PRINCIPALES METHODES
DE CLASSIFICATION NON SUPERVISEE

Livres :

- *Approche pragmatique de la classification: arbres hiérarchiques, partitionnements.* J-P. Nakache, J. Confais. Technip 2004 (site internet)
- *Finding Groups in Data: An Introduction to Cluster Analysis*. Kaufman, L. and Rousseeuw, P.J. (1990). Wiley, New York.

Sites :

- http://zoonek2.free.fr/UNIX/48_R_2004/all.html
- http://www.aliquote.org/articles/tech/multvar/multvar.html
- http://www.jstatsoft.org/v01/i04 (article en ligne)

Calcul du tableau de dissimilarité

Les différentes méthodes ci-dessous utilisent un tableau de dissimilarité à l'exception de MONA. La construction de ces tableaux peut être réalisée dans R avec :

- **dist(x, method = "euclidean", diag = FALSE, upper=FALSE,p=2)**
  Available distance measures are (written for two vectors *x* and *y*):
  - `euclidean`: Usual square distance between the two vectors (2 norm).
  - `maximum`: Maximum distance between two components of *x* and *y* (supremum norm)
  - `manhattan`: Absolute distance between the two vectors (1 norm).
  - `canberra`: *sum(|x_i - y_i| / |x_i + y_i|)*. Terms with zero numerator and denominator are omitted from the sum and treated as if the values were missing.
  - `binary`: (aka *asymmetric binary*): The vectors are regarded as binary bits, so non-zero elements are 'on' and zero elements are 'off'. The distance is the *proportion* of bits in which only one is on amongst those in which at least one is on.
  - `minkowski`: The *p* norm, the *p*th root of the sum of the *p*th powers of the differences of the components.

- **`dist.dudi(dudi, amongrow = TRUE)`**
  - dudi représente un schéma de dualité (acp, afc ...)
  - amongrow=T indique la dimension sur laquel le calcul doit être réalisé
  - dist.dudi(dudi.cpa(cidre),amongrow=TRUE) réalise le calcul de la matrice des distances entre les individus
  - dist.dudi(dudi.coa(csp),amongrow=F) réalise le calcul de la matrice des distances entre sur les profils colonnes.

- **daisy(x, metric = c("euclidean", "manhattan", "gower"),stand = FALSE)**
  - x : numeric matrix or data frame. Dissimilarities will be computed between the rows of `x`. Columns of mode `numeric` (i.e. all columns when `x` is a matrix) will be recognized as interval scaled variables, columns of class `factor` will be recognized as nominal variables, and columns of class `ordered` will be recognized as ordinal variables. Other variable types should be specified with the `type` argument. Missing values (`NA`s) are allowed.
  -

I CLASSIFICATION PAR PARTITION (kMEANS PAM FANNY)

Objectif : On considère un ensemble de n objets. On dispose :

- **cas 1 :** soit chaque objet est caractérisé par des variables à partir desquelles une dissimilarité peut être calculée : kmeans.
- **cas2 :** soit d'une matrice de dissimilarité : PAM, CLARA, nuées dynamiques.

L'objectif de la méthode est alors de déterminer une partition de cet ensemble en k classes, k étant fixé a priori.

A Méthodes des k-means (centres mobiles)

Les objets sont réparties en k classes de façon à minimiser la fonction objective

$$C = \sum_{j=1}^{k} \sum_{i \in I_j} p_i \, d^2(i, g_j)$$

Algorithme k means

1. Choisir k centres mobiles (en général au hasard)
2. Affecter chaque objet au centre le plus proche de façon à minimiser C.
3. Calculer les centres de gravités des classes définissant de nouveaux centres mobiles.
4. On reprend les étapes 1 à 3 jusqu'à obtenir une partition stable.

Avec la méthode de Forgy, les centres mobiles initiaux sont tirés au sort parmi les n objets (voir le cours).
Avec la méthode de MacQueen, les centres mobiles sont recalculés à chaque réaffectation d'un objet.

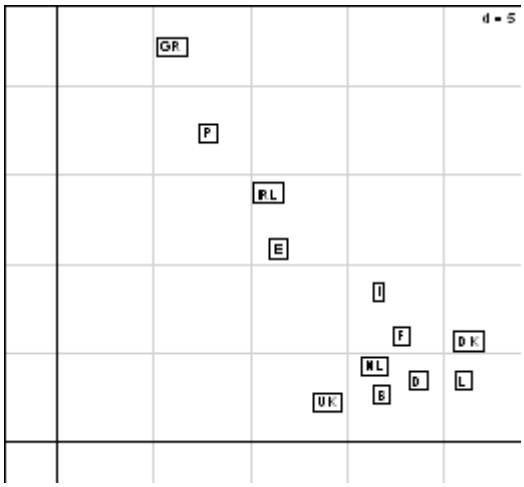
Avec R, on utilise la fonction :

```
kmeans(x, centers, iter.max = 10, nstart = 1,
       algorithm = c("Hartigan-Wong", "Lloyd", "Forgy","MacQueen"))
```

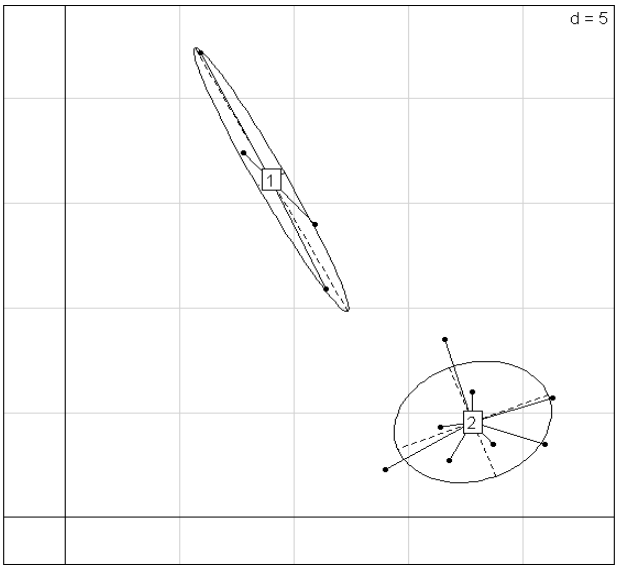
- x est un tableau numérique : objet $\times$ variables numériques
- kmeans(x,k)$cluster est un vecteur définissant la classe de chaque objet

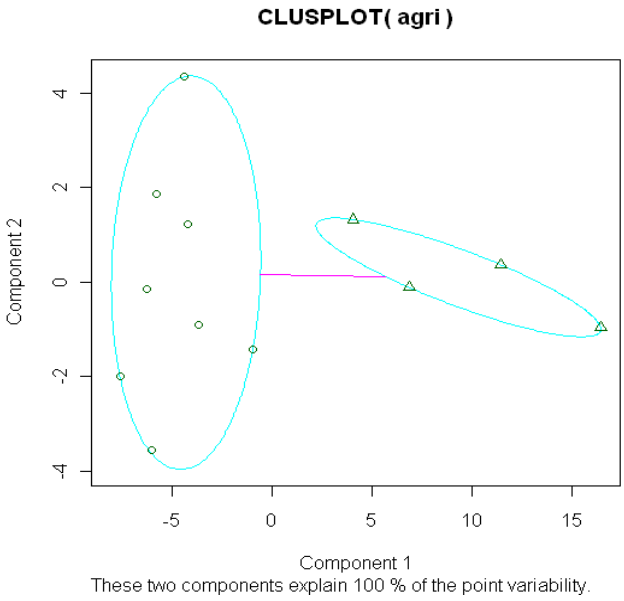
Exemple : Le tableau décrit pour chaque pays de la CEE la production agricole par unité d'exploitation (x) et le pourcentage de la population active employée dans l'agriculuture (y).

pays	x	y
B	16.8	2.7
DK	21.3	5.7
D	18.7	3.5
GR	5.9	22.2
E	11.4	10.9
F	17.8	6.0
IRL	10.9	14.0
I	16.6	8.5
L	21.0	3.5
NL	16.4	4.3
P	7.8	17.4
UK	14.0	2.3




```
> kmeans(agri,2)
K-means clustering with 2 clusters of sizes 4, 8
Cluster means:       x       y
                1  9.000 16.1250
                2 17.825  4.5625
Clustering vector:        B  DK   D  GR   E   F IRL   I   L  NL   P  UK
                          2   2   2   1   1   2   1   2   2   2   1   2
Within cluster sum of squares by cluster: [1] 90.76750 71.91375

s.class(agri,as.factor(kmeans(agri,2)$cluster))
clusplot(agri,as.factor(kmeans(agri,2)$cluster))
```





B – Partitionnement autour de médoïdes : PAM CLARA

1. Méthode PAM (partition around medoids)

Principe général : On définit k objets représentatifs des classes, appelés médoïdes, situés au centre des classes. Le médoïde est l'objet pour lequel la dissimilarité moyenne par rapport aux autres objets de la classe est la plus faible.

algorithme
Etape 1 :

- Le premier médoïde est l'objet minimisant la fonction objective

$$C = \sum_{i=1}^{n} d\left(i, m_1\right)$$

- Le second médoïde est l'objet, autre que m_1, minimisant la même fonction objective
- On détermine ainsi les k premiers médoïds.
- Chaque objet est ensuite affecté au groupe dont le médoïde est le plus proche.

Etape 2

- Pour tout couple d'objet (i,j) tel que i soit un des k médoïdes et j un objet autre qu'un médoïdes. On permute i et j si la fonction objective baisse.
- On répéte l'étape précédente jusqu'à stabilisation de la partition.

Silhouette d'une classe :
Pour chaque objet, on calcule le coefficient $s_i = \left(b_i - a_i\right)/max\left(a_i, b_i\right)$ avec a_i la dissimilarité moyenne de i au sein de son groupe et b_i la dissimilarité moyenne de i par rapport au groupe le plus voisin de i (dissimilarité moyenne la plus faible entre i et les objets de l'autre groupe).
On représente alors dans chaque classe la silhouette des objets et on visualise la bonne affectation des objets (s_i>0,5 par exemple).

Avec R, on utilise la fonction suivante :

```
pam(x, k, diss = F, metric= »euclidean », stand=F)
```

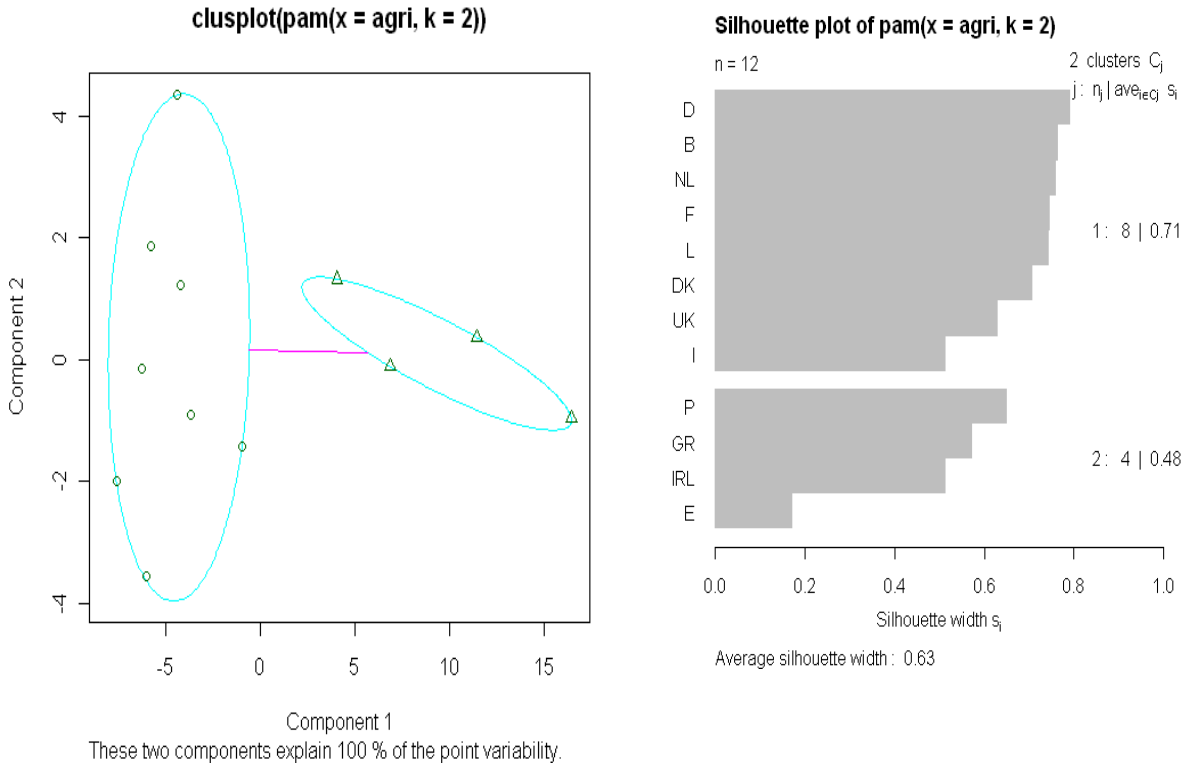
- x est un tableau numérique (diss=F ou un tableau de dissimilarité (diss=T).

Exemple : On reprend l'exemple précédent.

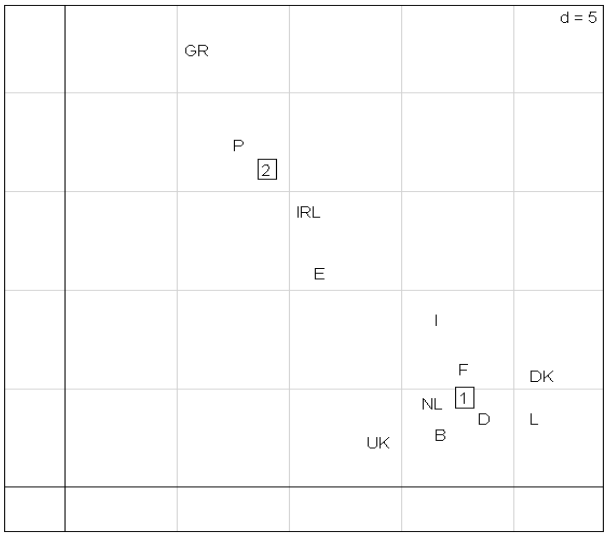
```
> pam(agri,2)
Medoids:          ID    x    y
                  D  3 18.7  3.5
                  P 11  7.8 17.4
Clustering vector:        B  DK   D  GR   E   F IRL   I   L  NL   P  UK
                          1   1   1   2   2   1   2   1   1   1   2   1
Objective function:        build     swap
                         3.429317 3.360610
```

```
Si x est un tableau numérique, la fonction pam transforme ce tableau en un
tableau de dissimilarité. Cela revient ici à faire pam(dist(agri),2).
```

```
plot(pam(agri,2))
```




```
s.label(agri,boxes=FALSE)
s.label(kmeans(agri,2)$centers,add.plot=TRUE)
```



Dans la première figure les médoïdes sont P et D et les centres obtenus avec kmeans sont notés 1 et 2.

2. méthode CLARA (Clustering LARge Application)

La fonction PAM nécessite un grand nombre de calcul. En présence d'un grand nombre d'objets, on utilise la méthode CLARA.

Etape 1 :

- Un échantillon est prélevé aléatoirement de l'ensemble.
- On utilise l'algorithme PAM pour déterminer les k médoïdes.
- On affecte ensuite les autres objets aux médoïdes les plus proches.
- On caractérise la partition par la valeur de la fonction objective correspondante.

Etape 2 :

- On répète l'étape 1 plusieurs fois et on retient les médoïdes et la partition correspondante qui minimise la fonction objective.

Avec R, on utilise la fonction :

```
clara(x, k, metric = "euclidean", stand = FALSE, samples = 5,
      sampsize = min(n, 40 + 2 * k))
```

C – Partitionnement autour d'un noyau : Méthode des nuées dynamiques

La distance entre deux classes est définies par :

$$D\left(I_j, I_{j'}\right) = \sum_{i \in I_j} \sum_{i' \in I_{j'}} d\left(i, i'\right)$$

Algorithme nuées dynamiques

1. On sélection k sous ensembles N_j^0 de q objets parmi les n objets de E de tel sorte que $N_j^0 \cap N_{j'}^0 = \emptyset$ pour j différent de j'. Les N_j^0 sont appelés noyaux.
2. On affecte alors les objets i de E aux classes j dont la distance $D\left(i, N_j^0\right)$ est minimale. On obtient alors une partition de E, $P^0 = \left(P_1^0, \ldots, P_k^0\right)$
3. On détermine dans chaque classe P_j^0 les q objets, notés N_j^1 qui minimise $D(N_j^1, P_j^0)$.
4. On reconduit les étapes 2 et 3 jusqu'à obtenir une partition stable.

D Classification floue : FANNY

Principe :

Les méthodes précédentes affecté à chaque objet une classe unique. La classification floue définit pour chaque objet un coefficient d'appartenance (memberships) à chaque classe qui indique la proximité de cet objet au groupe correspondant.

Soit u_{ij} le coefficient d'appartenance de l'objet i à la classe j. u_{ij} vérifie :

- $u_{ij} \geqslant 0$ pour tout couple ij
- $\sum_{j=1}^{k} u_{ij} = 1$

La fonction objective est : $C = \sum_{j=1}^{k} \sum_{i,i'=1}^{n} u_{(ij)}^2 u_{(i',j)}^2 d(i,i')/2 \sum_{i'=1}^{n} u_{(i',j)}^2$

Les valeurs u_{ij} sont calculés à l'aide d'un algorithme afin de minimiser cette fonction.

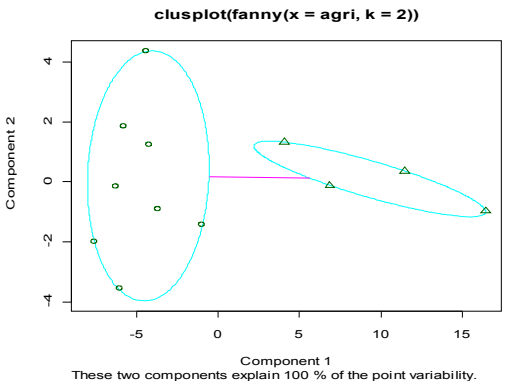
Coefficient de Dunn : ce coefficient égal $\sum_{i=1}^{n} \sum_{j=1}^{k} \frac{u_{(ij)}^2}{n}$ indique si la classification obtenue est proche d'une classification « dure » si il est proche de 1 (appartenance à une seule classe) ou « floue » si il est proche de $\frac{1}{k}$.

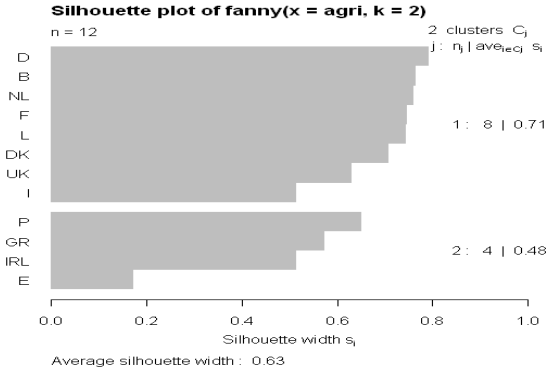
Avec R, on utilise la fonction :

```
fanny(x, k, diss = inherits(x, "dist"), memb.exp = 2,
      metric = c("euclidean", "manhattan", "SqEuclidean"))
```

```
> fanny(agri,2)
m.ship.expon.        2
objective     20.89447
tolerance        1e-15
iterations          22
converged            1
maxit              500
n                   12
Membership coefficients (in %, rounded):
    [,1] [,2]
B     90   10
DK    85   15
D     93    7
GR    18   82
E     31   69
F     90   10
IRL   15   85
I     73   27
L     87   13
NL    91    9
P     10   90
UK    80   20
Fuzzyness coefficients:
dunn_coeff normalized
 0.7486474  0.4972948
Closest hard clustering:
  B  DK   D  GR   E   F IRL   I   L  NL   P  UK
  1   1   1   2   2   1   2   1   1   1   2   1
```

plot(fanny(agri,2))





II CLASSIFICATION HIERARCHIQUE (AGNES DIANA MONA)

Ces méthodes peuvent s'appliquer à des tableaux de dissimilarités ou des tableaux numériques. Les algorithmes construisent des partitions emboitées (hierarchies) avec un nombre k de partitions variant de n à 1 pour une classification hierarchique ascendante (agglomerative) ou de 1 à n pour une classification hierarchique descendante (divisive).

A Classification ascendante hierarchique : AGNES (agglomrative nesting)

Algorithme :

- Les deux objets ou classes les plus proches sont regroupés en une nouvelle classe.
- La distance des autres objets ou classes par rapport à cette nouvelle classe sont calculés à l'aide d'un critère d'agrégation :
  - Méthode de la moyenne non pondérée UGPMA

    $$d(j, j') = \frac{1}{n_j n_{j'}} \sum_{i \in I(j)} \sum_{i' \in I(j')} d_{ii'}$$

    avec j, j' représentent les classes et i, i' les objets
    $I(j)$ représente les indices des objets appartenant à j.

  - Méthode du saut minimal

    $d(j, j') = min(\ d_{ii'}, i \in I(j),\ i' \in I(j'))$

  - Méthode du saut maximal

    $d(j, j') = min(\ d_{ii'}, i \in I(j),\ i' \in I(j'))$

  - Méthode de Ward

    Pour chaque regroupement possible, on calcule la variation d'inertie intraclasse en résultant.

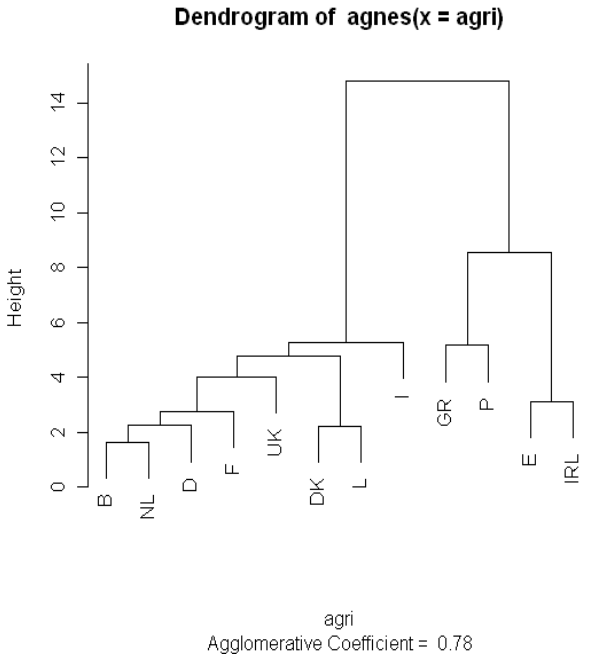
Avec R, les fonctions sont :

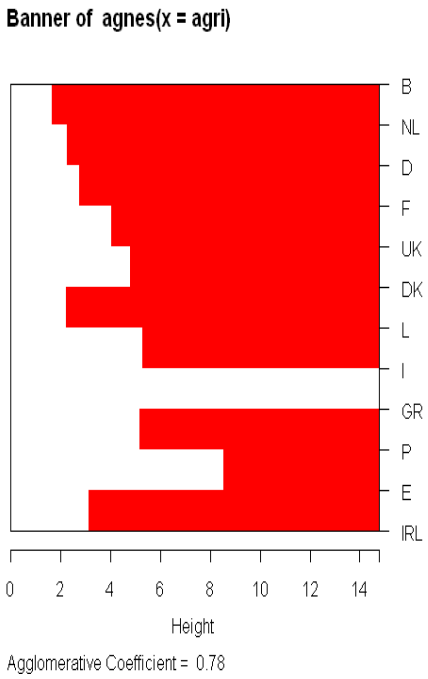
```
hclust(d, method = "complete", members=NULL)
ou mieux

agnes(x, diss = inherits(x, "dist"), metric = "euclidean",
      stand = FALSE, method = "average")
```

Exemple : toujours agri

```
> agnes(agri)
Agglomerative coefficient:  0.7818932
Order of objects:
 [1] B   NL  D   F   UK  DK  L   I   GR  P   E   IRL
Height (summary):
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.649   2.509   4.027   4.966   5.228  14.780
```

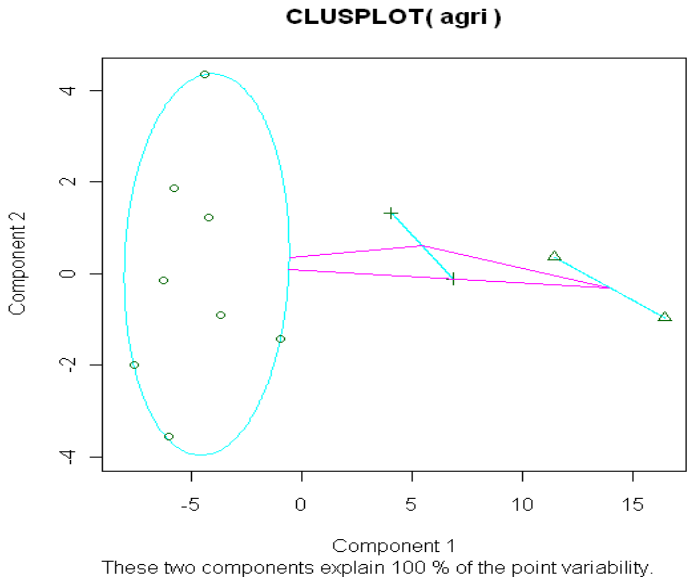
plot(agnes(agri))





Le graphique banner indique la dissimilarité du regroupement de deux classes. Par exemple B et NL ont une dissimilarité faible (lire de gauche à droite) alors qu'entre le premier groupe (B à I) et le second (GR à IRL) elle est très forte. Le coefficient d'agglomération (AC) est le pourcentage de la figure colorée (pour faire simple). C'est un indicateur de la structure des données. Plus il est élevé, plus les dissimilarités entre objets sont grandes.

```
clusplot(agri,cutree(agnes(agri),k=3))
```



B Classification descendante hierarchique : DIANA (DIvisive ANAlysing)

Algorithme :

- Les objets sont tous regroupés au sein d'une même classe.
- Dans la classe C présentant la plus grande dissimilarité entre deux objets, on sépare ces objets en deux classes A et B.
- Les objets de la classe C scindée en deux sont affectés à l'un ou l'autre des deux classes A ou B créées suivant l'alogorithme suivant :

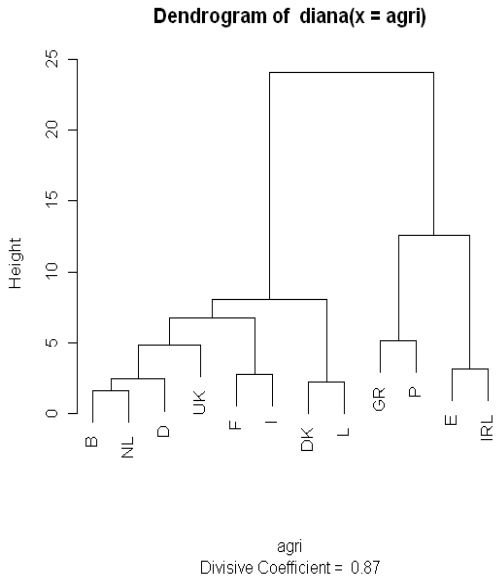
- A est au départ constitué de tous les objets de C, B est vide.
- Pour chaque objet i de A, on calcule la dissimilarité moyenne aux autres objets de A. On affecte l'objet m ayant la plus forte dissimilarité moyenne dans le groupe B. On a alors A= A \{m} et B={m}
- our chaque objet de A, on calcule la dissimilarité moyenne à A et à B. L'objet ayant la plus forte différence d(i,A)-d(i,B) est affecté au groupe B si la différence est positive sinon l'algorithme s'arrête.

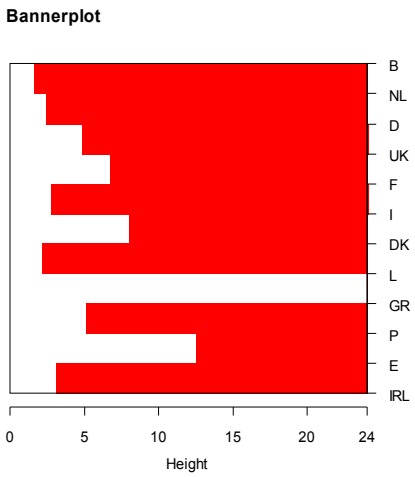
Avec R, la fonction est :

```
diana(x, diss = inherits(x, "dist"), metric = "euclidean",
      stand = FALSE, method = "average")
```

Exemple : toujours agri

```
> diana(agri)
Merge:
      [,1] [,2]
 [1,]   -1  -10
 [2,]   -2   -9
 [3,]    1   -3
 [4,]   -6   -8
 [5,]   -5   -7
 [6,]    3  -12
 [7,]   -4  -11
 [8,]    6    4
 [9,]    8    2
[10,]    7    5
[11,]    9   10
Order of objects:
 [1] B   NL  D   UK  F   I   DK  L   GR  P   E   IRL
Height:
 [1]  1.649242  2.435159  4.850773  6.723095  2.773085  8.052950  2.220360
 [8] 24.035391  5.162364 12.567418  3.140064
Divisive coefficient:
[1] 0.8711062

plot(diana(agri))
```





C Classification monothétique : MONA (MONothetic Analysing)

Cette méthode est conçue pour les données binaires. Il repose sur le principe de la classification hierarchique par divisibilité mais ne recquiert pas le calcul d'une matrice d dissimilarité.

Algorithme :

- A chaque étape l'algorithme utilise une des variables pour séparer les objets suivant la valeur 0 ou 1. On choisit la variable la plus fortement associée aux autres variables en calculant le coefficiant d'association $A_{jj'}$:

  $A_{jj'} = |a_{ij}\ d_{ij} - b_{ij}\ d_{ij}|$

  avec

  - a_{ij} le nombres d'objets vérifiant $x_{ij} = x_{ij'} = 0$
  - $d_{jj'}$ le nombres d'objets vérifiant $x_{ij} = x_{ij'} = 1$
  - b_{ij} le nombres d'objets vérifiant $x_{ij} = 0$ et $x_{ij'} = 1$
  - $c_{jj'}$ le nombres d'objets vérifiant $x_{ij} = 1$ et $x_{ij'} = 0$

  On en déduit la mesure totale d'association par $A_j = \sum_{j'} A_{jj'}$

- On poursuit ainsi l'algorithme en changeant de variables jusqu'à ne plus disposer de nouvelles variables ou à n'obtenir que des singletons.

Avec R, la fonction est :

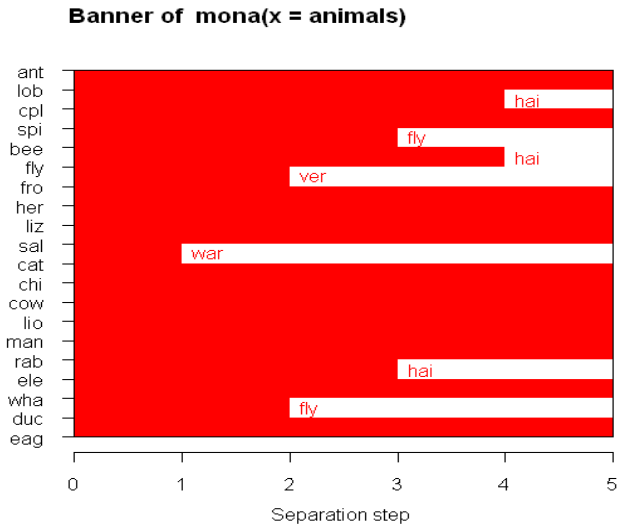
```
diana(x, diss = inherits(x, "dist"), metric = "euclidean",
      stand = FALSE, method = "average")
```

Exemple : Tableau animals décrivant la présence ou l'absence de 6 caractères.

```
A data frame with 20 observations on 6 variables:
```

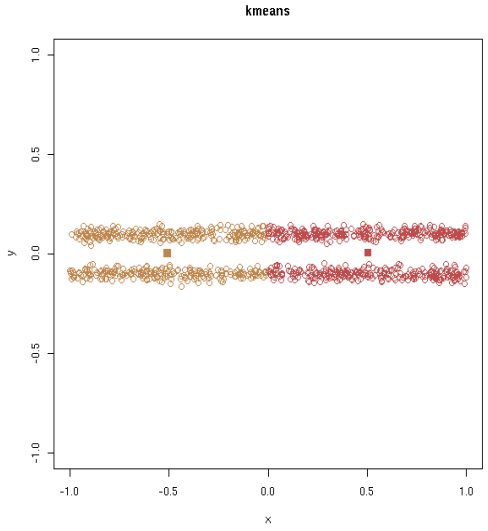
v1 war warm-blooded
v2 fly can fly
v3 ver vertebrate
v4 end endangered na
v5 gro live in group na
v6 hai have hair

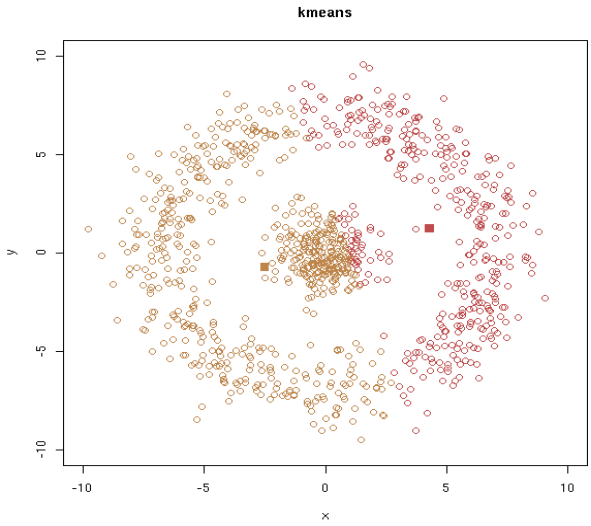
```
Data(animals)
mona(animals)
> mona(animals)
Revised data:
    war fly ver hai
ant   0   0   0   0
bee   0   1   0   1
cat   1   0   1   1
   .....
Order of objects:
 [1] ant lob cpl spi bee fly fro her liz sal cat chi cow lio man rab ele wha duc eag
Variable used:
 [1] NULL hai NULL fly hai ver NULL NULL NULL war NULL NUL NUL NUL NUL hai NULL fly  NULL
Separation step:
 [1] 0 4 0 3 4 2 0 0 0 1 0 0 0 0 0 3 0 2 0
```



III CLASSIFICATION PAR VOISINAGE DENSE (kernel)

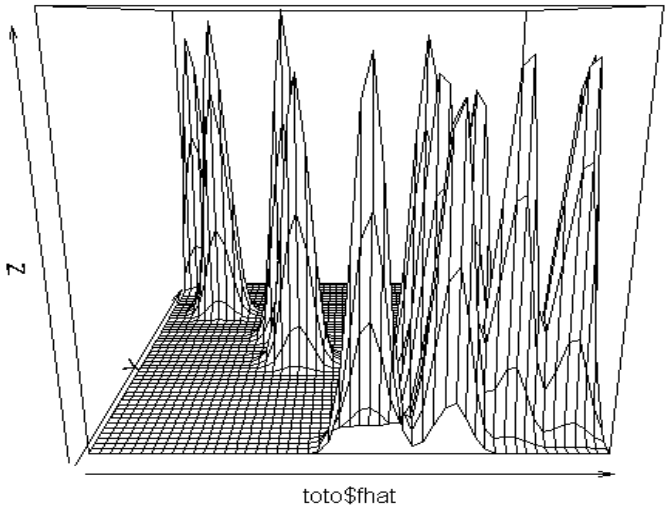
Dans les méthodes précédentes, les algorithme considères les ams comme sphérique pour la métrique utilisée et convexe. Elles ne sont plus adaptées pour des classes plus complexes.



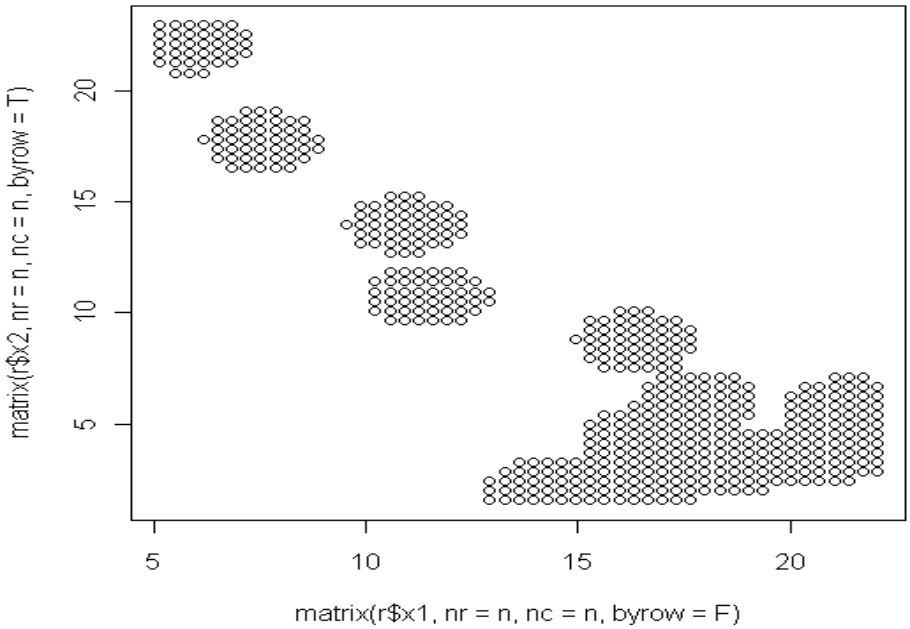


Dans R

```
help.search("density")
# Suggests:
#   KernSur(GenKern)          Bivariate kernel density estimation
#   bkde2D(KernSmooth)        Compute a 2D Binned Kernel Density Estimate
#   kde2d(MASS)               Two-Dimensional Kernel Density Estimation
#   density(mclust)           Kernel Density Estimation
#   sm.density(sm)            Nonparametric density estimation in 1, 2 or 3 dim

library(KernSmooth)
r <- bkde2D(agri, bandwidth=c(.5,.5))
persp(r$fhat)
```




```
n <- length(r$x1)
plot(matrix(r$x1,nr=n,nc=n,byrow=F),matrix(r$x2,nr=n,nc=n,byrow=T),col=r$fhat>.0
01)
```



On recherche alors les ensembles connexes.

IV Méthodes mixtes de classification

La méthode mixte de classification, décrite par Lebart (1984) permet de classer un grand nombre

d'individus. Elle se déroule en trois temps :

1. **k-means** : On effectue pour commencer une réallocation itérative des individus avec choix aléatoire des pôles d'attraction initiaux, en un grand nombre de classes. En principe, on choisit comme nombre de classes le dixième de l'effectif de départ. Ce premier classement permet de réaliser par la suite une classification ascendante hiérarchique sur un nombre plus restreint d'individus.

2. **Classification Ascendante Hiérarchique** : On utilise les barycentres des classes obtenues par réallocation itérative en leur donnant un poids égal à la somme des poids des individus de la classe. A partir de ces barycentres et de leur poids, on réalise une classification hiérarchique selon le critère du saut de Ward. En fonction de l'arbre de classification obtenu, on détermine en combien de classes il convient de regrouper l'ensemble des individus.

3. **k-means** : Pour améliorer la classification obtenue avec `cahd`, on effectue une deuxième réallocation itérative sur l'ensemble, éventuellement pondéré, des individus avec comme noyaux de départ les barycentres des classes déterminées par la classification hiérarchique précédente. Cette méthode vise à augmenter l'inertie inter-classe. De ce point de vue, la classification ne peut donc que s'améliorer.