Lecture 1 : Cluster Analysis

Cluster analysis is an exploratory data tool for solving classification problems. Its object is to sort individuals (plants, cells, genes, ...) into groups, or clusters, such that the degree of association is strong between members of the same cluster and weak between members of different clusters. Among a large unstructured data set, it may reveal associations and structure in data which, even not previously evident, may be sensible and usefull once founded. In mathematical terms, cluster a data set consists consists in finding the *best* partition of a data set in the sens of some criterium. But it is important to realize that it is usually impossible to look at all the partitions of the data set and choose the best ones, because the number of partitions is usually much to large : there exist for example more than 10 millions of partitions of 14 individuals in 4 clusters). Thus one can not hope more than finding an as good as possible partition by an iterative method. There exist mainly two kinds of alogorithms, namely *hierarchical analysis* (that produces special hierarchical trees called *dendrograms*) and *K-means clustering* (which can be generalized in *neuronal methods*, that is even more efficient). The K-means algorithm is an example of what is called *unsupervised learning algorithms.*

Distance Matrix : Usually, the n individuals that we want to classify are known through the values of p variables that have been measured on each of them. Thus the data set is simply a data matrix $X = (x_i^j)_{1\leq i\leq n, 1\leq j\leq p}$. But the first thing to do to be able to classify individuals is to choose a distance (or more generally a dissimilarity) between two individuals, namely $d(x_i, x_{i'})$. Than one can build up an $n \times n$ symmetric matrix,

$$\Delta = (\delta_{ii'})_{1\leq i,i'\leq n} = d(x_i, x_{i'})$$

called the *distance matrix.* The choice of the distance is important because the result of the clustering analysis usually depends on it. If the individuals may be represented as elements of a p dimensional euclidien space (that is, if the variables are non categorical) than the most commonly chosen type of distance is the euclidien distance $d(x_i, x_{i'})^2 = \sum_{j=1}^p (x_i^j - x_{i'}^j)^2$. But it can also be the *city-block distance* $d(x_i, x_{i'}) = \sum_{j=1}^p |x_i^j - x_{i'}^j|$ or more generally the *power distance* $d(x_i, x_{i'})^r = \sum_{j=1}^p (x_i^j - x_{i'}^j)^s$, where r and s are user-defined parameters.

Huygens' Formula : Assume that the set of individuals Γ is the union of q clusters Γ_1, Γ_2, $\ldots\Gamma_q$ and denote by $\mathcal{I}(\Gamma)$ and $\mathcal{I}(\Gamma_k)$, for $k = 1, \ldots, q$, the inertia of the set Γ and the inertia of its clusters. Let us call the sum of the inertia of all clusters

$$\mathcal{I}_{within}(\Gamma) = \mathcal{I}(\Gamma_1) + \ldots + \mathcal{I}(\Gamma_q)$$

the *within-clusters inertia* of Γ. Let $\overline{x}_k$ be the centroïd of Γ_k and π_k its weight, wich is equal to the sum of the weights ω_i of the individuals belonging to Γ_k. Then the inertia of the set of the weighted points $(\overline{x}_k, \pi_k)$ is called the *between-clusters inertia* of Γ, denoted by $\mathcal{I}_{between}(\Gamma)$. We have the following result called the *Huygens' Formula :*

Proposition 1 *The total inertia $\mathcal{I}(\Gamma)$ of a cloud of points wich is the union of the clusters Γ_1, Γ_2, $\ldots\Gamma_q$ is the sum of its within-cluster inertia and its between-clusters inertia :*

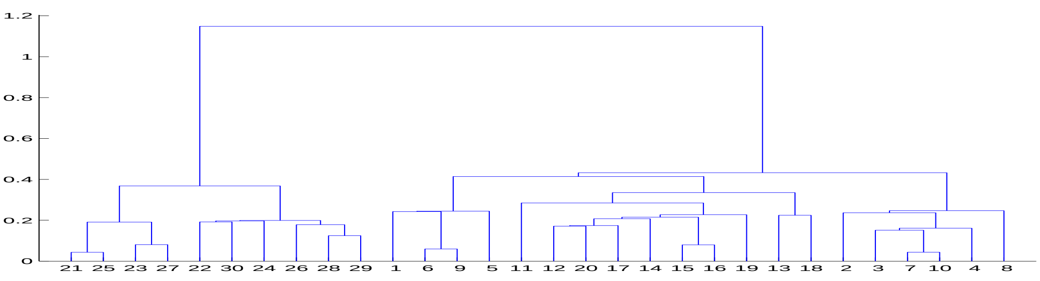
$$\mathcal{I}(\Gamma) = \mathcal{I}(\Gamma_1 \dot{\cup} \Gamma_2 \dot{\cup} \ldots \dot{\cup} \Gamma_q) = \mathcal{I}(\Gamma_1) + \mathcal{I}(\Gamma_2) + \ldots + \mathcal{I}(\Gamma_q) + \mathcal{I}_{between}(\Gamma) = \mathcal{I}_{within}(\Gamma) + \mathcal{I}_{between}(\Gamma).$$

This formula can easily be shown for 1-dim cloud and then generalized to any dimension using Pythagoras' formula.

One consequence of this theorem is that, as the total inertia $\mathcal{I}(\Gamma)$ does not depend on the partition of Γ into clusters, than a partition that maximizes the within-clusters inertia (making the association within members of a cluster as strong as possible) will automatically minimize the between-clusters inertia (making the association between different clusters as low as possible).

Hierarchical Clustering (HC) : One begins with the trivial partition of the set for which any individual is a cluster. At each step of the algoritm the closest two clusters are merged in a single cluster producing a new partition with one less cluster. Therefore a measure of dissimilarity between clusters must be defined to choose which clusters are the closest. There are many possibilities, for exemple, if Γ_k and $\Gamma_{k'}$ are two clusters, $D(\Gamma_k, \Gamma_{k'}) = \text{Min}\,\{d(x_i, x_{i'}), x_i \in \Gamma_k, x_{i'} \in \Gamma_{k'}\}$ is called the *single linkage*, $D(\Gamma_k, \Gamma_{k'}) = \text{Max}\,\{d(x_i, x_{i'}), x_i \in \Gamma_k, x_{i'} \in \Gamma_{k'}\}$ is called the *complete linkage*, but one should preferably use the *Ward linkage* defined by

$$D(\Gamma_k, \Gamma_{k'}) := \frac{\pi_k \pi_{k'}}{\pi_k + \pi_{k'}} d(\overline{x}_k, \overline{x}_{k'}).$$



Indeed it can be shown that this distance measures precisely the loss of between-clusters inertia produced in merging the two clusters in a single one (or the gain of within-clusters inertia produced in merging them). **Dendrogram :** Usually one represents the HC by a binary tree, the root of which represent the initial data set Γ itself, each node represents a cluster of Γ, the terminal nodes represent the individuals. Each non terminal node (parent) has two daughter nodes that represent the two clusters that were merged. The height of each node is proportional to the distance between its two daughters. Thus with the choice of the Ward linkage the height in the dendrogram is simply proportional to the within-clusters inertia of the data set Γ. Notice that at the first step of the algorithm the within-clusters inertia is 0 and the between-clusters inertia is the total inertia. But at the end it is the converse. At each step, the one decreases and the other increases and the algorithm chooses to merge two clusters in order to produce the maximal loss of between-clusters inertia (or to produce the maximal gain of within-clusters inertia). It is up to the user to decide how to cut the dendrogram, that means to decide which level (if any) actually represents the *best* clustering of the initial data set.
K-mean Clustering : For this family of methods, also called *dynamic centers* methods) we have to figure out first how many clusters we want. Let q be this number of clusters (denoted by K originally). The simplest algorithm starts with a random sample of q points (initialisation of the *centers*) belonging to the data set Γ, $\{c_1^0, c_2^0, \ldots, c_q^0\}$. From these q points one defines a first partition of the set $\Gamma = \Gamma_1 \dot{\cup} \Gamma_2 \dot{\cup} \ldots \dot{\cup} \Gamma_q$, just puting in the cluster Γ_k the individuals that are closer to c_k^0 than to the other $c_{k'}^0$. This means that, for all $k = 1, 2, \ldots, q$, the cluster Γ_k is given by

$$\Gamma_k = \left\{x_i | d(x_i, c_k^0) = \text{Min}\left\{d(x_i, c_{k'}^0), k' = 1, 2, \ldots, q\right\}\right\}.$$

Then one replaces each of the center c_k^0 by the centroïd of the cluster Γ_k. The new centers are denoted by $\{c_1^1, c_2^1, \ldots, c_q^1\}$. Then the next step just repeats what as been done in the previous step but beginning with the new centers. Each step produces a new partition of the data set and the important point is to show that the new one corresponds to a within-cluster inertia lower (or equal) to the previous one. Usually the user will stop the algorithm either if two successive steps induce no new modification of the partition or if the within-clusters inertia does no longer decrease noticeably (or may be after a given number of steps).

This algorithm belongs to the family of *iterative descent methods* because it move at each step from one partition of the set of all partitions of Γ to another in such a way that the value of some criterion (here the within-clusters inertia of the set) improve from its previous value. As the set of all partitions of Γ is finite (even if it is huge), this algorim does converge. Unfortunately, it may converge to some local optimum which may be highly suboptimal when compared to the global optimum. For the user, this problem will probably appear when he will realize that the optimum given by the algorithm does depend on the choice of the initial sample of centers $c_1^0, c_2^0, \ldots, c_q^0$.

This is one reason to introduce a modified algorithm called *dynamic K-means method.* As for the simple K-means algorithm, one begins with q centers $c_1^0, c_2^0, \ldots, c_q^0$ chosen as a random sample and then, at each step, only one new point x_i of Γ is chosen, it is assigned to one of the existing centers, say c_k^m if it is done at the m^{th} step, and then one replaces this center c_k^m by the centroïd of the set $\{x_i, c_k\}$ called c_k^{m+1}. We will see in a futur lecture why this modified algorithm is a *statistical learning* method and how it can be generalized to the so called *neural network approach.*