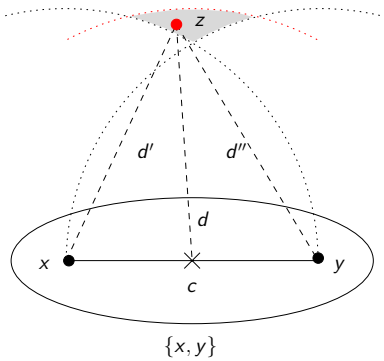
# Mathematical Models and Applications

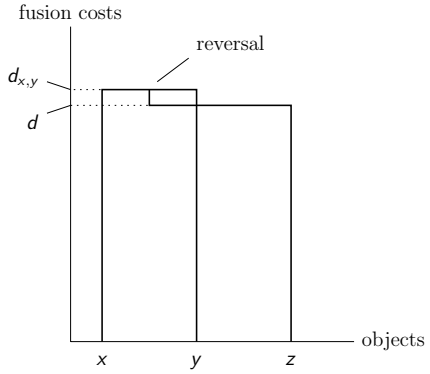
## An introduction to Clustering Analysis

Pedro Cristiano Silva

Instituto Superior de Agronomia

2019-20



# Outline

- Introduction
- Dissimilarity measures
- Clustering methods
- Validation

# Main bibliography

**Main reference** An important part of this course material is based on the textbook


L P. Legendre, L. Legendre, *Numerical Ecology* (2003)


**Other important references**


C J. Cadima, *Multivariate Statistics*, course notes (2009/10)

E M. Ester, H.P. Kriegel, J. Sander, X. Xu, *Density Based algorithm for discovering clusters in large spatial databases with noise* (1996)

Ev B. S. Everitt, *Cluster Analysis* (1993)

M C. Manning, P. Raghavan, H. Shutze, *An Introduction to Information Retrieval* (2009)

R A.C Rencher, *Methods of Multivariate Analysis* (2002)

T Theodoridis, S. and Koutroumbas, K., Pattern Recognition (2009)

# 1. INTRODUCTION

# What is clustering?

- *Oirginally developed by the biologists Robert Sokal and Peter Sneath in their seminal paper 'Principles of Numerical Taxonomy' in 1963, as method to classify organisms into species, given a set of proeminent features*
- *A quest for discontinuities in data*
- *It is a method of unsupervised classification*
- *No priori information on the groups is assumed*
- *Does not involve predicting*

# Clustering

**Clustering** consists of partitioning a collection of objects/variables/descriptors/etc, such that:

- Each object belongs to one and only one subset (cluster) of the partition
- Objects belonging to the same cluster are more similar (**internal cohesion**) than objects belonging to distinct clusters (**external separation**), given the variables considered

The definition above is called **hard** or **crisp** clustering.
There is also a notion of **fuzzy** clustering where the objects belong to a given set with a certain degree of membership

*Clustering always* **imposes some kind structure on the data**, *even when no special structure or discontinuities are present! For instance, many clustering techniques tend to form globular clusters, e.g., with elliptical or spherical shapes*

# A more formal definition of clustering

Given a collection of $N$ objects, $X = \{x_1, \ldots, x_N\}$, one seeks a partition of X into $K$ non empty disjoint sets (the *clusters*),

$$X = \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_K$$

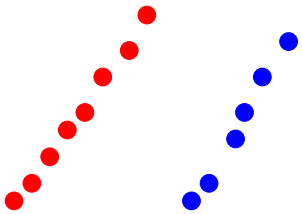
such that, given the resemblance notion considered, it

- maximizes the **internal homogeneity or cluster cohesion**, or equivalently, it minimizes the *intra-cluster variability* - objects belonging to the same cluster should share the same features
- it maximizes the **external heterogeneity or cluster separation**, i.e., it maximizes the *inter-cluster separability* - objects belonging to distinct clusters should be very dissimilar and have clear distinguished features

# Examples

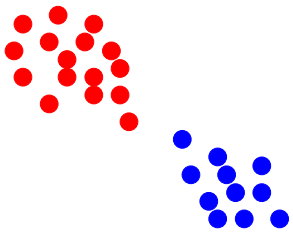
**strong internal cohesion**

**strong separation**



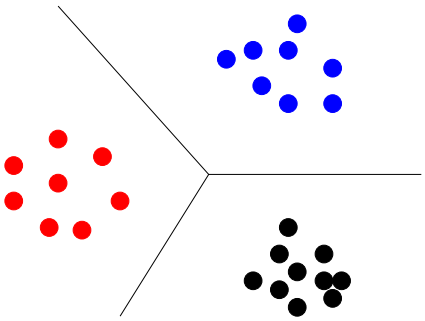
**weak internal cohesion**

**strong separation**

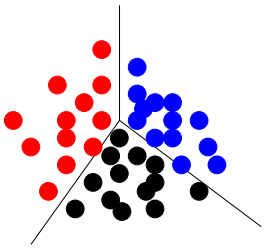


**strong internal cohesion**

**weak separation**



**clear clustering structure**



**artificial clustering structure**

## Huge solution space...

The possible number of partitions of $N$ elements into $K$ clusters $(1 \leq K \leq N)$ equals

$$\xi(N, K) = \frac{1}{K!} \sum_{j=1}^{K} \binom{K}{j} (-1)^{K-j} j^N,$$

which is a huge number, known as Stirling of second kind, even for relatively small values of $N$ and $K$, making impossible to to find the best partition by exhaustion For instance, for the total number partitions of a set with 25 elements into 8 clusters gives

$$\xi(25, 8) = 69022372111836858$$

# Steps in the clustering analysis process

- **Variables/features selection**
  To choose the best variables to encode as much as possible the information concerning the task, avoiding redundancy (highly correlated variables), but at the same time being parsimony
  Some questions arise:
  - which types of variables is more appropriated: continuous, categorical, ordinal, binary, . . . ?
  - standartize/normalize the variables ?
- **Clustering model**:
  Which combination of clustering method and distance/dissimilarity ?
- **Cluster validation**
  **internal**: how many groups ? how to assess the quality/stability of the clusters ?
  **external**: how the clustering results compare with the outcomes obtained using different clustering models or it compares with known information
- **Interpretation of the results**:
  Are the outcomes interpretable in the context of the problem ?
  How to associate the most important variables/features to the groups (for instance, obtained performing a clustering analysis on the individuals projected on the two first axes of a PCA) ?

# Cluster model

A cluster model result depends on

- **the notion of distance/dissimilarity** between individuals and clusters: should be adequate to the type of variables involved and to the type of results sought
- **the clustering method**: should take into account the type of structure/shape of the clusters sought (rounded shape/arbitrary shape/...) and characteristics of the method itself (sensitivity to outliers/noise/ldots), computational issues (is scalable for large datasets), etc...

When two or more clustering models may be appropriate one should compare the outputs of such models to seek for common patterns that emerge from several clustering models - **robust solutions**

# Example - iris flower dataset

The well konwn iris flower dataset contains the sepal and petal lengths and widths (in cm) of 150 iris flowers

| | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width |
|---|---|---|---|---|
| 1 | 5.1 | 3.5 | 1.4 | 0.2 |
| 2 | 4.9 | 3.0 | 1.4 | 0.2 |
| 3 | 4.7 | 3.2 | 1.3 | 0.2 |
| 4 | 4.6 | 3.1 | 1.5 | 0.2 |
| 5 | 5.0 | 3.6 | 1.4 | 0.2 |
| 6 | 5.4 | 3.9 | 1.7 | 0.4 |
| 7 | 4.6 | 3.4 | 1.4 | 0.3 |
| 8 | 5.0 | 3.4 | 1.5 | 0.2 |
| 9 | 4.4 | 2.9 | 1.4 | 0.2 |
| 10 | 4.9 | 3.1 | 1.5 | 0.1 |

- How to measure the distance between two iris flowers ?
- Standardize (z-score normalization) or normalize (min-max scalling) the variables in order that the differences between all variables contribute equally ?
- How to measure the distance between the variables ?

# Example - a freshwater fish dataset in West Africa

In the biogeography it is common to use biological markers (the species) to distinguish between sites (the river basins)

| | annectens | ansorgi | bichir | endlicheri |
|---|---|---|---|---|
| GAMBIE | 1 | 0 | 1 | 0 |
| GEBA | 0 | 1 | 1 | 1 |
| CRUBAL | 0 | 1 | 0 | 0 |
| KONKOURE | 0 | 0 | 0 | 0 |
| KOLENTE | 0 | 0 | 0 | 0 |
| LSCARC | 0 | 0 | 0 | 0 |
| ROKEL | 0 | 0 | 0 | 0 |

- Which type of variable/feature is the most appropriate to encode this type data ?
- Given the type of variable chosen how to measure the resamblance between river basins ?
- How similar are the fish species with respect to their distribution in the sites ?

# Example - a two way contigency table

Contingency table of the country of residence and primary language spoken by 1000 inhabitants. It corresponds to a dataset of size 1000 described by two qualitative variables with 5 modalities (categories) each

| | English | French | Spanish | German | Italian | Total |
|---|---|---|---|---|---|---|
| Canada | 688 | 280 | 10 | 11 | 11 | 1000 |
| USA | 730 | 31 | 190 | 8 | 41 | 1000 |
| England | 798 | 74 | 38 | 31 | 59 | 1000 |
| Italy | 17 | 13 | 11 | 15 | 944 | 1000 |
| Switz. | 15 | 222 | 20 | 648 | 95 | 1000 |
| Total | 2248 | 620 | 269 | 713 | 1150 | 5000 |

(source: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3718710/)

- How similar are the countries given the spoken languages ?
- How similar are the languages given their distribution by the countries ?

# 2. (DIS)SIMILARITY MEASURES

# Dissimilarity measures between individuals

A **dissimilarity measure** on a set $X$ is a real function

$$d : X \times X \to \mathbb{R},$$

such that, for all $x, y, \in X$, we have

- $d(x, y) \geq 0$
- $d(x, y) = 0$
- $d(x, y) = d(y, x)$ (*symmetric*)

If additionally, $d$ verifies the *triangle inequality*

- $d(x, z) \leq d(x, y) + d(y, z)$ for all $x, y, z \in \mathbb{R}$,

$d$ is called a **distance** or **metric**.

# Dissimilarity measures for quantitative data

Given $p \in ]0, +\infty]$, the **Minkowski** or $L_p$**-norm** of an element $x = (x_1, \ldots, x_d) \in \mathbb{R}^d$ is defined as

$$\|x\|_p = \left( \sum_{i=1}^{d} |x_i|^p \right)^{\frac{1}{p}},$$

The **Minkowski dissimilarity** between $x, y \in \mathbb{R}^d$ is defined as

$$D_p(x, y) = \|x - y\|_p$$

For $p \geq 1$ this dissimilarity is actually a distance

## Important cases of Minkowski distance: $p = 1, 2, \infty$

- If $p = 2$ we get the usual **Euclidean metric** or $\ell_2$-**distance**:

$$\begin{aligned} D_2(x, y) &= \sqrt{(x - y)^T (x - y)} \\ &= \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_d - y_d)^2} \end{aligned}$$

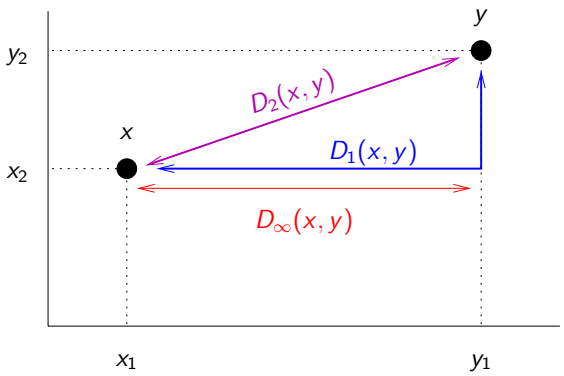
- If $p = 1$ we get the **Manhattan** city block or $\ell_1$-**distance**:

$$D_1(x, y) = \sum_i |x_i - y_i|.$$

- If $p = \infty$ we get the **maximum metric** or Chebyshev or $\ell_\infty$-**distance**:

$$\begin{aligned} D_\infty(x, y) &= \lim_{p \to \infty} D_p(x, y) \\ &= \max_i \{|x_i - y_i|\}. \end{aligned}$$

# Relation among the Minkowski distances




For all $x, y \in \mathbb{R}^2$ we have

$$D_1(x,y) \geq D_2(x,y) \geq D_\infty(x,y)$$

In general, if $1 \leq p < q \leq \infty$,

$$D_p(x,y) \geq D_q(x,y)$$

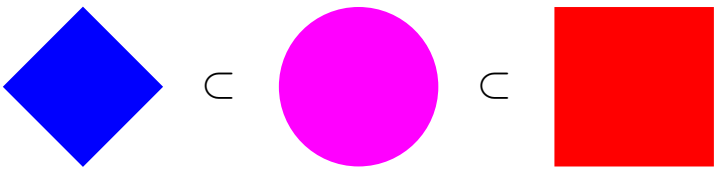
for all $x, y \in \mathbb{R}^N$

# Relation among the Minkowski balls

For the 2-dimensional ball with norm 1, i,e, for the set of points lying at a distance inferior or equal than one from the origin,

$$B(0) = \{x \in \mathbb{R}^2 : \|x\| \leq 1\},$$

w.r.t. the Minkowski norms $D_1$, $D_2$ and $D_\infty$, the relations among the metrics of the previous slide yield the following inclusions among the 1-balls for these norms,



As before, the property extends to the $n$-dimensional case

## Important cases of Minkowski distance: $p = 1, 2, \infty$

- If $p = 2$ we get the usual **Euclidean metric** or $\ell_2$-**distance**:

$$\begin{aligned} D_2(x, y) &= \sqrt{(x - y)^T (x - y)} \\ &= \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_d - y_d)^2} \end{aligned}$$

- If $p = 1$ we get the **Manhattan** city block or $\ell_1$-**distance**:

$$D_1(x, y) = \sum_i |x_i - y_i|.$$

- If $p = \infty$ we get the **maximum metric** or Chebyshev or $\ell_\infty$-**distance**:

$$\begin{aligned} D_\infty(x, y) &= \lim_{p \to \infty} D_p(x, y) \\ &= \max_i \{|x_i - y_i|\}. \end{aligned}$$

For small values of $p$ the relative weights of the variable differences are approximately equal while for greater values of $p$, the larger the difference the more the variable is important

# The canberra distance

If $x, y$ are $N$-dimensional vectors one can define the so-called **canberra** distance

$$d(x, y) = \sum_{i=1}^{N} \frac{|x_i - y_i|}{|x_i| + |y_i|}$$

- This distance is a weighted version of the Manhattan distance that is more sensitive to small values
- This can be usefull when data contains large and small values and the differences between the small values should also be taking into account

# Generalized euclidean distances

Let $\Sigma$ be a *symmetric positive definite matrix* (which is necessarily invertible) of order $n$. Given $x, y \in \mathbb{R}^n$, we define the distance

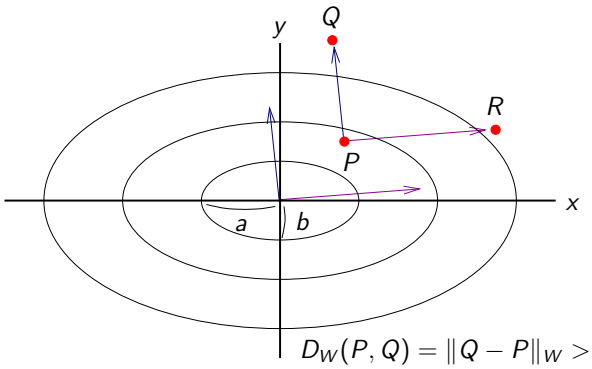
$$d_\Sigma(x, y) = \sqrt{(x-y)^T \Sigma^{-1} (x-y)}$$

- If $\Sigma = I_n$ (identity matrix), $d_\Sigma(x, y)$ is the usual Euclidean distance between $x$ and $y$
- If $\Sigma$ is the diagonal matrix $\begin{bmatrix} a^2 & 0 \\ 0 & b^2 \end{bmatrix}$, we get the weighted euclidean distance,

$$d_\Sigma(x, y) = \sqrt{\frac{(x_1 - y_1)^2}{a^2} + \frac{(x_2 - y_2)^2}{b^2}}$$

The points at a distance one from the origin lie in an ellipse of semi-axes $a$ and $b$

# Generalized euclidean distance




$$D_W(P,Q) = \|Q-P\|_W > 2 > D_W(P,R) = \|R-P\|_W$$

# Mahalanobis distance

Let $X$ be a set of observations in $\mathbb{R}^n$ with mean $\mu$ and **variance-covariance** matrix $\Sigma$. The **Mahalanobis distance** between two observations $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ of $X$ is the generalized euclidean distance

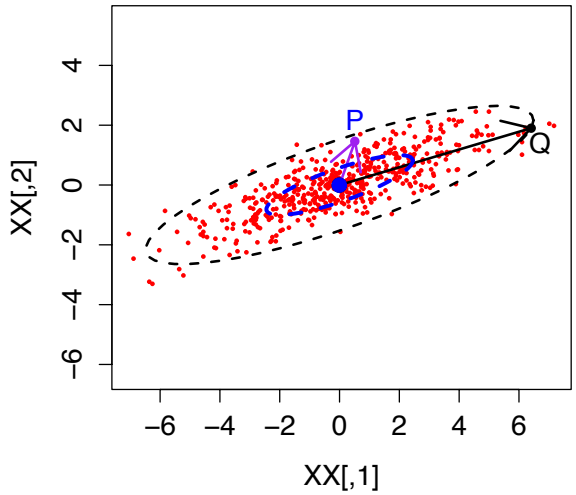
$$d_\Sigma(x, y) = \sqrt{(x - y)^T \Sigma^{-1} (x - y)}$$

- If the variables are uncorrelated, i.e., $\Sigma$ is a diagonal matrix containing only the variances, $d_\Sigma(x, y)$ equals the euclidean distance between the standardized variables, i.e., the standardized euclidean distance

$$d_\Sigma(x, y) = \sqrt{\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{\sigma_i^2}}$$

- The Mahalanobis distance is invariant under scale transformations on the variables
- The observations at distance one from the mean lie in an ellipsoid centred at the mean with semi-axes equal to the standard deviations of the variables
- The Mahalanobis distances are 'smaller' along the directions of greater variability

# Mahalanobis distance for correlated variables




Euclidean distance between $P$ and $Q$: $d(P, Q) = \sqrt{(Q - P)^T (Q - P)} = 5.907128$

Mahalanobis distance between $P$ and $Q$:

$$d_\Sigma(P, Q) = \sqrt{(Q - P)^T \Sigma^{-1} (Q - P)} = 3.570066$$

# Dissimilarity measures for binary data

$x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ binary vectors

$a$: nr components where both variables take value 1 (positive agreement)

$b$: nr of components where $x$ take value 1 and $y$ value 0 (disagreement)

$c$: nr of components where $x$ take value 0 and $y$ value 1 (disagreement)

$d$: nr of components where both variables take value 0 (negative agreement)

- **Simple matching** (counts double-zeroes, is suitable if 0-1 represent equally valued attributes like male-female):

$$S(x,y) = \frac{a+d}{a+b+c+d} \qquad D(x,y) = 1 - S(x,y) = \frac{b+c}{a+b+c+d}$$

- **Jaccard coefficient** (does not count double zeroes, is suitable if 0-1 represent unequal valued attributes, like species presences-absences):

$$J(x,y) = \frac{a}{a+b+c} \qquad D(x,y) = 1 - J(x,y) = \frac{b+c}{a+b+c}$$

- **Gower and Legendre coefficient** (take values in [-1,1]):

$$S(x,y) = \frac{a+d-(b+c)}{a+b+c+d} \qquad D(x,y) = \sqrt{1 - S^2(x,y)}$$

# Example

Assume that we have two binary variables representing the presence (1) and absence (0) of two species at 16 spots:

Sp1=c(0,1,1,1,0,0,0,0,0,0,0,0,0,1,0,0)
Sp2=c(0,1,0,0,1,0,0,0,0,1,0,0,0,0,0,1)

How similar are the two species with regard to their distribution in the 16 spots ?
Note that $a = 1$, $b = 3$, $c = 3$ and $d = 9$

- Simple matching: $\frac{a+d}{a+b+c+d} = 10/16$
- Jaccard coefficient: $\frac{a}{a+b+c} = 1/7$
- Gower and Legendre coefficient: $\frac{a+d-(b+c)}{a+b+c+d} = 4/16$

The (asymmetrical) Jaccard coefficient seems to be more suitable to determine homogeneous groups of species with respect to their distribution at a given collection of sites

R

```
# The R function dist with the method ‘‘binary’’ computes the
dissimilarity as d(x,y) = 1 - S(x,y), where S is the Jaccard coefficient
d = dist(cbind(x,y),method=‘‘binary’’,diag=FALSE,upper=FALSE,p=2)
# 6/7
```

# Example: a case with nominal variables

Consider again the two-way contingency table of country of residence and primary language spoken by 1000 inhabitants

| | English | French | Spanish | German | Italian | Total |
|---|---|---|---|---|---|---|
| Canada | 688 | 280 | 10 | 11 | 11 | 1000 |
| USA | 730 | 31 | 190 | 8 | 41 | 1000 |
| England | 798 | 74 | 38 | 31 | 59 | 1000 |
| Italy | 17 | 13 | 11 | 15 | 944 | 1000 |
| Switz. | 15 | 222 | 20 | 648 | 95 | 1000 |
| Total | 2248 | 620 | 269 | 713 | 1150 | 5000 |

(source: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3718710/)

# Row profiles

From the previous contigency table one computes the row profiles by dividing each row by row total

| | English | French | Spanish | German | Italian | |
|---|---|---|---|---|---|---|
| Canada | 0.688 | 0.280 | 0.010 | 0.011 | 0.011 | 1.000 |
| USA | 0.730 | 0.031 | 0.190 | 0.008 | 0.041 | 1.000 |
| England | 0.798 | 0.074 | 0.038 | 0.031 | 0.059 | 1.000 |
| Italy | 0.017 | 0.013 | 0.011 | 0.015 | 0.944 | 1.000 |
| Switz. | 0.015 | 0.222 | 0.020 | 0.648 | 0.095 | 1.000 |
| Average row profile | 0.450 | 0.124 | 0.054 | 0.143 | 0.230 | 1.000 |

# $\chi^2$-metric

- The distance between the rows (countries) $r_k$ and $r_\ell$ can be defined using the $\chi^2$-metric,

$$d(r_k, r_\ell) = \sqrt{\sum_{i=1}^{N} \frac{p_{k,i}^2}{p_{\bullet,i}}}$$

  Here $N = 5$ is the number of columns, $p_{i,j}$ is the relative frequency of row $i$ in column $j$ and $p_{\bullet,j}$ is the average relative frequency for column $j$

- For instance, the squared distance $d^2(r_1, r_5)$ between *Canada* and *Switzerland* equals

$$\frac{(0.688-0.015)^2}{0.450} + \frac{(0.280-0.222)^2}{0.124} + \frac{(0.010-0.020)^2}{0.054} + \frac{(0.011-0.648)^2}{0.143} + \frac{(0.011-0.095)^2}{0.230} = 1.975782$$

- The division of the squared terms by the expected relative frequencies, weighting each column in the inverse proportion of its average frequency, allows that small differences between rare categories are not "smashed" by the larger differences between common categories

# Clustering variables

- An usual similarity notion between two variables $x$ and $y$ is Pearson's correlation

$$\rho = \frac{cov(x, y)}{\sigma_x \, \sigma_y}$$

  This similarity can be transformed into a dissimilarity using the transformation $d = \sqrt{1 - \rho^2}$, which take values in the interval $[0, 1]$
- Highly linearly correlated variables (positively or negatively) will have $d \approx 0$ while for uncorrelated variables $d \approx 1$
- Alternatively, we can define $d = (1 - \rho)/2$. In this case the strengh of the linear relationship and the direction are both accounted
- We can use the above dissimilarity measures to cluster variables. Each cluster will consist of a set of variables highly correlated. This can be useful to detect redundancies and can give an idea of the number of principal dimensions of data
- Actually, for each cluster we can define a synthetic variable called **latent variable**, which is the linear combination of the variables in the group, that minimizes the sum of the dissimilarities $(1 - \rho^2)$ with respect to all variables (a kind of centroid of the cluster of variables)

# 3. CLUSTERING METHODS

# Clustering methods

**Distance-based models** rely only on pairwise dissimilarities between individuals. Two major groups of methods can be found:

- **Hierarchical methods** - produce a *nested* structure of partitions. It does not require the number of clusters to be known *a priori*:
  - *Agglomerative clustering* (bottom-up strategy) - It starts from the partition consisting of one individual per cluster (singletons) until it aggregates all individual in the same cluster: single, complete, average, McQuitty, centroid, median, Ward, ...
  - *Divisive clustering* (top-down strategy) - it proceeds in the opposite way and it is usually more computacional demanding, thus being more rarely used
- **Partitional methods** - produce *flat* (non-nested) partitions. Usually tries to maximize some intra-cluster homogeneity / inter-cluster heterogeneity criterion. It requires the number of clusters to be known *a priori*: K-means, K-medoids ... Unlike the hierarchical methods two individuals that are aggregated together at a given step can be desaggregated at a posterior stage

# Clustering methods (cont.)

Other types of methods include:

- **Density-based clustering**: seek for high density regions of points (clusters) separated by low density of points (noise)
- **Model-based clustering** assumes that some model (or mixture of models) generates the data
- **Constrained-clustering**: accounts for other type of information, such as spatial relationships between individuals (for instance, contiguity relationships between cells in a map)
- **Fuzzy**: the same individual can belong to several classes with a certain degree of membership (probability)

# Hiearchical agglomerative clustering algorithm

## Algorithm

**Input**: *the proximity matrix containing the pairwise dissimilarities between $N$ individuals $x_1, \ldots, x_N$*

- *Starts with $N$ clusters containing a single object each (singletons);*
- *Merges the least dissimilar pair of clusters, i.e., the pair of clusters with smallest fusion cost into a new cluster and updates the proximity matrix (reducing its order by one);*
- *Repeats step 2 until only one cluster remains ($N - 1$ steps).*

**Output**: *a sequence of length $N - 1$ encoding the merged clusters and their fusion costs*

## Hierarchical agglomerative clustering methods

The computation of the dissimilarity between two clusters (i.e., fusion cost) during the clustering process depends on the aggregation method:

- Single-linkage or nearest-neighbor
- Complete-linkage or furthest-neighbor
- Average (UPGMA)
- Weighted average or McQuitty (WPGMA)
- Centroid (UPGMAC)
- Median (WPGMC)
- Ward or mininum-variance clustering

# Updating formula for HAC

- For all the aggregration methods above, the dissimilarity between the most recently merged clusters, say $\mathcal{C}' \cup \mathcal{C}''$, and each one of the remaining clusters $\mathcal{C}$, $D(\mathcal{C}' \cup \mathcal{C}'', \mathcal{C})$ can be determined in terms of the pairwise dissimilarities $D(\mathcal{C}', \mathcal{C})$ and $D(\mathcal{C}'', \mathcal{C})$, i.e., in terms of the data from the previous proximity matrix, and therefore can be done using a recursive algorithm with a convenient updating formula
- Unlike many other statistical methods the clustering analysis using a HAC does not require the knowledge of the original dataset. *Only the proximity matrix containing the initial pairwise distances is required!*

# Dendrogram

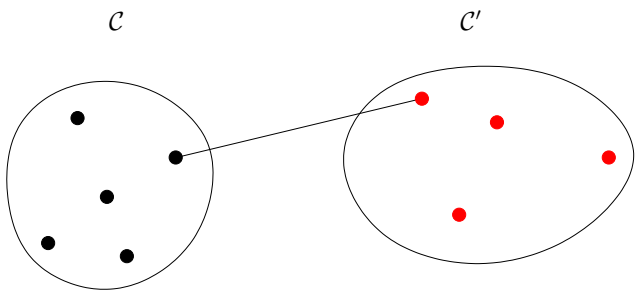
The sequence of length $N-1$ of the merged clusters and their fusion costs can be graphically represented in a special tree graph called **dendrogram**

- *Dendrograms* are tree-like diagrams made of branches that join terminal nodes (*leaves*)
- *Branches* represent clusters and the heights at which the branches are connected represent fusion costs. *Leaves* represent objects
- The *lifetime* of a branch is the difference of fusion costs between the moments it appears and the moment it is aggregated

# The simplest HAC method is the single linkage method

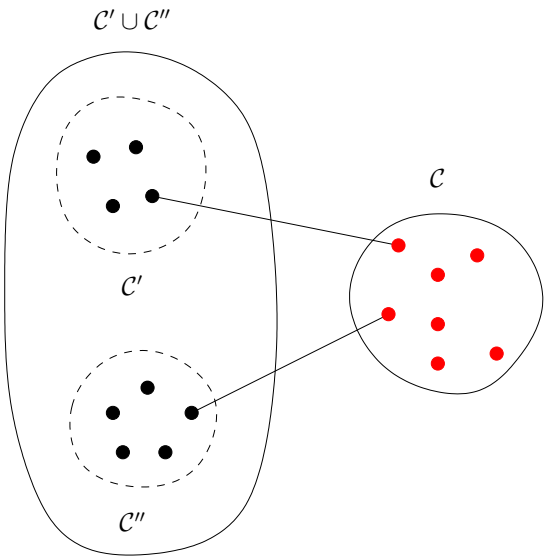
The most basic HAC algorithm is single-linkage The fusion cost between two clusters $\mathcal{C}$ and $\mathcal{C}'$ is defined as the distance between their nearest pair of points (one in each cluster), i.e.,

$$D(\mathcal{C}, \mathcal{C}') = \min_{x \in \mathcal{C}, x' \in \mathcal{C}'} d(x, x')$$



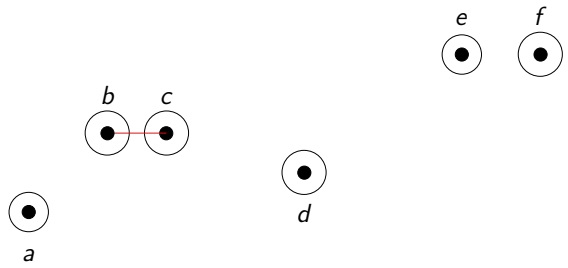
# Updating formula for the single linkage

$$D(\mathcal{C}' \cup \mathcal{C}'', \mathcal{C}) = \min\{D(\mathcal{C}', \mathcal{C}), D(\mathcal{C}'', \mathcal{C})\}$$



# Example of single-linkage clustering: step -1

$X = \{a, b, c, d, e, f\}$




PROXIMITY MATRIX

| | a | b | c | d | e |
|---|---|---|---|---|---|
| b | 0.7 | | | | |
| c | 1.0 | 0.3 | | | |
| d | 1.8 | .1.3 | 0.9 | | |
| e | 2.9 | 2.4 | 1.9 | 1.3 | |
| f | 3.4 | 2.8 | 2.4 | 1.7 | .5 |

At the initial step all clusters are singletons

Next step merges the clusters $\{b\}$ and $\{c\}$

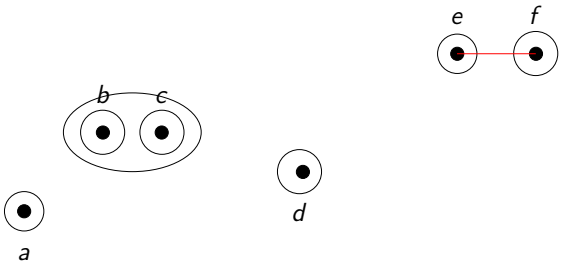
with fusion cost 0.3 (the least dissimilar pair) and in each dashed box

the minimum value is chosen, reducing the proximity matrix order by one,

and defining the dissimilarities between each one of the singletons and the new formed cluster $\{b, c\}$

# Step -2

$X = \{a, b, c, d, e, f\}$

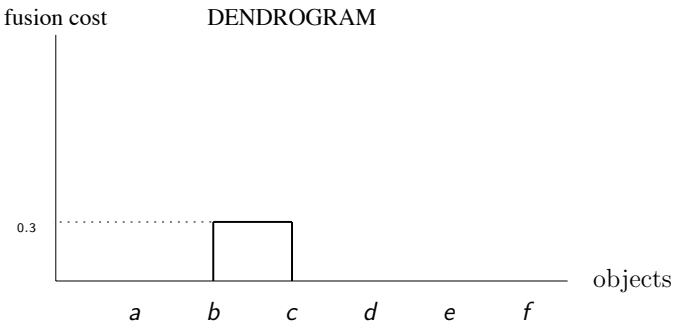



PROXIMITY MATRIX

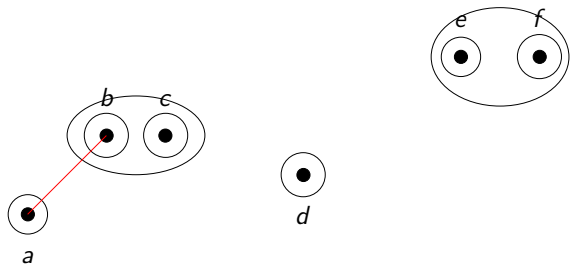
| | $a$ | $\{b, c\}$ | $d$ | $e$ |
|---|---|---|---|---|
| $\{b, c\}$ | 0.7 | | | |
| $d$ | 1.8 | 0.9 | | |
| $e$ | 2.9 | 1.9 | 1.3 | |
| $f$ | 3.4 | 2.4 | 1.7 | 0.5 |

Next step merges the singletons $\{e\}$ and $\{f\}$

with fusion cost 0.5



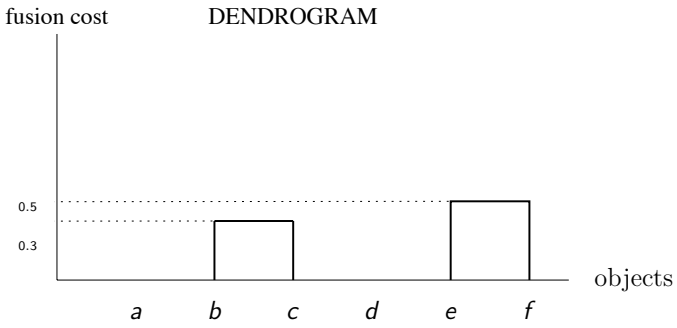
# Step - 3



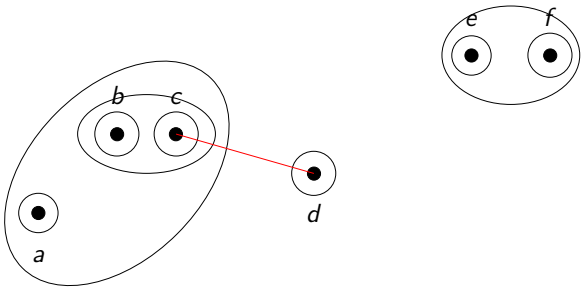

PROXIMITY MATRIX

| | $a$ | $\{b,c\}$ | $d$ |
|---|---|---|---|
| $\{b,c\}$ | 0.7 | | |
| $d$ | 1.8 | 0.9 | |
| $\{e,f\}$ | 2.9 | 1.9 | 1.3 |

Next step merges the pair of clusters $\{a\}$ and $\{b,c\}$ with fusion cost 0.7



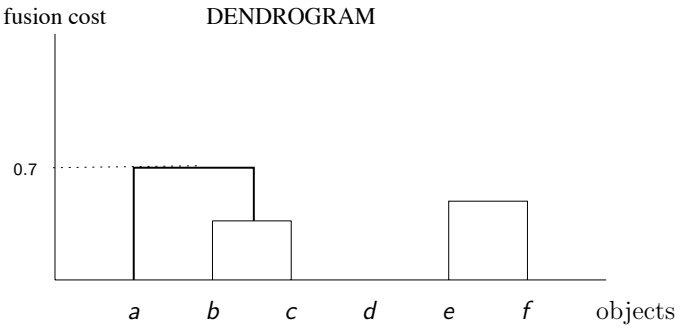
# Step - 4



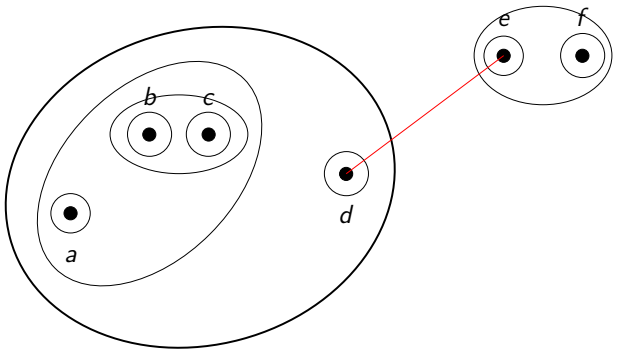

PROXIMITY MATRIX

| | $\{a, b, c\}$ | $d$ |
|---|---|---|
| $d$ | 0.9 | |
| $\{e, f\}$ | 1.9 | 1.3 |

Next step merges the clusters $\{a, b, c\}$ and $\{d\}$ with fusion cost 0.91



# Step - 5

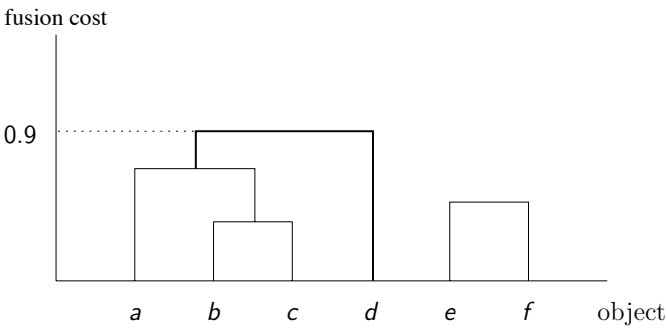



Next step is the final one and merges the clusters $\{a, b, c, d\}$ and $\{e, f\}$ with fusion cost 1.3

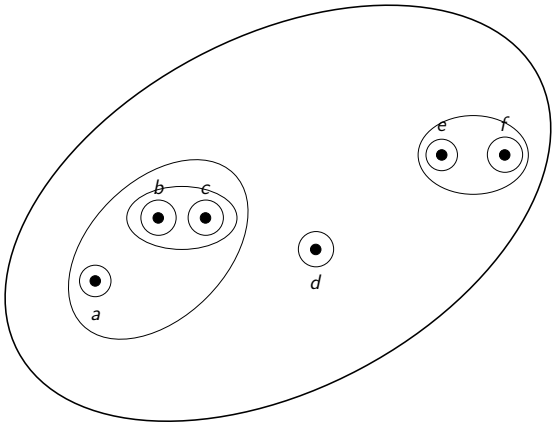
PROXIMITY MATRIX

| | $\{a, b, c, d\}$ |
|---|---|
| $\{e, f\}$ | 1.3 |

DENDROGRAM



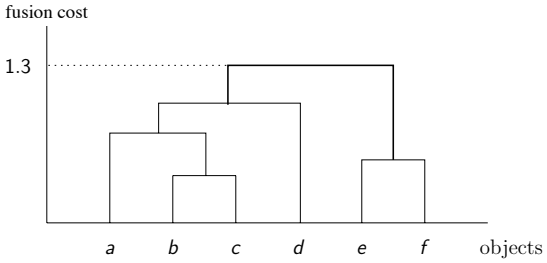
# step - 6




PROXIMITY MATRIX

EMPTY

DENDROGRAM



# The R function hclust

It performs hierarchical aglomerative clustering using several aggregation criterion methods and admits an arbitrary dissimilarity matrix as input

**input**: a *dissimilarity matrix d* and the clustering *method* among the options, "ward", "single", "complete" (default), "average", "mcquitty", "median" or "centroid".

**value**: the function returns an object of the class *hclust*, which consists of a list including, among others, the following elements:
*merge*: a $(n-1) \times 2$ matrix indicating the clusters being merged
*heigth*: the list of fusion costs

R

```
hc<-hclust(d, method=''complete'', members=NULL)
plot(hc) or plot(hc, hang=-1) to plot the dendrogram with all leaves at the same height
```

# Example

## R (SL-1)

```
X<-matrix(c(0,0,0.5,0.5,0.85,0.5,1.75,0.25,2.75,1,3.25,1),
nrow=6,byrow=TRUE)
d<-dist(X)
SL<-hclust(d, method="single")
SL$height
[1] 0.375 0.5 0.707 0.91 1.25
SL$merge
[,1] [,2]
[1,] -2 -3 (merges singletons {2} with {3})
[2,] -5 -6 ( merges singletons {5} with {6})
[3,] -1 1 (merges singleton {1} with cluster {2,3})
[4,] -4 3 (merges singleton {4} with cluster {1,2,3})
[5,] 2 4 (merges clusters {5,6} with cluster {1,2,3})
# The number with minus sign refers to a singleton ID,
# otherwise refers to the step number where the cluster was aggregated
plot(SL)
plot(SL, hang=-1)
# try with horiz=TRUE
```

# Where to cut the dendrogram?

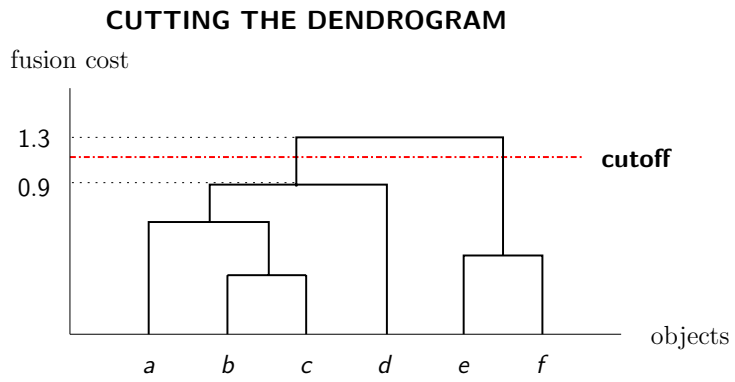
A cut in a dendrogram at a given height $\tau$ produces the (flat) partition into the clusters whose fusion cost is smaller than or equal to $\tau$
Usually one seeks cuts in the dendrogram such that:

- **splits high consecutive height differences (high lifetimes)** to get high inter-cluster heterogeneity
- **as close to the leaves as possible** to get high intra-class homogeneity

Some caution has to be applied regarding the decision where to cut the dendrogram (and what is the "best" number of clusters). With some methods (for instance, the Ward method), the dendrogram height distances tend to be higher when larger clusters are merged
Several internal validity indices can be used complementarly to estimate the optimal number of clusters

# Example

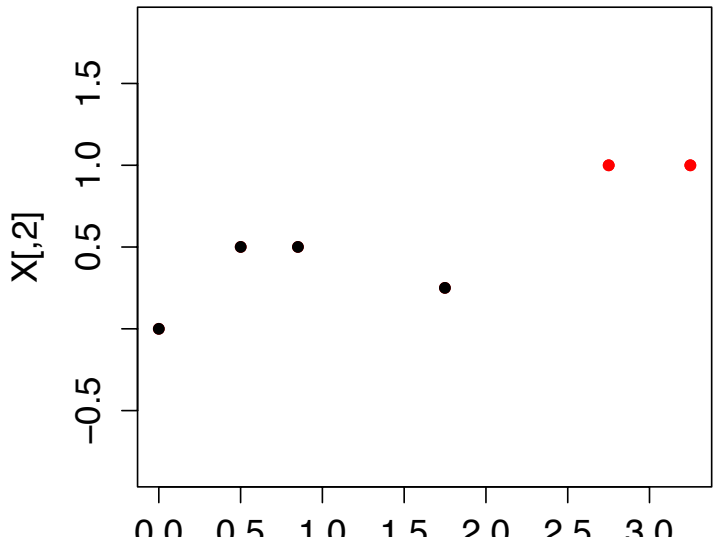
For instance to obtain a partition into 2 clusters we have to cut the dendrogram at some height in the interval $]0.9, 1.3[$, yielding the clusters $\mathcal{C} = \{a, b, c, d\}$ and $\mathcal{C}' = \{e, f\}$



# Cutting the dendrogram in R

## R (SL-1-cutree)

```
SL<-hclust(X,method="single")
part<-cutree(SL,2) # 2 clusters
# # or
part<-cutree(SL,h=1.1) # h is the height
part

plot(X,type="p",cex=0.8,pch=16, col=part,asp=TRUE)
```



# Contraction of the attribute space

- In single-linkage two clusters are merged at a fusion cost $\tau$ if and only if there is a pair of objects, one in each cluster, with pairwise distance inferior or equal than $\tau$
- As the cluster growths it becames more and more easier to incorporate new elements in a cluster, as if the distances between objects were getting smaller and smaller (actually, the distance of a point to any point in the cluster becames the distance to the nearest point of the cluster)
- New individuals tend to aggregate to existing clusters, often producing elongated clusters (chain effect)

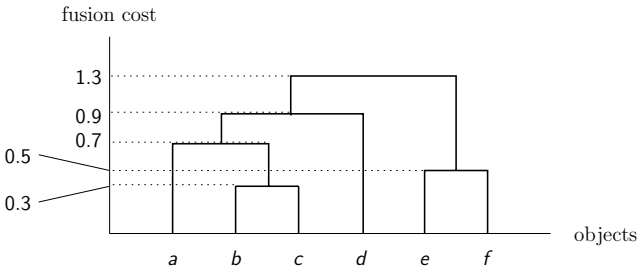
## Cophenetic distance

- Introduced by Sokal and Rohlf
- It assess how well the dendrogramatic distance preserve the original distances
- Measures the goodness-of-fit of the adjustment of the dendrogramatic distances, i.e., cophenetic distances, to the original dataset
- It is considered an internal validation criterion for evaluating the efficiency of various clustering techniques, particularly for hierarchical methods

# Cophenetic distance

*The* **cophenetic distance** *between two individuals $x$ and $y$ with respect to a given HAC is the merging cost at which $x$ and $y$ became members of the same cluster, during the course of hierachical clustering. Cophenetic distances are distances in the usual sense, i.e., they are dissimilarities that verify the triangle inequality, under the assumption of monotonicity*

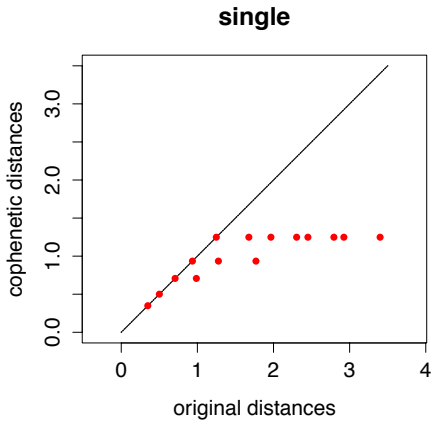
Any dendrogram can be uniquely represented by its matrix of cophenetic distances. This matrix can be used to compare distinct classifications




| | $a$ | $b$ | $c$ | $d$ | $e$ |
|---|---|---|---|---|---|
| $b$ | 0.7 | · | · | · | · |
| $c$ | 0.7 | 0.3 | · | · | · |
| $d$ | 0.9 | 0.9 | 0.9 | · | · |
| $e$ | 1.3 | 1.3 | 1.3 | 1.3 | · |
| $f$ | 1.3 | 1.3 | 1.3 | 1.3 | 0.5 |

# Shepard-like diagram

The Shepard-like diagram of the previous example shows that the cophenetic distances are smaller than the original ones, due to the *contraction of the space of attributes*.




**Shepard diagrams** *are scatter plots commonly used to visualize comparisons between distances*

# Distortion measures - Cophenetic Pearson Coefficient

The **cophenetic correlation coefficient** (CPCC) is the Pearson's correlation between the original and the cophenetic distances (using half of the proximity matrix), i.e.,

$$CPCC = \frac{\text{cov}(D, C)}{\sigma_D \sigma_C} = \frac{\sum_{i<j}(d_{ij} - \bar{d})(c_{ij} - \bar{c})}{\sqrt{\sum_{i<j}(d_{ij} - \bar{d})^2 \sum_{i<j}(c_{ij} - \bar{c})^2}}$$

where $C = (c_{ij})$ and $D = (d_{ij})$ are the vectors containing the original and cophenetic distances

- CPCC is considered an internal validation criterion for hierarchical clustering
- A high value of the CPCC means that the cophenetic distances are a good portray of the original distances
- The cophenetic correlation usually ranges between 0.6 and 0.95.
- Cophenetic correlations between 0.7 and 0.8 are considered *reasonable good*, between 0.8 and 0.9 *good* and above 0.9 *very good*.

# Distortion measures - Cophenetic Spearman Coefficient

Another distortion measure is **Cophenetic Spearman's rank order correlation coefficient** (CSCC), which only depends on the ranks of the variables and is computed as the Pearson's correlation between the respective ranked variables $rk(C) = (c'_{ij})$ and $rk(D) = (d'_{ij})$ defined by the vectors of original and cophenetic distances,

$$CSCC = \frac{\text{cov}(rk(D), rk(C))}{\sigma_{rk(D)}\sigma_{rk(C)}} = \frac{\sum_{i<j}(d'_{ij} - \bar{d})(c'_{ij} - \bar{c'})}{\sqrt{\sum_{i<j}(d'_{ij} - \bar{d'})^2 \sum_{i<j}(c'_{ij} - \bar{c'})^2}}.$$

- Unlike the Pearson correlation coefficient, Spearman's rank order correlation coefficient can be applied to compare original and cophenetic dissimilarities even if there is no linear relation between both dissimilarities
- A Spearman's rank order correlation close to 1 signifies that we have a strong positive relationship between the ranks of original and of cophenetic distances, i.e., the higher the original distances the higher the corresponding cophenetic distances and vice-versa. If the rank order correlation is close to -1 an opposite behavior occurs

# Shepard diagram and cophenetic correlations in R

R (SL-3)

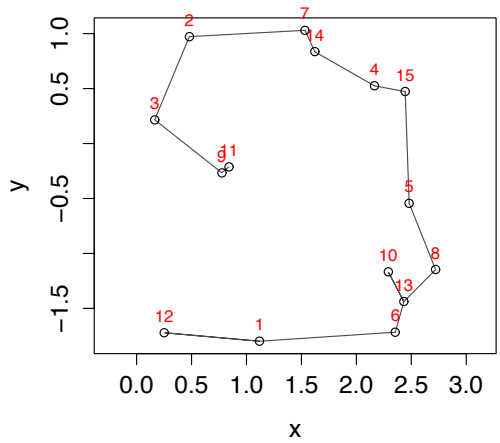
```
coph.SL<-cophenetic(SL) # an object of type dist

cor.coph<-cor(coph.SL,d)

cor.coph

sp.coph<-cor(rank(coph.SL),rank(d))

sp.coph

plot(seq(0,3.5,.01),seq(0,3.5,.01),pch=16,type="p",
cex=.2, main="single", asp=T, xlab="original distances",
ylab="cophenetic distances")

points(d,coph.SL,asp=1,xlim=c(0,5),ylim=c(0,5),pch=16,
col="red",type="p")
```

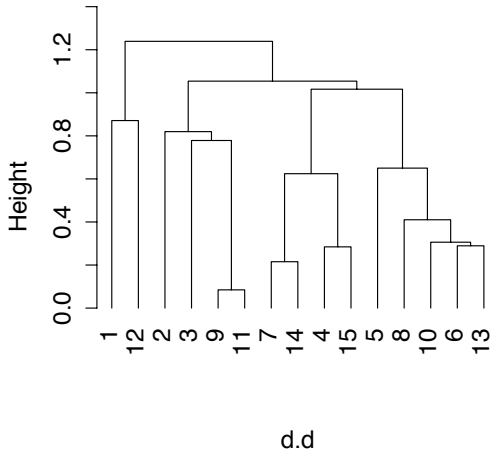
# MST

- An minimum spanning tree (MST) of a set of points in a $d$-dimensional Euclidean space is a tree connecting all points with line segments such that the sum of the lengths of the segments is minimal
- MST are not necessarily unique (if we have ties in the dissimilarity matrix)
- The concept can also be defined for abstract graphs

# Single-Linkage and MST algorithm

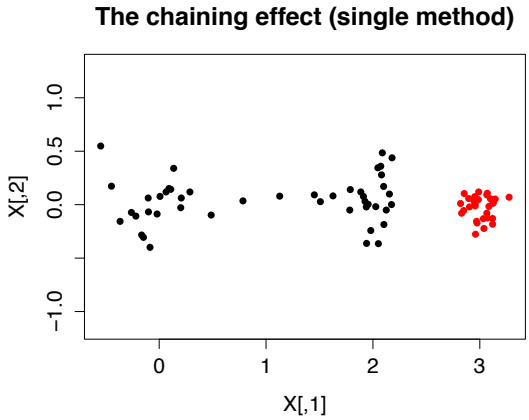
The hierarchical clustering of a cloud of points with the single linkage can be obtained in the following way: consider an MST connecting all points and sequencially aggregate the clusters whose pair of nearest neighbors correspond to vertices of the edges of the MST, ordered from the smallest to the largest length

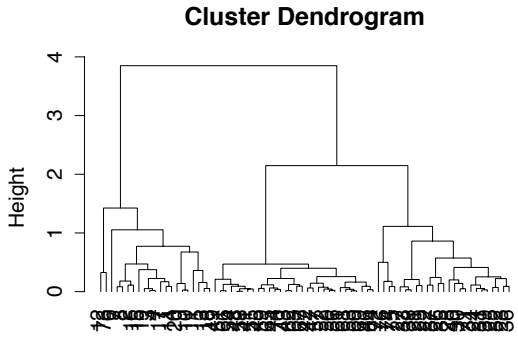


# Chaining effect

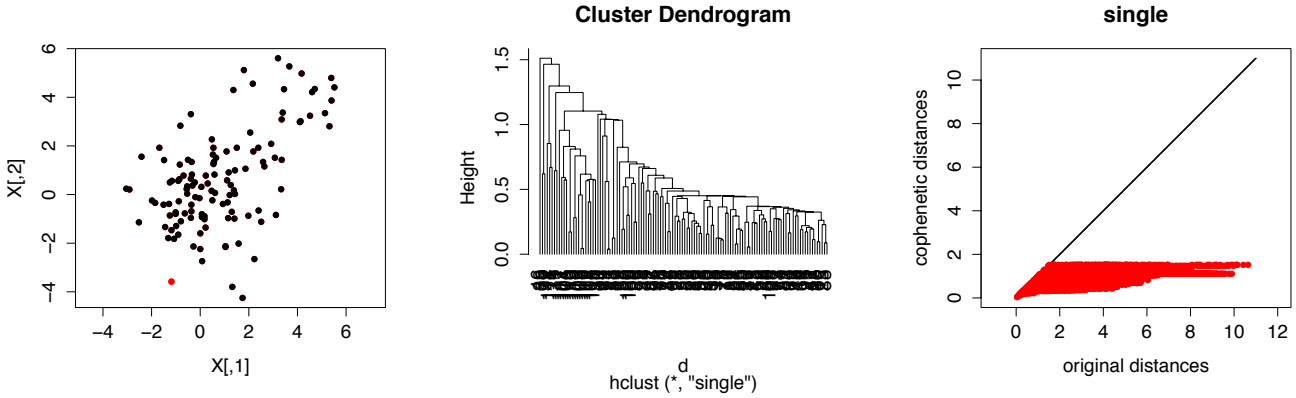
The chaining effect is usually produced by the existence of a few intermediate points between clusters, giving rise to elongated clusters connecting distant points



# Chaining effect

- In the next example the single linkage produced a 2-partition with an elongated cluster and a singleton (red dot). The chaining effect is clearly visible in the dendrogram where long chains of nearby points are aggregated
- All points are aggregated at very lower costs: the maximum heights of the red bars in the shepard diagram is very low (below 2), while the original distances range from 0 to above 10



# Single-linkage clustering - summary

**Pros**

- Can detect arbitrary cluster shapes
- Can be applied to large datasets, since it is computationally efficient - there are polynomial-time algorithms
- Invariant under monotonic transformations of the proximity matrix - only ordinal properties, i.e., rank orders, are important
- Emphasis clusters separation
- Insensitive to ties in the proximity matrix

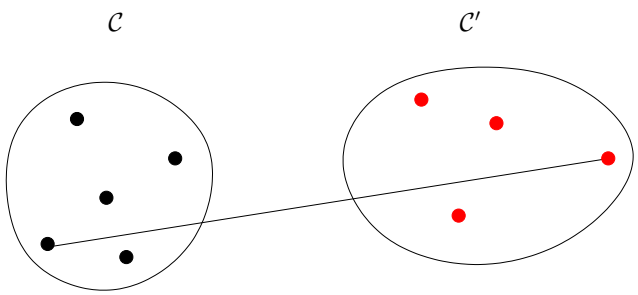
**Cons**

- Contracts the space of attributes
- Sensitive to observation errors and noise
- Suffers from the chain effect, producing elongated clusters, often originating very unbalanced clusters

# Complete-linkage clustering model

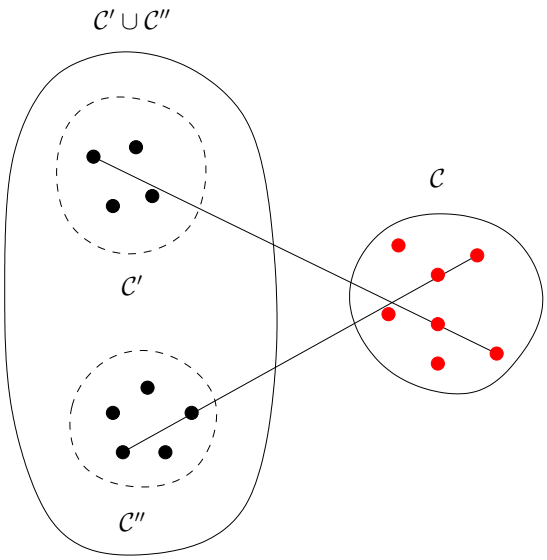
The complete-linkage or farthest sorting is the opposite of nearest-neighbor clustering algorithm The fusion cost between two clusters $\mathcal{C}$ and $\mathcal{C}'$ in this method is defined as the distance between the farthest pair of points, one in each cluster, that is,

$$D(\mathcal{C}, \mathcal{C}') = \max_{x \in \mathcal{C}, x' \in \mathcal{C}'} d(x, x')$$



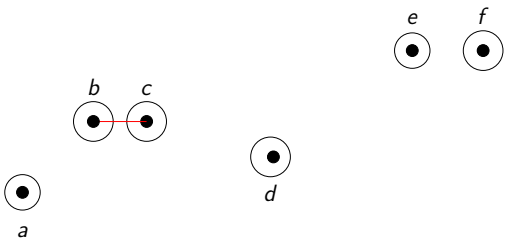
# Updating formula for complete linkage

$$D(\mathcal{C}' \cup \mathcal{C}'', \mathcal{C}) = \max\{D(\mathcal{C}', \mathcal{C}), D(\mathcal{C}'', \mathcal{C})\}$$



# Complete-linkage: step -1

$X = \{a, b, c, d, e, f\}$



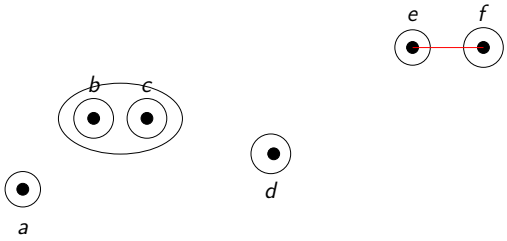

First pair to be merged is $\{b, c\}$ with fusion cost 0.3

PROXIMITY MATRIX

| | a | b | c | d | e |
|---|---|---|---|---|---|
| b | 0.7 | | | | |
| c | 1.0 | 0.3 | | | |
| d | 1.8 | 1.3 | 0.9 | | |
| e | 2.9 | 2.4 | 1.9 | 1.3 | |
| f | 3.4 | 2.8 | 2.4 | 1.7 | .5 |

# Step -2

$X = \{a, b, c, d, e, f\}$

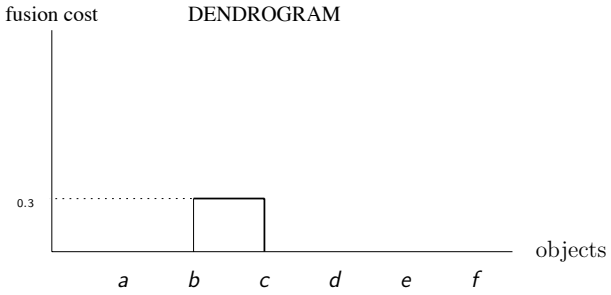



PROXIMITY MATRIX

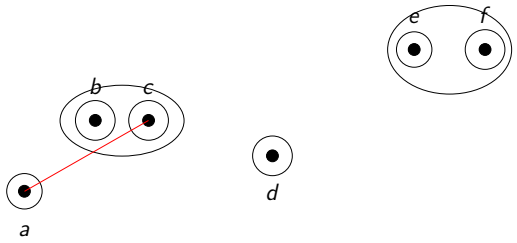
| | $a$ | $\{b, c\}$ | $d$ | $e$ |
|---|---|---|---|---|
| $\{b, c\}$ | 1.0 | | | |
| $d$ | 1.8 | 1.3 | | |
| $e$ | 2.9 | 2.4 | 1.3 | |
| $f$ | 3.4 | 2.8 | 1.7 | 0.5 |

Next pair to be merged is $\{e\}$ and $\{f\}$

with fusion cost 0.5



# Step - 3

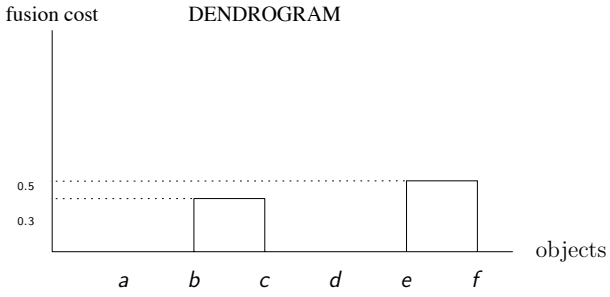



Next pair to be merged is $\{a\}$ and $\{b, c\}$

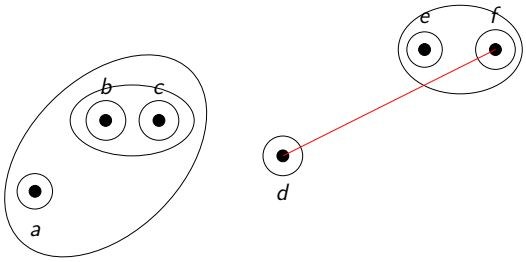
with fusion cost 1.0

PROXIMITY MATRIX

| | $a$ | $\{b, c\}$ | $d$ |
|---|---|---|---|
| $\{b, c\}$ | 1.0 | | |
| $d$ | 1.8 | 1.3 | |
| $\{e, f\}$ | 3.4 | 2.8 | 1.7 |



# Step - 4

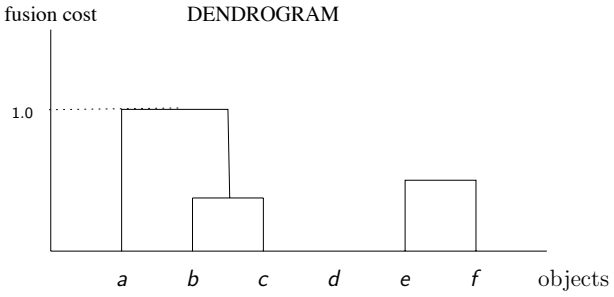



PROXIMITY MATRIX

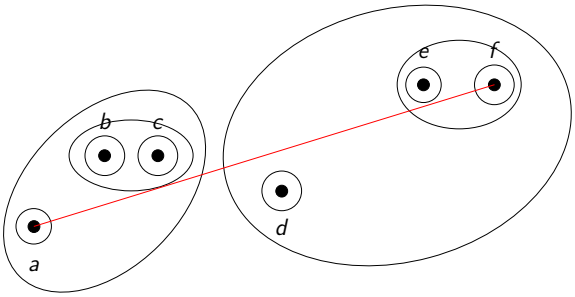
| | $\{a, b, c\}$ | $d$ |
|---|---|---|
| $d$ | 1.8 | |
| $\{e, f\}$ | 3.4 | 1.7 |

Next pair to be merged $\{d\}$ and $\{e, f\}$

with fusion cost 1.7



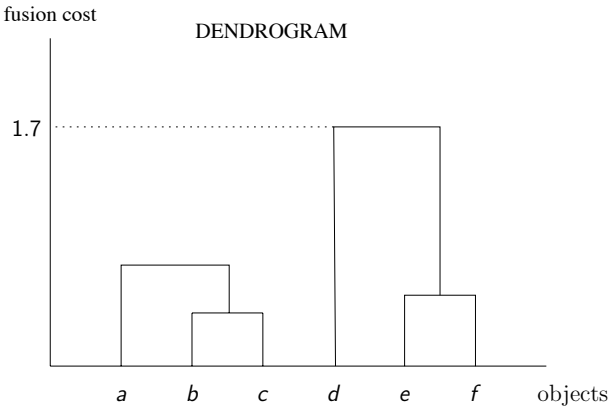
# Step - 5



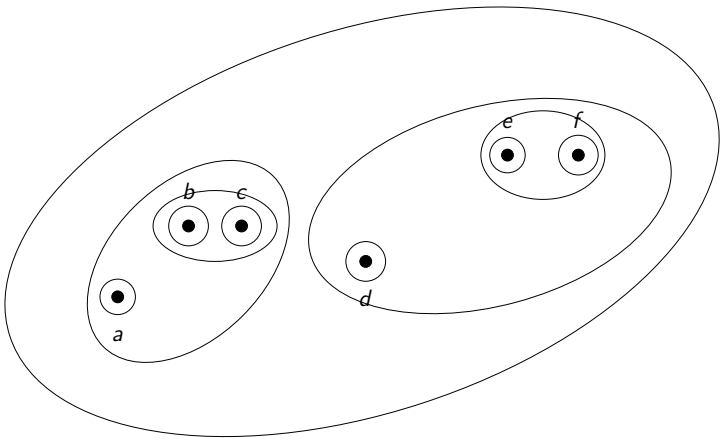

Next step merges (final) merges the pair of clusters $\{a, b, c\}$ and $\{d, e, f\}$ with fusion cost 3.4

PROXIMITY MATRIX

| | $\{a, b, c\}$ |
|---|---|
| $\{d, e, f\}$ | 3.4 |



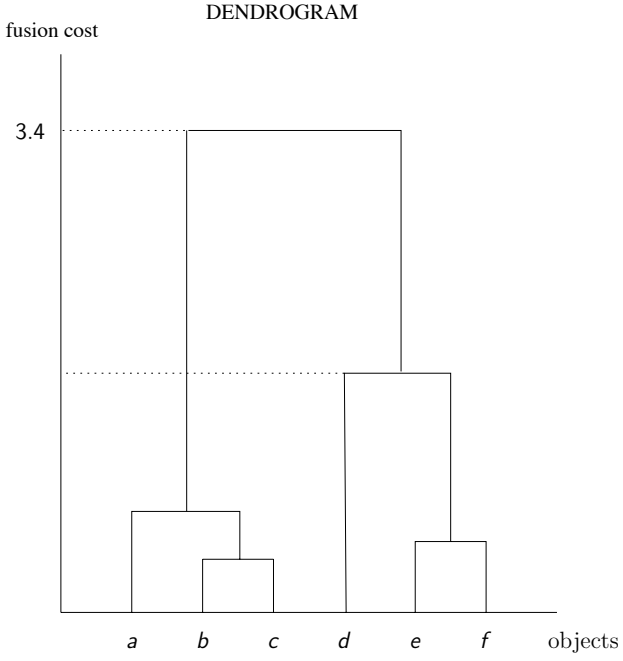
# step - 6




PROXIMITY MATRIX

EMPTY

DENDROGRAM

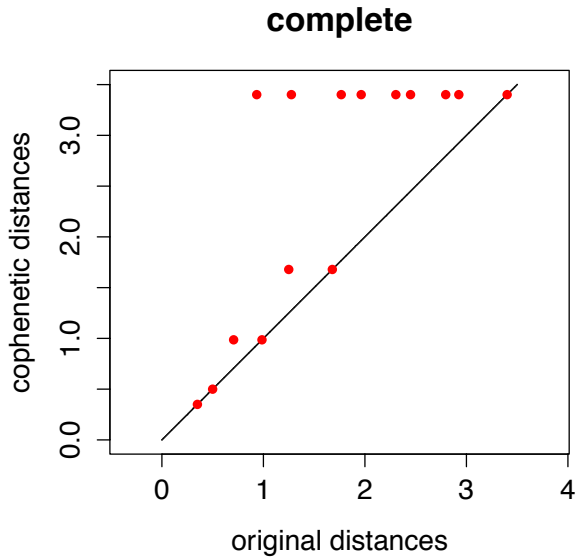


# Dilation of the attribute space

- In complete-linkage two clusters are merged at a fusion cost $\tau$ if and only if all elements of one cluster are at a distance inferior to or equal than $\tau$ with respect to all elements of the other cluster.
- As the cluster growths it becames more and more harder to incorporate new elements in a cluster, as if the distances between objects were getting greater and greater. All aggregations tend to occur at small dissimilarities and consecutive dendrogram heights tend to become larger
- This produces a "dilation effect" in the attribute space that can be observed comparing the original proximity matrix and the cophenetic matrix in a scatterplot shepard-like diagram

# Shepard-like diagram and the cophenetic correlation

The Shepard-like diagram of the previous example shows that the cophenetic distances in complete-linakge clustering are greater than the original ones, due to the *dilation of the space of attributes*




The cophenetic and Spearman rank order correlation coefficients are, respectively, $CPCC = 0.75$ and $CSCC = 0.79$

## Complete-linkage clustering - summary

**Pros**

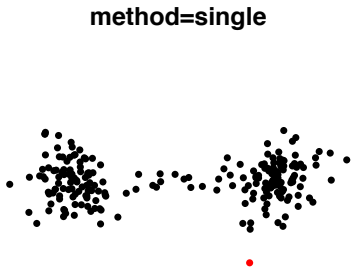
- Favors compactness - tend to form tight spherical clusters with small diameters
- Invariant under monotonic transformations of the proximity matrix - only the dissimilarity ranks are important.

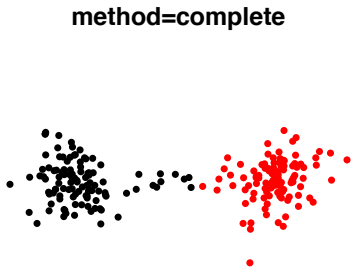
**Cons**

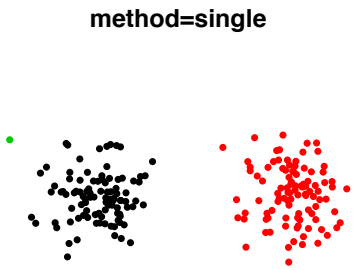
- Dilates the space of attributes
- The decision of aggregate two cluster only relies on one individual in each cluster
- Sensitive to outliers
- Cannot detect arbitrary cluster shapes

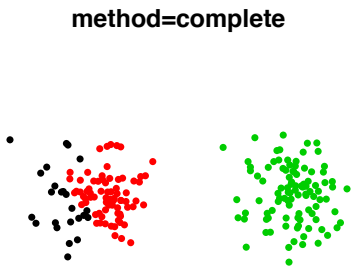
# Noise and outliers: single vs complete methods

Next examples show that single clustering method is more sensitive to noise than complete, whereas the opposite occurs with outliers



# Average clustering

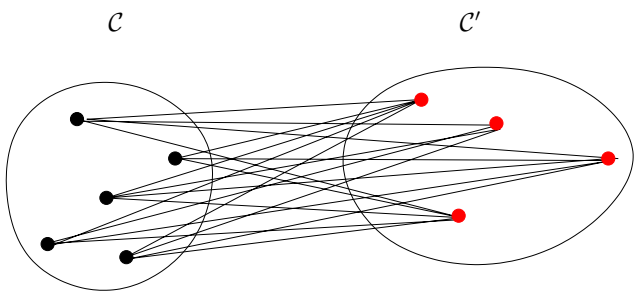
- In-between the contraction effect of single linkage method and the dilation effect of complete linkage method, we have the *unweighted pair group method average* (UPGMA) method, simply known as **average method**, which conserves the metric properties of the attribute space
- This method usually presents the best cophenetic correlation coefficient among the three methods, but is not invariant under monotonic transformations of the proximity matrix
- Average clustering methods sometimes refer to family methods, average (UPGMA), Mcquitty (WPGMA), centroid (UPGMC) and median (WPGMC)

# Average fusion cost

The fusion cost between two clusters $\mathcal{C}$ and $\mathcal{C}'$ is defined as the arithmetic average of the distances connecting one point in $\mathcal{C}$ with one point in $\mathcal{C}'$,

$$D(\mathcal{C}, \mathcal{C}') = \frac{\sum_{x \in \mathcal{C}} \sum_{x' \in \mathcal{C}'} d(x, x')}{n\, n'},$$

where $n = |\mathcal{C}|$ and $n' = |\mathcal{C}'|$



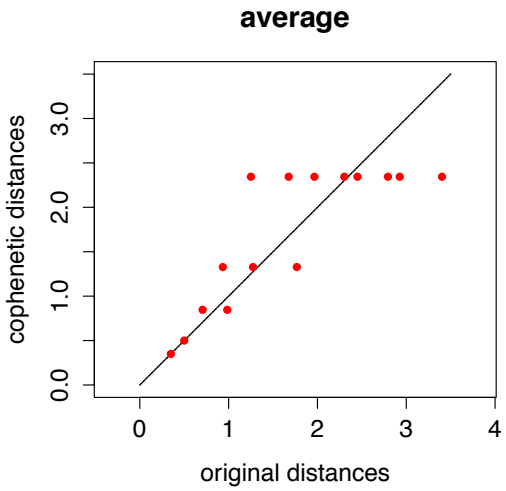
# Updating formula

$$\boxed{D(\mathcal{C}' \cup \mathcal{C}'', \mathcal{C}) = \frac{n' D(\mathcal{C}', \mathcal{C}) + n'' D(\mathcal{C}'', \mathcal{C})}{n' + n''}}$$

where $n' = |\mathcal{C}'|$, $n'' = |\mathcal{C}''|$

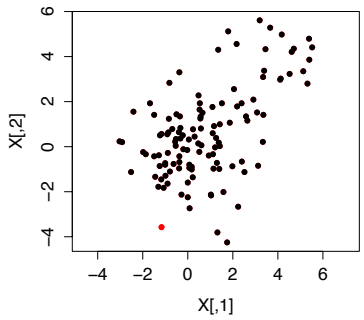
# Shepard-like diagram and the cophenetic correlation

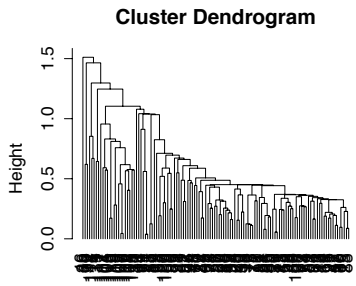
The Shepard-like diagram of the previous example shows that the cophenetic distances lie in both sides of the diagonal yielding a better portray of the original distances. This is also confirmed by the higher cophenetic correlation. This method conserves the metric in the space of attributes



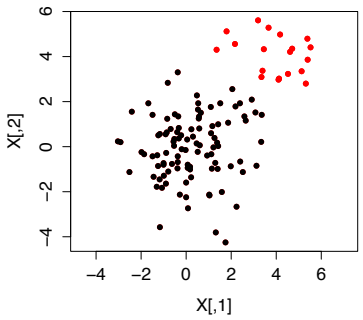

The Pearson and Spearman rank order cophenetic correlations are $CPCC = 0.83$ and $CSCC = 0.84$

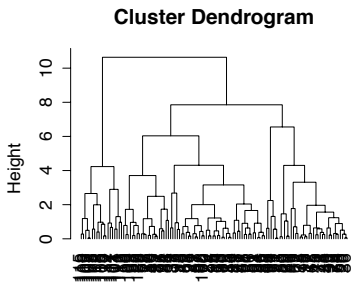
# Comparison single-complete-average



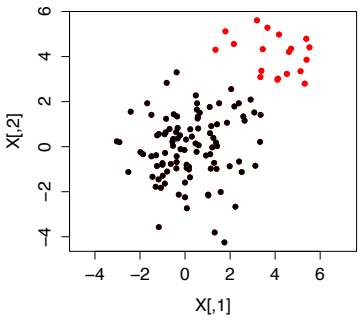



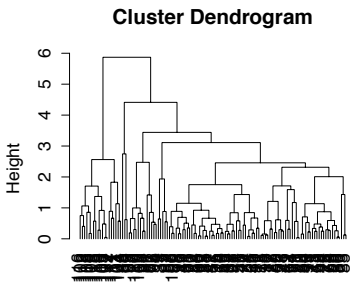

contracting



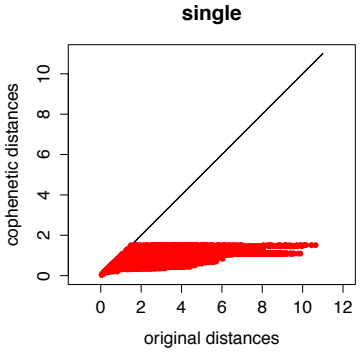

dilating



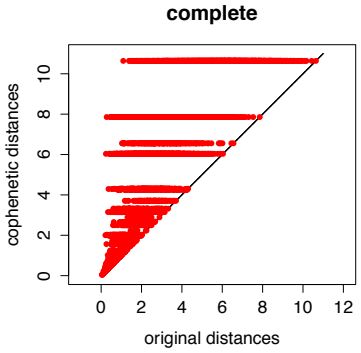

conserving

# Comparison single-complete-average (cophenetic)



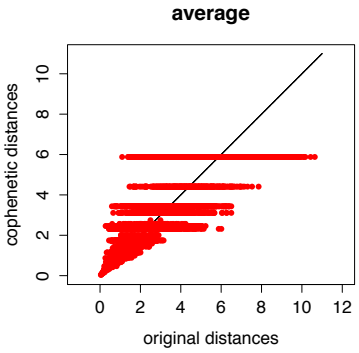

contracting

CPCC=0.69




dilating

CPCC=0.74



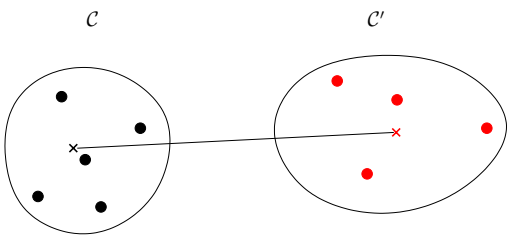

conserving

CPCC=0.82

# Centroid clustering model

In this method, also known as UPGMC (unweighted pair group method centroid) the **clusters are represented by their centroids** and the fusion cost between two clusters $\mathcal{C}_i$ and $\mathcal{C}_j$ is defined as the distance between the respective centroids

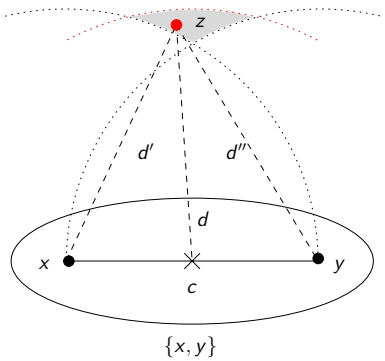
$$D(\mathcal{C}_i, \mathcal{C}_j) = \left\| \frac{1}{|\mathcal{C}_i|} \sum_{x_i \in \mathcal{C}_i} x_i - \frac{1}{|\mathcal{C}_j|} \sum_{x_j \in \mathcal{C}_j} x_j \right\|$$

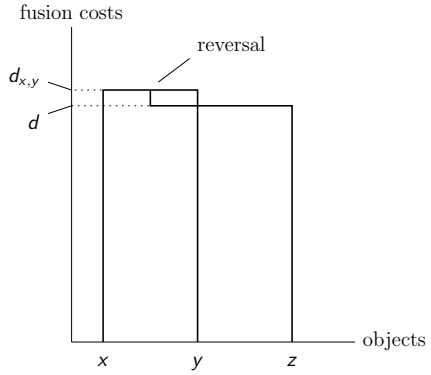



The centroid of the merged group will be $m_{ij} = \dfrac{n_i m_i + n_j m_j}{n_i + n_j}$

# Centroid clustering model - crossovers

In the centroid method the cophenetic distances may not verify the ultrametric property, giving rise to non-monotonic fusion distances with **crossovers** (alo called **inversions**) in the dendrogram
All circles have radii equal to the distance between $x$ and $y$, $d_{x,y}$.



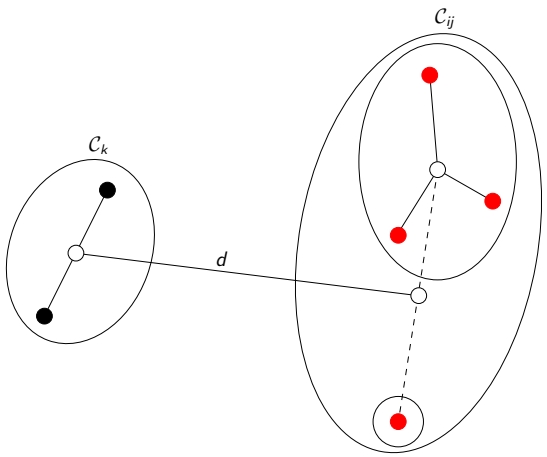

Since $z$ (red point) lie in the grey area, ouside the black circles, $d_{x,y} < d', d''$. Hence $x$ and $y$ are the first pair of objects to be merged.
Since $z$ lie inside the red circle centred at the centroid $c$ of $x$ and $y$,

$$D(\{x, y\}, z) = d_{c,z} < d_{x,y} = D(\{x\}, \{y\})$$

# Median clustering model (WPGMC)

In the centroid clustering if two clusters have very different sizes the centroid of the merged cluster tend to be close or even inside the largest cluster. The median clustering is a variant designed to correct this distortion effect Updating formula: the distance between a cluster $\mathcal{C}_k$ and the cluster $\mathcal{C}_{ij} = \mathcal{C}_i \cup \mathcal{C}_j$, is given by the distance of the centroid of $\mathcal{C}_k$ to the median point of the centroids of $\mathcal{C}_i$ and $\mathcal{C}_j$, i.e.,

$$D(\mathcal{C}_{ij}, \mathcal{C}_k) = \left\| \frac{m_i + m_j}{2} - m_k \right\|$$



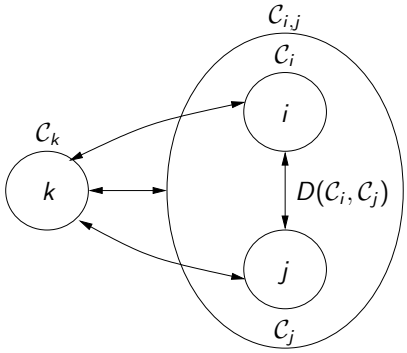
# Lance-Williams updating formula

All HAC methods considered so far can be implemented using a general formula to update the cluster dissimilarities after a merge step in terms of the dissimilarities prior to that fusion

Given clusters $\mathcal{C}_i$, $\mathcal{C}_j$ and $\mathcal{C}_k$, denote by $\mathcal{C}_{ij}$ the cluster obtained merging clusters $\mathcal{C}_i$, $\mathcal{C}_j$ and set

$$\boxed{D(\mathcal{C}_{ij}, \mathcal{C}_k) = \alpha_i D(\mathcal{C}_i, \mathcal{C}_k) + \alpha_j D(\mathcal{C}_j, \mathcal{C}_k) + \beta D(\mathcal{C}_i, \mathcal{C}_j) + \gamma |D(\mathcal{C}_i, \mathcal{C}_k) - D(\mathcal{C}_j, \mathcal{C}_k)|}$$

where $\alpha_i$, $\alpha_j$, $\beta$ and $\gamma$ are convenient parameters and $D(\cdot, \cdot)$ refer the distances squared distances given by the proximity matrix (see the chart in the next slide):



# Lance-Williams chart

| | $\alpha_i$ | $\alpha_j$ | $\beta$ | $\gamma$ | dissimilarity matrix | effect on metric | reversals |
|---|---|---|---|---|---|---|---|
| **single** | $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | $-\frac{1}{2}$ | $d_{ij}$ | very contracting | NO |
| **complete** | $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | $\frac{1}{2}$ | $d_{ij}$ | very dilating | NO |
| **average** (UPGMA) | $\frac{n_i}{n_i+n_j}$ | $\frac{n_j}{n_i+n_j}$ | 0 | 0 | $d_{ij}$ | conserving | NO |
| **McQuitty** (WPGMA) | $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | 0 | $d_{ij}$ | conserving | NO |
| **centroid** (UPGMC) | $\frac{n_i}{n_i+n_j}$ | $\frac{n_j}{n_i+n_j}$ | $\frac{-n_i n_j}{(n_i+n_j)^2}$ | 0 | $d_{ij}^2$ | conserving | can occur |
| **median** (WPGMC) | $\frac{1}{2}$ | $\frac{1}{2}$ | $-\frac{1}{4}$ | 0 | $d_{ij}$ | conserving | can occur |
| **Ward** | $\frac{n_i+n_k}{n_i+n_j+n_k}$ | $\frac{n_j+n_k}{n_i+n_j+n_k}$ | $-\frac{n_k}{n_i+n_j+n_k}$ | 0 | $d_{ij}^2$ | contracting | NO |

# Monotonicity and Crossover

We say that a clustering method satisfies the **monotonicity** condition if whenever two clusters $C_i$ and $C_j$ are merged into a cluster $C_s$ we have

$$D(C_k, C_s) \geq D(C_i, C_j) \qquad \forall k \neq i, j, s$$

This implies that the dendrogram cannot have crossovers

**Proposition**

*If in the Lance-Williams updating formula the parameters $\alpha_i, \alpha_j$ are nonnegative, $\alpha_i + \alpha_j + \beta \geq 1$, and either $\gamma \geq 0$ or $\max\{-\alpha_i, \alpha_j\} \leq \gamma \leq 0$, the corresponding clustering method satisfies the monotonicity condition*

From the Lance-Williams table we deduce immediately that the single, complete, average, McQuitty and Ward methods are in conditions of the proposition and therefore satisfy the monotonicity condition. Therefore their dendrograms cannot have crossovers

# Ultrametric property

We say that a distance $D$ verify the **ultrametric property** if it verifies the following stronger condition than triangle inequality: for all individuals $x, y, z$

$$D(x, z) \leq \max(D(x, y), D(y, z))$$

If the cophenetic distances $d_c(\cdot, \cdot)$ verify the ultrametric property, then cluster process verify the monotonicity condition. Actually, given $x \in C_i$, $y \in C_j$ and $z \in C_k$,

$$D(\mathcal{C}_i \cup \mathcal{C}_j, \mathcal{C}_k) = d_c(x, z) = d_c(y, z)$$

and we get

$$D(\mathcal{C}_i, \mathcal{C}_j) = d_c(x, y) \leq \max\{d_c(x, z), d_c(y, z)\} = D(\mathcal{C}_i \cup \mathcal{C}_j, \mathcal{C})$$

The single-linkage, complete-linkage and average cophenetic distances verify the ultrametric property.
Since the centroid can present inversions, its cophenetic distance is not ultrametric

# Ward's method

Let $X$ be a dataset of $N$ individuals, $x_1, \ldots, x_N$ in $p$ (quantitative) variables with mean $\bar{x}$. Given a partition of $X$ into $K$ clusters

$$X = \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_K$$

we define

$$SSQ_t = \sum_{i=1}^{N} \|x_i - \bar{x}\|^2 \text{ (total inertia)}$$

$$SSQ_b = \sum_{k=1}^{K} n_k \|m_k - \bar{x}\|^2 \text{ (between-clusters inertia)}$$

$$SSQ_w = \sum_{k=1}^{K} \sum_{x \in \mathcal{C}_k} \|x - m_k\|^2 \text{ (total within-clusters inertia),}$$

where $m_k$ is the centroid of cluster $\mathcal{C}_k$ and $n_k$ the number of its elements

*By Huygens theorem,*

$$SSQ_t = SSQ_b + SSQ_w$$

# Ward's method

In Ward's method **each cluster is represented by its centroid** and the goal is **to maximize the between-clusters inertia** $SSQ_b$ as the groups are being clustered. By Huygens theorem this is equivalent **to minimize the total within-group inertia** $SSQ_w$, i.e., to minimize the information loss with the replacement of the elements of each cluster by the cluster centroid At beginning all clusters have a single element and thus,

$$SSQ_t = SSQ_b, \qquad SSQ_w = 0$$

Note that $SSQ_b$ correspond to the sum squared pairwise distances In each step, Ward's method tries to merge the pair of clusters $\mathcal{C}_i$, $\mathcal{C}_j$ that produce the smallest increase in the total within-cluster inertia

# Increase in the sum of within-cluster inertia

Since

$$SSQ_w = \sum_{k=1}^{K} e_k^2,$$

where $e_k^2$ is the within group inertia of cluster $k$,i.e.,

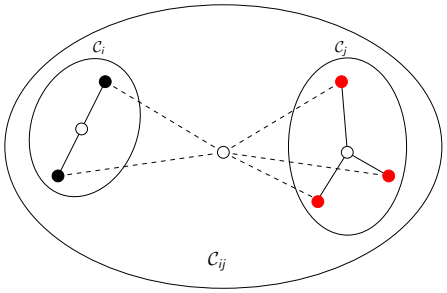
$$e_k^2 = \sum_{x \in \mathcal{C}_k} \|x - m_k\|^2 = \frac{\sum_{x,y \in \mathcal{C}_k} \|x - y\|^2}{2n_k}$$

When two clusters $\mathcal{C}_i$ and $\mathcal{C}_j$ are merged into a cluster $\mathcal{C}_{ij}$, the increase in $SSQ_w$ reduces to the following $\Delta SSQ_w$ statistic:

$$\Delta_{i,j} SSQ_w = e_{ij}^2 - e_i^2 - e_j^2$$

since all other within-group inertias are not affected

# Updating formula




It can be proved that

$$\Delta_{ij} SSQ_w = \frac{n_i n_j}{n_i + n_j} \|m_i - m_j\|^2,$$

In particular $\Delta_{ij} SSQ_w$ depends on the squared pairwise dissimilarities between the cluster centroids $m_i$, $m_j$ and the cluster sizes $n_i$ and $n_j$ One can use the above formula to determine the next pair to be clustered but it requires the original coordinates of the points (raw data) to be known. There is an alternative formula that only requires the (squared) pairwise distances

# E.g.: Ward clustering via LW updating formula

The LW updating formula for Ward's method is given by

$$D^2(\mathcal{C}_i \cup \mathcal{C}_j, \mathcal{C}_k) = \frac{(n_i + n_k) \cdot D^2(\mathcal{C}_i, \mathcal{C}_k) + (n_j + n_k) \cdot D^2(\mathcal{C}_j, \mathcal{C}_k) - n_k \cdot D^2(\mathcal{C}_i, \mathcal{C}_j)}{n_i + n_j + n_k}$$

where $n_i = |\mathcal{C}_i|$, $n_j = |\mathcal{C}_j|$ and $n_k = |\mathcal{C}_k|$
This expression returns the increase in the $SSQ_w$ statistic when clusters $\mathcal{C}_i \cup \mathcal{C}_j$ and $\mathcal{C}_k$ are merged, depending only on the clusters involved Let us exemplify on the dataset $X = \{a, b, c, d\} = \{1, 2, 4, 8\}$
The dissimilarity and squared dissimilarity matrices are, respectively,

$$\left[\begin{array}{c|ccc} D & a & b & c \\ \hline b & 1 & & \\ c & 3 & 2 & \\ d & 7 & 6 & 4 \end{array}\right], \qquad \left[\begin{array}{c|ccc} D^2 & a & b & c \\ \hline b & 1 & & \\ c & 9 & 4 & \\ d & 49 & 36 & 16 \end{array}\right]$$

The minimum of the squared distances is attained for $D^2(a, b)$ so the first pair to be clustered will be $a \cup b$ with squared fusion cost 1

# Ward clustering using LW updating formula (cont.)

$$
\begin{aligned}
D^2(a \cup b, c) &= \frac{2\,D^2(a,c) + 2\,D^2(b,c) - D^2(a,b)}{3} \\
&= \frac{2 \cdot 9 + 2 \cdot 4 - 1}{3} = \frac{25}{3}
\end{aligned}
$$

and

$$
\begin{aligned}
D^2(a \cup b, d) &= \frac{2\,D^2(a,d) + 2\,D^2(b,d) - D^2(a,b)}{3} \\
&= \frac{2 \cdot 49 + 2 \cdot 36 - 1}{3} = \frac{169}{3}
\end{aligned}
$$

$D^2(c,d)$ is not affected. Thus the new squared dissimilarity matrix is

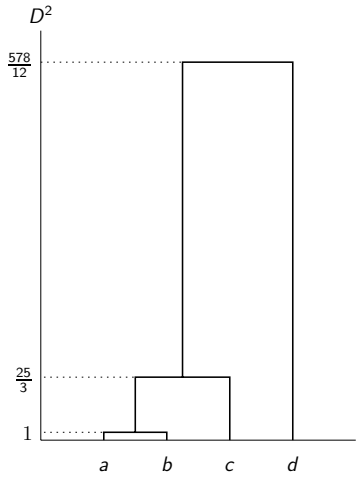
$$
\left[\begin{array}{c|cc}
D^2 & a \cup b & c \\
\hline
c & \frac{25}{3} & \\
d & \frac{169}{3} & 16
\end{array}\right]
$$

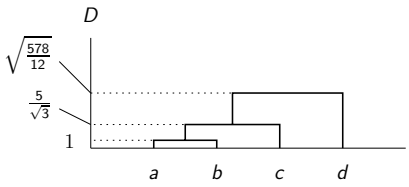
The minimum of the squared distances is attained for $D^2(a \cup b, c)$ so the next pair to be clustered will be $(a \cup b) \cup c$ with squared fusion cost $\frac{25}{3}$

# Ward clustering using LW updating formula (concl.)

$$
\begin{aligned}
D^2((a \cup b) \cup c, d) &= \frac{3\,D^2(a \cup b, d) + 2\,D^2(c, d) - D^2(a \cup b, c)}{4} \\
&= \frac{3 \cdot \frac{169}{3} + 2 \cdot 16 - \frac{25}{3}}{4} = \frac{578}{12}
\end{aligned}
$$

The dendrogram can be presented either using squared or not squared fusion costs. Its topology however does not change



## Computing using the R function *hclust*

R

```
X<-c(1,2,4,8)
d<-dist(X) # (euclidean) distance matrix
h.ward<-hclust(d,method="ward.D2")
h.ward$height
plot(h.ward, hang=-1)
```

## Ward's clustering method - summary

**Pros**

- Tend to create hyperspherical shape clusters, with approximately the same number of elements each (balanced)
- Conservative (some authors consider it contracting although less than single-linkage method)
- No crossovers
- Is regarded by some authors as the companion hierarchical method to use with correspondence analysis (CA) since it shares the same variance criterion

**Cons**

- Computationally intensive
- Cannot detect arbitrary cluster shapes
- Sensitive to outliers since it uses centroids

# HAC - summary

**Pros**

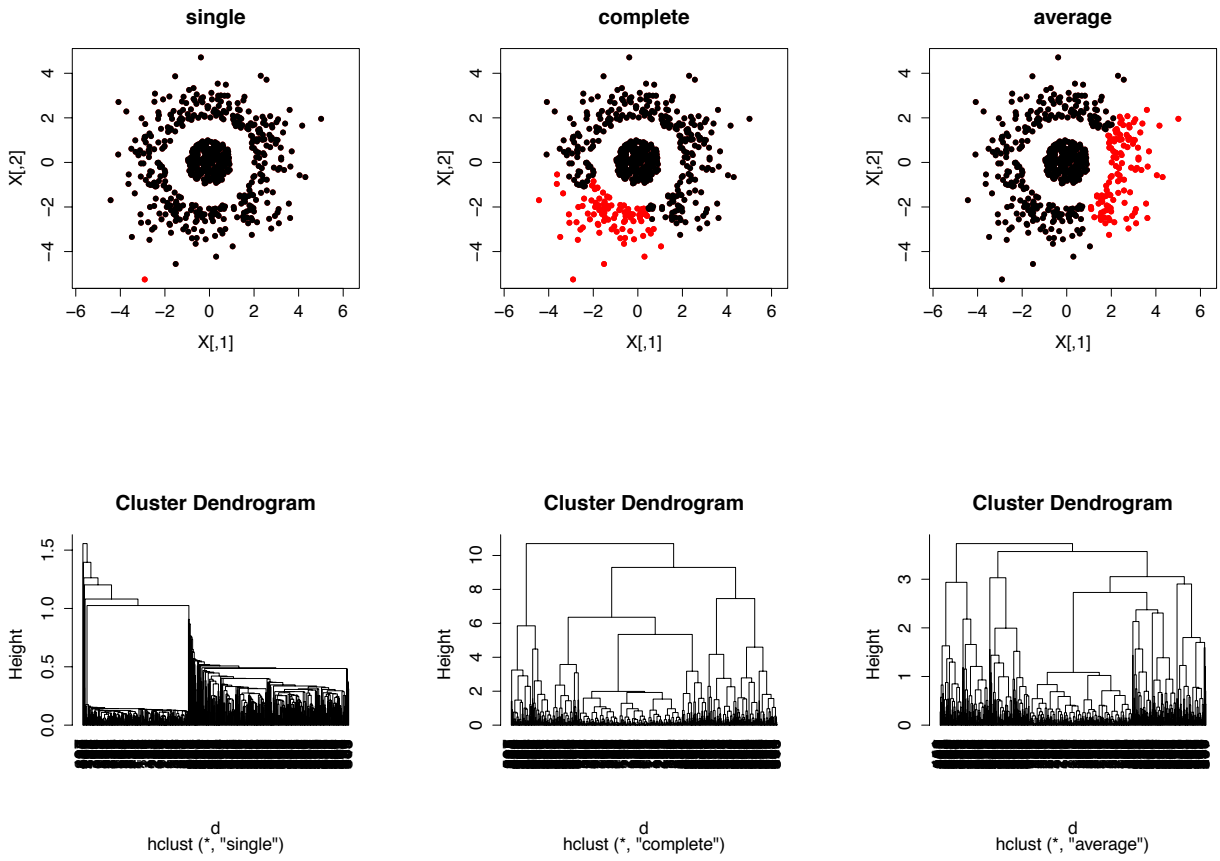
- The dendrogram provides "taxonomical information" on the clusters
- The number of clusters does not need to be defined *a priori*
- Many methods rely on a proximity matrix allowing almost any kind of resemblance notion

**Cons**

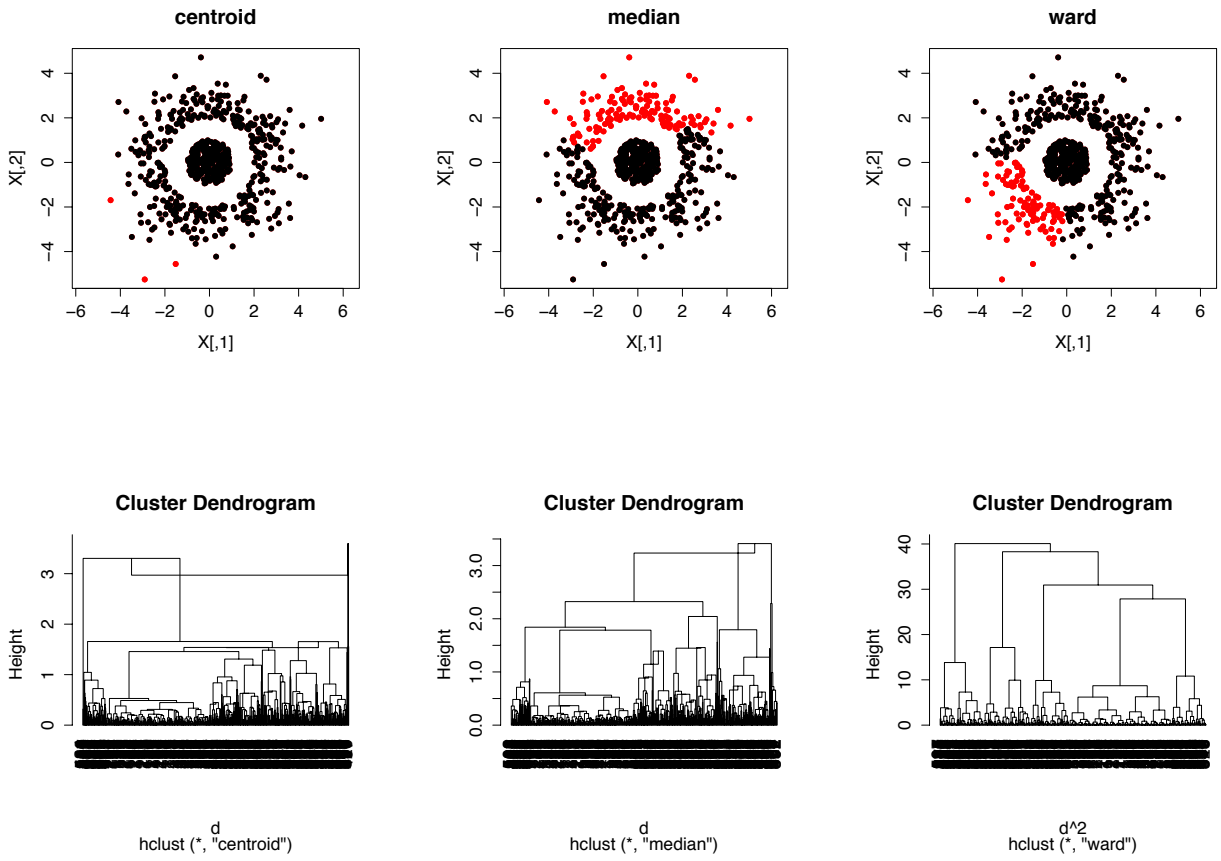
- **The aggregation of a point in a group at a given step cannot be revised, even if the point is misplaced in that group**
- Computationally demanding for large datasets since keeps track of a square matrix of order $n$ (number of individuals): time and space complexity of most algorithms are not better than $O(n^2 \log(n))$
- Dendrogram difficult to visualize and interpret for large datasets
- Most HAC algorithms are greedy and produce suboptimal solutions

Average and Ward (specially when groups have similar sizes) are often considered among the best overall HAC methods

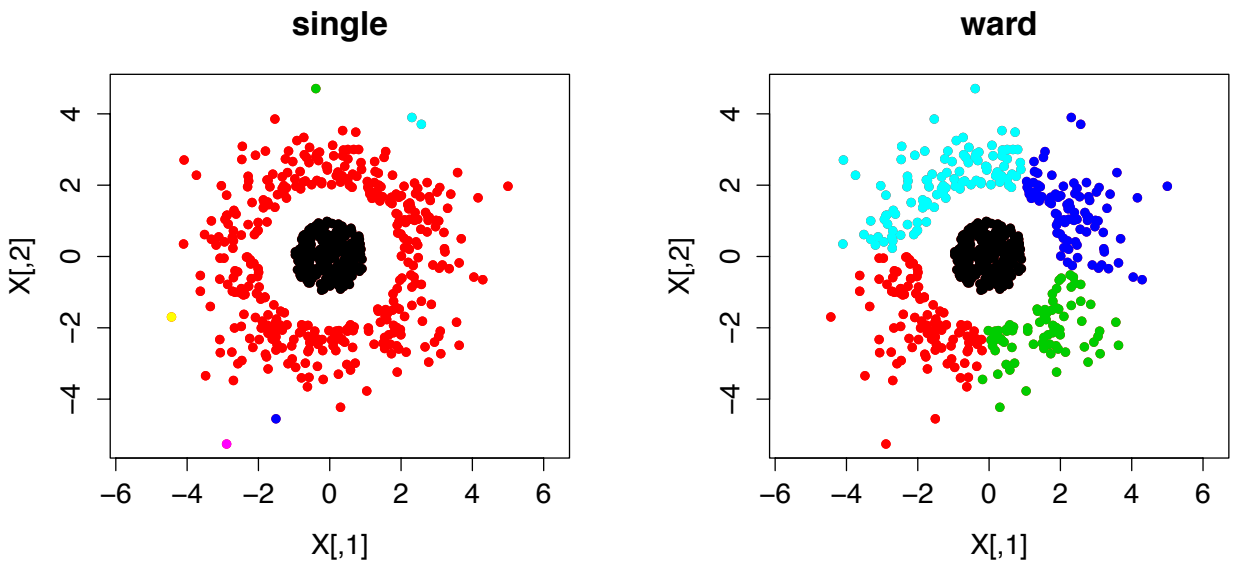
# Assessment of an example with 2 clusters



# Examples (cont.) where inversions are clearly visible



# Same example with 7 clusters



# A short remark on divisive hierarchical methods

- It starts from a cluster consisting of all objects and successively splits every cluster until all clusters have only one element (singletons).
- In each step the splits can be done according to the value of a unique variable **monothetic** (all objects of the same cluster must agree w.r.t. the that variable) or w.r.t. several variables **polythetic**
- Divisive clustering algorithms can be computationally demanding. In R can be performed using the function `diana` of cluster package

# Example

R

```
require(datasets)
require(cluster)
data(iris)
head(iris)
dist.iris<-dist(iris[-5])
iris.diana<-diana(d)
pltree(iris.diana)
```

# Nonhierarchical clustering

To find a single partition into $K$ clusters of a set of $N$ objects in a $p$ dimensional space Two types of criteria are commonly found:

- **Global criterion** such as to represent each cluster by a *type-object* (e.g., centroid, medoid) and to assign each object to the nearest *type-object*, optimizing some global criterion of internal homogeneity and external heterogeneity, such as, minimizing the within cluster inertia

  Usually requires a prior estimate of the number of clusters

  Examples: K-means and K-medoids (PAM) algorithms

- **Local criterion** such as to seek for higher density regions in data. May require to set some parameters

  Example: DBSCAN

# K-means

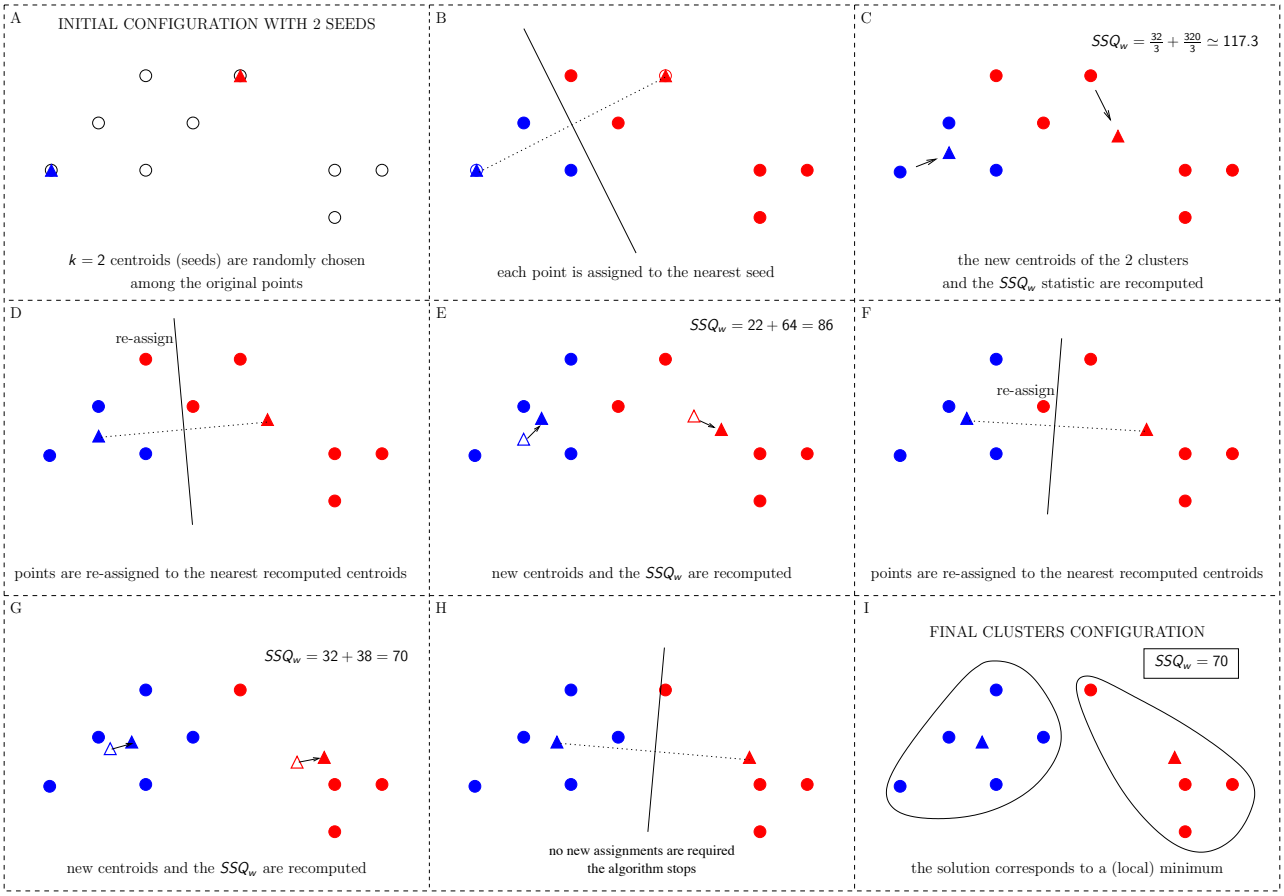
Shares the same global criterion with Ward's method: to minimize the total within-cluster sum of squares ($SSQ_w$) of a set of points partitioned into $K$ clusters in a $d$-dimensional space

**Algorithm (Lloyd)**

- *Starts with $K$ randomly chosen initial* **seeds** *representing initial candidates to centroids;*
- *Assigns each object to the nearest centroid*
- *Recomputes the centroids of the $K$ groups and use them as new seeds*
- *Repeat through steps 2-4 until the algorithm converges, i.e., until no new assigns of points to clusters occur*

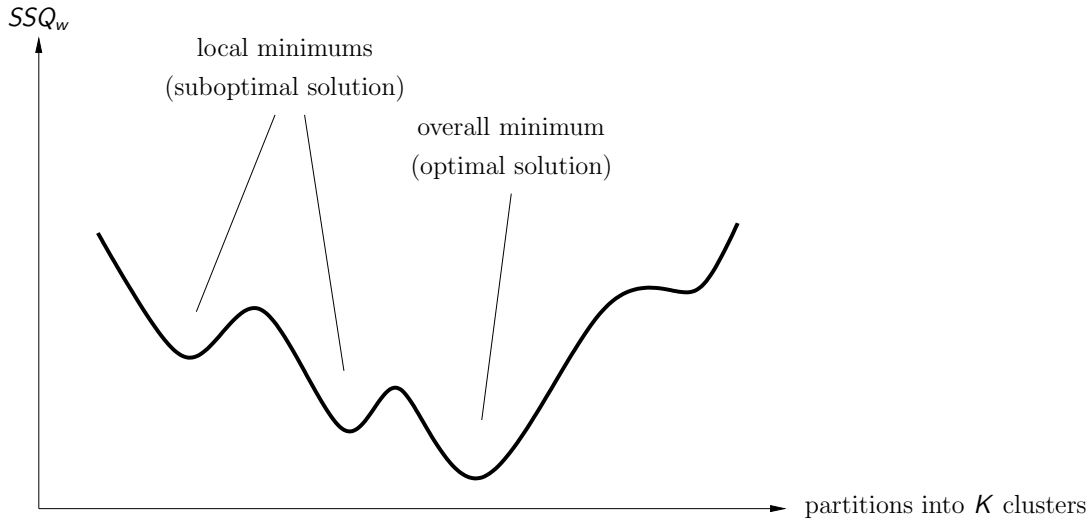
I can be proved that the cost $SSQ_w$ monotonically decreases during the course of the algorithm converging to a (possibly local) optimum

# K-means algorithm



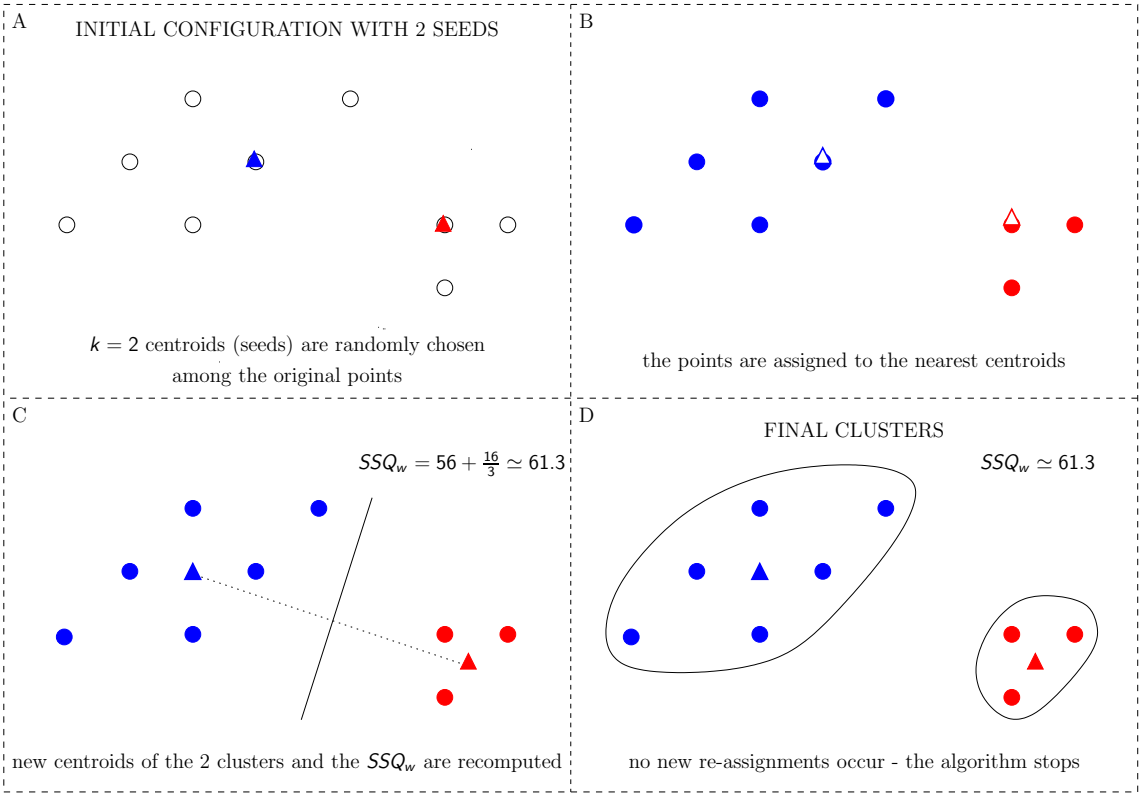
# K-means: local minimum problem

The clustering solution can be highly depend on the choice of the initial position of the centroids (seeds), and may converge to a **local** minimum



# Example

The solution found by the $K$-means algorithm in the previous example is not a global minimum. Actually, with new seeds the algorithm can converge to a solution that improves (i.e., lowers) the $SSQ_w$ statistic



## Possible strategies to find the overall minimum?

- To repeat the algorithm several times with randomized configurations of $K$ seed points and keep the configuration giving the smallest $SSQ_w$ value of the within-cluster inertia
- To provide initial configuration of the $K$ seed points close to the final solution relying on some real hypothesis
- To provide an initial configuration of seed points issued from some hierarchical aggregation method (e.g., Ward, average), for instance, their clusters centroids

# K-means in the plane and the Voronoi diagram

Given a set of $N$ points in the plane,

$$\{c_1, \ldots, c_N\}$$

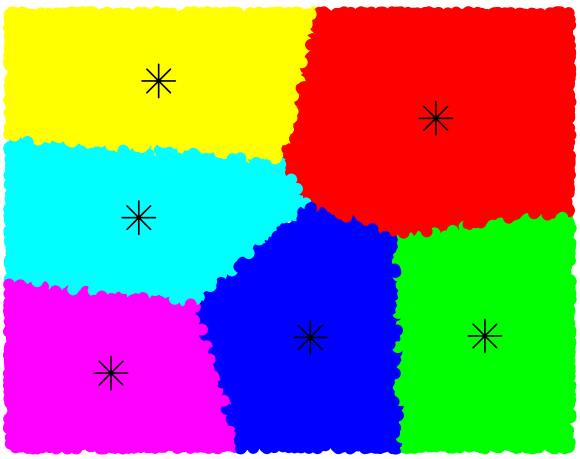
the Voronoi diagram is defined as the partition of the plane into $K$ convex regions, called **Voronoi cells**,

$$R_1, \ldots, R_K$$

such that each cell $R_i$ consists of the points of the that are closest to $c_i$ In each step of $K$-means algorithm the clusters correspond to set of points of $X$ belonging to the Voronoi cells defined be the $K$ centroids $c_1, \ldots, c_K$, which is called Lloyd's algorithm or Voronoi iteration

# The Voronoi partition and its centroids

This partition was originated applying $K$-means algorithm to a uniform distribution of points in the plane



# K-means: summary

- The optimizing function $SSQ_w$ is always monotonic decreasing, i.e., the intra-group inertia decreases in each step, converging to some (possibly local) optimum
- The number of iterations required to converge to an optimum is usually small (usually $\approx 10$ iterations are enough)
- Finding an optimal solution is $NP$-hard. Actually the time complexity is $O(n^{dK+1} \ln d)$, where $K$ denotes the number of clusters, $d$ the dimension and $N$ the number of points)
- It forms linearly separated clusters. In particular it cannot detect arbitrary cluster shapes
- Nearby points can end in distinct classes. Groups can end empty
- Sensitive to noise and outliers
- Requires some geometric notion of center/centroid. In particular cannot be applied to categorical data. Assumes the euclidean distance

# Computing $K$-means with R

The $K$-means clustering can be performed using the R function

kmeans(x, centers, iter.max = 10, nstart = 1, ...)

**x**: numeric matrix of data

**centers**: the number of clusters or a set of initial (distinct) cluster centres. If a number, a random set of (distinct) rows in x is chosen as the initial centres

**nstart**: if centers is a number, how many random sets should be chosen (repeat)
Returns a list with components:

**cluster**: A vector of integers (from 1:k) indicating the cluster to which each point is allocated.

**centers**: A matrix of cluster centers.

**totss**: The total sum of squares $SSQ$

**withinss**: Vector of within-cluster sum of squares, one component per cluster

**tot.withinss**: Total within-cluster sum of squares, i.e., sum(withinss) $SSQ_w$

**betweenss**: The between-cluster sum of squares, i.e. totss-tot.withinss $SSQ_b$

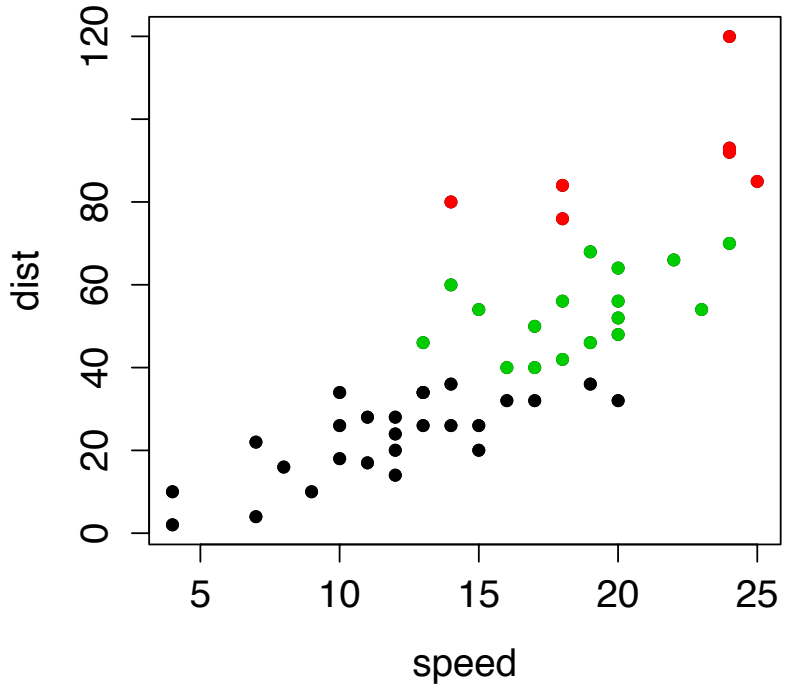
**size**: The number of points in each cluster

# Example

R

```
require(datasets)

data(cars)

?cars

head(cars)

cars.cl<-kmeans(cars, 3, nstart=100)
# 3 centers randomly chosen repeated 100 times

cars.cl

plot(cars,type=‘‘p’’,pch=16,cex=.5)

for(i in 1:50){points(cars[i,1],
cars[i,2],col=cars.cl$cluster[i], pch=16,type=‘‘p’’)}
```

# Clustering result



# Optimal number of clusters in $K$-means?

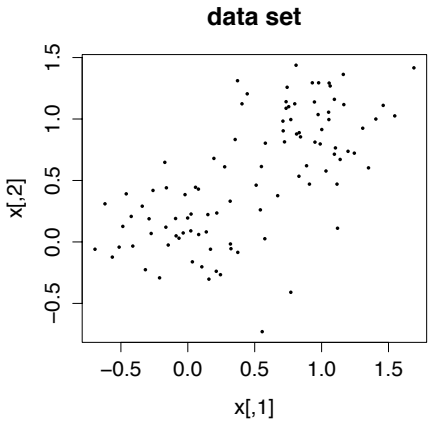
- As in the parcimony principle we seek for a good compromise between having a small number of clusters and minimizing the information loss due to replacing the observations by the clusters
- This is one of the most difficult tasks in the clustering analysis. No definitive answer can usually be given.
- Several indices have been proposed to estimate the number of clusters and to assess the internal cluster quality. Some of the most well-known indices include:
  - The $SSQ_w$ or equivalently the $R^2$ index
  - Calinski-Harabasz index
  - Silhouette coefficient
  - Davies-Boudin
  - Duhn index
  - ...

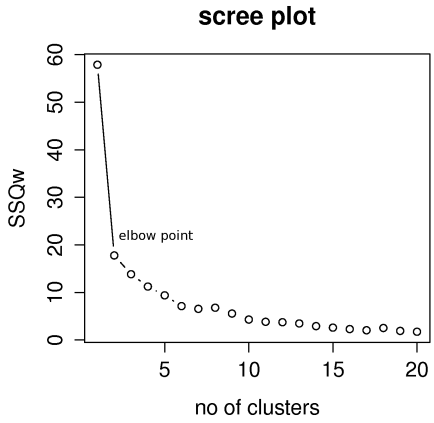
**R**

*Several cluster validity indices can be computed with the R function* `cluster.stats` *of* `fpc` *package or using the* `clustCrit` *or* `NbClust` *packages*

# Scree plot of $SSQ_w$ statistic

- A simple method consists in analysing the variation of $SSQ_w$ against the number of clusters in a scree plot, which is essentially equivalent, by Huygens's theorem, to study the variation of the percentage of the total inertia that is retained by the clusters, called the $R^2 = \frac{SSQ_b}{SSQ_t}$ *determination coefficient* (by analogy to the linear regression theory)
- This statistic is usually monotonically decreasing as the number of cluster increases. An elbow point in this plot indicating a high decrease in the $SSQ_w$ statistic, while further increasing the number of clusters only marginally improves (i.e., lowers) the statistics may provide a good estimate for the number of clusters



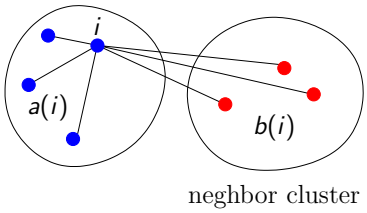



# Calinski-Harabaz index

- The **Calinski-Harabaz index** of a set of $N$ observations partitioned into $K$ clusters is defined as
$$CH(K) = \frac{N-K}{K-1} \cdot \frac{SSQ_b}{SSQ_w}$$
- To maximize $CH(K)$ is equivalent to maximize $SSQ_b$(i.e., to maximize cluster separation) and to minimize $SSQ_w$ (i.e., to maximize cluster cohesion). The optimal number of clusters is estimated as the number yielding the **largest** value for $CH(K)$
- Several studies suggest that Calinski-Harabaz index is one of the internal validation indices yielding the best results
- Corresponds to $F$-value of the *one-way ANOVA* with $K$ factors and is also known as the **variance ratio criterion** (VRC), more precisely, to the proportion if *explained variance* $\frac{SSQ_b}{K-1}$ over the *unexplained variance* $\frac{SSQ_w}{N-K}$
- Can be computed using the R function `calinhara` of the package `fpc`

# Silhouette coefficient

- For each observation $i$ compute average dissimilarity $a(i)$ between $i$ and the remaining points in its cluster
- For each one of the other clusters compute the average dissimilarity from $i$ to the points of that cluster and take the minimum $b(i)$ of these average dissimilarities
- The cluster for which the minimum $b(i)$ is attained, i.e., the cluster with lowest average dissimilarity w.r.t to observation $i$, is called the **neighbor cluster** of $i$



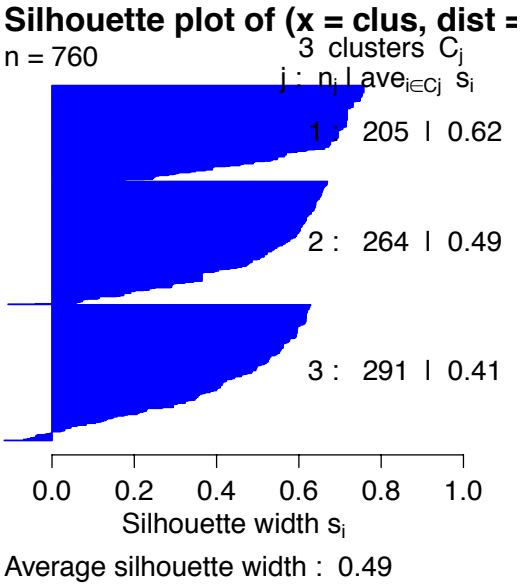

The *silhouette coefficient* of observation $i$ is defined as

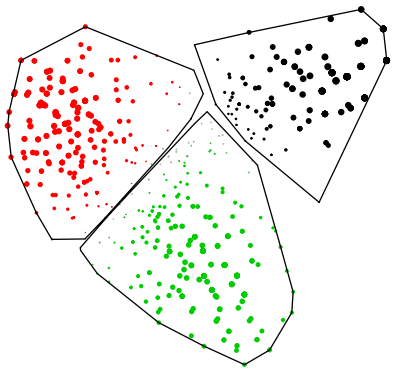
$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

## Interpretation of silhouette coefficients

- The denominator $\max\{a(i), b(i)\}$ is a normalization term allowing that the index vary in the range $[-1, 1]$
- Small values of $a(i)$ along with large values of $b(i)$ yield a silhouette coefficient close to one
- Likewise, large values of $a(i)$ along with small values of $b(i)$ yield a silhouette coefficient close to minus one
- Observations with silhouette coefficients **close to one are very well classified**
- Observations with silhouette coefficients **close to zero probably lie between clusters**
- Observations with **negative silhouette coefficientes are probably misplaced in their clusters**

# Silhouette plot



In the figure on the right the dot sizes are proportional to their silhouette coefficients. Larger dots lie in core regions of the clusters whereas smaller dots lie in border regions or between clusters

# Average silhouette width - an internal validity criterion

The **average silhouette width** (ASW) is defined as the average of the silhouette coefficients for all observations

- It assess both **cluster cohesion** and **cluster separation**
- It increases with a strong cluster separation (higher $b(i)$ values) and cluster tightness (small values of $a(i)$)

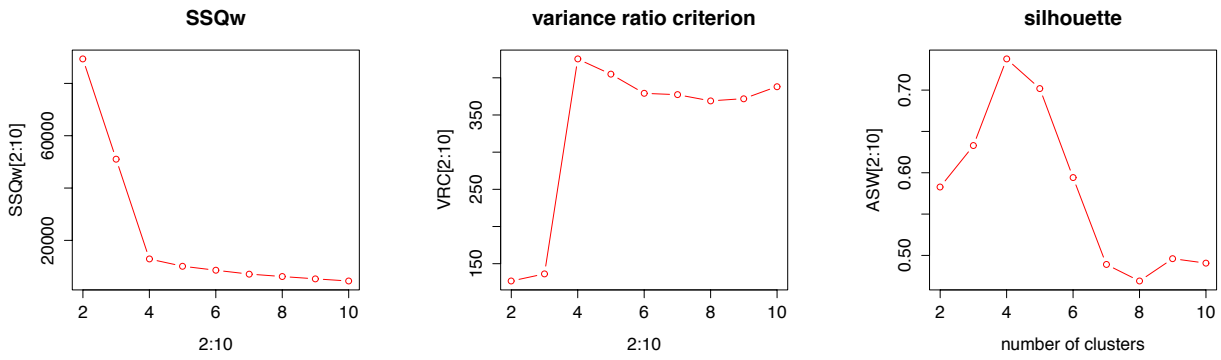
**Range of ASW**

- between 0.71 and 1.0: a **strong structure** has been found
- between 0.5 and 0.7: a **reasonabble strucuture** has been found
- between 0.26 and 0.5: the **structure is weak** and can be artificial
- below 0.25: **no substantial structure** has been found

The optimal number of clusters can be estimated maximizing the ASW

# Number of clusters?

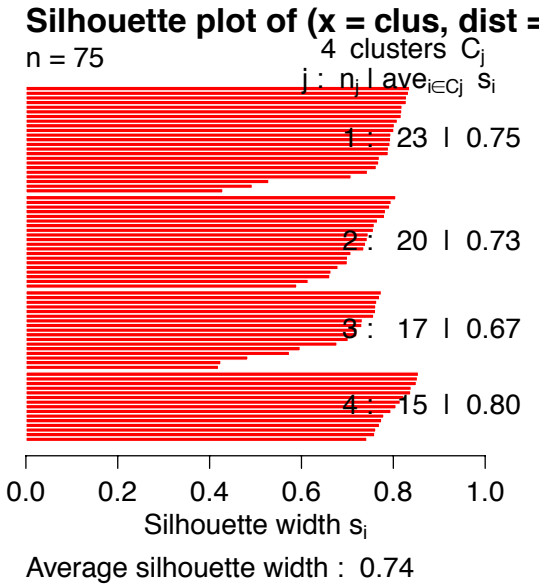
Applying the criteria $SSQ_W$ statistic, VRC and ASW to the Ruspini data, a popular dataset in clustering analysis, all criteria agree on 4 clusters



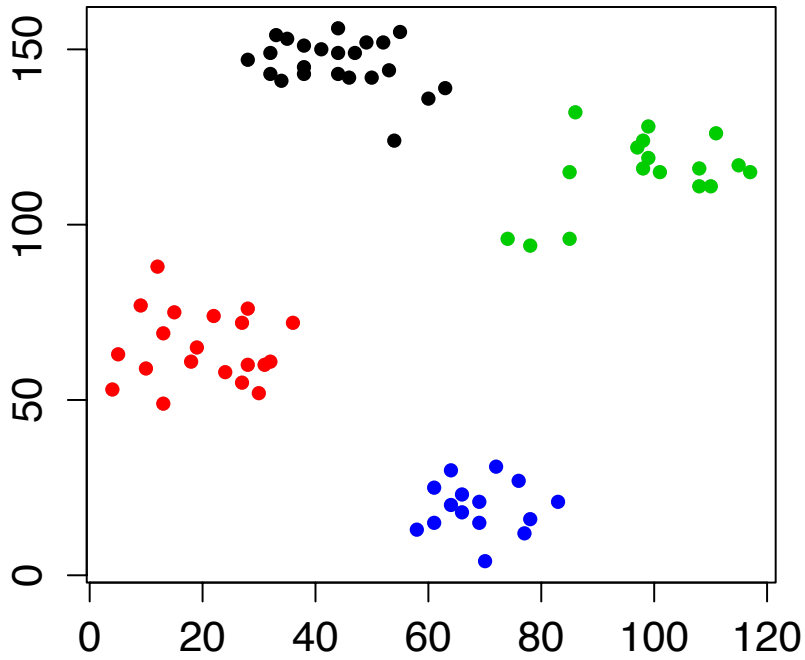

*A monotone increase in the VCR criterion over the number of clusters k indicates that no group structure is present. A monotone decrease suggests hierarchical relationships*

# (Internal) cluster validity

The average of the silhouette widths of the previous example is close to .75 suggesting that a strong clustering structure was found in Ruspini data. Since all silhouette coefficients are above .4 no points are misplaced in their clusters



# Ruspini plot into 4 clusters using the K-means algorithm



# K-medoids clustering

- The **optimization problem** is to find $K$ points in the dataset, called **medoids**, such that the sum of the distances of all non-medoid points to their nearest medoid points is minimal
- A **medoid** also called **exemplar** or **centrotype** is an element of the set whose average distance to every other element in the set is minimal, i.e., the most centrally located point in the set
- The algorithm uses medoids to represent clusters instead of centroids, since are less sensitive to outliers (why?)
- Can work with any notion of distance/dissimilarity
- The most well known algorithm for computing the $K$-medoids is the *greedy* algorithm PAM (Partitioning Around Medoids). May fail to converge to the overall optimum
- For large sets, there are the more efficient algorithm CLARA (Clustering LARge Applications), which applies PAM to samples, and CLARANS (Randomized CLARA)
- To handle categorical data there is a variant called the *K-mode algorithm*

# Partition around medoids (PAM) algorithm

## Algorithm

- *Start with a set $M$ of $K$ initial medoids randomly chosen, $m_k$, $k = 1, \ldots, K$;*
- *Set $O = X \setminus M$ containing the non-medoid points, $o_j$, $j = 1, \ldots, \ell$;*
- *For every pair $(k, j)$ compute the* total swapping cost *of replacing the medoid $m_k$ by the non-medoid $o_j$,*

$$T_{k,j} = \sum_{i \neq j} T_{k,j,i}$$

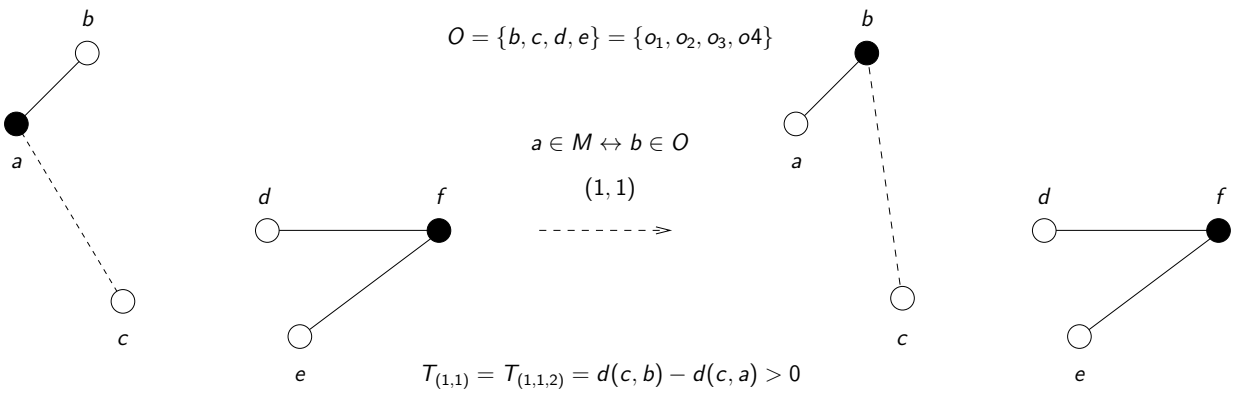
*where $T_{k,j,i}$ is the swapping cost accounted for object $o_i \neq o_j$, and selects the pair $(k, j)$ that minimizes $T_{k,j}$*

- *If $T_{k,j} < 0$ perform the swap, update the sets $M$ and $O$ and return to step 3;*
- *If $T_{k,j} \geq 0$, the objective cannot be improved and the algorithm stops*

# Example of a swapping step

$M = \{a, f\} = \{m_1, m_2\}$

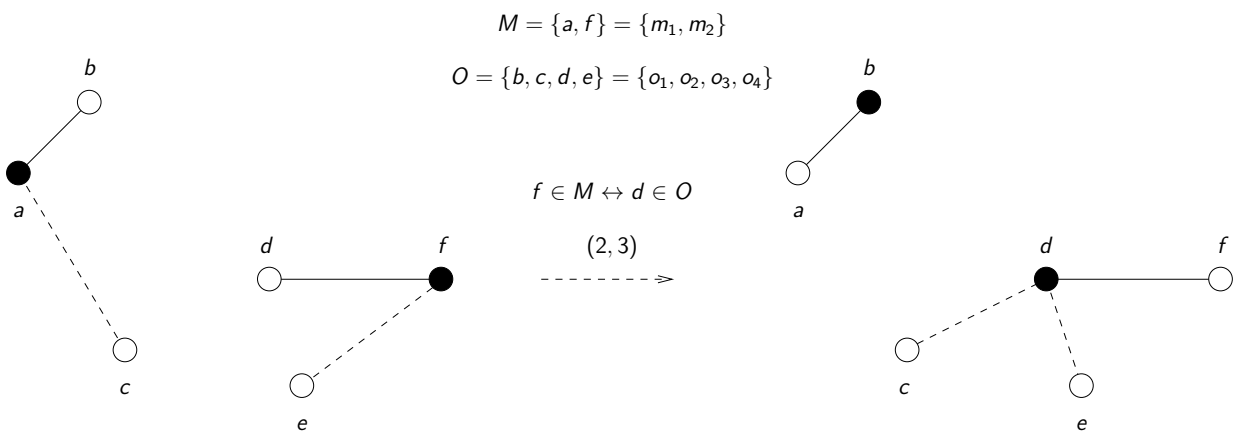
$O = \{b, c, d, e\} = \{o_1, o_2, o_3, o4\}$




$T_{(1,1)} = T_{(1,1,2)} = d(c, b) - d(c, a) > 0$

**Total swapping cost is positive - does not improve the objective function (sum of the edges lengths)**

# Example of a swapping step (cont.)




$$T_{(d,f)} = T_{(d,f,c)} + T_{(d,f,e)} = d(c,d) - d(c,a) + d(e,b) - d(e,f) < 0$$

**Total swapping cost is negative - it improves the objective function**

It can be seen that no other swap can improve this result. The sets $M$ and $O$ are then updated to $M = \{b, d\}$ and $O = \{a, c, e, f\}$ Continuing to perform the swaps, no further improvement occur. The algorithm stops yielding the clusters

$$\mathcal{C}_1 = \{a, b\}, \qquad \mathcal{C}_2 = \{c, d, e, f\}$$

# Computing $K$-medoids with R

The partition around medoids clustering algorithm can be performed using the R function of the `cluster` package
(it accepts metrics distinct from the euclidean metric)

pam(x, k, diss = inherits(x, "dist"), metric = "euclidean", ...)

**x**: data matrix or dataframe or a dissimilarity matrix or object

**k**: is the number of clusters

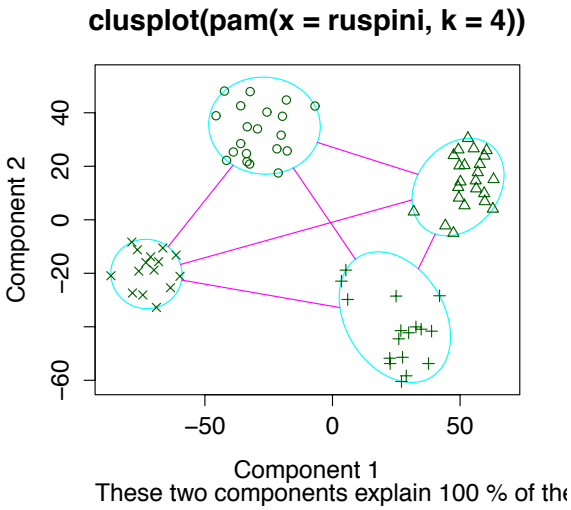
...

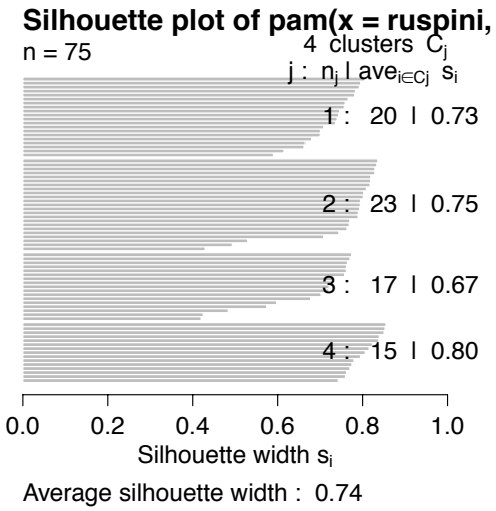
Returns an object of the class *pam* consisting of a list with several components

R

```
require(datasets)
data(ruspini)
clus.PAM<-pam(ruspini,4) # perform the clustering with 4 clusters
clusplot(clus.PAM) # displays the clusters and the silhouette plot
cls<-clus.PAM$cluster # vector of classes for all elements
mdd<-clus.PAM$medoids # vector of coordinates of the medoids of the
final 4 clusters
```

# Ruspini data 4-partition using PAM

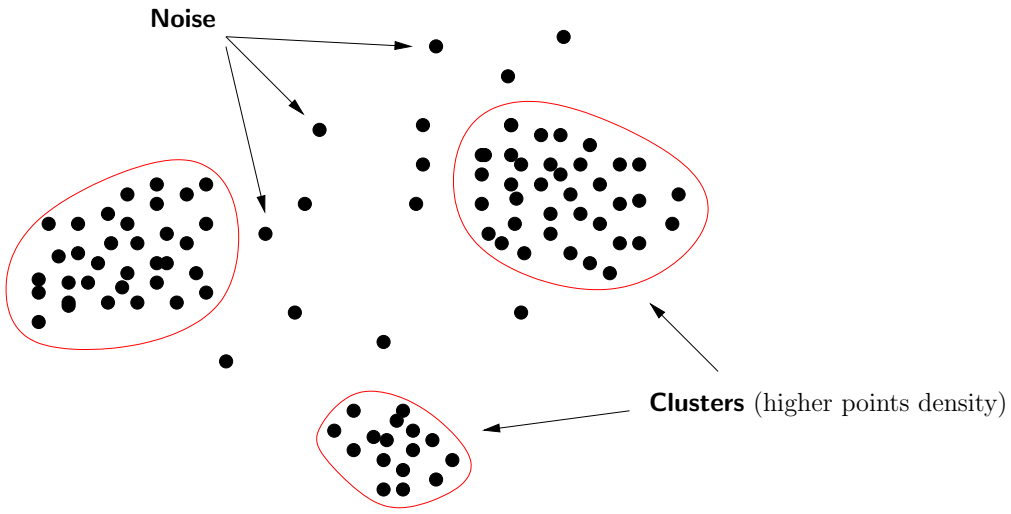



The R `clustplot` function displays the projection of the centered variables onto the principal factorial plane of the PCA, allowing to visualize high dimensional data

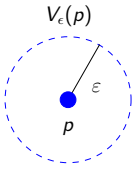
# Density-based clustering: DBSCAN

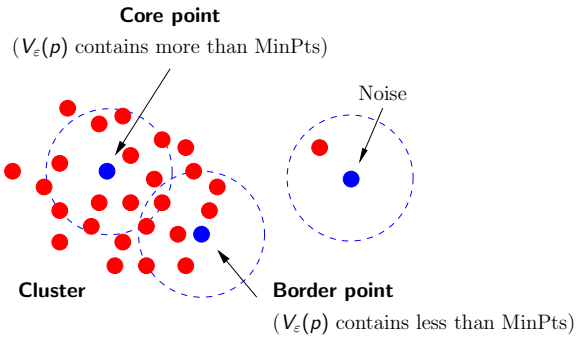
The DBSCAN algorithm implements the very natural idea that clusters are regions with high density of points separated by regions with low density of points (noise)



# How to define a region with a minimum density of points?

- Density is the number of points per area unit. A naïve idea could be to define a cluster $\mathcal{C}$ as the region where each point $p$ has an neighborhood of radius $\varepsilon$, $V_\varepsilon(p)$ with at least a pre-defined minimum number of points $\mathrm{MinPts}$ of $\mathcal{C}$
- The definition works for **core points** but may fail for **border points**, where the number of points in the $\varepsilon$-neighborhood can be considerable lower. A border point in a cluster should however possess at least one core point in its $\varepsilon$-neighborhood
- Points in low density regions with no core points in their $\varepsilon$-vicinity are considered **noise** w.r.t. $(\varepsilon, \mathrm{MinPts})$

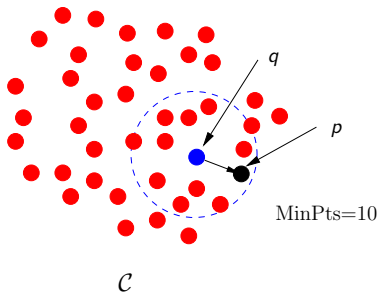


# Directly density reachable relation

We say that a point $p$ is **directly density reachable** from a point $q$ w.r.t. $(\varepsilon, \mathrm{MinPts})$ and we denote by $p \leftarrow q$, if

- $p \in V_\epsilon(q)$
- $|V_\epsilon(q)| \geq \mathrm{MinPts}$

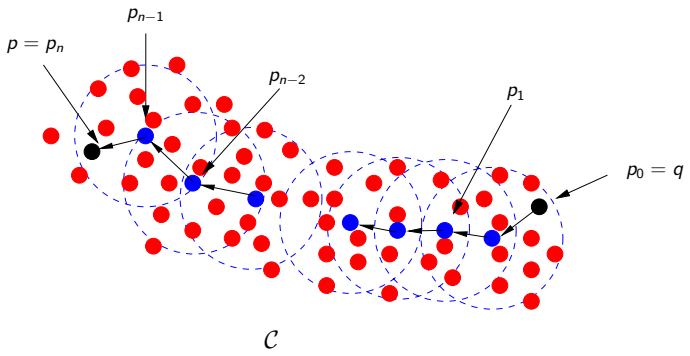



*The relation directly density reachable is not symmetric (in general). Why? Give an example.*

# Density reachable relation

We say that a point $p$ is **density reachable** from a point $q$ w.r.t. $(\varepsilon, \mathrm{MinPts})$ if there is a sequence $q = p_0, p_1, \ldots, p_n = p$ s.t. each $p_{i+1}$ is directly reachable from $p_i$, i.e.,

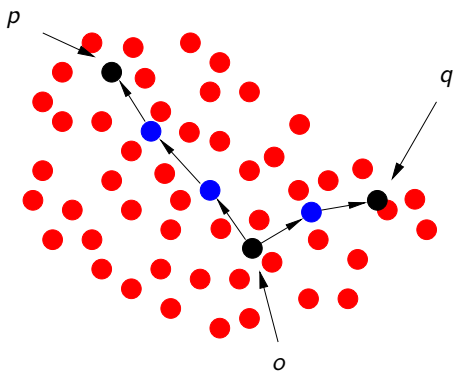
$$p = p_n \leftarrow p_{n-1} \leftarrow p_{n-2} \leftarrow \cdots \leftarrow p_1 \leftarrow p_0 = q$$




*This definition extends the previous one. Moreover, it is not symmetric and may fail if both points p and q lie at the boundary a cluster*

# Density connected cluster

We say that points $p$ and $q$ are **density connected** w.r.t. $(\varepsilon, \mathrm{MinPts})$ if there is a point $o$ such that $p$ and $q$ are both density reachable from $o$




$\mathcal{C}$

# Clusters and noise

**(cluster)**

*A* **cluster** *w.r.t.* $(\varepsilon, \mathrm{MinPts})$ *is a region* $\mathcal{C}$ *of pairwise density connect points that cannot be enlarged with new points (maximal)*

*It can be proved that a cluster* $\mathcal{C}$ *w.r.t.* $(\varepsilon, \mathrm{MinPts})$ *consists exactly of the set of points that can be density-reachable from any of its core points, i.e., from any of its points* $q \in \mathcal{C}$ *such that* $|V_\varepsilon(q)| \geq \mathrm{MinPts}$*. Each one of this core points can be considered as a seed defining the cluster. Points of the cluster where* $|V_\varepsilon(q)| < \mathrm{MinPts}$ *are called* **border points**

**(noise)**

*Let* $\mathcal{C}_1, \ldots, \mathcal{C}_k$ *be the clusters of a set* $X$ *w.r.t.* $(\varepsilon, \mathrm{MinPts})$*. Points not belonging to any cluster are called* **noise** *or* **outliers**, *i.e.,* **noise points** *belong to the set*

$$X \setminus (\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_k)$$

# The algorithm

**Algorithm (DBSCAN)**

*Assume* $(\varepsilon, \mathrm{MinPts})$ *given*

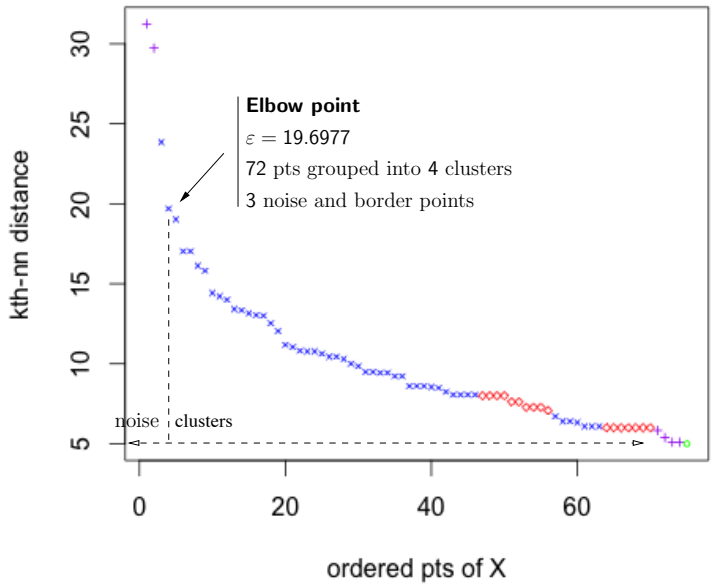
- *Consider a point* $p$ *in* $X$*;*
- *If* $p$ *has not been visited and* $p$ *is a core point, i.e.,* $|V_\varepsilon(p)| \geq \mathrm{MinPts}$*, determine the cluster formed by all points density reachable from* $p$*;*
- *Return to step 1 until all points of* $X$ *have been visited;*
- *Points that are not density reachable from any core point are considered noise points*

The R function `dbscan` of package `fpc` implements the DBSCAN algorithm

# A simple estrategy to estimate $\varepsilon$ and $\mathrm{MinPts}$

- Choose $\mathrm{MinPts} \geq \dim X + 1$. The default for the R `dbscan` function is $\mathrm{MinPts} = 5$. For two dimensional databases $\mathrm{MinPts} = 4$ is usually appropriated.
- For each element $x \in X$ determine the $k$-th nearest neighbor $y$ of $x$, with $k = \mathrm{MinPts}$. Let $\varepsilon_x = d(x, y)$ be the distance from $x$ to $y$. Then the $\varepsilon_x$-neighborhood of $x$ will have at least $k = \mathrm{MinPts}$ elemens, with equality for most $x$;
- Plot the elements by decreasing values of $\varepsilon_x$ called *sorted $k$-th nearest neighbor distance function*. This graph gives an idea of the distribution of densities on the dataset. If the dataset has clusters with similar densities the graph should exhibit a steady decrease for cluster points (specially core points), while for noise points, having their $k$-th nearest neighbor considerable farther may produce an abrupt change.
  An elbow point in this plot usually provides a good candidate for the parameter $\varepsilon$. Points on the left of this elbow point will (usually) correspond to mostly to noise points and border points;
- If several points are candidates, smaller choices of the $\varepsilon$ parameter yield more tighter clusters although with more unclassified points (i.e. noise points)

# The sorted $k$-th distance function for the Ruspini example

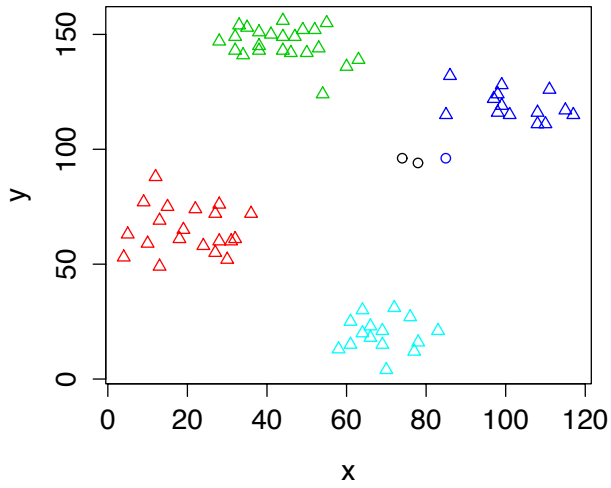



$\varepsilon$ values in the $y$-axis corresponding to blue, red, purple and green points yielding 4, 5, 3 and 1 clusters respectively. Try values for the $\varepsilon$ parameter in each one these cases and interpret the results

# The partition obtained with DBSCAN algorithm

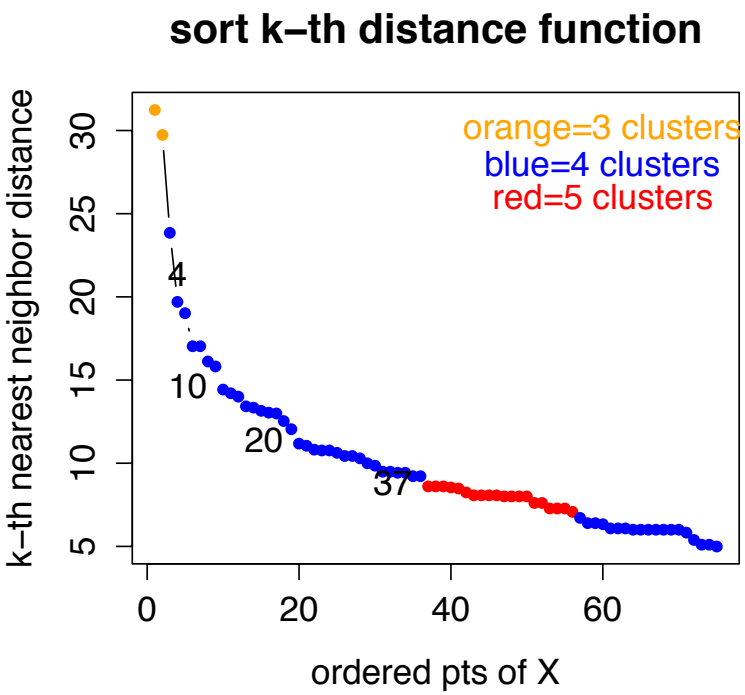
`dbs.ruspini<-dbscan(ruspini, MinPts=5,` $\varepsilon = 19.69$`)`

`plot(dbs.ruspini,ruspini)`



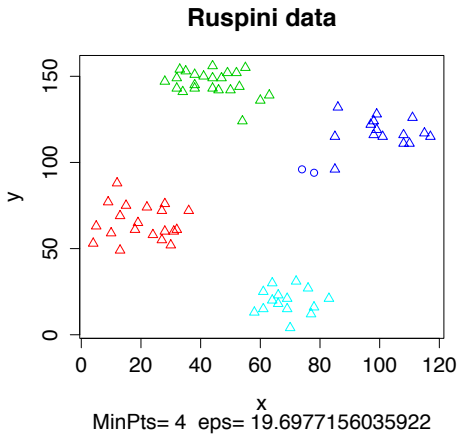

(core points are displayed as colored triangles, border points as colored circles and noise points as black circles)

# Ruspini example revisited with $\mathrm{MinPts}$=4 and 4 $\varepsilon$ values

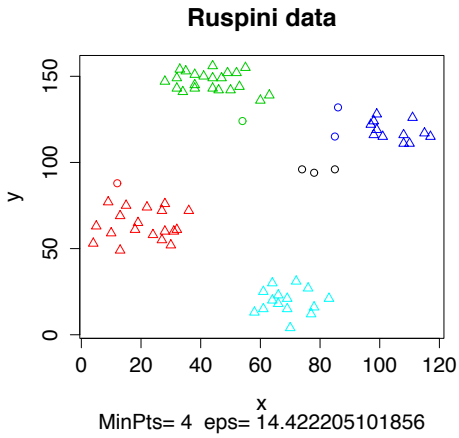



The points marked with labels, 4, 10, 20 and 37 correspond to $\varepsilon = 19.69$, $\varepsilon = 14.42$ (elbow point), $\varepsilon = 11.18$ and $\varepsilon = 8.60$

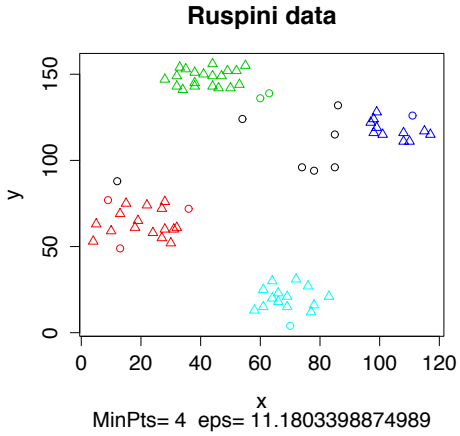
# Clustering outputs corresponding to the 4 $\varepsilon$ values



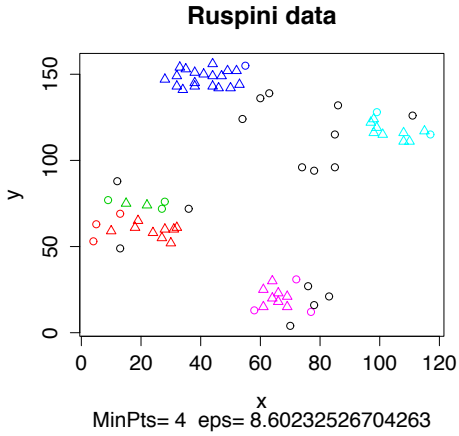

MinPts= 4 eps= 19.6977156035922




MinPts= 4 eps= 14.422205101856




MinPts= 4 eps= 11.1803398874989




MinPts= 4 eps= 8.60232526704263

# Summary of density-based clustering

- Discover clusters with arbitrary shapes
- Good efficiency in large datasets ($o(n \log n)$ time complexity)
- Handles noise points and is robust to outliers
- The number of clusters is not required *a priori*

- The optimal parameters $\varepsilon$ and $\mathrm{MinPts}$ have to be estimated
- Can be difficult or impossible to find a good combination of parameters $(\varepsilon, \mathrm{MinPts})$ when the dataset contains regions of very distinct density

## Non-hierarchical clustering - summary

**Pros**

- **Can reallocate an individual that was misplaced in its cluster**
- Computationally efficient ($K$-means)
- Can improve the objective function obtained with some hierarchical methods (e.g., $K$-means vs Ward)

**Cons**

- The number of clusters (or some other parameters) has to be estimated *a priori*
- No taxonomic type of relationship between clusters is obtained and no dendrogram is produced
- Some methods work only with "geometric" data and may require the euclidean distance

# 4. COMPARING PARTITIONS

# Motivation

- Several clustering analyses of the same data can be done using distinct meaningful combinations of clustering methods and resemblance notions;
- Clustering analyses having a high degree of agreement may suggest that the common patterns produced by these methods is robust;
- If the clustering structure is known *a priori* and it is important to assess how well the clustering method was able to reproduce this structure;
- It is very difficult (if not impossible or meaningless) to match each cluster of a partition with the correct cluster of the other partition
- The usual way is to compute the number of pairs of individuals that both clustering methods agree to assign in the same/distinct class

# Rand index

- Assume that $N$ individuals are classified by two distinct clustering methods. The total number of pairs of individuals is $\binom{N}{2}$. Denote by:

  $A$: number of pairs classified in the same class in both partitions

  $B$: number of pairs classified in the same [distinct] class for the first [second] partition

  $C$: number of pairs classified in the distinct [same] class for the first [second] partition

  $D$: number of pairs classified in distinct classes in both partitions

- The Rand index is a simple concordance index used as external validition index and is defined as

$$RI = \frac{A + D}{\binom{N}{2}} = \frac{A + D}{A + B + C + D}$$

  where $A+D$ is number of agreements for both partitions

- It ranges from 0 *(total disagreement)* to 1 *(total agreement)*

# Rand index

- To each partition of a set of $N$ individuals, $x_1, \ldots, x_N$ we associate a binary vector of length $\binom{N}{2}$, where the component corresponding to pair $(i, j)$ is equal 1 if $x_i$ and $x_j$ are assigned in the same class and 0 otherwise
- The Rand index of two partitions is nothing more than the simple matching index between the binary vectors associated to these partitions
- *Note that the number of groups in each partition can be distinct*

# Rand index: example

$X = \{a, b, c, d, e, f, g\}$

Partition 1: $a\ b\ e \mid c \mid d\ f$

| | $a$ | $b$ | $c$ | $d$ | $e$ |
|---|---|---|---|---|---|
| $b$ | 1 | · | · | · | · |
| $c$ | 0 | 0 | · | · | · |
| $d$ | 0 | 0 | 0 | · | · |
| $e$ | 1 | 1 | 0 | 0 | · |
| $f$ | 0 | 0 | 0 | 1 | 0 |

Partition 2: $a\ c \mid b\ d \mid e\ f$

| | $a$ | $b$ | $c$ | $d$ | $e$ |
|---|---|---|---|---|---|
| $b$ | 0 | · | · | · | · |
| $c$ | 1 | 0 | · | · | · |
| $d$ | 0 | 1 | 0 | · | · |
| $e$ | 0 | 0 | 0 | 0 | · |
| $f$ | 0 | 0 | 0 | 0 | 1 |

Contigency table between partition 1 and partition 2:

| | 1 | 0 | |
|---|---|---|---|
| 1 | 0 | 4 | 4 |
| 0 | 3 | 8 | 11 |
| | 3 | 12 | 15 |

$$\Rightarrow \qquad RI = \frac{0+8}{15}$$

# Correction for chance: adjusted Rand index

The Rand index present some issues:

- The expected value of Rand index between random partitions with the same number of elements in each class is not constant (e.g. equal to 0)
- It is highly depend on the number of clusters. For instance, even if the clusterings are independent, the index will converge to 1 as the number of clusters increase

To correct these issues the so-called **adjusted Rand index** as proposed

$$ARI = \frac{RI - Expected\ RI}{Max - Expected\ RI} = \frac{\binom{N}{2}(A+D) - U}{\binom{N}{2}^2 - U}$$

where $U = (A+B)(A+C) + (C+D)(B+D)$, assuming the generalized hypergeometric distribution as null hypothesis (keeping the clusters sizes) Gives value 0 for independent clusterings and 1 for identical clusterings. May give negative values indicating quite low agreement. More difficult to interpret than the more simple Rand index

# Computing the adjusted Rand index in R

To compute the adjusted Rand index of the two partitions in 3 classes,

$$\mathcal{P}_1 : \quad a\ b\ e \mid c \mid d\ f \qquad\qquad \mathcal{P}_1 : \quad a\ c \mid b\ d \mid e\ f,$$

we encoded these partitions as vectors

$$(1,1,2,3,1,3), \qquad (1,2,1,2,3,3),$$

representing the classes of the elements $a, b, c, d, e, f$

R

```
require(mclust)
adjustedRandIndex(c(1,1,2,3,1,3),c(1,2,1,2,3,3))
-0.2962963
# 2 randoms vectors of length 100 consisting
# of elements in 1,...,10
p1<-sample(1:10,100,replace=TRUE)
p2<-sample(1:10,100,replace=TRUE)
adjustedRandIndex(p1,p2) # should be approximately zero!
```