    0  0.2  0.4 29.2  1.5  1.8  1.5  0.7 11.0 30.5  8.7  5.2  3.1  0.3
HRV  0.4  0.6    0  0.1  0.0 14.1  0.3  1.7  0.1  0.4  2.1 25.6  2.3  2.2  1.2  0.0
CZE  1.9  3.3    0  0.2  0.3 27.2  0.6  3.8  0.7  0.5  4.4 50.8  5.0  4.8  1.7  0.2
DNK  0.8  2.2    0  0.1  0.1 16.3  0.3  1.4  0.5  0.9  4.9 12.2  4.6  2.4  0.9  0.1
EST  0.1  0.2    0  0.0  0.0  3.8  0.1  0.1  0.0  0.3  0.3  7.9  0.3  0.6  0.4  0.0
    CS2M CS2N CS2P
AUT  0.2  0.3    0
BEL  0.5  0.3    0
HRV  0.1  0.1    0
CZE  0.2  0.2    0
DNK  0.3  0.1    0
EST  0.0  0.0    0
```

On calcule la distance euclidienne entre les deux premiers pays du fichier : l'Autriche (AUT) et la Belgique (BEL) :

$$d_{12} = \sqrt{\sum_{j=1}^{19} (x_{1j} - x_{2j})^2}$$

```
sqrt(sum((mame[1,]-mame[2,])^2))
```

```
[1] 13.49074
```

que l'on peut retrouver à l'aide de la fonction `dist` :

```
dmame <- dist(mame)
as.matrix(dmame)[1,2]
```

```
[1] 13.49074
```

Exercice. Faire la même vérification entre l'Allemagne (ligne 9, DEU) et la France (ligne 8, FRA).

3 Distances et dendrogrammes

3.1 Principe général

En classification hiérarchique, on distingue les méthodes ascendantes et les méthodes descendantes. Les méthodes ascendantes créent une partie en regroupant deux parties existantes. Les méthodes descendantes divisent au contraire une partie existante pour en faire deux nouvelles.

Pour regrouper, il faut un critère. Au début, il est naturel de regrouper les deux individus les plus proches au sens de la dissimilarité de départ. Mais immédiatement après cette opération, on peut regrouper soit des individus, soit un individu et une classe, soit, un peu plus tard, deux classes. Plusieurs stratégies peuvent alors s'insérer dans le schéma général :

Etape 1. On dispose d'une matrice de dissimilarités entre n individus. Chaque individu donne une partie réduite à lui-même à laquelle on attribue la valeur 0. Prendre la plus petite valeur de cette matrice et faire avec le couple correspondant une partie à deux éléments. Attribuer à cette nouvelle partie une valeur positive. On a alors n-1 parties.

Exemple

	2	1	3	4	5	8	6	7	13	9	10	11	12
2	0	3	3	6	6	12	12	12	12	12	12	12	12
1	3	0	**1**	6	6	12	12	12	12	12	12	12	12
3	3	**1**	0	6	6	12	12	12	12	12	12	12	12
4	6	6	6	0	4	12	12	12	12	12	12	12	12
5	6	6	6	4	0	12	12	12	12	12	12	12	12
8	12	12	12	12	12	0	5	5	11	11	11	11	11
6	12	12	12	12	12	5	0	2	11	11	11	11	11
7	12	12	12	12	12	5	2	0	11	11	11	11	11
13	12	12	12	12	12	11	11	11	0	10	10	10	10
9	12	12	12	12	12	11	11	11	10	0	7	9	9
10	12	12	12	12	12	11	11	11	10	7	0	9	9
11	12	12	12	12	12	11	11	11	10	9	9	0	8
12	12	12	12	12	12	11	11	11	10	9	9	8	0

On regroupe les individus 1 et 3 en un couple $A = \{1, 3\}$ et $h(A) = 1$.

Etape 2. A chaque pas, on a m parties et une valeur $h(i)$ associée à chacune d'entre elles. Regrouper deux d'entre elles sur le critère **M** et attribuer à la réunion une valeur h supérieure ou égale à la valeur des deux composantes.

Exemple
On calcule une nouvelle matrice de dissimilarités en remplaçant les individus 1 et 3 par le couple **A**. On remarque que pour tout individu conservé, les distances au couple **A** sont égales (triangle isocèle).

	2	**A**	4	5	8	6	7	13	9	10	11	12
2	0	3	6	6	12	12	12	12	12	12	12	12
A	3	0	6	6	12	12	12	12	12	12	12	12
4	6	6	0	4	12	12	12	12	12	12	12	12
5	6	6	4	0	12	12	12	12	12	12	12	12
8	12	12	12	12	0	5	5	11	11	11	11	11
6	12	12	12	12	5	0	**2**	11	11	11	11	11
7	12	12	12	12	5	**2**	0	11	11	11	11	11
13	12	12	12	12	11	11	11	0	10	10	10	10
9	12	12	12	12	11	11	11	10	0	7	9	9
10	12	12	12	12	11	11	11	10	7	0	9	9
11	12	12	12	12	11	11	11	10	9	9	0	8
12	12	12	12	12	11	11	11	10	9	9	8	0

On regroupe les individus 6 et 7 en un couple $B = \{6, 7\}$ et $h(B) = 2$.

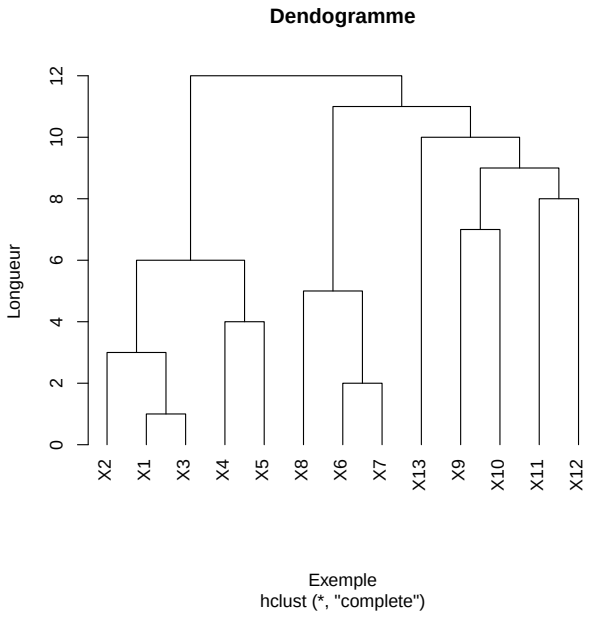
Etape 3. Recommencer jusqu'à ce qu'il ne reste que la classe regroupant le tout et lui attribuer une valeur supérieure à toutes les autres.

Exemple

	2	**A**	4	5	8	**B**	13	9	10	11	12
2	0	**3**	6	6	12	12	12	12	12	12	12
A	**3**	0	6	6	12	12	12	12	12	12	12
4	6	6	0	4	12	12	12	12	12	12	12
5	6	6	4	0	12	12	12	12	12	12	12
8	12	12	12	12	0	5	11	11	11	11	11
B	12	12	12	12	5	0	11	11	11	11	11
13	12	12	12	12	11	11	0	10	10	10	10
9	12	12	12	12	11	11	10	0	7	9	9
10	12	12	12	12	11	11	10	7	0	9	9
11	12	12	12	12	11	11	10	9	9	0	8
11	12	12	12	12	11	11	10	9	9	8	0

On regroupe les individus 1, 3 et 2 en un couple $C = \{1, 2, 3\}$, $h(C) = 3$ et ainsi de suite.

Les regroupements successifs peuvent être représentées par un **arbre** ou **dendogramme**.



En conclusion, chaque procédé qui définit **M** et h, respectivement le choix pour le regroupement et la fonction de valuation, donne une classification hiérarchique particulière. Parmi les procédés les plus répandus figurent d'abord ceux qui sont basés sur les distances entre parties.

3.2 Exemples

On considère une variable mesurée sur quatre individus.

```
 w <- c(0,1,2.1,3.3)
 w <- data.frame(w)
 w
    w
1 0.0
2 1.0
3 2.1
4 3.3
 (dw <- dist(w))
    1   2   3
2 1.0
3 2.1 1.1
4 3.3 2.3 1.2
 as.matrix(dw)
    1   2   3   4
1 0.0 1.0 2.1 3.3
2 1.0 0.0 1.1 2.3
3 2.1 1.1 0.0 1.2
4 3.3 2.3 1.2 0.0
```

La recherche d'une classification sur les individus dépend de différentes valeurs de h pour un même critère d'aggrégation $\mathbf{M}$. La fonction `hclust` de R en propose plusieurs et on va étudier les plus courantes.

1. Lien simple

Saut minimum = lien simple = single linkage = single $d(A,B) = min(d(a,b))$

Le résultat s'obtient à l'aide de la fonction `hclust`.

```
 hclust(dw,"single")
Call:
hclust(d = dw, method = "single")
Cluster method   : single
Distance         : euclidean
Number of objects: 4
```

Les valeurs retournées par la fonction `hclust` sont :

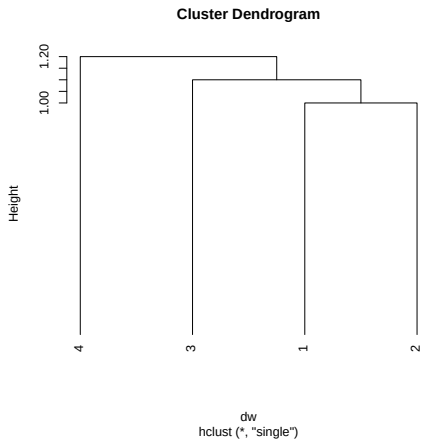
```
 res1 <- hclust(dw,"single")
 names(res1)
[1] "merge"       "height"      "order"       "labels"      "method"
[6] "call"        "dist.method"
 unclass(res1)
$merge
     [,1] [,2]
[1,]   -1   -2
[2,]   -3    1
[3,]   -4    2
$height
[1] 1.0 1.1 1.2

$order
[1] 4 3 1 2

$labels
NULL

$method
[1] "single"

$call
hclust(d = dw, method = "single")

$dist.method
[1] "euclidean"
```

`res1$merge` contient les différentes étapes de regroupement des individus. Une entrée négative indique un regroupement de singletons ; une entrée positive indique un regroupement de classes.
Dans l'exemple, la ligne 1 signifie que les deux singletons $\{1\}$ et $\{2\}$ ont été regroupés ; la ligne 2 indique que le singleton $\{3\}$ a été regroupé avec la première classe $\{1, 2\}$; la ligne 3 indique que le singleton $\{4\}$ a été regroupée avec la seconde classe $\{1, 2, 3\}$.

`res1$height` contient les longueurs des branches associant des individus et/ou des groupes d'individus entre eux.
Dans l'exemple, les hauteurs retenues pour les noeuds de la classification sont 1.0, 1.1 et 1.2.

Le dendogramme s'obtient par :

```
plot(hclust(dw,"single"),hang=-1)
```



2. Lien complet

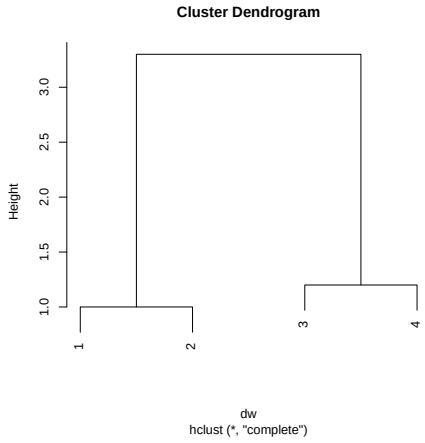
Agrégation par le diamètre = lien complet = complete linkage = complete
$d(A, B) = max(d(a, b))$

```
unclass(hclust(dw,"complete"))
```

```
$merge
     [,1] [,2]
[1,]   -1   -2
[2,]   -3   -4
[3,]    1    2
$height
[1] 1.0 1.2 3.3

$order
[1] 1 2 3 4

$labels
NULL

$method
```

```
[1] "complete"

$call
hclust(d = dw, method = "complete")

$dist.method
[1] "euclidean"
```

```
plot(hclust(dw,"complete"))
```

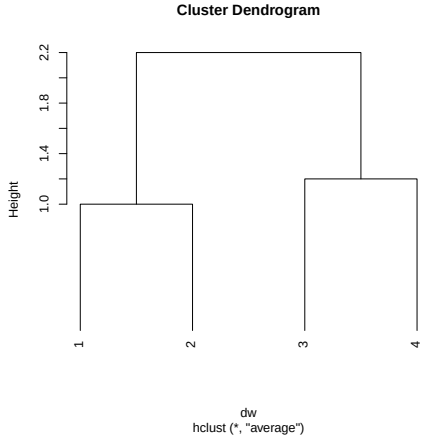


3. Lien moyen

Lien moyen = Unweighted Pair Group Method of Agregation (UGPMA) = average $d(A, B) = mean(d(a, b))$

```
plot(hclust(dw,"average"),han=-1)
unclass(hclust(dw,"average"))
```

```
$merge
     [,1] [,2]
[1,]   -1   -2
[2,]   -3   -4
[3,]    1    2
$height
[1] 1.0 1.2 2.2

$order
[1] 1 2 3 4

$labels
NULL

$method
[1] "average"

$call
hclust(d = dw, method = "average")

$dist.method
[1] "euclidean"
```



3.3 Exercice

On considère les données de mortalité contenues dans `mame`.

1. Examiner les données à l'aide par exemple de boites à moustaches et répondre à la question : vaut-il mieux travailler sur les données brutes ? les données centrées ? les données normées ?
2. Calculer les distances entre les sites selon une méthode associée aux données quantitatives.
3. Construire une classification hiérarchique ainsi que le dendogramme associé.
4. Commenter.

4 Une fonction de valuation particulière : le critère de Ward

C'est souvent le meilleur critère. On va détailler son fonctionnement mais pour en savoir plus encore, consulter l'excellent ouvrage de Lebart, Morineau, Piron [3].

Agrégation de Ward = Moment d'ordre 2 = Inertie minimale

4.1 Distances et variance

Le critère de Ward s'appuie sur la forte connexion entre les notions de distances et de variance. On a :

$$var_{\mathbf{p}}(\mathbf{x}) = \sum_{i=1}^{n} p_i (x_i - \bar{x})^2 = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} p_i p_j (x_i - x_j)^2$$
$$var(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2 = \frac{1}{2n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (x_i - x_j)^2$$

D'où la généralisation en terme d'inertie :

$$Iner\,(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i - \bar{\mathbf{x}}\|^2 = \frac{1}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\|\mathbf{x}_i - \mathbf{x}_j\|^2$$

D'où la généralisation en terme d'hétérogénéité :

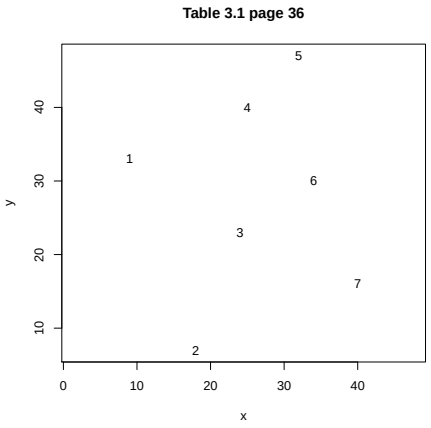
$$Heter\,(\mathbf{\Omega}) = \frac{1}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2$$

avec $\mathbf{\Omega}$ une collection de n objets et d_{ij}^2 le carré de la distance de l'objet i à l'objet j. On peut mesure de l'hétérogénéité dans une partie (inertie intra-classe) ou entre parties (inertie inter-classe). On peut faire de la statistique avec des matrices de distances entre objets.

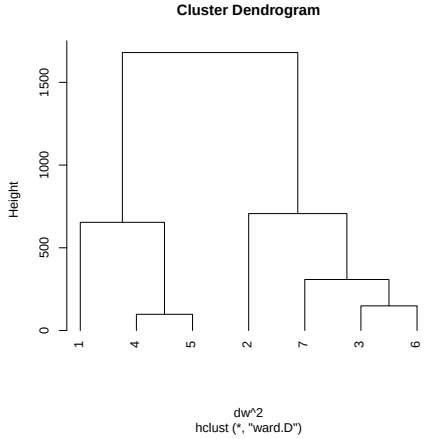
4.2 Principe

On utilise l'exemple page 36 de l'ouvrage de référence de Gordon [2].

```
x <- c(9,18,24,25,32,34,40)
y <- c(33,7,23,40,47,30,16)
(w <- cbind(x,y))
plot(w,type="n",asp=1, main = "Table 3.1 page 36")
text(w[,1],w[,2],1:7)
```




```
dw <- dist(w)
dw^2
hc1 <- hclust(dw^2,"ward.D")
unclass(hc1)
plot(hc1,hang=-1)
```



La matrice de départ est considérée comme la matrice de l'hétérogénéité de tous les groupements initiaux possibles. Au départ, l'inertie totale vaut l'inertie inter-classe et l'inertie intra-classe est nulle. L'objectif de l'algorithme est d'agréger les individus ou les classes afin de faire varier le moins possible l'inertie intra-classe à chaque étape. Ceci revient à rendre minimale la perte d'inertie inter-classe résultant de l'agrégation de deux parties. d_{ij}^2 est la valeur dont augmentera l'inertie intra-classe dans le regroupement si on passe d'une partition en n parties à un élément à une partition en n - 1 parties en groupant i et j.
Exemple

```
     1    2    3    4    5    6
2  757
3  325  292
4  305 1138  290
5  725 1796  640   98
6  634  785  149  181  293
7 1250  565  305  801 1025  232
```

Comme 4 et 5 sont groupés, on met à jour la matrice de l'hétérogénéité des groupements maintenant possibles. Elle a une ligne et une colonne en moins et toutes les valeurs des classes non modifiées sont conservées. On a seulement besoin de la valeur de l'hétérogénéité **nouvelle** engendrée par le groupement au pas suivant de $C_i \cup C_j$ (le groupement qu'on vient d'opérer) avec C_k, une classe quelconque héritée du tour précédent. Si on utilise une distance euclidienne en raisonnant sur les centres de gravité des classes, on trouve que l'accroissement de l'inertie intra-classe vaut :

$$I\left(C_i \cup C_j, C_k\right) = \frac{n_i + n_k}{n_i + n_j + n_k} I\left(C_i, C_k\right) + \frac{n_j + n_k}{n_i + n_j + n_k} I\left(C_j, C_k\right) - \frac{n_k}{n_i + n_j + n_k} I\left(C_i, C_j\right)$$

avec n_i le nombre d'objets (individus) dans la partie C_i. Ainsi, au lieu de chercher les deux éléments les plus proches, on cherche les éléments pour lequel cet accroissement est minimal. Par exemple :

$$I\left(\{4,5\}, \{3\}\right) = \frac{2}{3} I\left(\{4,3\}\right) + \frac{2}{3} I\left(\{5,3\}\right) - \frac{1}{3} I\left(\{4,3\}\right) = \frac{2}{3} 290 + \frac{2}{3} 640 - \frac{1}{3} 98 = 587.3$$

D'où le nouvel indice entre parties (`a` est le regroupement de `4` et `5`) :

```
     1    2     3     a      6
2  757
3  325  292
a  654 1923.3 587.3
6  634  785   149    283.3
7 1250  565   305   1184.7  232
```

On recommence (`b` est le regroupement de `3` et `6`) :

```
     1      2     b     a
2  757
b  589.7  668.3
a  654   1923.3 587.3
7 1250    565   308.3 1184.7
```

Le tableau complet est dans Gordon [2] page 84. Tous les justificatifs sont dans Benzécri *et al* [1] (2.5.2 p. 187). On retiendra qu'une méthode prend tout aussi bien $d_{ij}, \sqrt{d_{ij}}, d_{ij}^2, ...$ en entrée. C'est un reproche qu'on fait souvent à ce type de méthodes qui est peu contrariant sur les input. Avec le critère de Ward, la justification euclidienne implicite rend logique l'usage des carrés d'une distance euclidienne.

4.3 Exercices

Exercice 1

On considère à nouveau les données de mortalité en Europe contenues dans l'objet `mame`. Comparer les classifications obtenues sur la matrice de distances, la racine carrée de la matrice de distances ou le carré de la matrice de distances euclidiennes.

Exercice 2

On considère les données de mortalité en Europe contenues dans l'objet `mame`. Construire et commenter la classification associée à la méthode de Ward sur la matrice de distances construite au paragraphe 2.

5 Recherche d'une partition

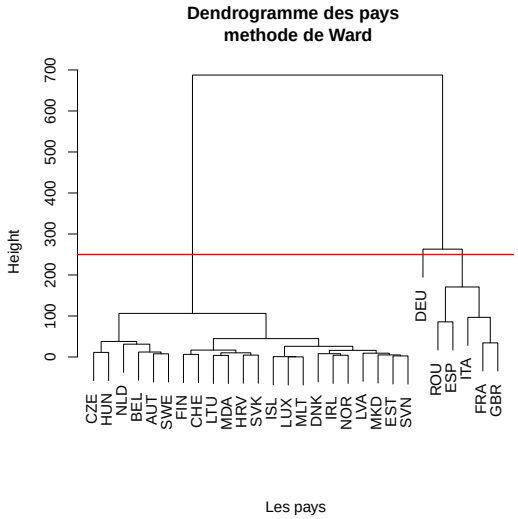
La recherche d'une partition revient à couper un arbre à partir soit d'une hauteur donnée, soit d'un nombre de classes défini. Pour ce faire, on utilise la fonction bien nommée `cutree()`.

5.1 Exemple de la mortalité dans les pays européens

On note `hexo2` la classification - méthode de Ward - des pays obtenue dans le paragraphe 4.3.2. On choisit de couper l'arbre à la hauteur 250.

```
plot(hexo2, main="Dendrogramme des pays \n methode de Ward", xlab = "Les pays", sub = "")
abline(h=250, col="red", lwd=1.5)
parti <- cutree(hexo2,h=250)
parti
```

```
AUT BEL HRV CZE DNK EST FIN FRA DEU HUN ISL IRL ITA LVA LTU LUX MLT NLD NOR MDA ROU
  1   1   1   1   1   1   1   2   3   1   1   1   2   1   1   1   1   1   1   1   2
SVK SVN ESP SWE CHE MKD GBR
  1   1   2   1   1   1   2
```

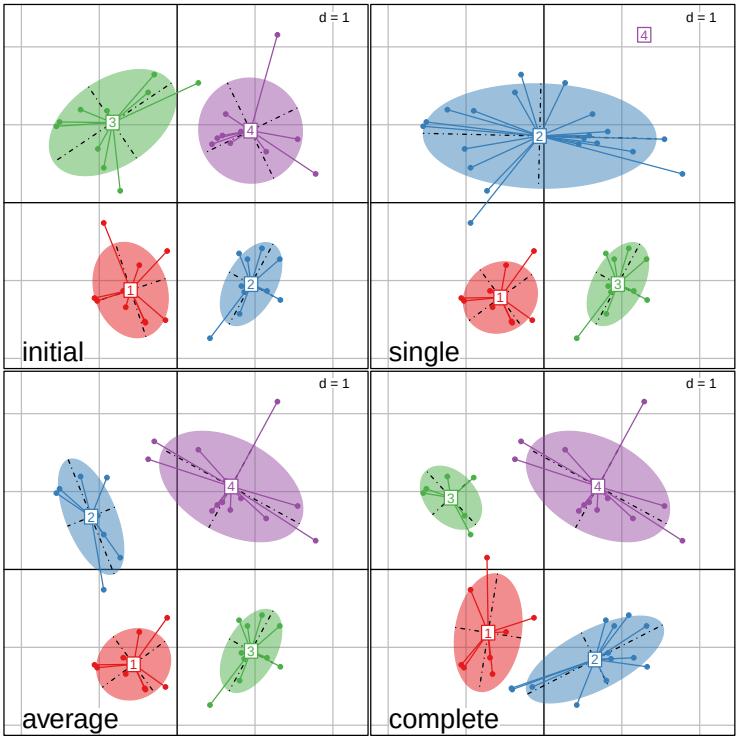


Interpréter.

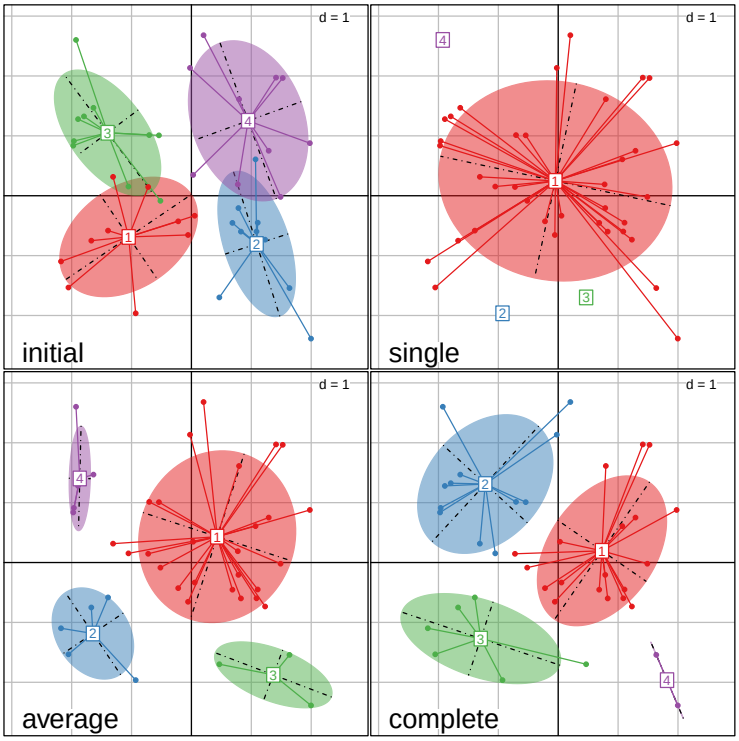
5.2 Introduction à la fonction `kmeans`

On peut toujours trouver légitime de partager en paquets un ensemble de points même régulièrement répartis dans l'espace. Le problème est de ne pas faire d'erreurs grossières, lesquelles se voient bien en dimension 2, mais se cachent sans peine en dimension quelconque.

```
library(mvtnorm)
fc <- function(sd) {
         x1 <- rmvnorm(10, mean = c(-1, -1), sig=diag(sd, 2))
          x2 <- rmvnorm(10, mean = c(1, -1), sig=diag(sd, 2))
      x3 <- rmvnorm(10, mean = c(-1, 1), sig=diag(sd, 2))
      x4 <- rmvnorm(10, mean = c(1, 1), sig=diag(sd, 2))
      x <- rbind(x1,x2,x3,x4)
      init <- factor(rep(1:4,rep(10,4)))
#
gfi <- s.class(x, init, psub.text="initial",psub.cex=2, col=TRUE, plot=FALSE)
#
h0 <- hclust(dist(x),"single")
parti <- as.factor(cutree(h0,k=4))
gfs <- s.class(x, parti, psub.text="single", psub.cex=2, col=TRUE, plot=FALSE)
#
h0 <- hclust(dist(x),"average")
parti <- as.factor(cutree(h0,k=4))
gfa <- s.class(x, parti, psub.text="average", psub.cex=2, col=TRUE, plot=FALSE)
#
h0 <- hclust(dist(x),"complete")
parti <- as.factor(cutree(h0,k=4))
gfc <- s.class(x, parti, psub.text="complete", psub.cex=2, col=TRUE, plot=FALSE)
#
ADEgS(list(gfi,gfs,gfa,gfc), layout=c(2,2))
}
fc(sd=0.25)
```

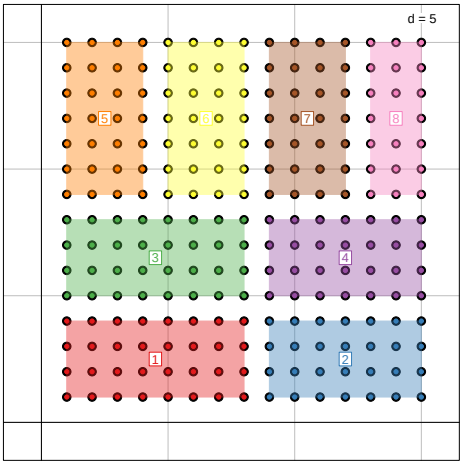

d = 1
initial
single
average
complete

```
fc(sd=0.5)
```


d = 1
initial
single
average
complete

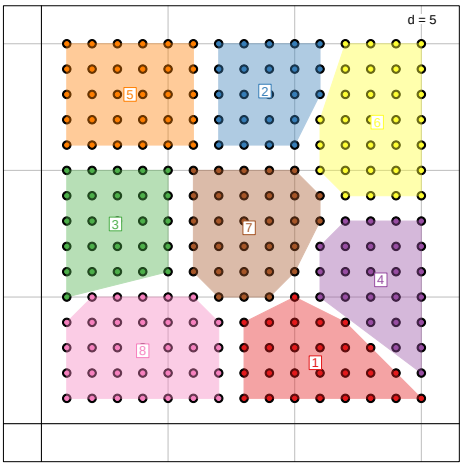
Avec une grille régulière :

```
w <- expand.grid(1:15,1:15)
#g1 <- s.label(w, plabels.cex=0, ppoints.cex =2)
#g2 <- s.class(as.data.frame(w),as.factor(cutree(hclust(dist(w)^2,"single"),8)),
#         chullSize=1, ellipseSize=0, starSize=0, col=TRUE)
#g1+g2
g3 <- s.label(w,plabels.cex=0,ppoints.cex=2, plot=FALSE)
g4 <- s.class(as.data.frame(w),as.factor(cutree(hclust(dist(w)^2,"ward.D2"),8)),
        chullSize=1, ellipseSize=0, starSize=0, col=TRUE, plot=FALSE)
g3+g4
```

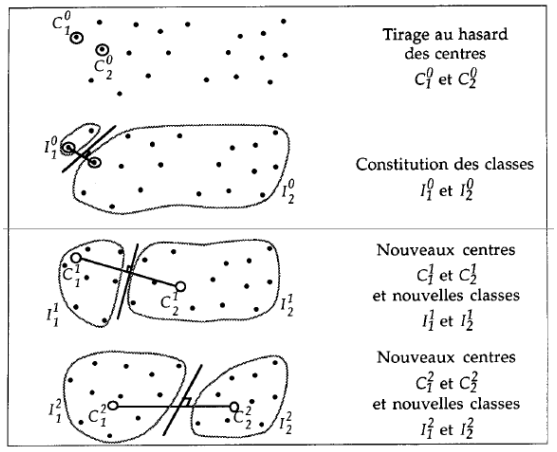


On doit pouvoir faire mieux :

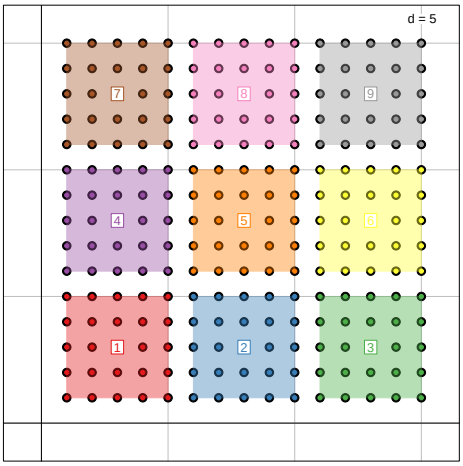
```
g5 <- s.label(w,plabels.cex=0,ppoints.cex=2, plot=FALSE)
g6 <- s.class(as.data.frame(w),as.factor(kmeans(w,8)$cluster),chullSize=1,
        ellipseSize=0, starSize=0, col=TRUE, plot=FALSE)
g5+g6
```



C'est mieux, mais pas toujours la même chose ! Il s'agit d'une agrégation autour des centres mobiles. La figure suivante résume parfaitement la situation :
La fonction calcule à chaque étape les centres de gravité des classes puis réaffecte chaque point au centre le plus proche. Elle accepte en entrée soit le nombre de classes (dans ce cas, la première série de centres est tirée au hasard), soit une liste de points qui serviront de centres de départ.




```
g7 <- s.label(w, plabels.cex=0,ppoints.cex=2, plot=FALSE)
cent <- expand.grid(c(3,8,13),c(3,8,13))
g8 <- s.class(as.data.frame(w),as.factor(kmeans(w,cent)$cluster),chullSize=1,
      ellipseSize=0, starSize=0, col=TRUE, plot=FALSE)
g7+g8
```



Sur les données de mortalité., on choisit de définir 4 groupes.

```
res4 <- kmeans(mame, 4)
names(res4)
```

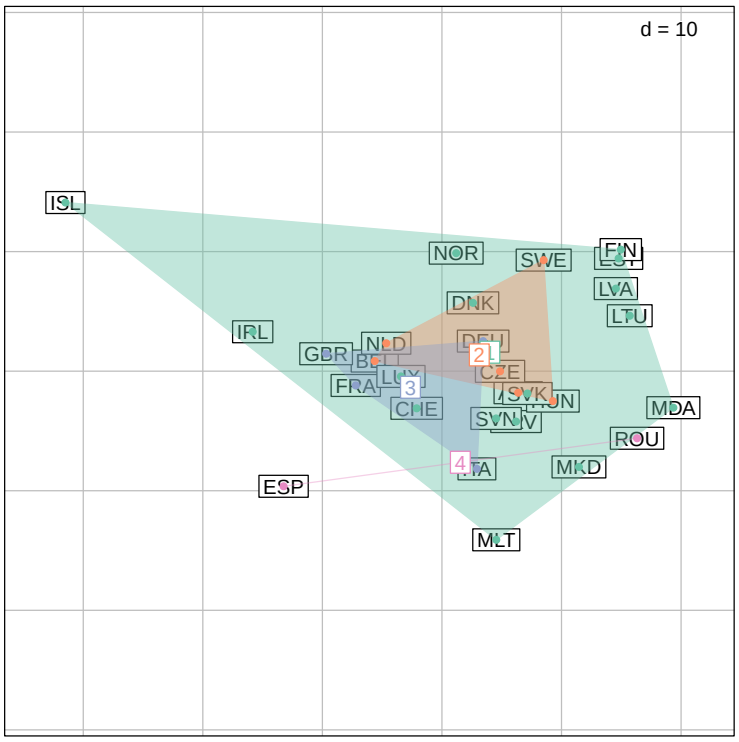
```
[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

```
res4$cluster
```

```
AUT BEL HRV CZE DNK EST FIN FRA DEU HUN ISL IRL ITA LVA LTU LUX MLT NLD NOR MDA ROU
  2   2   1   2   1   1   1   3   3   2   1   1   3   1   1   1   1   2   1   1   4
SVK SVN ESP SWE CHE MKD GBR
  1   1   4   2   1   1   3
```

Bien que rien ne laisse penser que la répartition spatiale des pays puisse être liée à la mortalité, on choisit de réaliser une représentation graphique permettant de visualiser les classes à partir de ces coordonnées *via* leur capitale. Ainsi,

```
library(RColorBrewer)
couleur <- brewer.pal(4,"Set2")
coordxy <- read.table("Coord28paysEurope.txt", h=TRUE)
gcapital <- s.label(coordxy[,c(5,4)], labels=coordxy$Code, porigin.include=FALSE, plot=FALSE)
gcluster <- s.class(coordxy[,c(5,4)], as.factor(res4$cluster),chullSize=1, ellipseSize=0, starSize=0, col=coul
gcapital+gcluster
```



Réaliser plusieurs fois l'analyse et comparer les résultats avec ce qui a été obtenu avec la méthode de Ward. Conclure.

Références

[1] J.P. Benzecri. *L'analyse des données. T.1 : La taxinomie.* Dunod, Paris, 1973.

[2] A.D. Gordon. *Classification.* Chapman & Hall, London, 2 edition, 1999.

[3] L. Lebart, A. Morineau, and M. Piron. *Statistique exploratoire multidimensionnelle.* Dunod, Paris, 1995.

Annexe

Cette annexe contient l'ensemble des matrices de distances calculées selon les différentes valeurs de h dans l'exemple 3.2.

1. Lien Simple

	1	2	3	4
1	0	1.0	2.1	3.3
2	1.0	0	1.1	2.3
3	2.1	1.1	0	1.2
4	3.3	2.3	1.2	0

On regroupe les singletons 1 et 2 : $A = \{1, 2\}$

	A	3	4
A	0		
3	1.1	0	
4	2.3	1.2	0

On regroupe le singleton 3 avec la classe A : $B = \{1, 2, 3\}$

	B	4
B	0	
4	1.2	0

2. Lien Complet

	1	2	3	4
1	0	1.0	2.1	3.3
2	1.0	0	1.1	2.3
3	2.1	1.1	0	1.2
4	3.3	2.3	1.2	0

On regroupe les singletons 1 et 2 : $A = \{1, 2\}$

	A	3	4
A	0		
3	2.1	0	
4	3.3	1.2	0

On regroupe le singleton 3 avec la classe A : $B = \{1, 2, 3\}$

	B	4
B	0	
4	3.3	0

3. Lien Moyen

	1	2	3	4
1	0	1.0	2.1	3.3
2	1.0	0	1.1	2.3
3	2.1	1.1	0	1.2
4	3.3	2.3	1.2	0

On regroupe les singletons 1 et 2 : $A = \{1, 2\}$

	A	3	4
A	0		
3	1.6	0	
4	2.8	1.2	0

On regroupe les singletons 3 et 4 : $B = \{3, 4\}$

	A	B
A	0	
B	2.2	0