Fiche TD avec le logiciel R : tdr23

Dendrogrammmes

D. Chessel, A.B. Dufour & J.R. Lobry

Introduction à la représentation d'une hiérarchie de partitions.

Table des matières

1 Introduction

La variance s'écrit aussi :

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

D'où la généralisation (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 (hétérogénéité) :

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

où $\mathbf{\Omega}$ est une collection de n objets et d_{ij}^2 le carré de la distance de l'objet i à l'objet j. On peut faire de la statistique avec des matrices de distances entre objets.

2 Distances et dendrogrammes

2.1 Calculer une matrice de distances

Étudier la fonction `dist()` de la librairie `stats`. Extrait de la documentation (`?dist`) :

```
dist                    package:stats                    R Documentation

Distance Matrix Computation

Description:

     This function computes and returns the distance matrix computed by
     using the specified distance measure to compute the distances
     between the rows of a data matrix.
```

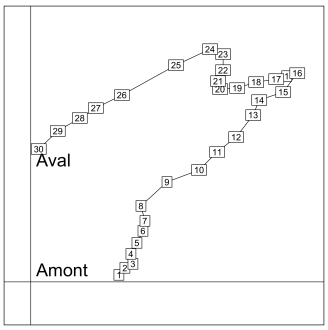
Récupérer le jeu de données `doubs` [**?**] de la bibliothèque `ade4` :

```
library(ade4)
data(doubs)
```

Consulter la documentation (`?doubs`) :

```
doubs                   package:ade4                    R Documentation
Pair of Ecological Tables
Description:
     This data set gives environmental variables, fish species and
     spatial coordinates for 30 sites.
Format:
     doubs is a list with 3 components.
     mil is a data frame with 30 rows (sites) and 11 environmental
          variables.
     poi is a data frame with 30 rows (sites) and 27 fish species.
     xy is a data frame with 30 rows (sites)  and 2 spatial
          coordinates.
```

30 sites sont régulièrement disposés le long du Doubs, affluent de la Saône qui passe à Besançon. A gauche, un schéma de la région indique la position du Doubs :



Refaire la figure de droite avec :

```
margesordi <- par("mar")
par(mar=c(0.1,0.1,0.1,0.1))
s.traject(doubs$xy,grid = F)
s.label(doubs$xy,add.plot=T)
text(0,10,"Amont", cex =2, pos = 4)
text(0,120,"Aval", cex =2, pos = 4)
par(mar=margesordi)
```

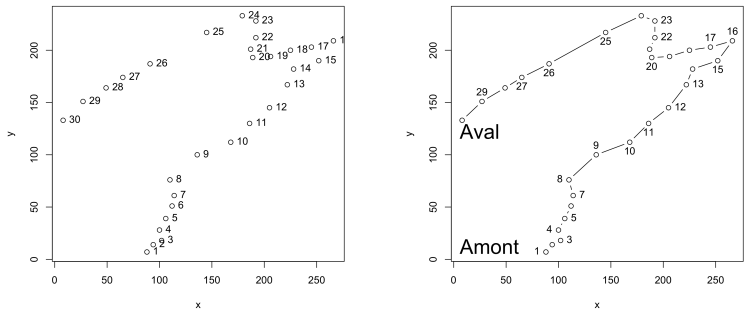
On peut s'exercer à l'étiquetage des figures. Le plus simple est (ci-dessous, à gauche) :

```
plot(doubs$xy)
text(doubs$xy, label = rownames(doubs$xy), pos = 4)
```

Ce n'est pas très joli à cause des superpositions. On raffine avec la fonction interactive `identify()` en cliquant (en bas, à gauche, en haut ou à droite pour choisir la position du label) de quelques points d'intérêt.

```
txtst <- identify(doubs$xy, pos = TRUE)
```

Vous devez obtenir un résultat du type (ci-dessous à droite) :



Intéressons nous maintenant au tableau `doubs$poi` :

```
poi <- doubs$fish
names(poi)
```

```
 [1] "Cogo" "Satr" "Phph" "Neba" "Thth" "Teso" "Chna" "Chto" "Lele" "Lece" "Baba"
[12] "Spbi" "Gogo" "Eslu" "Pefl" "Rham" "Legi" "Scer" "Cyca" "Titi" "Abbr" "Icme"
[23] "Acce" "Ruru" "Blbj" "Alal" "Anan"
```

On a affaire à 27 espèces de Poissons (tableau 1). La cinquième est l'ombre commun. Vous pouvez chercher la photo des autres.

`http://www.peche.org/poissons_eau_douce/ombre.jpg`

Vérifions à la main le calcul des distances de `dist()`, par exemple entre la première et la deuxième station :

1	CHA	chabot	*Cottus gobio*
2	TRU	truite	*fario Salmo trutta fario*
3	VAI	vairon	*Phoxinus phoxinus*
4	LOC	loche franche	*Nemacheilus barbatulus*
5	OMB	ombre commun	*Thymallus thymallus*
6	BLA	blageon	*Leuciscus soufia*
7	HOT	hotu	*Chondrostoma nasus*
8	TOX	toxostome	*Chondrostoma toxostoma*
9	VAN	vandoise	*Leuciscus leuciscus*
10	CHE	chevesne	*Leuciscus cephalus*
11	BAR	barbeau	*Barbus fluviatilis*
12	SPI	spirlin	*Alburnoides bipunctatus*
13	GOU	goujon	*Gobio gobio*
14	BRO	brochet	*Esox Lucius*
15	PER	perche	*Perca fluviatilis*
16	BOU	bouvière	*Rodeus sericeus*
17	PSO	perche soleil	*Lepomis gibbosus*
18	ROT	rotengle	*Scardinius erythrophtalmus*
19	CAR	carpe	*Cyprinus carpio*
20	TAN	tanche	*Tinca tinca*
21	BCO	brême commune	*Abramis brama*
22	PCH	poisson-chat	*Ictalurus melas*
23	GRE	grémille	*Gymnocephalus cernua*
24	GAR	gardon	*Rutilus rutilus*
25	BBO	brême bordelière	*Blicca bjoerkna*
26	ABL	ablette	*Alburnus alburnus*
27	ANG	anguille	*Anguilla anguilla*

TABLE 1 : Code des espèces du tableau doubs$poi.

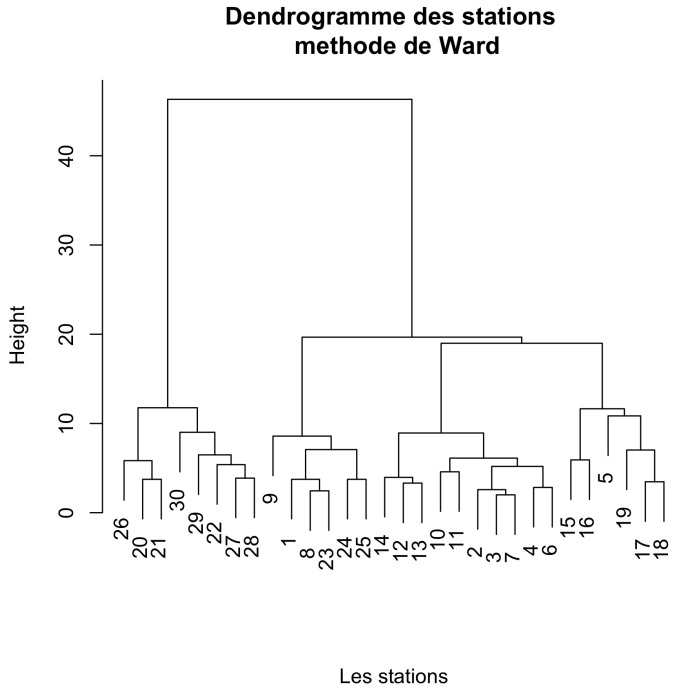
```
sum((poi[1,]-poi[2,])^2)
[1] 29
dpoi <- dist(poi)
as.matrix(dpoi)[1,2]^2
[1] 29
```

Faire la même vérification entre la troisième et la quatrième station. Etudier les distances disponibles pour les données en présence-absence (`dist.binary()`).

2.2 Tracer un dendrogramme

Utiliser la fonction `hclust()` :

```
h0 <- hclust(d = dpoi, method = "ward.D2")
plot(h0, main="Dendrogramme des stations \n methode de Ward", xlab = "Les stations", sub = "")
```



Ce graphique est un dendrogramme, représentation d'une hiérarchie de partitions valuée.

- ⋆ *Hiérarchie de partitions*
- ⋆ *Agrégation hiérarchique*
- ⋆ *Distances entre groupes*

```
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)
dw    # voir methods("print") print
```

```
   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
```

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

Les objets hclust :

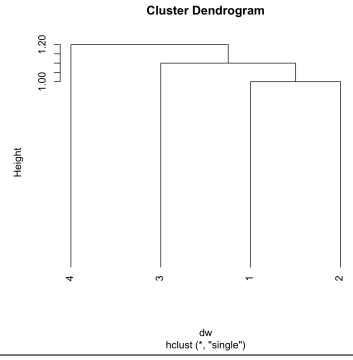
```
hclust(dw,"single")
```

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

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

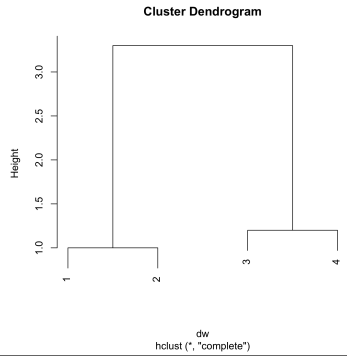
```
$merge
     [,1] [,2]
[1,]   -1   -2
[2,]   -3    1
[3,]   -4    2
$height
[1] 1.0 1.1 1.2

$order
[1] 4 3 1 2

$labels
NULL

$method
[1] "single"

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

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

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



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

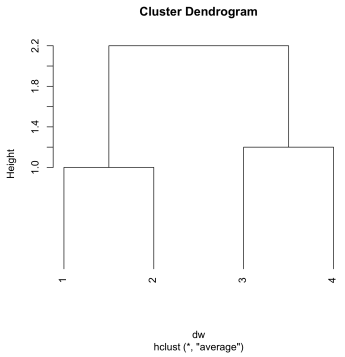
```
 unclass(hclust(dw,"complete"))
$merge
     [,1] [,2]
[1,]   -1   -2
[2,]   -3   -4
[3,]    1    2
$height
[1] 1.0 1.2 3.3

$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"))
```



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

```
 plot(hclust(dw,"average"),han=-1)
 unclass(hclust(dw,"average"))
$merge
     [,1] [,2]
[1,]   -1   -2
```

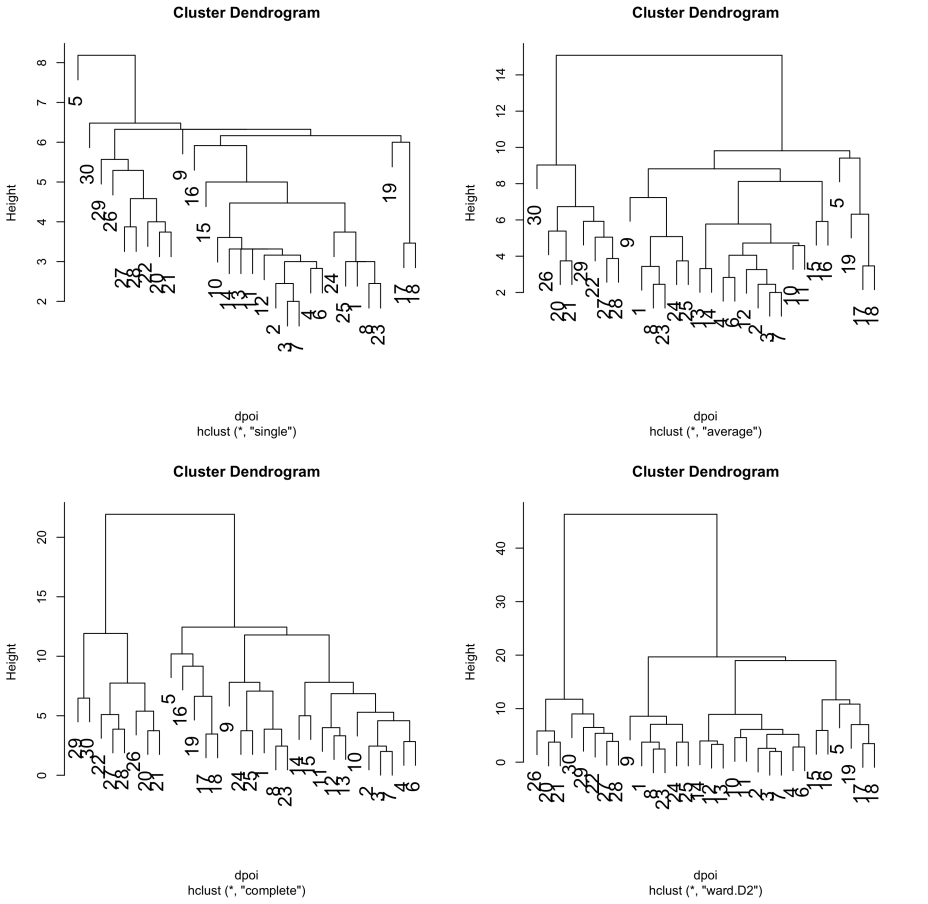
```
[2,]   -3   -4
[3,]    1    2
$height
[1] 1.0 1.2 2.2

$order
[1] 1 2 3 4

$labels
NULL

$method
[1] "average"

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

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



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

Pour en savoir plus, consulter l'excellent ouvrage de Lebart, Morineau et Piron [**?**].

```
old.par <- par(no.readonly = TRUE)
par(mfrow=c(2,2))
plot(hclust(dpoi,"single"),cex=1.5)
plot(hclust(dpoi,"average"),cex=1.5)
plot(hclust(dpoi,"complete"),cex=1.5)
plot(hclust(dpoi,"ward.D2"),cex=1.5)
par(old.par)
```



Extrait de la documentation (`?hclust`) :

> A number of different clustering methods are provided. _Wards_ minimum variance method aims at finding compact, spherical clusters. The _complete linkage_ method finds similar clusters. The _single linkage_ method (which is closely related to the minimal spanning tree) adopts a friends of friends clustering strategy. The other methods can be regarded as aiming for clusters with characteristics somewhere between the single and complete link methods.

3 Dendrogrammes et classification

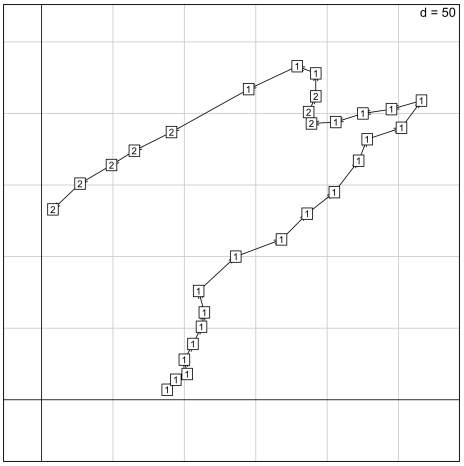
Couper un arbre se fait avec `cutree()`. On peut couper à la hauteur voulue ou pour un nombre de classes choisi. La fonction mérite bien son nom.

```
parti <- cutree(hclust(dpoi,"ward.D2"),h=20)
parti
```

```
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  2  2  2  1  1  1  2  2  2
29 30
 2  2
```

```
s.traject(doubs$xy,clab=0)
s.label(doubs$xy,lab=as.character(parti),add.p=T,cla=0.75)
round(data.frame(lapply(split(poi,parti),function(x) apply(x,2,mean))),dig=2)
```

```
      X1   X2
Cogo 0.68 0.00
Satr 2.55 0.12
Phph 2.95 0.38
Neba 2.95 1.00
Thth 0.64 0.12
Teso 0.82 0.12
Chna 0.23 1.62
Chto 0.64 1.50
Lele 1.14 2.25
Lece 1.41 3.12
Baba 0.59 3.75
Spbi 0.45 2.12
Gogo 0.91 4.38
Eslu 0.64 3.25
Pefl 0.64 2.75
Rham 0.23 3.50
Legi 0.23 3.00
Scer 0.18 2.12
Cyca 0.18 2.62
Titi 0.59 4.00
Abbr 0.05 3.12
Icme 0.00 2.25
Acce 0.23 4.12
Ruru 1.09 4.88
Blbj 0.09 3.62
Alal 0.77 5.00
Anan 0.14 3.00
```



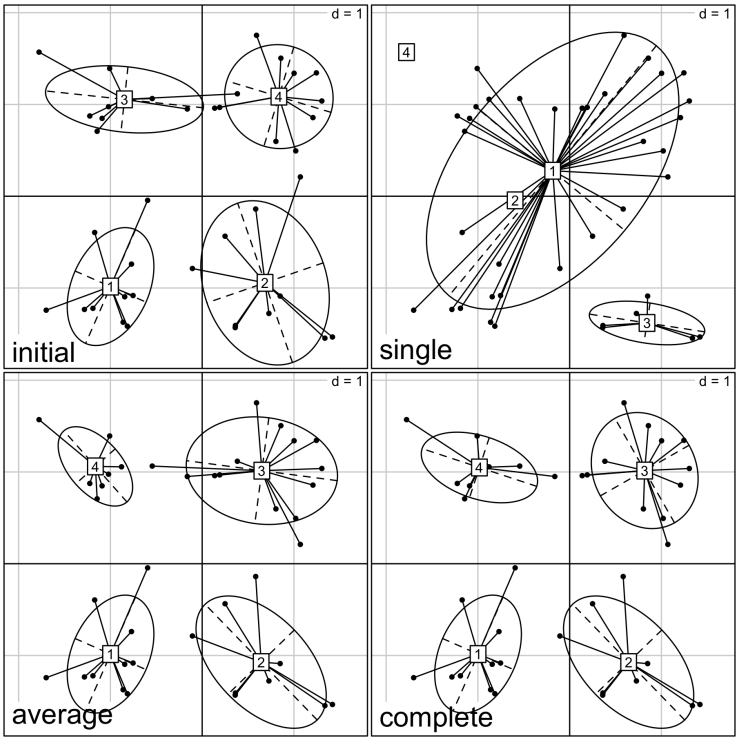
Interpréter.

4 Simulations

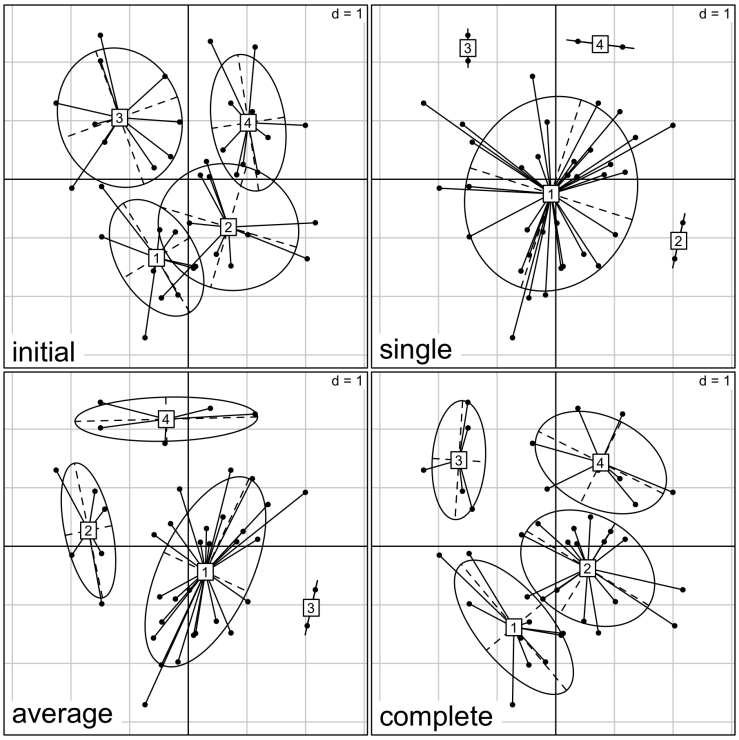
Ecrire une fonction pour explorer la stabilité des classifications par coupe de dendrogrammes. Par exemple (`rmvnorm()` est disponible dans `mvtnorm`) :

```
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)))
      old.par <- par(no.readonly = TRUE)
      par(mfrow=c(2,2))
```

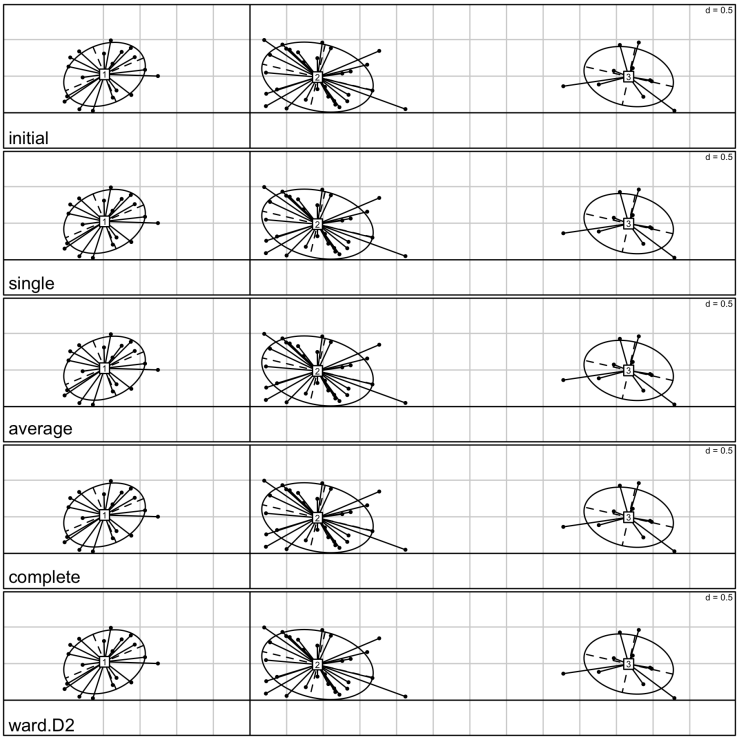
```
      s.class(x,init,sub="initial",csub=2)
      h0 <- hclust(dist(x),"single")
      parti <- as.factor(cutree(h0,k=4))
      s.class(x,parti,sub="single",csub=2)
      h0 <- hclust(dist(x),"average")
      parti <- as.factor(cutree(h0,k=4))
      s.class(x,parti,sub="average",csub=2)
      h0 <- hclust(dist(x),"complete")
parti <- as.factor(cutree(h0,k=4))
      s.class(x,parti,sub="complete",csub=2)
      par(old.par)
}
fc(sd=0.25)
```




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




```
fu <- function(sd) {
      fu1 <- function(method) {
            h0 <- hclust(dist(x),method)
            parti <- as.factor(cutree(h0,k=3))
            s.class(x,parti,sub=method,csub=2,ylim=c(0,1))
      }
      x1 <- rnorm(20,m=-2,sd=sd)
      x2 <- rnorm(30,m=1,sd=sd)
      x3 <- rnorm(10,m=5,sd=sd)
      y <- runif(60)
      x <- cbind(c(x1,x2,x3),y)
      init <- factor(rep(1:3,c(20,30,10)))
      old.par <- par(no.readonly = TRUE)
      par(mfrow=c(5,1))
      s.class(x,init,sub="initial",csub=2)
      fu1("single")
      fu1("average")
      fu1("complete")
      fu1("ward.D2")
      par(old.par)
}
fu(0.5)
```



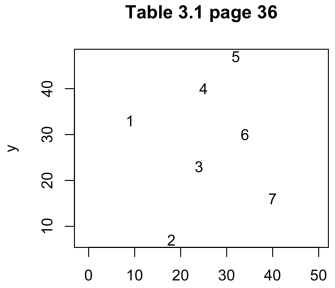
Voir les célèbres iris de Fisher (`?iris`) :

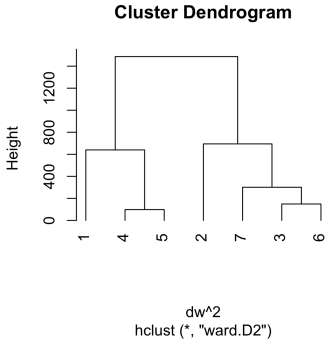
```
pairs(iris)
plot(hclust(dist(iris[,1:4]),"ward.D2"))
```

5 Critère de Ward

C'est souvent le meilleur. On détaille son fonctionnement. Utiliser l'exemple page 36 de l'ouvrage de de référence de Gordon [**?**].

```
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.D2")
unclass(hc1)
plot(hc1,hang=-1)
```





La matrice de départ est considérée comme la matrice de l'hétérogénéité de tous les groupements de départ possible. d_{ij}^2 est la valeur dont diminuera l'inertie intra-classe si on passe d'une partition en n parties à un élément à une partition en n - 1 parties en groupant i et j.

```
     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 :

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

Par exemple :

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

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

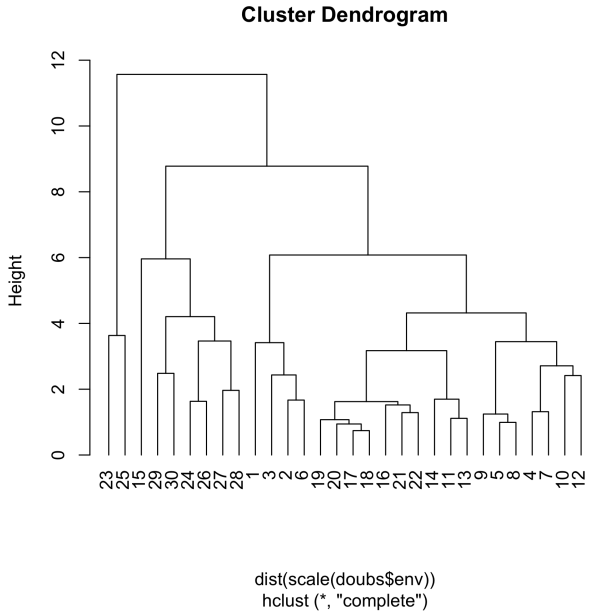
```
     1    2   3     a     5       6
2  757
3  325  292
a  654 1923 587.3
6  634  785 149   181   283.3
7 1250  565 305   801  1184.7  232
```

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 [**?**] p. 84. Tous les justificatifs dans Benzecri [**?**] (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.

```
plot(hclust(dist(scale(doubs$env))),hang=-1)
```




```
plot(hclust(dist(scale(doubs$env))^2),hang=-1)
```