UNSUPERVISED LEARNING

1. INTRODUCTION

This article presents a review of traditional and current methods of classification in the framework of unsupervised learning, in particular cluster analysis and self-organizing neural networks. Both are vector quantization methods aiming at minimizing the distance between an input vector and its representation. The learning is unsupervised as no predefined cluster structure of the input data is assumed. The review of cluster analysis methods covers *hard clustering*, hierarchical and nonhierarchical, whose aim is to assign exact (with membership degree equal to 1) units (objects) to clusters; *fuzzy clustering*, where the membership degree of a unit to a cluster is allowed to stay in the interval [0; 1]; *mixture clustering*, a model-based clustering consisting in fitting a mixture model to data and identifying each cluster with one of its components. All these methods are reviewed in all the variants related to the presence of complex or big data structures or to the presence of outliers.

The *self-organizing maps* are also presented as artificial neural network, the cells (neurons) of which become specifically tuned to various input data patterns or classes of patterns through an unsupervised learning process. The resulting vector quantization process allows clustering of the input data.

References 1,2,3 deeply elaborate the topic.

2. CLUSTER ANALYSIS

In this section, we describe the principal methods and theoretical approaches of cluster analysis.

2.1. Hard Clustering

Given a set of units, the aim of hard cluster analysis is to assign each unit to only one cluster so that units within each cluster are similar to one another with respect to the observed variables, and the units in different clusters are dissimilar. Clustering methods are classified as hierarchical clustering and nonhierarchical clustering (or partitional clustering) methods, based on the properties of the generated clusters (1,4). Hierarchical clustering groups data by means of a sequence of partitions, either starting with one cluster with all units and then splitting it into smaller clusters or starting with each unit forming a separate cluster and then merging similar clusters into larger clusters. The former is known as divisive clustering, and the latter as agglomerative clustering. Both agglomerative and divisive clustering methods organize data into the hierarchical structure on the basis of appropriate proximity measures (i.e., distance measures (see Section "Distance Measures"), dissimilarity measures, similarity indices). In Section "Distance Measures", we focus our attention only on the agglomerative approach. Nonhierarchical clustering (see Section "Nonhierarchical Clustering (Partitioning Clustering)") directly divides data units into some prespecified number of clusters without the hierarchical structure. See, for more details, References 1,4.

Distance Measures.. Let $\mathbf{X} = \{x_{ij} : i = 1, \dots, n;\ j = 1, \dots, J\} = \{\mathbf{x}_i = (x_{i1}, \dots, x_{is}, \dots, x_{iJ})' : i = 1, \dots, n\}$ be a data matrix, where x_{ij} represents the jth quantitative variable observed on the ith unit and $\mathbf{x}_i$ represents the vector of the variables observed for the ith unit. Clustering methods are based on measuring distances among units.

A *distance function* $d : A \times A \to \mathbb{R}$ is a real-valued function on a real vector space A satisfying the following properties (i, i', $i'' = 1, 2, ..., n$):

1) $d(\mathbf{x}_i, \mathbf{x}_i) = 0\ \forall \mathbf{x}_i \in A$
2) $d(\mathbf{x}_i, \mathbf{x}_{i'}) > 0\ \forall \mathbf{x}_i \neq \mathbf{x}_{i'}$
3) $d(\mathbf{x}_i, \mathbf{x}_{i'}) = d(\mathbf{x}_{i'}, \mathbf{x}_i)\ \forall \mathbf{x}_i, \mathbf{x}_{i'} \in A$
4) $d(\mathbf{x}_i, \mathbf{x}_{i''}) \leq d(\mathbf{x}_i, \mathbf{x}_{i'}) + d(\mathbf{x}_{i'}, \mathbf{x}_{i''})\ \forall \mathbf{x}_i, \mathbf{x}_{i'}, \mathbf{x}_{i''} \in A$

The distance class of Minkowski is (1):

$$ {}_r d_{ii'} = \left[\sum_{j=1}^{J} |x_{ij} - x_{i'j}|^r \right]^{\frac{1}{r}} \qquad r \geq 1 $$

where x_{ij} and $x_{i'j}$ represent the jth variables observed, respectively, in the ith and i'th unit ($i, i' = 1, \dots, n$). For $r = 1$, we have the city block distance (Manhattan distance):

$$ {}_1 d_{ii'} = \sum_{j=1}^{J} |x_{ij} - x_{i'j}| $$

and for $r = 2$, we have the Euclidean distance, probably the most commonly used distance measure in cluster analysis:

$$ {}_2 d_{ii'} = \left[\sum_{j=1}^{J} (x_{ij} - x_{i'j})^2 \right]^{\frac{1}{2}}. $$

The Canberra distance is similar to the Manhattan distance. The distinction is that the absolute difference between the observed variables is divided by the sum of the absolute variable values prior to summing:

$$ {}_c d_{ii'} = \sum_{j=1}^{J} \frac{|x_{ij} - x_{i'j}|}{|x_{ij} + x_{i'j}|}. $$

The Mahalanobis distance takes into account the association between pairs of variables as measured by the covariance:

$$ {}_M d_{ii'} = \left[(\mathbf{x}_i - \mathbf{x}_{i'})^T \mathbf{S}^{-1} (\mathbf{x}_i - \mathbf{x}_{i'}) \right]^{\frac{1}{2}} $$

where $\mathbf{S}$ is the covariance matrix among the variables.

See Reference 1 for more details on the distance measures and their use in the cluster analysis.

Hierarchical Clustering. The most widely used among the hierarchical methods are agglomerative methods. They produce a sequence of partitions of the data; the first consisting n single-member "clusters"; the last consisting a single cluster containing all n units, at the end of merge

J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*.
DOI: 10.1002/047134608X.W8379

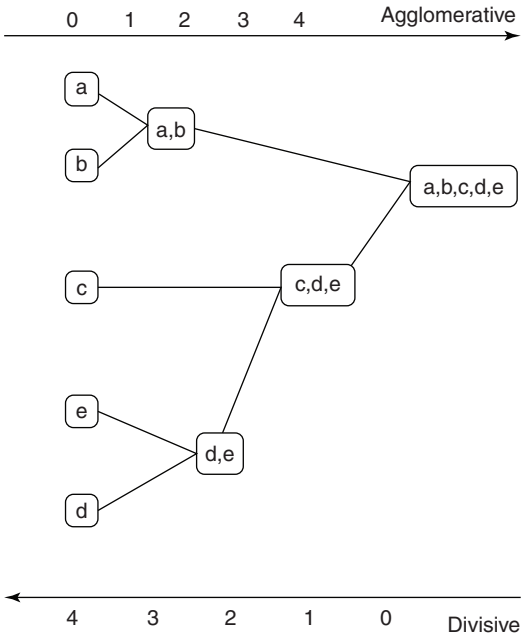

Figure 1. Agglomerative–divisive hierarchical clustering of units a,b,c,d,e (1).

operations forcing all units into the same group (1), as shown in Figure 1.

The general agglomerative clustering can be summarized by the following procedure (4):

1. Start with the partition with n singleton clusters. Calculate the proximity matrix, for example, distance matrix, for the n clusters.
2. Combine clusters C_k and $C_{k'}$ whose distance is minimal to form a new cluster $C_{kk'}$, where $d(C_k, C_{k'}) = \min d(C_p, C_q)$ $1 \leq p,\ q \leq n,\ p \neq q$, where $d(\cdot, \cdot)$ is the distance function.
3. Update the distance matrix by computing the distances between the cluster $C_{kk'}$ and the other clusters.
4. Repeat steps 2 and 3 until only one cluster remains.

The core of the procedure is the definition of the distance function between two clusters at the basis of the formation of a new cluster. There exists a large number of distance function definitions between a cluster C_q and a new cluster $C_{kk'}$ formed by merging two clusters C_k and $C_{k'}$. Some methods for defining distance functions are shortly described in Table 1 and displayed in Figure 2.

Single linkage, complete linkage, and average linkage consider all units of a pair of clusters when calculating their intercluster distance, and they are also called graph methods. The others are called geometric methods because they use geometric centers to represent clusters and determine their distances (4). See Reference 1 for the features and properties of these methods and experimental comparative studies.

The graphical representation of the results of hierarchical clustering is a particular graph called dendrogram. The dendrogram is a tree-structured graph. At the bottom

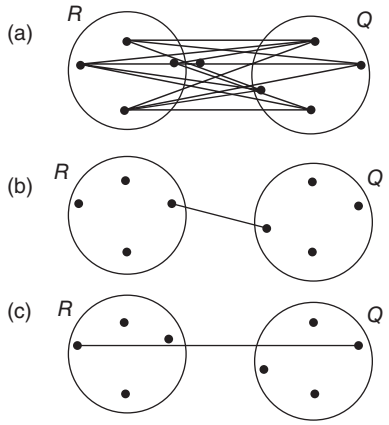

Figure 2. Distance between two clusters R and Q. (a) Average. (b) Single. (c) Complete (1).

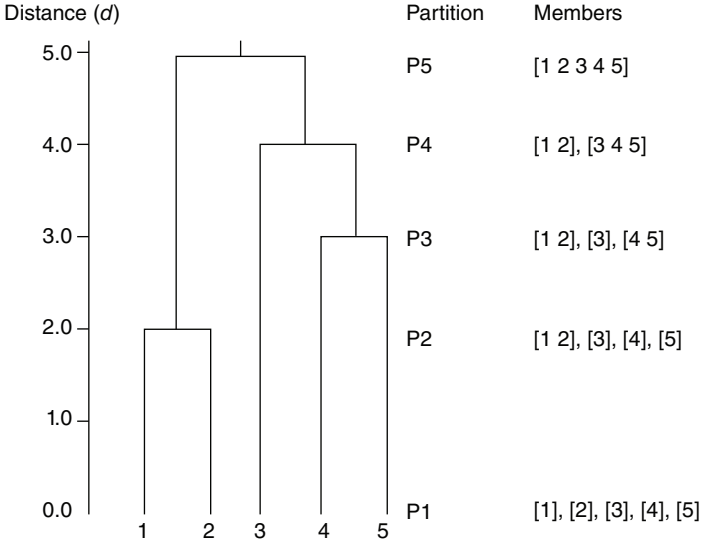

Figure 3. Example of dendrogram in hierarchical clustering of units 1,2,3,4,5 (5).

of the tree all units form a separate cluster; the root node at the top represents the whole data set in a single cluster and each leaf node is regarded as a cluster. The dendrogram visualizes the clustering agglomerative process from bottom to top through horizontal lines taken on the vertical axes at the height of the distance of the linked clusters.

The intermediate nodes thus describe the extent to which the units are proximal to each other. The ultimate clustering results can be obtained by cutting the dendrogram at the desired level of distance (or number of groups). An example of dendrogram is shown in Figure 3. This representation provides very informative description and a visualization of the potential data clustering structures, especially when real hierarchical relations exist in the data (1,4).

As far as the choice of the optimal partition (optimal number of clusters) is concerned, different cluster validity criteria are used for hierarchical and partitioning methods. In particular, for hierarchical clustering methods the optimal partition is achieved by selecting one of the solutions

Table 1. Some Agglomerative Clustering Methods

Single linkage method (also known as nearest-neighbor method)	The single linkage method uses the smallest distance among the pairs of units of two clusters to define intercluster distance. Single linkage clustering tends to generate elongated clusters, which causes the chaining effect (1). As a result, noise may produce merging of two clusters with different properties. However, if the clusters are separated far from each other, the single linkage method works well
Complete linkage method	The complete linkage method uses the farthest distance among the pairs of units of two clusters to define inter-cluster distance
Group average linkage method (also known as the unweighted pair group method average, i.e., UPGMA)	The distance between two clusters is defined as the average of the distances between all pairs of units in the two clusters
Weighted average linkage method (also known as the weighted pair group method average, i.e., WPGMA)	Similar to UPGMA, the average linkage is also used to calculate the distance between two clusters. The difference is that the distances between the newly formed cluster and the rest are weighted based on the number of data units in each cluster
Centroid linkage method (known as the unweighted pair group method centroid, i.e., UPGMC)	Two clusters are merged on the basis of the distance of their centroids (means)
Ward's method (also known as the minimum variance method)	Clusters are merged aiming at minimizing the increase of the so-called within-class sum of the squared errors

in the sequence representing the hierarchy, equivalent to cutting a dendrogram at a particular height (sometimes termed the best cut). This defines a partition such that clusters below that height are distant from each other by at least that amount, and the appearance of the dendrogram can thus informally suggest the number of clusters. Large changes in merging levels are taken to indicate the best cut. Other criteria based on the investigation of the dendrogram are described in Reference 1. More formal approaches to the problem of determining the number of clusters have been reviewed by authors in References 1, 6.

Nonhierarchical Clustering (Partitioning Clustering). In contrast to hierarchical clustering, which yields a sequence of partitions into clusters by iterative fusions or divisions, nonhierarchical or partitioning clustering assigns a set of data units to c clusters without any hierarchical structure, thus requiring previous knowledge about the number of clusters. The process is accomplished by optimizing a criterion function, usually by minimization of an objective function representing the variability of the clusters within (4). The best-known and popular nonhierarchical clustering method is the c-means clustering. Another very common partitioning method is the c-medoids clustering. In the following, we present briefly these methods and the cluster validity criteria for determining the optimal number of clusters that in these methods have to be prespecified.

c-Means Clustering Method. The c-means clustering method (7) is one of the best known and most popular clustering methods. The optimal partition of the data in c-means clustering is obtained by minimizing the sum of squared error criterion shown in equation 1 with an iterative optimization procedure, which belongs to the category of hill climbing algorithms (4). The basic clustering procedure of c-means clustering is summarized as follows (1,4):

1. Initialize a c-partition randomly or on the basis of some prior knowledge. Calculate the cluster prototypes (*centroids* or *means*) (i.e., calculate the mean of the variables in each cluster considering only the observations belonging to each cluster).
2. Assign each unit in the data set to the nearest cluster by using an appropriate distance measure between each unit and centroids.
3. Recalculate the cluster prototypes (centroids or mean) based on the current partition.
4. Repeat steps 2 and 3 until no change is required for each cluster.

Mathematically, the *c-means clustering method* is formalized as follows:

$$\min : \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}\, d_{ik}^2 = \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}\, \|\mathbf{x}_i - \mathbf{h}_k\|^2$$

$$\sum_{k=1}^{c} u_{ik} = 1,\ u_{ik} \geq 0, u_{ik} = \{0,1\} \tag{1}$$

where u_{ik} indicates the membership degree of the ith unit to the kth cluster; $\mathbf{h}_k = (h_{k1}, \ldots, h_{kj}, \ldots, h_{kJ})'$ represents the kth centroid, where h_{kj} indicates the jth component (jth variable) of the kth centroid vector; $u_{ik} = \{0,1\}$, that is, $u_{ik} = 1$ when the ith unit belongs to the cth cluster; $u_{ik} = 0$ otherwise; $d_{ik}^2 = \|\mathbf{x}_i - \mathbf{h}_k\|^2$ indicates the squared Euclidean distance between the ith unit and the centroid of the kth cluster. For more details see Reference 1.

c-Medoids Clustering Method. By considering the *c-medoids clustering method* (5,8), units are classified into clusters where the prototype of each cluster is the so-called *medoid*, that is, a unit belonging to the cluster representative of the units of the cluster. Each medoid represents the prototypal features of the clusters and then synthesizes the characteristics of the units belonging to

each cluster. Following the *c-medoids clustering method*, the objective function to be minimized is represented by the sum (or mathematically equivalent, average) of the dissimilarity of units to their closest representative unit. The *c-medoids clustering method* first computes a set of representative units, called *medoids*. After finding the set of medoids, each unit of the data set is assigned to the nearest medoid. The algorithm suggested by Kaufman and Rousseeuw (5) for *c-medoids clustering method* proceeds in two phases: (*i*) selection of c "centrally located" units to be used as initial medoids and (*ii*) swapping of a selected (as a medoid) with an unselected unit, if the objective function can be reduced by the swap. This is continued till the objective function can no longer be decreased. Then, by considering a set of n units by $\mathbf{X}$ (set of the observations) and a subset of $\mathbf{X}$ with c units by $\tilde{\mathbf{X}}$ (set of the medoids) (where $c << n$), the formalization of the model is as follows:

$$\min : \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}\, \tilde{d}_{ik}^2 = \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}\, \|\mathbf{x}_i - \tilde{\mathbf{x}}_k\|^2 \qquad (2)$$

$$\sum_{k=1}^{c} u_{ik} = 1,\ u_{ik} \geq 0, u_{ik} = \{0, 1\}$$

where u_{ik} indicates the membership degree of the ith unit to the kth cluster; $u_{ik} = \{0, 1\}$, that is, $u_{ik} = 1$ when the ith unit belongs to the cth cluster; $u_{ik} = 0$ otherwise; $\tilde{d}_{ik}^2 = \|\mathbf{x}_i - \tilde{\mathbf{x}}_k\|^2$ indicates the squared Euclidean distance between the ith unit and the medoid of the kth cluster.

Some Cluster Validity Criteria. Nonhierarchical clustering requires previous knowledge about the number of clusters. Useful cluster validity criteria for determining the number of clusters are as follows.

Calinski and Harabasz criterion: Calinski and Harabasz (9) suggest taking the value of c, the number of clusters, which corresponds to the maximum value of C_c:

$$C_c = \frac{\text{trace}\,(\mathbf{B})}{(c-1)} : \frac{\text{trace}\,(\mathbf{W})}{(n-c)}$$

where $\mathbf{B}$ is the between groups dispersion matrix and $\mathbf{W}$ is the within-group dispersion matrix. The evaluation of this criterion at a given number of groups, g, requires knowledge of the group membership to determine the matrices $\mathbf{B}$ and $\mathbf{W}$. Notice that $\mathbf{T} = \mathbf{W} + \mathbf{B}$, where $\mathbf{T}$ indicates the total dispersion matrix. See Reference 1 for more details. In general, different clustering methods give rise to different number of groups (1).

Silhouette criterion (10): A unit $i \in (1, \ldots, n)$ belonging to cluster $k \in (1, \ldots, c)$ is considered, meaning, for example, by a c-means clustering algorithm that the ith unit is closer to the centroid of the kth cluster than to any other centroid. Let the average (squared Euclidean) distance of the ith unit to all other units belonging to cluster k be denoted by a_{ik}. Also, let the average distance of this unit to all units belonging to cluster k', $k' \neq k$, be denoted by $d_{ik'}$.

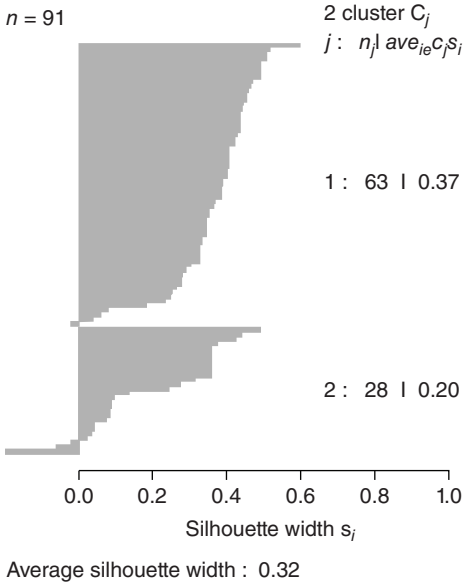

Figure 4. Silhouette width for each unit, average for cluster, average I_{CS} (1).

Finally, let $b_{ik'}$ be the minimum $d_{ik'}$ computed over $k' = 1, \ldots, c, k' \neq k$, which represents the dissimilarity of the ith unit to its closest neighboring cluster. Then, the silhouette of ith unit is defined as follows:

$$s_i = \frac{b_{ik'} - a_{ik}}{\max\{a_{ik}, b_{ik'}\}}$$

where the denominator is a normalization term. Evidently, the higher s_i, the better the assignment of ith unit to cth cluster.

The silhouette defined as the average of s_i over $i = 1, \ldots, n$ is:

$$I_{\text{CS}} = \frac{1}{n}\sum_{i}^{n} s_i.$$

The best partition is achieved when the silhouette is maximized, which implies minimizing the intra cluster distance (a_{ik}) while maximizing the intercluster distance ($b_{ik'}$). In Figure 4, the silhouette widths for each unit s_i, their average for cluster, and the total average I_{CS} are represented for the two cluster partition of a data set of 91 university students preferences and attitudes toward video games (1). The silhouette widths for each cluster are ordered from the highest to the smallest.

Co-clustering, Comparison Clustering, Consensus Clustering, Strategy of Analysis. *Co-clustering* (biclustering or two-mode clustering) is a technique that allows simultaneous clustering of rows (units) and columns (variables) of a data matrix $\mathbf{X}$. The goal of co-clustering is to generate biclusters/co-clusters: a subset of rows that exhibit similar behavior across a subset of columns (11).

Comparison clustering deals with a variety of methods proposed to measure the similarity of two clustering partitions. Some of them – among which the Rand Index – are based on counting the number of pairs in agreement (same cluster)/disagreement (different clusters) in the compared

partitions. Other indices have their foundation on concepts from information theory (12).

Consensus clustering regards methods aiming at combining multiple partitions of the same set of units into a *consensus partition*. In the literature, there are three main approaches for obtaining a *consensus partition*: the *constructive*, the *axiomatic*, and the *optimization* approach. The most natural way for defining consensus of partitions is by the optimization approach that maximizes some similarity measure between the consensus partition and each of the base partitions (13).

Some *strategies of analysis* combine regression analysis and cluster analysis. Clusterwise linear regression is a multivariate statistical procedure that attempts to cluster units with the objective of minimizing the sum of the error sums of squares for the within-cluster regression models (14). Other strategies of analysis combine data reduction methods and cluster analysis. These methods can use a *sequential approach* or a *simultaneous* approach. Among the strategies of analysis in the sequential approach there are *Tandem Analysis type 1*, that is, application of a factorial method on the data matrix and, sequentially, c-means on the score matrix, and *Tandem Analysis type 2*, that is, application of c-means on the data matrix and, sequentially, a factorial method on the centroid matrix (15). The simultaneous approach applies simultaneously dimension reduction and cluster analysis on the data matrix identifying the best partition of the units, described by the best orthogonal linear combinations of the variables according to an optimization criterion. It can be applied to quantitative or qualitative data. Among the strategies of analysis in this approach there are reduced K-means (RKM)/factorial K-means (FKM) and multiple correspondence K-means (MCKM), respectively, for quantitative and qualitative data (16,17).

Available Software. In R, clustering methods are implemented in the following libraries, for example:

- *kmeans.ddR* (*https://CRAN.R-project.org/package=kmeans.ddR*)
- *NbClust* (*https://CRAN.R-project.org/package=NbClust*)
- *stats* (*https://cran.r-project.org*)

2.2. Fuzzy Clustering

Fuzzy c-Means (FcM) Clustering. Let $\mathbf{X} = \{x_{ij} : i = 1, \ldots, n;\ j = 1, \ldots, J\} = \{\mathbf{x}_i = (x_{i1}, \ldots, x_{ij}, \ldots, x_{iJ})' : i = 1, \ldots, n\}$ be a data matrix, where x_{ij} represents the jth quantitative variable observed on the ith unit and $\mathbf{x}_i$ represents the vector of the variables observed for the ith unit. The FcM clustering method proposed in Reference 18 is formalized in the following way:

$$\min : \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}^m d_{ik}^2 = \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}^m \|\mathbf{x}_i - \mathbf{h}_k\|^2 \qquad (3)$$

$$\sum_{k=1}^{c} u_{ik} = 1,\ u_{ik} \geq 0$$

where u_{ik} denotes the membership degree of the ith unit to the kth cluster; $d_{ik}^2 = \|\mathbf{x}_i - \mathbf{h}_k\|^2$ is the squared Euclidean distance between the ith unit and the centroid of the kth cluster; $\mathbf{h}_k = (h_{k1}, \ldots, h_{kj}, \ldots, h_{kJ})'$ represents the kth centroid, where h_{kj} indicates the jth component (jth variable) of the kth centroid vector; $m > 1$ is a parameter controlling the fuzziness of the partition (for the selection of m, see Reference 19). The standard c-means (cM) clustering method (7) is obtained by setting $m = 1$ in equation (3), (see Section 3). The optimal iterative solutions obtained by solving the constrained optimization problem equation (3) with the Lagrangian multipliers method are (see Reference 18):

$$u_{ik} = \left(\sum_{k'=1}^{c} \left[\frac{\|\mathbf{x}_i - \mathbf{h}_k\|}{\|\mathbf{x}_i - \mathbf{h}_{k'}\|} \right]^{\frac{2}{m-1}} \right)^{-1}, \qquad \mathbf{h}_k = \frac{\sum_{i=1}^{n} u_{ik}^m \mathbf{x}_i}{\sum_{i=1}^{n} u_{ik}^m}. \quad (4)$$

Reasons for adopting a fuzzy clustering approach are discussed in the literature. As remarked in Reference 20, fuzzy clustering approach offers advantages over classic hard clustering approach. First, the fuzzy clustering methods are computationally more efficient because heavy changes in the value of cluster membership are less likely to occur during the estimation procedures (21). Second, fuzzy clustering has been shown to be less affected by local optima problems (22). Finally, the memberships indicate whether there is a second-best cluster almost as good as the best cluster generally not possible with traditional clustering methods (1).

A data set motivating fuzzy clustering is the butterfly data set (Fig. 5). It consists of 15 points; 3 data points form a bridge between the wings of a butterfly. The results of applying Ruspini's algorithm (18) are listed as membership functions reported in Figure 5 (membership to the second cluster is the complement to 1 of the membership to the first cluster). At the bottom of Figure 5, the memberships to the two clusters (shown as continuous although discrete) as functions of the horizontal coordinate illustrate the way in which fuzzy clustering smooths hard clustering. The point representing the geometric centroid of the data has membership of 0.5 in each fuzzy cluster: progressing away from the "core" of each wing, memberships become more and more distinct.

For more details, see Reference 19.

Cluster Validity. In the FcM clustering method (eq. 3), before computing the membership degrees and the centroids iteratively, by means of (eq. 4), a suitable number of clusters c has to be selected. Many cluster–validity criteria have been suggested. For a review on fuzzy cluster validity criteria see, among others, References 23, 24.

The Xie–Beni Criterion. A widely used cluster validity criterion for selecting c is the *Xie–Beni criterion* (25):

$$\min_{c \in \Omega_c} : I_{\mathrm{XB}} = \frac{\sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}^m \|\mathbf{x}_i - \mathbf{h}_k\|^2}{n \min_{k,k'} \|\mathbf{h}_k - \mathbf{h}_{k'}\|^2} \qquad (5)$$

where Ω_c represents the set of possible values of c ($c < n$).

The numerator of I_{XB} represents the *total within-cluster distance*. It is the objective function J of FcM clustering method. The ratio J/n is a measure of the *compactness* of the fuzzy partition. The smaller this ratio, the more compact a partition with a fixed number of clusters and any

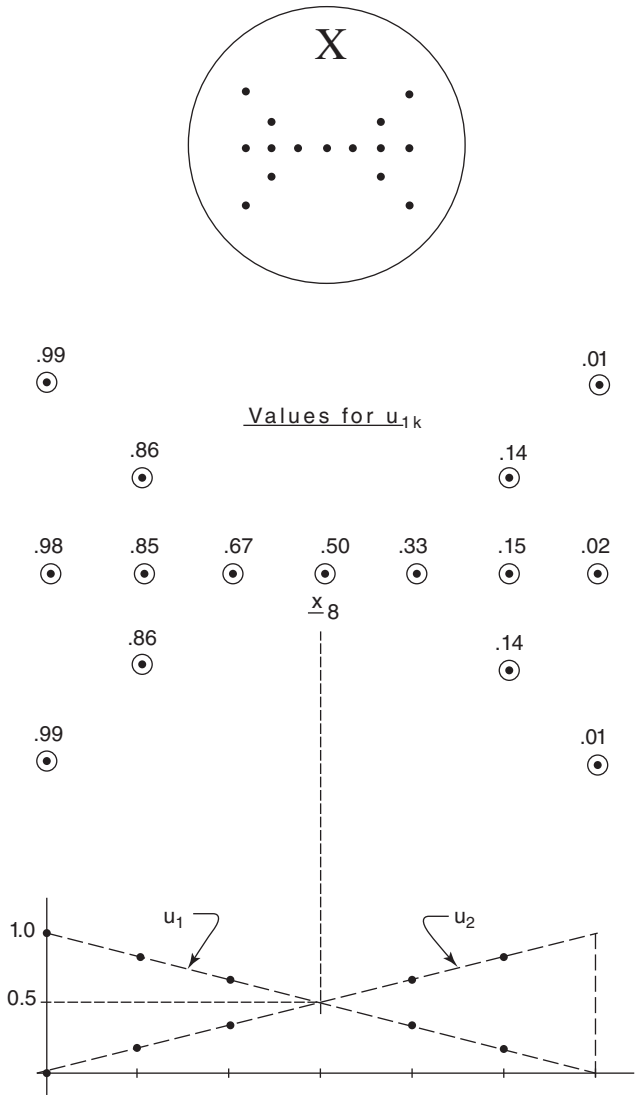

Figure 5. Butterfly data: membership assignment using Ruspini's algorithm (18).

number of data units. The minimum squared distance between centroids in the denominator of I_{XB} is a measure of the *separation* of the fuzzy partition. The greater this distance, the more separate a data partition with a fixed number of clusters. Therefore, for a fixed number of clusters, the partition with the smaller I_{XB} is chosen.

The Silhouette Criterion. Another interesting cluster validity procedure is the fuzzy extension of the *Silhouette criterion* (26) (see Section 3).

The fuzzy silhouette makes explicit use of the fuzzy partition matrix $\mathbf{U} = \{u_{ik} : i = 1, \ldots, n;\ k = 1, \ldots, c\}$. It considers the information on the membership degrees contained in the fuzzy partition matrix **U** by stressing importance of units concentrated in the vicinity of the cluster prototypes (high membership) while reducing importance of units lying in overlapping areas (small membership). The fuzzy silhouette (I_{FS}) is defined as follows:

$$I_{FS} = \frac{\sum_{i=1}^{n}(u_{ik} - u_{ik'})^{\gamma} s_i}{\sum_{i=1}^{n}(u_{ik} - u_{ik'})^{\gamma}}, \tag{6}$$

where u_{ik} and $u_{ik'}$ are the first and second largest elements of the ith row of the fuzzy partition matrix, respectively, and $\gamma \geq 0$ is a weighting coefficient. The effect of varying this parameter on the weighting terms in equation 6 is investigated in Reference 26.

As remarked by Campello and Hruschka (26), the fuzzy silhouette (eq. 6) differs from I_{CS} "for being a weighted average (instead of an arithmetic mean) of the individual silhouettes s_i. The weight of each term is determined by the difference between the membership degrees of the corresponding unit to its first- and second-best matching fuzzy clusters, respectively. In this way, a unit in the near vicinity of a cluster prototype is given more importance than another unit located in an overlapping area (where the membership degrees of the units to two or more fuzzy clusters are similar)."

With respect to other well known validity criteria based uniquely upon the fuzzy partition matrix (such as the partition coefficient), the fuzzy silhouette (eq. 6) takes into account the geometrical information related to the data distribution by means of the term s_i.

Fuzzy c-Medoids (FcMd) Clustering. The c-medoids clustering method has been introduced in the fuzzy framework as fuzzy c-medoids (FcMd) clustering (27,28).

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_n\}$ be a set of n units (data matrix) and let $\tilde{\mathbf{X}} = \{\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_i, \ldots, \tilde{\mathbf{x}}_c\}$ be a subset of $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_n\}$ with cardinality c.

The FcMd clustering method is formalized as follows:

$$\min : \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}^{m}\tilde{d}_{ik}^{2} = \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}^{m}\|\mathbf{x}_i - \tilde{\mathbf{x}}_k\|^2 \tag{7}$$

$$\sum_{k=1}^{c} u_{ik} = 1,\ u_{ik} \geq 0$$

where $\tilde{d}_{ik}^{2} = \|\mathbf{x}_i - \tilde{\mathbf{x}}_k\|^2$ indicates the squared Euclidean distance between the ith unit and the medoid of the kth cluster.

Solving the constrained optimization problem (eq. 7) by means of the Lagrangian multiplier method the local optimal solutions are as follows (28):

$$u_{ik} = \left(\sum_{k'=1}^{c}\left[\frac{\|\mathbf{x}_i - \tilde{\mathbf{x}}_k\|}{\|\mathbf{x}_i - \tilde{\mathbf{x}}_{k'}\|}\right]^{\frac{2}{m-1}}\right)^{-1}. \tag{8}$$

FcMd clustering method belongs to the class of partitioning around medoids (PAM) procedures. In FcMd clustering, each cluster is represented by an observed unit and not by an artificial one (prototype, i.e., centroid). The possibility of obtaining observed representative prototypes in the clusters is very appealing and important for the interpretation of the selected clusters in many applications. In fact, as remarked by Kaufman and Rousseeuw (5) "in many clustering problems one is particularly interested in a characterization of the clusters by means of typical or representative units. These are units that represent

the various structural aspects of the set of units being investigated. These representative units not only provide a characterization of the clusters, but can often be used for further work or research, especially when it is more economical or convenient to use a small set of c units."

FcMd clustering method does not depend on the order in which the units are presented (except when equivalent solutions exist, which very rarely occurs in practice). This is not the case for many other algorithms present in the literature (5).

FcMd clustering exhibits more robustness to the presence of outliers with respect to the c-means version because a medoid is less influenced by outliers or other extreme values than a mean. Thus, FcMd can be considered more robust than its possible c-means version. However, as remarked by García-Escudero and Gordaliza (29,30), the FcMd provides only a timid attempt to alleviate the negative effects of the presence of outliers in the dataset, hence only a mild robustification of the FcM.

The medoids $\tilde{\mathbf{X}} = \{\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_i, \ldots, \tilde{\mathbf{x}}_n\}$ obtained by minimizing the objective function in equation 7 provide a fuzzy partition via equation 8. However, the objective function in equation 7 cannot be minimized by means of the alternating optimization algorithm, because the necessary conditions cannot be derived by differentiation with respect to the medoids. A fuzzy clustering solution minimizing the objective function in equation 7 can nonetheless be obtained following the heuristic algorithm of Reference 31 for a crisp version of the objective function in equation 7 (28).

As for the classical case, the algorithm utilized for equations 7 and 8 falls in the category of alternating cluster estimation paradigm (32). The algorithm does not guarantee to find the global minimum. Thus, more than one random start is suggested.

The algorithm utilized for equations 7 and 8 is based on an exhaustive search for the medoids, which with large datasets could be too computationally heavy. The computational complexity of FcMd can be reduced by considering the "linearized" algorithm introduced in References 28, 33. In this way, when updating the medoids for cluster k only the subset of units with the higher membership degree in cluster k are considered.

Since the medoid always has a membership of 1 in the cluster, raising its membership to the power m has no effect. Thus, when m is high, the mobility of the medoids may be lost. For this reason, a value between 1 and 1.5 for m is recommended (34).

Fuzzy Relational Clustering. The FcM-based clustering methods and their variants consider the case where the vector of observed variables is available for each unit in the data set. In many real cases, the input data takes the form of a $(n \times n)$-pairwise dissimilarity matrix, each element of which indicates the dissimilarity between a pair of units. Relational clustering aims at identifying clusters using this information.

In a fuzzy framework, there exists a large variety of clustering techniques for such settings (35,36,37). In the literature, the first fuzzy clustering method for relational data (*fuzzy relational clustering*) has been proposed by Trauwaert (38) and successively extended by Kaufman and Rousseeuw (5). This clustering method can be formalized as follows:

$$\min : \sum_{k=1}^{c} \frac{\sum_{i,i'=1}^{n} u_{ik}^m u_{i'k}^m d_{ii'}}{2 \sum_{i'=1}^{n} u_{i'k}^m} \tag{9}$$

where u_{ik} and $u_{i'k}$ represent, respectively, the membership degrees of the ith and i'th units to the kth cluster and $d_{ii'}$ indicates a dissimilarity measure between each pair of ith and i'th units. In equation (9), any type of dissimilarity measures (city-block distance, Lagrange distance, and so on) can be used. The factor 2 in the denominator compensates the duplicity (5) of each term in the multiple sums.

A local optimal solutions of equation (9) can be found by using the Lagrangian multiplier method, by taking into account the Kuhn–Tucker conditions (see, e.g., Reference 39). Notice that the optimization of equation (9) is not alternating optimization, but simply optimization. For the detailed technical description of the numerical algorithm used for equation (9), see, for example, Reference 5.

We remark that the principal advantage of the fuzzy relational clustering approach is that we can utilize any type of dissimilarity measure in the clustering framework. With the choice in the fuzzy relational algorithm the squared Euclidian distance is used as a dissimilarity measure, it corresponds to the FcM (for the proof, see Reference 5).

Other useful readings on fuzzy relational clustering can be found in References 28, 40.

Possibilistic Clustering. In FcM, the constraint of summing up to one of the membership degrees for each unit (see eq. 3) may give rise to meaningless results, especially in the presence of noise. Following the Possibility Theory (41) by dropping the normalization constraint $\sum_{k=1}^{c} u_{ik} = 1$ a more intuitive assignment of degrees of membership is obtained. The possibilistic c-means (PcM) clustering model provides "degrees of compatibility" of the units with each of the clusters (42).

In the possibilistic perspective, u_{ik} represents the degree of possibility of unit i belonging to cluster k or, in other terms, the degree of "compatibility" of the profile $\mathbf{x}_i$ with the characteristics of cluster k embodied the related prototype $\mathbf{h}_k$. The FcM objective function is consequently modified by introducing a "penalization" term that takes care of the balance between the fuzziness of the clustering structure and the "compactness" of the clusters.

Two possible implementations of the PcM objective function (see Reference 43) are as follows:

$$\sum_{i=1}^{n} \sum_{k=1}^{c} u_{ik}^m \|\mathbf{x}_i - \mathbf{h}_k\|^2 + \sum_{k=1}^{c} \eta_k \sum_{i=1}^{n} (1 - u_{ik})^m \tag{10}$$

$$\sum_{i=1}^{n} \sum_{k=1}^{c} u_{ik}^m \|\mathbf{x}_i - \mathbf{h}_k\|^2 + \sum_{k=1}^{c} \eta_k \sum_{i=1}^{n} (u_{ik} \log u_{ik} - u_{ik}) \tag{11}$$

where η_k is a tuning parameter associated with cluster k, weighting its contribution to the penalization function. For details on η_k see References 43, 44.

Krishnapuram and Keller (43) argue that the possibilistic approach provides a "mode-seeking" clustering

procedure, to be confronted with the "partition-seeking" property of FcM. Thus, PcM clustering methods tend to be more robust with respect to noise, as compared to FcM techniques. Minimization of equation (10), leads to the following solutions:

$$u_{ik} = \frac{1}{1 + \left(\frac{\|\mathbf{x}_i - \mathbf{h}_k\|^2}{\eta_k} \right)^{\frac{1}{m-1}}} \quad . \tag{12}$$

The solutions obtained using objective functions such as equations (10) and (11) are mainly affected by the η_k parameters. Limitations in the use of PcM algorithms are due to the possibility of "coincident clusters" (45). A proper initialization of the parameters is required for the algorithm to work (42).

An alternative possibilistic clustering approach has been given by Yang and Wu (46):

$$\sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}^m \|\mathbf{x}_i - \mathbf{h}_k\|^2 + \frac{\beta}{m^2\sqrt{c}} \sum_{i=1}^{n}\sum_{k=1}^{c} (u_{ik}^m \log u_{ik}^m - u_{ik}^m) \tag{13}$$

where $\frac{\beta}{m^2\sqrt{c}}$ is a suitable tuning parametric function (46). In this case the iterative solutions are as follows:

$$u_{ik} = \exp\left(-\frac{m\sqrt{c}\|\mathbf{x}_i - \mathbf{h}_k\|^2}{\beta} \right). \tag{14}$$

Possibilistic clustering methods based on a "partitioning around medoids" approach can be obtained substituting in the previous methods the medoids $\tilde{\mathbf{x}}_k$ to the centroids $\mathbf{h}_k$.

Robust Fuzzy Clustering. In this section, three robust fuzzy clustering models able to neutralize the negative effects of noise and outliers data in the clustering process are introduced. They are robust variants of FcM or FcdM.

Fuzzy Clustering with Noise Cluster. The fuzzy c-means clustering with noise cluster (FcM-NC) neutralizes the negative effects of noise and outliers data in deviating the centroids from their true positions introducing the so-called *noise cluster*, a cluster collecting units far away from the natural c clusters in the data. FcM-NC has been initially proposed by Davé (47), which uses a criterion similar to Ohashi (48), and later extended by Davé and Sen (49). The *noise cluster* is not explicitly associated with a prototype, but to a *fictitious prototype* (*noise prototype*) at a constant distance (*noise distance*) from every unit in the data. A unit belongs to a *real cluster* only if its distance from a prototype (centroid) is lower than the noise distance; otherwise, the object belongs to the noise cluster.

FcM-NC clustering method can be formalized as follows:

$$\min : \sum_{i=1}^{n}\sum_{k=1}^{c-1} u_{ik}^m \|\mathbf{x}_i - \mathbf{h}_k\|^2 + \sum_{i=1}^{n} \delta^2 \left(1 - \sum_{k=1}^{c-1} u_{ik} \right)^m \tag{15}$$

$$\sum_{k=1}^{c} u_{ik} = 1,\ u_{ik} \geq 0$$

where δ is a suitable scale parameter, the so-called *noise distance*, to be chosen in advance. Such parameter plays the role to increase (for high values of δ) or to decrease (for low values of δ) the emphasis of the "noise component" in the minimization of the objective function in equation (15), for example, $\delta^2 = \lambda[n(c-1)]^{-1}[\sum_{i=1}^{n}\sum_{k=1}^{c-1}\|\mathbf{x}_i - \mathbf{h}_k\|^2]$, where λ is a scale multiplier that needs to be selected depending on the type of data.

It has to be observed that the model provides c clusters, $(c-1)$ of which are "real" cluster. The difference in the second term of the objective function shown in equation (15) expresses the membership degree of each unit to the noise cluster, and shows that the sum of the membership degrees over the first $(c-1)$ clusters is lower than or equal to 1. Indeed, the membership degree (u_{i*}) of the ith object to the *noise cluster* is defined as $u_{i*} = 1 - \sum_{k=1}^{c-1} u_{ik}$ and the usual constraint of the FcM ($\sum_{k=1}^{c-1} u_{ik} = 1$) is not required. Thus, the membership constraint for the *real clusters* is relaxed to $\sum_{k=1}^{c-1} u_{ik} \leq 1$. This allows noise object to have small membership values in *good clusters* (40).

By solving equation (15), we obtain

$$u_{ik} = \left[\sum_{k'=1}^{c-1} \left[\frac{\|\mathbf{x}_i - \mathbf{h}_k\|}{\|\mathbf{x}_i - \mathbf{h}_{k'}\|} \right]^{\frac{2}{m-1}} + \left[\frac{\|\mathbf{x}_i - \mathbf{h}_k\|}{\delta} \right]^{\frac{2}{m-1}} \right]^{-1}.$$

A FcMd version of equation (15), called FcMd-NC, can be obtained considering the medoid $\tilde{\mathbf{x}}_k$ instead of the centroid $\mathbf{h}_k$.

Fuzzy Clustering with Exponential Distance. The fuzzy c-means clustering with exponential distance (FcM-Exp) neutralizes the negative effects of noise and outliers data in deviating the centroids from their true positions introducing in the objective function the exponential distance, resulting in the following objective function to be minimized:

$$\min : \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}^m \left[1 - \exp\left\{ -\beta \|\mathbf{x}_i - \mathbf{h}_k\|^2 \right\} \right] \tag{16}$$

$$\sum_{k=1}^{c} u_{ik} = 1,\ u_{ik} \geq 0$$

where $m > 1$ is a weighting exponent that controls the fuzziness of the obtained partition.

Following Wu and Yang (50), the local optimal solutions for the objective function in equation (16) are as follows:

$$u_{ik} = \left(\sum_{k'=1}^{c} \left[\frac{1 - \exp\left\{ -\beta \|\mathbf{x}_i - \mathbf{h}_k\|^2 \right\}}{1 - \exp\left\{ -\beta \|\mathbf{x}_i - \mathbf{h}_{k'}\|^2 \right\}} \right]^{\frac{1}{m-1}} \right)^{-1} \tag{17}$$

and

$$\mathbf{h}_k = \frac{\sum_{i=1}^{n} u_{ik}^m \exp\left\{ -\beta \|\mathbf{x}_i - \mathbf{h}_k\|^2 \right\} \mathbf{x}_i}{\sum_{i=1}^{n} u_{ik}^m \exp\left\{ -\beta \|\mathbf{x}_i - \mathbf{h}_k\|^2 \right\}}. \tag{18}$$

The exponential distance assigns small influence in determining the centroids to outliers being a monotone increasing function of the distance (50). Following Wu and Yang (50), β is set as the inverse of the total variance of the data.

Wu and Yang (50) showed that the c-means clustering model based on the exponential distance is more robust than the model based on the Euclidean norm.

Wu and Yang (50) also used the fixed-point iterative method to solve $\mathbf{h}_k$ in equation (18).

The medoids version (i.e., FcMd-Exp) of the FcM-Exp model is obtained considering the medoid $\tilde{\mathbf{x}}_k$ instead of the centroid $\mathbf{h}_k$.

Trimmed Fuzzy Clustering. The trimmed fuzzy c-means clustering model (Tr-FcM) neutralizes the negative effects of noise and outliers data in deviating the centroids from their true positions by adopting the "impartial trimming" procedure (29,51) to identify the units more distant from the data. The procedure is said to be "impartial" because the trimming is led by the data. This approach is also suitable to detect both "outlying clusters" (outliers grouped in one small cluster) and "radial outliers" (isolated outliers) (52).

Given a trimming size α that ranges between 0 and 1, the double minimization problem is the following:

$$\min_{Y} \min_{u_{ik}} : \sum_{i=1}^{H(\alpha)} \sum_{k=1}^{c} u_{ik}^{m} \|\mathbf{x}_i - \mathbf{h}_k\|^2 \qquad (19)$$

$$\sum_{k=1}^{c} u_{ik} = 1,\ u_{ik} \geq 0$$

where u_{ik} is the membership degree of the ith unit to the $k$the cluster; $m > 1$ is the fuzziness parameter –the greater the value of m the more fuzzy is the obtained partition–; Y ranges on all the subsets of the objects $\{\mathbf{x}_i = (x_{i1}, \ldots, x_{is}, \ldots, x_{ip})' : i = 1, \ldots, n\}$, containing $H(\alpha) = \lfloor n \cdot (1 - \alpha) \rfloor$ units ($\lfloor . \rfloor$ is the integer part of a given value). Using the above described trimming rule we allow for a proportion α of units to be left unassigned (51). Notice that equation (19) includes FcM as a limit case when $\alpha = 0$. Then, each non-trimmed object is allocated into the cluster corresponding to its closest centroid.

The local optimal solutions are as follows:

$$u_{ik} = \left(\sum_{k'=1}^{c} \left[\frac{\|\mathbf{x}_i - \mathbf{h}_k\|}{\|\mathbf{x}_i - \mathbf{h}_{k'}\|} \right]^{\frac{2}{m-1}} \right)^{-1}, \qquad \mathbf{h}_k = \frac{\sum_{i=1}^{H(\alpha)} u_{ik}^{m} \mathbf{x}_i}{\sum_{i=1}^{H(\alpha)} u_{ik}^{m}}. \qquad (20)$$

For more details, see, for example, Reference 53.

Also, in this case the partitioning around medoids version is easily obtained.

Other Fuzzy Clustering Methods: Gustafson–Kessel Clustering, Fuzzy Shell Clustering, Kernel-Based Fuzzy Clustering. The *Gustafson–Kessel clustering* method replaces the Euclidean distance by a cluster-specific Mahalanobis distance (Section "Distance Measures"), adapting various sizes and forms of the clusters, to extract from the data more information than the methods based on the Euclidean distance. For cluster c, the associated Mahalanobis distance between unit i and prototype $\mathbf{h}_c$ of cluster c is

$$_{M}d_{ic} = \left[(\mathbf{x}_i - \mathbf{h}_c)^T \mathbf{S}_c^{-1} (\mathbf{h}_c - \mathbf{x}_j) \right]^{\frac{1}{2}}$$

where $\mathbf{S}_c$ is the covariance matrix among the variables in the cth cluster. The objective function and the update equations of the prototypes of the Gustafson–Kessel algorithm are the same as Fuzzy c-means with the replacement of the Euclidean distance with the Mahalonobis distance. In addition there is an update equation for the covariance matrices of each cluster that are modified to incorporate the fuzzy information (54). Some techniques have been proposed to improve the calculation of the fuzzy covariance matrix in the Gustafson–Kessel clustering algorithm.

Fuzzy shell clustering is a generalization of fuzzy cluster analysis – in particular of the fuzzy c-means algorithm – to *shell like* clusters, that is, clusters that lie in nonlinear subspaces (55) and resemble shells or surfaces with no interior points. The Euclidean distance is replaced by other distances to allow the comparison between input data and prototypes. Fuzzy shell algorithm can detect ellipses, quadrics, and so on. There is a large number of fuzzy shell clustering algorithms that use different kinds of prototype and different distance measures. For example, the *fuzzy c ellipsoidal shell algorithm* searches for shell clusters with the shape of ellipses, ellipsoids, or hyperellipsoids. At this point a distance between an input unit and the closest ellipse is introduced (55).

Kernel-based variants of *fuzzy clustering* modify the distance function to handle nonvectorial data, such as trees, sequences, or graphs without modifying completely the standard algorithm. The essence of kernel-based methods involves performing an arbitrary nonlinear mapping from the original feature space to the *kernel space*. There are two major forms of kernel-based fuzzy clustering. The first one comes with prototypes constructed in the feature space and the second one with prototypes retained in the kernel space thus requiring an inverse mapping from kernel space to feature space. The optimization of the objective function, the updating rules for the membership degrees, and the derivation of the prototypes are obtained as in fuzzy c-means and depend on the specific selection of the kernel function (56).

Fuzzy Co-clustering, Comparison of Fuzzy Clustering, Consensus of Fuzzy Clustering, Strategy of Analysis. In a fuzzy framework, some fuzzy *co-clustering* methods have been proposed by Frigui and Nasraoui (57).

With regard to *comparison of fuzzy clustering*, a useful criterion for comparing each pair of fuzzy partitions obtained by fuzzy methods is the fuzzy Rand index. It is a fuzzy extension of the original Rand index based on the comparison of agreements (consistent classifications) and disagreements (inconsistent classifications) of the two partitions, the fuzzy partition and the hard partition (58).

Consensus of fuzzy clustering arises by the availability of different fuzzy partitions of the same set of objects and it can be relevant to obtain a *consensus* partition that summarizes the information contained in the different fuzzy partitions. The most natural way for obtaining fuzzy consensus of fuzzy partitions is by the optimization approach: it considers a criterion that measures the distance between the set of fuzzy partitions in the profile and a fuzzy classification, and one seeks a fuzzy consensus classification that optimizes the stated criterion (59).

The *strategies af analysis* described in Section "Co-clustering, Comparison Clustering, Consensus Clustering, Strategy of Analysis" apply to fuzzy data. In particular,

fuzzy clusterwise regression analysis (60), and the combination of data reduction and fuzzy cluster analysis (61) have been proposed.

Available Software. In R, fuzzy clustering methods are implemented in the following libraries, for example

- *clue* (*https://CRAN.R-project.org/package=clue*)
- *cluster* (*https://CRAN.R-project.org/package=cluster*)
- *clustrd* (*https://CRAN.R-project.org/package=clustrd*)
- *e1071* (*https://CRAN.R-project.org/package=e1071*)
- *kml* (*https://CRAN.R-project.org/package=kml*)
- *skmeans* (*https://CRAN.R-project.org/package=skmeans*)
- *vegclust* (*https://CRAN.R-project.org/package=vegclust*)

2.3. Clustering of Nonstandard Data

In this section, clustering methods of nonstandard data, in particular based on the fuzzy approach, are reviewed. In the previous section, we have analyzed FcM and its variants for standard data structures, that is, standard quantitative/numerical data.

Here, we focus our attention on data feature-based variants of FcM, that is, fuzzy clustering methods for nonstandard data. In particular, we consider fuzzy clustering methods for data with different nature (i.e., fuzzy data, symbolic data, interval data, categorical data, text data, time and/or spatial data, three-way data, sequence data, functional data, network data, directional data, mixed data) and data with particular structural features (i.e., outlier data, incomplete data, big data).

Fuzzy Data. In the last years, a great deal of attention has been paid to fuzzy cluster analysis for imprecise/vague data, where the impreciseness/vagueness is modelled following a fuzzy approach (fuzzy data). For a formal definition of fuzzy data, see, for example, Reference 62. Hathaway et al. (63) analyzed heterogeneous fuzzy data by utilizing fuzzy clustering. Pedrycz et al. (64) suggested nonparametric methods for fusing heterogeneous fuzzy data. Yang and Ko (65) developed fuzzy clustering methods for univariate fuzzy data. Yang and Liu (66) extended the Yang–Ko's clustering methods to conical fuzzy vectors. Auephanwiriyakul and Keller (67) suggested a linguistic fuzzy clustering method for fuzzy data based on the extension principle and the decomposition theorem. Yang et al. (68) proposed a fuzzy clustering method for fuzzy and symbolic data (see also Section "Symbolic Data"), by defining a "composite" dissimilarity measure. Hung and Yang (69) developed a robust fuzzy clustering method for univariate fuzzy data based on exponential-type distance measure. D'Urso and Giordani (70) suggested a fuzzy clustering method for symmetrical fuzzy data by using a "weighted" dissimilarity for comparing pairs of fuzzy data that is composed of two distances, the so-called center distance, and spread distance. The method tunes automatically the influence of the two components of the fuzzy data for calculating the center and spreads centroids in the fuzzy clustering procedure. Coppi et al. (42) proposed two clustering models for fuzzy data by adopting, respectively, fuzzy and possibilistic approaches. Coppi and D'Urso (71,72) suggested fuzzy clustering methods for fuzzy time-varying data. For an application of the fuzzy clustering for fuzzy data, see, for example, Reference 73. For a survey on fuzzy clustering for fuzzy data see References 62, 74–76.

In a fuzzy framework, an interesting nonhierarchical clustering for symbolic data has been proposed by El-Sonbaty and Ismail (77). El-Sonbaty and Ismail (77), analyzing different types of symbolic data, remarked that the fuzzy methodological approach improve the performance of the clustering process, giving more meaning and easier interpretation of the results obtained from their clustering method.

Interval-Valued Data. Interval-valued data refers to variables observed in the form of intervals, rather than single numbers. There are different studies regarding the clustering of interval-valued data. For instance, by defining a suitable distance measure for interval-valued data and following a fuzzy approach, D'Urso and Giordani (78) - assuming that an interval-valued datum is represented by the center and the radius of the interval (the radius is the distance between the center and lower/upper bound of the interval) – suggested a robust FcM clustering method for classifying interval–valued data. The peculiarity of this method is the capability of managing outlier interval-valued data by reducing the effects of such outliers in the clustering process. Notice that, in the interval case, the concept of outlier data involves both the center and the width (the radius) of an interval. Other useful references are References 79–81.

We observe that, since interval-valued data can be considered as a particular case of fuzzy data (fuzzy data with uniform membership function) or a particular case of symbolic data, the fuzzy clustering methods for fuzzy and symbolic data (see Sections "Fuzzy Data" and "Symbolic Data") can also be suitably utilized for classifying interval-valued data.

Categorical Data. Recently, in the clustering literature, increasing attention has been paid to cluster analysis for categorical data, since this task is of great practical relevance in several fields. Several methods for categorical data have been suggested. Among them, the *k*-modes clustering method proposed by Huang (1997) is one of the most efficient clustering algorithm. It uses a dissimilarity measure between two categorical data defined by the total mismatches of the corresponding categories instead of the Euclidean distance and *means* instead of *modes* for cluster prototypes. A fuzzy version of the *k*-modes clustering algorithm has been proposed by Huang and Ng (1999). In this method each pattern is allowed to have memberships in all clusters rather than just a distinct membership to a single cluster. The membership matrix provides more information to help the users to decide the core and boundary objects of clusters. Lee and Pedrycz (82) introduced a generalization of the *k*-modes type clustering

algorithm with fuzzy p-mode prototypes. The fuzzy p-mode algorithm incorporates a weighting scheme for the dissimilarity measure by which each category is automatically assigned with a weight measuring its individual contribution for the clusters. Recently, a fuzzy clustering method with between-cluster information for categorical data has been suggested by Bai et al. (83). For other fuzzy clustering methods for categorical data, see References 84–86.

Symbolic Data. Symbolic data occur as multi valued (as in lists) interval-valued or categorical-valued observations. There are different non-fuzzy clustering methods for classifying symbolic data. Most of these techniques are based on hierarchical methodologies, which use the concept of agglomerative or divisive methods as the core of the algorithm.

Textual Data (Text Data). In the recent years, text data has gradually become a new research topic. Among them, the study of text clustering has attracted wide attention. Different fuzzy clustering methods have been suggested in the literature. For instance, Krishnapuram et al. (27) introduced a fuzzy k-medoids algorithm with application to web document. Runkler and Bezdek (37), considering distance measures for text strings (i.e., Levenshtein distance), proposed a fuzzy clustering method. Recently, Deng et al. (87) proposed fuzzy clustering for text data. They introduced the feature evaluation method to reduce the dimension of the text vector, and therefore they introduced the high power sample point set, the field radius, and weight to calculate the initial clustering center of the text and to keep the clustering results stable. Finally, they use the edit distance to recalculate the sample points on the boundary value among the clusters and to determine the type of sample points and optimize the clustering results.

Time Series Data. Several fuzzy clustering for time-varying data have been suggested. For instance, following a *model-based approach*, (88) proposed an autoregressive model-based FcMd clustering method and some of its variants for classifying univariate time series. By adopting a partitioning around medoids approach, D'Urso et al. (89) proposed GARCH-based FcMd clustering methods for classifying financial time series. Following a *feature-based approach*, Maharaj et al. (90) proposed a wavelet-based FcM clustering for univariate time series, and successively D'Urso and Maharaj (91) suggested different wavelet-based fuzzy clustering methods for multivariate time series. In Reference 92, Maharaj and D'Urso introduced different fuzzy clustering methods of univariate time series in the frequency domain. Following an *observation-based approach*, different FcM clustering methods (also robust methods) for classifying multivariate time trajectories were proposed. Coppi and D'Urso (93) suggested an entropy-based fuzzy clustering for time trajectories. They introduced a FcMd clustering for time trajectories. Coppi and D'Urso (71,72,94,95) suggested, respectively, FcM, FcMd, and entropy-based clustering methods for fuzzy multivariate time trajectories. See also References 53, 96–99. For more details on time series clustering –also in a fuzzy framework– see Caiado et al. (100).

Spatial Data. In the literature, many works devoted to the development of clustering methods for spatial units have been suggested. The peculiarity of these techniques consists in their capability to suitably deal with the distinguishing characteristics of *spatial data*, that is, spatial dependence and spatial heterogeneity.

The fuzzy clustering methods for spatial data can be classified with respect to the objects to be clustered:

1. *Geographical Areas* (usually defined by means of administrative boundaries): In this class, the clustering methods aim at determining clusters of geographical areas such that the within cluster dispersion is minimized with the additional assumption that the configuration of the obtained clusters should satisfy particular spatial constraints (e.g., that the obtained clusters are formed by spatially contiguous areas). The empirical evidence suggests that spatial data are often characterized by positive spatial autocorrelation: neighboring sites tend to have similar features. If such spatial autocorrelation affects the observed data, this should be explicitly considered in the clustering method (instead of arbitrarily ignoring it) so that the resulting clusters may detect it (101). For an example of fuzzy clustering methods belonging to this class, see Reference 102.
2. *Pixels* (image segmentation): In this class, the clustering methods basically aim at assigning the pixels (i.e., the observation objects) in an image to different clusters according to their features. The standard clustering methods do not take into account the information given by the spatial distribution of the pixels, but only the one given by the observed features. To overcome this problem, clustering algorithms have been adapted by suitably taking into account spatial information (101). Among the spatial fuzzy clustering methods belonging to this class, we mention the methods suggested by Tolias and Panas (103,104), Pham and Prince (105), Liew, et al. (106,107), Pham (108), and Liew and Yan (109–111).

We remark that a possible way for extending the FcM clustering method to spatial data consists of adding a suitable spatial penalty term in the objective function of the clustering method. A reasonable choice for the spatial penalty term has been developed by Pham (108). Such a proposal has been introduced for solving the image segmentation problem. However, it also appears to be applicable to the case of geographical areas (101).

Spatial–Time Data. By considering two types of dissimilarity measures for multivariate trajectories – that is, the *cross sectional dissimilarity* that compares the instantaneous (positional) features of the trajectories and the *longitudinal dissimilarity* that captures the differences concerning the evolutive features (i.e., the "variational" patterns) of the trajectories measured by means of their velocities - and following a *fuzzy approach*, Coppi et al. (101)

proposed two types of objective function-based fuzzy clustering methods for classifying spatial units on the basis of multivariate time-varying empirical information (*spatial–time data*): the cross-sectional fuzzy c-means clustering for spatial-time data (CS-FcM-ST) and the longitudinal fuzzy c-means clustering for spatial-time data (L-FcM-ST). In particular, for the CS-FcM-ST, the objective function is constituted by two terms:

- The *instantaneous within cluster dispersion term*. It is a measure of the within cluster (cross-sectional) dissimilarities of the multivariate trajectories with respect to the centroids, appropriately weighted by membership degrees. Therefore, by minimizing this term we maximize the internal cohesion of the clusters, conditional on allowing for a certain degree of flexibility as indicated by the fuzziness parameter m.
- The *spatial penalty term* (*spatial regularization term*). The aim of this term is the following: for each spatial object i and each generic cluster k, the sum of the membership degrees of the contiguous/neighboring spatial objects in all the clusters except cluster k is constrained to be as small as possible.

For analytical details of these methods see Reference 101. Another interesting reference is Reference 112.

Three-Way Data. Many fuzzy clustering methods have been proposed for three-way data arrays (i.e., arrays of the type objects × variables × occasions). Interested readers may refer to References 113, 114. Other examples of fuzzy clustering for three-way data can be found, for instance, in References 59, 101, 115, 116.

Sequence Data. In the context of human activity pattern analysis based on "virtual" (e.g., web usability) or physical movements (i.e., grocery shopping activity, pedestrian urban activity), it is very interesting to classify persons by considering their activity patterns, that is, their individual behaviors in an actual or virtual domain represented by sequences (paths). Following a fuzzy approach, D'Urso and Massari (117) proposed some clustering methods for classifying individuals by taking into account their activity behaviors. A fuzzy approach is suitable for sequence data, since sequences (e.g., sequences of human activities) are typically characterized by switching behaviors, which are likely to produce overlapping clusters. D'Urso and Massari (117) adopted a partitioning around medoids strategy since in human activity patterns analysis it is useful to represent each cluster by means of a not fictitious prototypes (i.e., medoids). To measure pairwise distances among all sequence pairs they make use of the Levenshtein distance, which allows for the comparison between sequences of different length and explicitly takes into account the sequential nature of the data. In particular, they proposed a FcMd clustering for sequence data and two of its robust versions based, respectively, on noise cluster and on trimming technique. Suggestive applications of the three suggested methods to shopping path, Web usage mining, travel behavior, tourists path, and skiers paths are also shown in their paper.

Runkler and Bezdek (37) suggested a relational clustering algorithm by introducing an alternating cluster estimation procedure for relational data, that is, relational alternating cluster estimation (RACE) that is very similar to ACE (alternating cluster estimation); it is useful for relational matrices, when the starting point is represented, for example, by a distance matrix. We remark that in ACE (and RACE), membership degrees and prototypes are specified by the user. In the Runkler–Bezdek's method, the RACE algorithm is specifically devoted to web mining. In particular, in a web content mining context, the authors consider distance measures for text strings, that is, Levenshtein distance; in a web log mining framework, they introduce a graph-based distance measure and the Levenshtein distance for graph traversal sequences, in particular for web page sequences. The D'Urso–Massari's methods can be adopted for a wide range of types of human activity patterns (e.g., grocery shopping paths, travel behavior paths, tourist behavior paths, skiers paths, pedestrian activity paths, web log paths, web content paths, eye tracking paths, mouse tracking paths, and so on). Furthermore, we remark the two robust clustering methods suggested by D'Urso and Massari (117) are more resistant to disruptive effects of outliers in path data than RACE-based clustering algorithm proposed by Runkler and Bezdek (37).

Functional Data. Functional data are multivariate data with an ordering on the dimensions, thus a collection of functions. Typical examples include time series data such as weather data and human growth data. Functional data analysis has recently attracted many researchers. In the clustering literature, several non-fuzzy methods for classifying functional data have been proposed. However, recently, also in the fuzzy framework, some methodological proposals for classifying in a fuzzy manner functional data have been suggested. For instance, we point out the fuzzy clustering method proposed by Tokushige et al. (118).

Network Data. Network data refer to the structure of a communication network modeling social interaction. As remarked by Liu (119), "in recent years an explosive growth of interest and activity on the structure and dynamics of complex networks has appeared. This is partly due to the influx of new ideas, particularly ideas from statistical mechanics, to the subject, and partly due to the emergence of interesting and challenging new examples of complex networks such as the internet and wireless communication networks". In this regard to find the best partition of a large and complex network into a small number of clusters has been addressed in many different ways. Following a fuzzy approach, Liu (119) proposed a partitioning formulation, which is extended from a deterministic framework for network partition based on the optimal prediction of a random walker Markovian dynamics. See Reference 119 for more details.

Directional Data. In directional data the data, are represented by observed *directions*. The directions are regarded as points on the circumference of a circle in two

dimensions or on the surface of a sphere in three dimensions. Directional data are often met in astronomy, biology, and medicine. Fuzzy clustering is a useful tool for classifying directional data. Yang and Pan (120) suggested a fuzzy clustering method, called the fuzzy c-directions clustering method, applying the class of fuzzy classification maximum likelihood procedures to two-dimensional von Mises distribution (the von Mises distribution is the most used probability density on directional data).

Mixed Data. In many real situations, we may have datasets with mixed types of data, that is, quantitative data, categorical data, symbolic and fuzzy data, time and/or spatial-varying data, and so on.

Yang, et al. (68) proposed fuzzy clustering methods with feature vectors, including numeric, symbolic, and fuzzy data. Chatzis (121) introduced an extension of the FcM algorithm to allow for handling data with mixed numeric and categorical variables. For other recent references on fuzzy clustering with mixed data (i.e., numeric and categorical variables) see, for example, References 122, 123.

Incomplete Data. In classification, an interesting topic is the cluster analysis with incomplete datasets, that is datasets with missing values. In a fuzzy framework, interesting methods have been introduced by Hathaway and Bezdek (124,125). A kernel-based FcM clustering method for incomplete data has been proposed by Zhang and Chen (126). Different approaches to fuzzy clustering of incomplete data are illustrated in Timm et al. (127). Honda and Ichihashi (128) proposed two methods for partitioning an incomplete dataset with missing values into several linear clusters by extracting local principal components. The first method is an extension of the fuzzy c-varieties clustering that can be regarded as the method for local principal component analysis of fuzzy covariance matrices. The second method is a simultaneous application of fuzzy clustering and principal component analysis (strategy of analyses) of fuzzy correlation matrices. Both methods estimate prototypes ignoring only missing values and they need no data preprocessing such as elimination of samples with missing values or imputation of missing cases.

Big Data. *Big data* are any data that you cannot load into your computer's working memory (129). Huber (130) classified dataset sizes as follows: *tiny* (10^2 bytes), *small* (10^4 bytes), *medium* (10^6 bytes), *large* (10^8 bytes), *huge* (10^{10} bytes), *monster* (10^{12} bytes), *big* ($10^{>12}$ bytes) (added by Hathaway and Bezdek (131)). As remarked by Havens et al. (129), there are two main approaches for clustering very large data: distributed clustering based on various incremental styles and clustering a sample found by either progressive or random sampling. Each approach has been applied in the context of FcM clustering of very large data. The most well-known fuzzy clustering method for very large data is the *generalized extensible fast* FcM (geFFcM) proposed by Hathaway and Bezdek (131). This method utilizes statistics-based progressive sampling to produce a reduced dataset that is large enough to capture the overall nature of the data. Thus, the algorithm clusters this reduced dataset and non-iteratively extends the partition to the full dataset. However, as remarked by Hevens et al. (129), the sampling method utilized in geFFcM can be inefficient and, in some cases, the data reduction is not sufficient for very large data. Hence, these authors adapted geFFcM into a simple *random sampling plus extension* FcM (rseFcM) algorithm. Other leading algorithms include *single-pass* FcM (spFcM) (132) and *online* FcM (oFcM) (133), which are incremental algorithms to compute an approximate FCM solution. The *bit-reduced* FcM (brFcM) (134) algorithm uses a binning strategy for data reduction. Successively, a kernel-based strategy called *approximate kernel* FCM (akFcM) developed by Chitta et al. (135,136), relies on numerical approximation that uses sampled rows of kernel matrix to estimate the solution to a c-means problem.

Other fuzzy clustering methods for very large data are the *fast* FcM (fFcM) suggested by Shankar and Pal (137), in which FcM is applied to larger and larger nested samples until there is little change in the solution, and the *multistage random* FcM developed by Cheng et al. (138), which combines fFcM with a final literal run of FcM on the full dataset. Both these schemes are more in the spirit of acceleration, rather than scalability, as they both contain a final run on the full dataset (129). Other algorithms that are related, but were also developed for efficiency, include those proposed by Cannon et al. and Kolen et al. (139,140). Finally, we remark *fast kernel* FcM (fkFcM) proposed by Liao and Lin (141).

2.4. Clustering with Other Types of Uncertainty Management

In addition to fuzzy and possibilistic clustering, in the literature there are other clustering approaches for managing the uncertainty in the clustering process, that is, we remark: shadowed clustering (142), rough set-based clustering (143), intuitionistic fuzzy clustering (144), evidential clustering or credal clustering or belief clustering (145), credibilistic clustering (146), type-2 fuzzy clustering (147), neutrosophic clustering (148), hesitant fuzzy clustering (149), interval-based fuzzy clustering (150), picture fuzzy clustering (151). See also Reference 152 for a deep review on these clustering approaches.

3. MODEL-BASED UNSUPERVISED CLUSTERING

3.1. Mixture Clustering

Cluster analysis can be based on probability models of the data. Model-based clustering consists of fitting a mixture model to data and identifying each cluster with one of its components. For continuous data, the most common component distribution is a multivariate Gaussian (or Normal) distribution. Model-based clustering assumes that the multivariate observations $\mathbf{x} = (x_1, \cdots, x_p)$ are a sample from a finite mixture density $p(x/\theta) = \sum_{k=1}^{K} p_k f_k(x/\theta_k)$ where f_k and θ_k are the density and the parameters of the kth component in the mixture ($\theta = \theta_1, \cdots, \theta_K$) and p_k is the probability that an observation belongs to the kth component ($p_k \geq 0$, $\sum_{k=1}^{K} p_k = 1$). For estimation purposes, the mixture model is often expressed in terms of complete data, including

the groups to which the observation belongs. The complete data are $\mathbf{y} = (y_1, \cdots, y_n) = ((x_1, z_1), \cdots, (x_n, z_n))$, where the missing data are $\mathbf{z} = (z_1, \cdots, z_n)$, with $\mathbf{z}_i = (z_{i1}, \cdots, z_{iK})$ indicating the binary vectors such that $z_{ik} = 1$ if x comes from group k. The $\mathbf{z}_i$'s define a partition of the observed data in sets of $\mathbf{x}_i$ such that $z_{ik} = 1$. Geometric features (shape, volume, orientation) of the clusters are determined by the covariance matrices of the densities of the components. Banfield and Raftery (153) proposed a general framework for geometric constraints in multivariate normal mixtures by parametrizing the covariance matrix of each component through eigenvalue decomposition so that three parameters of the eigenvalue decomposition correspond to shape, size, and orientation of the clusters (by size the volume occupied by the cluster in the multidimensional space rather than the number of units it contains is intended). Banfield and Raftery also proposed methods for parameter estimation for clusters with non Gaussian distribution. The standard methodology to estimate the finite mixture parameters corresponding to each cluster in the presence of incomplete data consists of using the EM (expectation maximization) algorithm (154). The clustering is then done by assigning each unit to the cluster to which it is most likely to belong a posteriori, conditionally on the selected model and its estimated parameters.

Banfield and Raftery applied model-based clustering on a diabetes dataset containing three measurements, the area under a plasma glucose curve, the area under a plasma insulin curve, and steady-state plasma glucose curve for each of 145 subjects. The dataset is considered a standard introductory example for model-based clustering. The subjects were clinically diagnosed into three groups: normal, chemically diabetic, and overtly diabetic. The two-dimensional projection of the data shows a three-dimensional shape of a boomerang with two wings and a fat middle (Figure 6). One of the wings corresponds to patients with overt diabetes, the other wing is composed primarily of patients with chemical diabetes and the fat middle is composed of normal patients.

The data are modeled with a trivariate Gaussian distribution with a different covariance matrix for each component. The corresponding three-group classification matches the three clinically diagnosed groups with 90% accuracy.

For a survey of probabilistic models in the literature see References 155, 156. For reviews of model-based clustering, see References 157–159. A limitation of model-based clustering with high-dimensional data is the growth of the number of parameters of each component of the mixture. Their use can be limited for non Gaussian, high dimensional very large datasets. See also References 160, 161.

3.2. Available Software

In R, mixture clustering methods are implemented in the following library:

- *mclust* (*https://CRAN.R-project.org/package=mclust*)

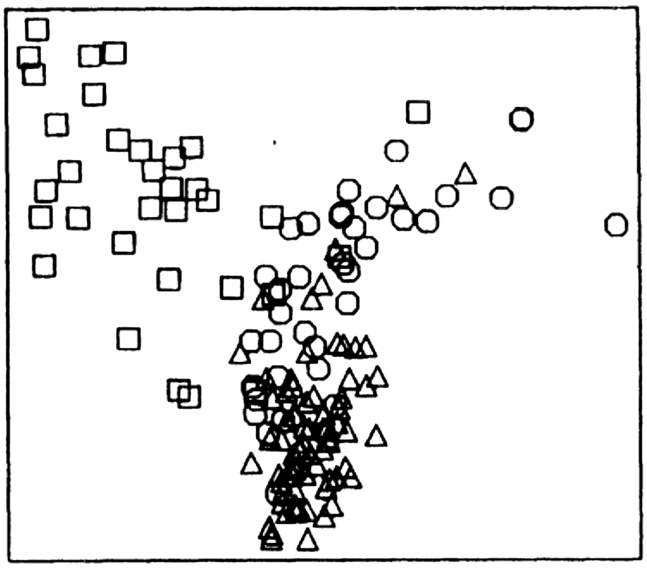

Figure 6. Two-dimensional projection of diabetes data set (circle chemical, square overt, triangle normal) (153).

4. UNSUPERVISED ARTIFICIAL NEURAL NETWORKS: THE SELF-ORGANIZING MAP

4.1. Neural Modeling and Early Work

The network architectures used to model neural systems can roughly be divided in two categories, supervised and unsupervised networks. *Supervised* networks are feedforward networks of cells (neurons) that transform sets of input data into sets of output data. The desired input–output transformation is determined by supervised adjustment of the system parameters. *Unsupervised* networks are competitive self-organizing networks of cells in which neighboring cells compete in their activities by means of mutual lateral interactions, and develop adaptively into specific sensors of different input patterns.

The self-organizing map belongs to the second category. It is an artificial neural network, the cells of which become specifically tuned to various input patterns or classes of patterns through an unsupervised learning process. In the basic version, only one cell or local group of cells at a time gives the active response to the current input. The locations of the responses tend to become ordered in a coordinate system created over the network for different input features. The spatial location or coordinates of a cell in the network then correspond to a particular domain of input pattern. Each cell or local cell group acts like a separate *sensor* for the same input. The interpretation of the input information does not produce an input–output transformation as in supervised networks, but gives rise to an active response in a spatial location.

The self-organizing map is "neural" as it behaves like various areas of the brain specialized to different cognitive functions. Some researchers in the 1950s (162) found that certain single neural cells in the brain respond selectively to some specific sensory stimuli. These cells are often organized into local groups (*brain maps)*, in which

their location corresponds to some feature value of a specific stimulus in an orderly manner.

Some biologists in the 1970s (163,164) tried to understand if feature-sensitive cells could also be formed in artificial systems automatically, by learning. Malsburg (165), and later Amari (166) implemented by the so-called competitively learning neural networks. In a subset of cells, adaptation of the strongest-activated cells made them become tuned to specific input. The above studies are of great theoretical importance because they involve a self-organizing tendency, but the ordering power they demonstrated was however still weak.

In 1981, Kohonen studied a process that seemed generally to produce globally well-organized maps, proposing the algorithm known as *The Self-Organizing Map* algorithm. Since then a wealth of contributions have been developed. In Oja et al. (167), many of them are collected and divided by type of contribution. The first few books edited by Kohonen are *Self-Organization and Associative Memory and Self-Organizing Maps* (3,168). Also, refer to References 169, 170.

4.2. The Self-Organizing Map

The Learning Rule. The two essential effects leading to spatially organized maps are as follows: (1) *Spatial concentration* of the network activity on the cell (or its neighborhood) that is best *tuned* to the present input (winner) and (2) further *sensitization or tuning* of the best-matching cell *and its topological neighbors* to the present input.

In biologically inspired neural network models, correlated learning by spatially neighboring cells can be implemented using various kinds of lateral feedback connection and other lateral interactions. In the process presented by Kohonen, lateral interaction is induced directly by defining a neighborhood set N_c around cell c. At each learning step all the cells within N_c are updated, whereas cells outside Nc are left intact. This neighborhood is centered around the cell for which the best match with input $\mathbf{x}$ is found. The width or radius of N_c can be time variable; in fact, for a global ordering, it has experimentally turned out to be advantageous to let N_c be very wide at the beginning and decreasing monotonically with time.

The explanation for this may be that a wide initial spatial resolution N_c in the learning process first induces a rough global order in the m_i values, after which narrowing the N_c improves the spatial resolution of the map. The global order is not altered afterward. It is even possible to end the process with $N_c = \{c\}$, that is, finally updating the best-matching unit (winner) only, in which case the process is reduced to simple competitive learning. Before this, however, the "topological order" of the map would have to be formed. The basic idea underlying what is called competitive learning is roughly as follows.

Assume a sequence of n statistical samples of a vectorial observable variable in the real J-dimensional space $\mathbf{x} = \mathbf{x}_i(t) \in \mathbb{R}^J$, $i = 1, 2 \ldots, n$ where t is the time coordinate, and a set of P vectors $\{\mathbf{m}_p(t) \in \mathbb{R}^J,\ p = 1, 2 \ldots, P\}$ (weight vectors or reference vectors or codebook vectors) each associated with a cell (neuron) of a network (topology, lattice, array) of P neurons. Each cell, beside the reference vector $\mathbf{m}_p$, has a (scalar or vectorial) location (coordinate) $\mathbf{r}_p$ dependent on the configuration of the network of neurons, one-dimensional or multidimensional. It is worth noting that the dimension J of the $\mathbf{m}_p$ vectors (the same of the input vectors) and the coordinate system $\mathbf{r}_p$ of the network arrangement of the cells may be different (Figure 7): the $\mathbf{m}_p$ can be multidimensional, whereas the cells may interconnect even in the lowest dimensional linear chain (scalar $\mathbf{r}_p$).

Assume that the $\mathbf{m}_p(0)$ are initialized with random values. If $\mathbf{x}(t)$ can somehow be simultaneously compared with each $\mathbf{m}_p(t)$ at each successive instant of time $t = 1, 2, 3, \ldots$, then the best-matching $\mathbf{m}_p(t)$ is updated to match even more closely the current $\mathbf{x}(t)$. If the comparison is based on some distance measure $d(\mathbf{x}, \mathbf{m}_p)$, updating $\mathbf{m}_p$ must be such that if $p = c$ is the index of the best-matching reference vector, then $d(\mathbf{x}, \mathbf{m}_c)$ is decreased, and all the other reference vectors $\mathbf{m}_p$, with $p \neq c$, are left intact.

In this way the different reference vectors tend to become specifically "tuned" to different variable domains of the input variable x.

The updating process or learning rule (in discrete time notation) may read:

$$\mathbf{m}_p(t+1) = \mathbf{m}_p(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{m}_p(t)] \quad p \in N_c$$
$$\mathbf{m}_p(t+1) = \mathbf{m}_p(t) \quad p \notin N_c \tag{21}$$

where $\alpha(t)$ is a scalar value $(0 < \alpha(t) < 1)$ decreasing with time. An alternative notation is to introduce a scalar "kernel" function $h_{cp}(t)$:

$$\mathbf{m}_p(t+1) = \mathbf{m}_p(t) + h_{cp}(t)[\mathbf{x}(t) - \mathbf{m}_p(t)]$$

where $h_{cp}(t) = 0$ outside N_c (the proper notation should be $h_{c(x)p}(t)$). A biological lateral interaction often has the form of a "bell curve." Denoting the coordinates of cells c and p by the vectors $\mathbf{r}_c$ and $\mathbf{r}_p$, respectively, a proper form for h_{cp} might be:

$$h_{cp}(t) = h_0 \exp -\frac{||\mathbf{r}_c - \mathbf{r}_p||^2}{\sigma^2}$$

with $h_0 = h_0(t)$ and $\sigma = \sigma(t)$ a suitable decreasing functions of time.

The computations for producing the ordered set of the SOM models can be implemented by either of the following main types of algorithms: (1) the reference vectors m_p are updated by a stepwise updating process in which the input vectors are applied to the algorithm one at a time, in a periodic or random sequence, for the time steps necessary to reach a reasonably stable state; (2) all of the input vectors are applied to the algorithm as one batch, and all of the reference vectors are updated in a single operation (batch algorithm).

Setting of the Parameters. Suggestions for the application of the algorithm in terms of learning rate, number of steps, value of $\alpha(t)$, topology, and size of the network can be found in Kohonen (171).

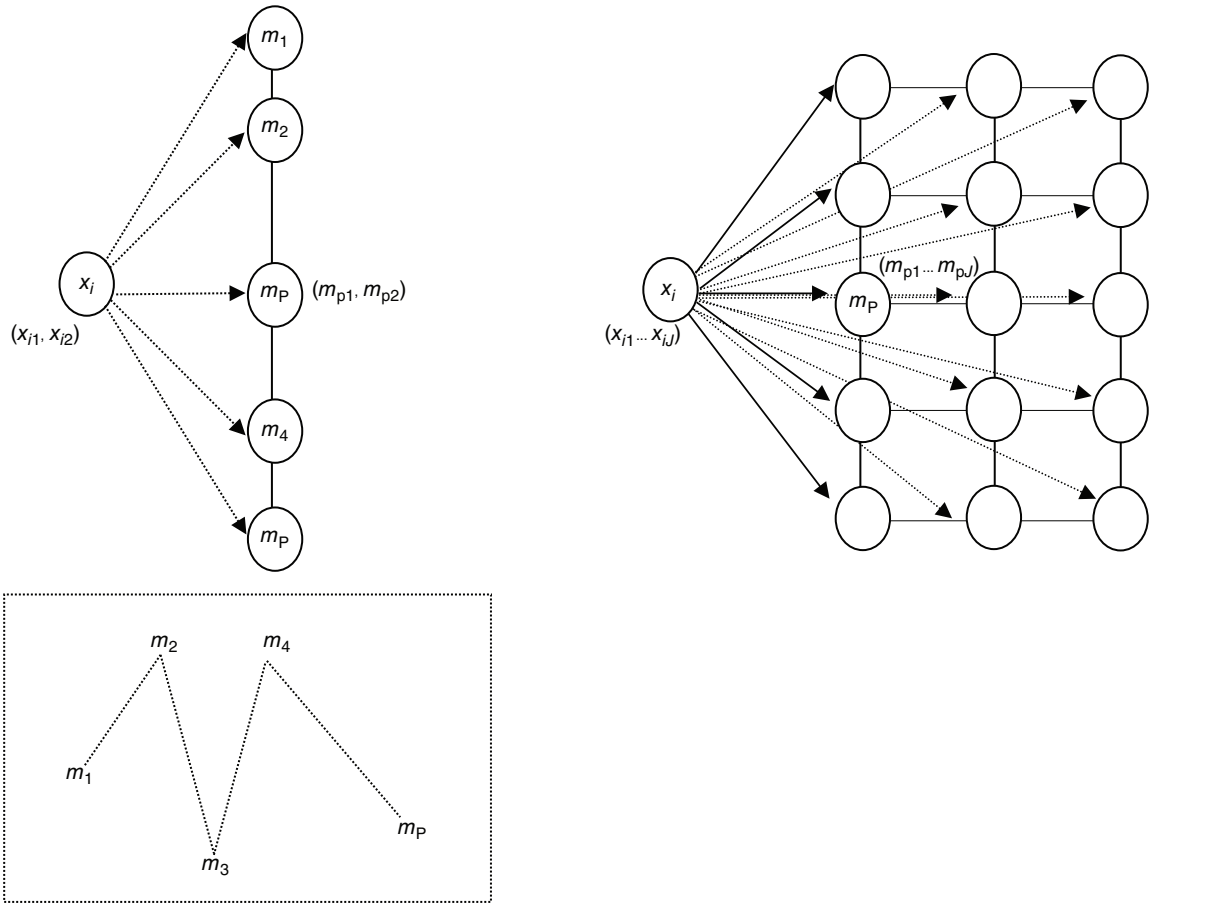

Figure 7. A two-dimensional input space mapped to a one-dimensional configuration of neurons (left: neural network top, input space bottom); a J-dimensional input space mapped to a two-dimensional configuration of neurons (right).

(1) The final statistical accuracy of the mapping depends on the number of time steps, which must be reasonably large. A rule of thumb is that, a good statistical accuracy is a number of time steps at least 500 times the number of network units. On the other hand, the dimension of the input has no on effect on the number of iteration steps.

(2) For approximately the first 1000 time steps, $\alpha(t)$ should start with a value that is close to unity, thereafter decreasing monotonically. The functional type of decreasing of $\alpha(t)$ can be linear, exponential, or inversely proportional to t. The ordering of the m_p occurs during the initial period, while the remaining steps are only needed for the fine adjustment of the map.

(3) The suggestion for the the choice of $N_c(t)$ is starting with a wide $N_c(0)$ and letting it decrease with time.

(4) In order to define the network structure and the number of cells of the network a preliminary visual inspection of $p(\mathbf{x})$ by, for example, Sammon projection is suggested.

Quality of Learning. The quality of learning in the SOM is measured through the *average expected quantization error* (AEQE) and the *expected distortion measure* (EDM), defined as:

$$\text{AEQE} = \int_{\mathbb{R}^J} d_g(\boldsymbol{m}_{c(x)}, \boldsymbol{x}) p(\boldsymbol{x}) \tag{22}$$

$$\text{EDM} = \int_{\mathbb{R}^J} \sum_{p=1}^{P} h_{p,\boldsymbol{c}(\boldsymbol{x})}(t) d_g(\boldsymbol{m}_p, \boldsymbol{x}) p(\boldsymbol{x}) \tag{23}$$

respectively, where d_g is a generic distance function (3), $\boldsymbol{x} \in \mathbb{R}^J$ is the input vector, $\boldsymbol{m}_{c(x)}$ is the weight vector closest to the input vector $\boldsymbol{x}$ according to d_g, and $h_{p,\boldsymbol{c}(\boldsymbol{x})}(t)$ is the degree of neighborhood between the locations of the neuron p and of the winner neuron c of $\boldsymbol{x}$.

The *average quantization error* (AQE) and the *distortion measure* (DM) are the sample counterpart of equations (22) and (23) and are defined as

$$\text{AQE} = n^{-1} \sum_{i=1}^{n} d_g(\boldsymbol{m}_{c(x_i)}, \boldsymbol{x}_i) \tag{24}$$

$$\text{DM} = \sum_{i=1}^{n} \sum_{p=1}^{P} h_{p,c(x_i)}(t) d_g(\boldsymbol{m}_p, \boldsymbol{x}_i). \tag{25}$$

Generally, in equations (22) and (24) the Euclidean distance ($\| \ldots \|$) is used as distance function d_g, while in equations (23) and (25), the squared Euclidean distance ($\| \ldots \|^2$) is adopted.

These measure can also be considered at the individual level, yielding the *individual quantization error* (IQE) and the *individual distortion measure* (IDM), respectively:

$$\text{IQE} = d_g(\boldsymbol{m}_{c(x_i)}, \boldsymbol{x}_i) \tag{26}$$

$$\text{IDM} = \sum_{p=1}^{P} h_{p,c(x_i)}(t) d_g(\boldsymbol{m}_p, \boldsymbol{x}_i). \quad (27)$$

The *ordering* ability of the SOMs-ID is measured through the analysis of the distances between the weight vectors and the related distances between their locations (closest neurons should have closest weight vectors).

The *topology preservation* ability of the SOMs-ID (closest input vectors should have closest neurons in the SOMs) is measured through the Spearman correlation coefficient between the ranks of the $I(I-1)/2$ distances between input vectors and the ranks of the distances of the weight vectors of the related closest neurons. Another measure of topology preservation is the topographic error that considers the ratio of input vectors for which the first and second best-matching cells are not adjacent. For other measures see Reference 172.

SOM Mathematics. Although the basic principles of the self-organizing systems are simple, the process behavior is difficult to be described in mathematical terms.

Convergence to an ordered state. In References 3, 168 the (self-)ordering of the weights is proved, restricting the considerations to a one-dimensional topology of neurons to each of which a scalar-valued input vector $\boldsymbol{x}$ is connected, showing that if $\boldsymbol{x}(t)$ is a random variable, considering the intermediate "states" (various types of partial sequences of the $\boldsymbol{m}_p$) of the process, then an index of disorder, $D = \sum_{p=2}^{P} |\boldsymbol{m}_p - \boldsymbol{m}_{p-1}| - |\boldsymbol{m}_1 - \boldsymbol{m}_P|$, more often decreases than increases in updating ($|x|$ denoting absolute value of x).

In Reference 173, (self-)ordering of the weights with respect to a one-dimensional topology and scalar-valued input vector is rigorously justified. The results hold for general metrics. The conditions regarding the learning rate under which convergence to an ordered state is obtained are $\sum_{s=0}^{\infty} \alpha(s) = \infty$, $\lim_{s\to\infty} \alpha(s) = 0$. See also References 174–177.

Generalizations of the (self-)ordering ability of the SOM to multidimensional input space and multidimensional topologies of the neurons have been considered.

With respect to the dimension of the input space, Budinich and Taylor (178) gave an intuitive necessary and sufficient condition of the decrease of D that applies to the case of multidimensional input space and one-dimensional topologies.

With respect to the topology of the network, Kohonen (3,168) assumes that in considering multidimensional topologies results similar to the one-dimensional case can be obtained. In Budinich and Taylor (178), the problems at the origin of ordering in higher dimensional topologies are intuitively explained.

Vector quantization. Vector Quantization (VQ) is a classical method, that produces an approximation to a continuous probability density function $p(\mathbf{x})$ of the vectorial input variable $\mathbf{x}$ using a finite number of codebook vectors $\mathbf{m}_p$, $p = 1, 2 \ldots, P$. Once the "codebook" is chosen, the approximation of $\mathbf{x}$ involves finding the reference vector $\mathbf{m}_c$ closest to $\mathbf{x}$. One kind of optimal placement of the $\mathbf{m}_p$ minimizes E, the expected rth power of the quantization error:

$$E = \int ||\mathbf{x} - \mathbf{m}_c||^r p(x) \mathrm{d}x \quad (28)$$

where $\mathrm{d}x$ is the volume differential in the $\mathbf{x}$ space, and the index $c = c(x)$ of the best matching reference vector (winner) is a function of the input vector $\mathbf{x}$:

$$\mathbf{m}_{c(x)} = \min_p \{||\mathbf{x} - \mathbf{m}_p||\}. \quad (29)$$

As far as the vector quantization ability of the SOM is concerned, Ritter (179) studies the probability density function of the weight vectors in simple cases showing that it approximates some monotonic function of the probability density function $p(\boldsymbol{x})$ of the J-dimensional continuous random variable $\mathbf{x}$.

Moreover in Reference 180 it is shown that the SOM learning process finds weight vectors minimizing the expected distortion measure and study under appropriate conditions.

At the end of the learning process each input is assigned to the closest reference vector, thus allowing clustering of the input data.

Simulations. The simulations presented are taken by Kohonen (3,168) and have been the first ones used to illustrate the effect that the reference vectors tend to approximate to the density function of the input vectors in an orderly faction. In these examples, the input vectors were chosen to be two-dimensional for visual display purposes, and their probability density function was arbitrarily selected to be uniform over the area demarcated by the borderlines (square or triangle). Outside the frame the density was zero. The vectors $\mathbf{x}(t)$ were drawn from this density function independently and at random, after which they caused adaptive changes in the reference vectors $\mathbf{m}_p$. The $\mathbf{m}_p$ vectors appear as points in the same coordinate system as that in which the $\mathbf{x}(t)$ are represented; in order to indicate to which cell $\mathbf{m}_p$ value belongs, the points corresponding to the $\mathbf{m}_p$ vectors have been connected by a lattice of lines conforming to the topology of the cells. A line connecting two reference vectors $\mathbf{m}_p$ and $\mathbf{m}_j$ is only used to indicate that the corresponding cells p and j are adjacent in the array of cells. In Figure 8, the arrangement of the two-dimensional cells is rectangular (square); whereas in Figure 9, the two-dimensional cells are interconnected in a linear chain.

4.3. Relation with Other Methods

It has been noted that in the SOM the dimension J of the $\mathbf{m}_p$ reference vectors (the same of the input vectors) and the coordinate system $\mathbf{r}_p$ of the network arrangement of the cells may be different. So the SOM is related with methods either of vector quantization or of dimensionality reduction. Among the others *clustering methods* and *projection methods* (linear and nonlinear) are considered. The c-means algorithm and the SOM algorithm are both vector quantization methods aiming at minimizing the distance between the input $\mathbf{x}$ and its representative.The

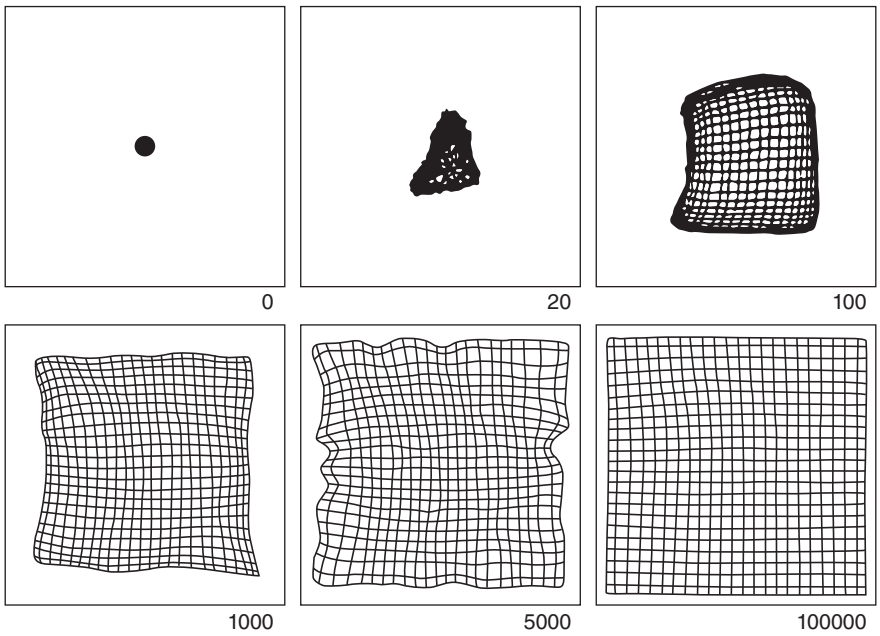

Figure 8. A square two-dimensional input space mapped to a one-dimensional configuration of neurons (168).

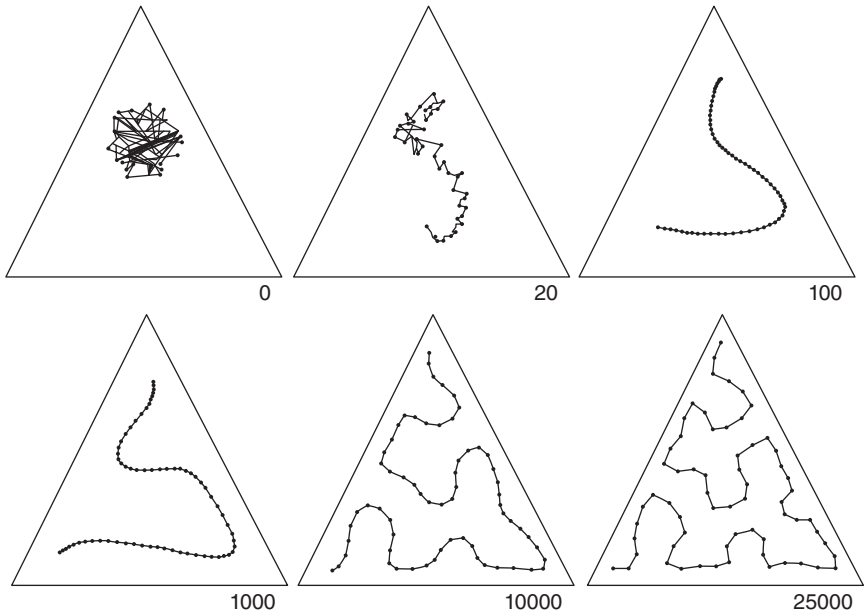

Figure 9. A triangular two-dimensional input space mapped to a one-dimensional configuration of neurons (168).

representative is the closest reference vector $\mathbf{m}_{cx}$ in the SOM and the prototype of the cluster in the c-means algorithm. With respect to the c-means clustering a side product of the SOM is the possibility to visualize the multivariate reference vectors $\mathbf{m}_p$ in the reduced low-dimensional (at least one-dimensional) coordinate system $\mathbf{r}_p$ of the related locations in the network arrangement. An advantage of the SOM is that the size of the network can be chosen much larger than the cluster structure of the input data as the clusters become visible on the network, thus overcoming the problem of the choice of the number of clusters in the c-means algorithm.

Linear (e.g., principal components analysis, PCA) and nonlinear (e.g., multidimensional scaling, MDS, or Sammon' mapping) projection methods can be used as methods of projection of the input vectors on a reduced space by preserving the variance of the data or the dissimilarities among the input vectors, respectively. The SOM represents the J-dimensional reference vectors associated with the neurons in the low-dimensional coordinate system $\mathbf{r}_p$

of the neuron network arrangement. Among all the previous dimensionality reduction methods, the essential difference between SOM and MDS is that the SOM tries to form a locally correct projection as a consequence of the lateral interaction between neurons, while MDS preserves all interpoint dissimilarities. Moreover, the projection in the SOM is a mapping into the coordinate system of the network arrangement of the reference vector and not the computation of artificial lower dimension variables as in PCA (181).

4.4. Variants, Developments, and Applications of the SOM

Kohonen proposed the LVQ (learning vector quantization) algorithm and its variants as supervisioned reward–punishment SOM in case of input vectors with known classification. The aims of the unsupervised and supervised learning processes are different. The unsupervised SOM is mainly intended to approximate the probability density function of the input by quantized reference vectors that are localized in the input space to minimize a quantization error functional. The supervised SOM (LVQ) in mainly intended to minimize the average expected misclassification probability (3,182).

A variety of versions of the basic SOM has been proposed. The SOM has been linked with density matching model and the point density that the SOM produces is linked to the density of the data. In the probabilistic SOM (183–185) a probabilistic mixture model is associated with the map, where each mixture component corresponds to a cell of the map, with related parameters. The optimal estimate of the parameters characterizing each mixture component is obtained by minimizing the Bregman divergences via partial differentials in respect to model parameters. The computation of the winning cell can be reformulated as the computation of the neuron that has the highest likelihood to have generated the observed input vector.

The Kernel method has been applied to the SOM (186). A kernel is a real function defined on couples of vectors in the input space. It is based on a (unknown or imaginary) nonlinear mapping function defined on each input vector x. The SOM is then operated entirely in the space defined by the kernel function. In hierarchical SOM GSOM (Growing SOM (187)) the topology of the SOM is dynamically defined in terms of size or structure of the map depending on intermediate results.

With regard to the input, an extension of the SOM for data imprecisely observed (self-organizing maps for imprecise data, SOMs-ID) has been proposed in which the learning algorithm is based on distances for imprecise data (188).

The main application areas of the self-organizing maps are, among the others, pattern recognition, robotics, processing of semantic information, industrial analyses and control, telecommunications, biomedical analyses, and finance. The spatial segregation of different responses and their organization into topologically related subsets result in a high degree of efficiency. By the end of the year 2005, more than 10,000 scientific works have been published that develop or apply the SOM (167,189,190).

SOMs have been applied to complex structures of data: fuzzy data (191), time series (192), interval data (188), three way data (193), high-dimensional data (187). See References 167, 189, 190.

4.5. Available Software

In R, self-organizing maps are implemented in the following libraries, for example,

- *kohonen* (*https://CRAN.R-project.org/package=kohonen*)
- *som* (*https://CRAN.R-project.org/package=som*)

BIBLIOGRAPHY

1. B. S. Everitt, S. Landau, and M. Leese. *Cluster Analysis*. 5th ed.; John Wiley & Sons, Inc.: New Jersey, 2011.
2. C. Hennig, M. Meila, F. Murtagh, and R. Rocci. *Handbook of Cluster Analysis*. Chapman and Hall, 2015.
3. T. Kohonen. *Self-Organizing Maps*. Springer, 1995.
4. R. Xu and D. Wunsch. *Clustering*. Jhon Wiley & Sons, Inc.: New Jersey, 2009.
5. L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, Inc.: New Jersey, 1990.
6. G. Milligan and M. Cooper. *Psychometrika*, **1985**, *50*(2), 150–179.
7. J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations, in *Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1; 1967, pp 281–297.
8. L. Kaufman and P. J. Rousseeuw. In *Statistical Data Analysis Based on the L1-Norm and Related Methods*, Dodge, Y. Ed.; 1987, pp 405–416.
9. T. Calinski and J. Harabasz. *Commun. Stat.*, **1974** *3*(1), pp 1–27.
10. P. J. Rousseeuw. *J. Comput. Appl. Math.*, **1987** *20*, pp 53–65.
11. I. Dhillon. Co-clustering Documents and Words Using Bipartite Spectral Graph Partitioning, in *Proc. of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*; ACM, 2001, pp 269–274.
12. N. Vinh, J. Epps, and J. Bailey. *J. Mach. Learn. Res.*, **2010**, *52*(11), pp 2837–2854.
13. S. Monti, P. Tamayo, and T. Mesirov, and J. Golub. *Mach. Learn.*, **2003**, *52*(1), pp 91–118.
14. W. S. DeSarbo and W. Cron. *J. Classif*, **1988**, *5*(2), pp 249–282.
15. P. Arabie, L. Hubert, W. Gaul, and D. Pfeifer. *From Data to Knowledge: Theoretical and Practical Aspects of Classification, Data Analysis, and Knowledge Organization: Studies in Classification, Data Analysis, and Knowledge Organization*. Springer-Verlag Berlin Heidelberg, 1996.
16. H. Bock, E. Diday, Y. Lechevallier, M. Schader, P. Bertrand, and B. e. Burtschy. *New Approaches in Classification and Data Analysis*. 1st ed.; Springer: Berlin Heidelberg, 1994.
17. M. van de Velden, A. D'Enza, and F. Palumbo. *Psychometrika*, **2017**, *82*(1), pp 158–185.
18. J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.*. Plenum Press, New York, 1981.
19. P. D'Urso. Fuzzy clustering. In *Handbook of Cluster Analysis*, Hennig, C., Meila, M., Murtagh, F., and Rocci, R. Eds.; Chapman and Hall, 2015; pp 545–573.

20. H. Hwang, W. S. DeSarbo, and Y. Takane. *Psychometrika*, **2007**, *72*(2), pp 181–198.
21. A. B. McBratney and A. W. Moore. *Agric. For. Meteorol*, **1985**, *35*(1), pp 165–185.
22. W. J. Heiser and P. J. F. Groenen. *Psychometrika*, **1997**, *62*(1), pp 63–83.
23. Y. Xu and R. G. Brereton. *Chemom. Intell. Lab. Syst.*, **2005**, *78*(1), pp 30–40.
24. W. Wang and Y. Zhang. *Fuzzy Sets Syst.*, **2007**, *158*(19), pp 2095–2117.
25. X. L. Xie and G. Beni. *IEEE Trans. Pattern Anal. Mach. Intell.*, **1991**, *13*(8), pp 841–847.
26. R. J. G. B. Campello and E. R. Hruschka. *Fuzzy Sets Syst.*, **2006** *157*(21), pp 2858–2875.
27. R. Krishnapuram, A. Joshi, and L. Yi. A Fuzzy Relative of the k-Medoids Algorithm with Application to Web Document and Snippet Clustering, in *1999 IEEE International Fuzzy Systems Conference Proceedings (FUZZ-IEEE'99)*, vol. 3, 1999; pp 1281–1286.
28. R. Krishnapuram, A. Joshi, O. Nasraoui, and L. Yi. *IEEE Trans. Fuzzy Syst.*, **2001**, *9*(4), pp 595–607.
29. L. A. García-Escudero and A. Gordaliza. . *J. Am. Stat. Assoc.*, **1999**, *94*(447), pp 956–969.
30. L. A. García-Escudero and A. Gordaliza. *J. Classif*, **2005**, *22*(2), pp 185–201.
31. K. S. Fu. *Syntactic Pattern Recognition and Applications*. Academic Press, San Diego, 1982.
32. T. A. Runkler and J. C. Bezdek. *IEEE Trans. Fuzzy Syst.*, **1999**, *7*(4), pp 377–393.
33. O. Nasraoui, R. Krishnapuram, A. Joshi, and T. Kamdar. *E-Commerce and Intelligent Methods*. Springer, 2002, pp 233-261.
34. T. Kamdar and A. Joshi. On Creating Adaptive Web Servers Using Weblog Mining. Technical report TR-CS- 00-05, Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, 2000.
35. R. J. Hathaway and J. C. Bezdek. *Pattern Recognit.*, **1994**, *27*(3), pp 429–437.
36. J. C. Bezdek, M. R. Pal, J. Keller, and R. Krisnapuram. *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Kluwer Academic Publishers, Norwell, MA, USA, 1999.
37. T. A. Runkler and J. C. Bezdek. *Int. J. Approx. Reason.*, **2003**, *32*(2), pp 217–236.
38. E. Trauwaert. In *Statistics Data Analysis based on the L1-Norm and Related Methods*, Dodge, Y., Ed.; North-Holland: Amsterdam, 1987, pp 417–426.
39. T. A. Runkler. In *Advances in Fuzzy Clustering and its Applications*, De Oliveira, J. V. and Pedrycz, W. Eds.; John Wiley & Sons, Ltd.: Chichester, 2007, pp 31–51.
40. R. N. Davé, and S. Sen. *IEEE Trans. Fuzzy Syst.*, **2002**, *10*(6), pp 713–727.
41. D. Dubois and H. M. Prade. *Possibility Theory*. Plenum press: New York, 1988.
42. R. Coppi, P. D'Urso, and P. Giordani. *Comput. Stat. Data Anal.*, **2012**, *56*(4), pp 915–927.
43. R. Krishnapuram and J. M. Keller. *IEEE Trans. Fuzzy Syst.*, **1996**, *4*(3), pp 385–393.
44. R. Kruse, C. Döring, and M.-J. Lesot. In *Advances in Fuzzy Clustering and its Applications*, De Oliveira, J. V. and Pedrycz, W., Eds.; John Wiley & Sons, Ltd.: Chichester, 2007, pp 3–30.
45. M. Barni, V. Cappellini, and A. Mecocci. *IEEE Trans. Fuzzy Syst*, **1996**, *4*, pp 393–396.
46. M.-S. Yang and K.-L. Wu. *Pattern Recognit.*, **2006**, *39*(1), pp 5–21.
47. R. N. Davé. *Pattern Recognit. Lett.*, **1991**, *12*(11), pp 657–664.
48. Y. Ohashi. Fuzzy clustering and robust estimation. In *Ninth Meeting of SAS Users Group International*, 1984.
49. R. N. Davé and S. Sen. Noise Clustering Algorithm Revisited, in *1997 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS'97)*; **1997**, pp 199–204.
50. K.-L. Wu and M.-S. Yang. *Pattern Recognit*, **2002** *35*(10), pp 2267–2278.
51. L. A. García-Escudero, A. Gordaliza, C. Matrán, and A. Mayo-Iscar. *Adv. Data Anal. Classif*, **2010**, *4*, pp 89–109.
52. L. A. García-Escudero, A. Gordaliza, and C. Matrán. *J. Comput. Graph. Stat.*, **2003**, *12*, pp 434–449.
53. P. D'Urso, R. Massari, C. Cappelli, and L. De Giovanni. *Chemom. Intell. Lab. Syst.*, **2017**, *13*(5), pp 583–604.
54. D. Gustafson and W. Geurts van Kessel. Fuzzy Clustering with a Fuzzy Covariance Matrix, in *1978 IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes;* 1978, pp 761–766.
55. F. Klawonn, R. Kruse, and H. Timm. In *Learning, Networks and Statistics*, Della Riccia, G., Lenz, H., and Kruse, R., Eds.; Springer: New York, 1997.
56. D. Graves and W. Pedrycz. *Fuzzy Sets Syst*, **2010**, *161*(4), pp 522–543.
57. H. Frigui and O. Nasraoui. Simultaneous Clustering and Attribute Discrimination, in Ninth *IEEE International Conference on Fuzzy Systems (FUZZ- IEEE 2000)* vol. 1, 2000, pp 158–163.
58. R. Campello. *Pattern Recognit. Lett.*, **2007**, *28*(7), pp 833–841.
59. A. D. Gordon and M. Vichi. *Psychometrika*, **2001**, *66*(2), pp 229–247.
60. T. Tan, H. Suk, H. Hwang, and J. Lim. *Adv. Data Anal. Classif.*, **2013**, *7*(1), pp 57–82.
61. M. Yamamoto and Y. Terada. *Comput. Stat. Data Anal*, **2014**, *79*, pp 133–148.
62. P. D'Urso. *Fuzzy Clustering of Fuzzy Data;*, **2007**, pp 155–189.
63. R. J. Hathaway, J. C. Bezdek, and W. Pedrycz. *J. Classif.*, **1996**, *4*, pp 270–281.
64. W. Pedrycz, J. C. Bezdek, R. J. Hathaway, and G. W. Rogers. *IEEE Trans. Fuzzy Syst.*, **1998**, *6*(3), pp 411–425.
65. M. Yang and C. Ko. *Fuzzy Sets Syst.*, **1996**, *84*(1), pp 49–60.
66. M. Yang and H. Liu. *Fuzzy Sets Syst.*, **1999**, *106*(2), pp 189–200.
67. S. Auephanwiriyakul and J. Keller. *IEEE Trans. Fuzzy Syst.*, **2002**, *18*(10), pp 563–582.
68. M. Yang, P. Hwang, and D. Chen. *Fuzzy Sets Syst.*, **2004**, *141*(2), pp 301–317.
69. W.-L. Hung and M.-S. Yang. *Fuzzy Sets Syst.*, **2005**, *150*(3), pp 561–577.
70. P. D'Urso and P. Giordani. *Comput. Stat. Data Anal.*, **2006**, *50*(6), pp 1496–1523.
71. R. Coppi and P. D'Urso. *Stat. Methods Appl.*, **2002**, *11*(1), pp 21–40.
72. R. Coppi and P. D'Urso. *Comput. Stat. Data Anal.*, **2003**. *43*(2), pp 149–177.

73. P. D'Urso, L. De Giovanni, and P. Spagnoletti. *Int. J. Mach. Learn. Cybern.*, **2013**, *4*(5), pp 487–504.
74. P. D'Urso. *Granul. Comput.*, **2017**, *2*(4), pp 225–247.
75. P. D'Urso, M. Disegna, R. Massari, and G. Prayag. *Knowl. Based Syst.*, **2015**, *73*, pp 335–346.
76. P. D'Urso, M. Disegna, R. Massari, and L. Osti. *Tour. Manag.*, **2016**, *55*, pp 297–308.
77. Y. El-Sonbaty and M. Ismail. *IEEE Trans. Fuzzy Syst.*, **1998**, *6*(2), pp 195–204.
78. P. D'Urso and P. Giordani. *Comput. Stat.*, **2006**, *21*, pp 251–269.
79. P. D'Urso, R. Massari, L. De Giovanni, and C. Capelli. *Fuzzy Optim. Decis. Mak.*, **2017**, *16*(1), pp 51–70.
80. P. D'Urso and J. Leski. *Pattern Recognit.*, **2016**, *58*, pp 49–67.
81. P. D'Urso, L. De Giovanni, and R. Massari. *Adv. Data Anal. Classif.*, **2015**, *9*(1), pp 21–40.
82. M. Lee and W. Pedrycz. *Fuzzy Sets Syst.*, **2009**, *160*(24), pp 3590–3600.
83. L. Bai, J. Liang, C. Dang, and F. Cao. *Fuzzy Sets Syst.*, **2013**, *4*(3), pp 393–396.
84. H. Ralambondrainy. *Pattern Recognit. Lett.*, **1995**, *16*(11), pp 1147–1157.
85. D. Kim, K. Lee, and D. Lee. *Pattern Recognit. Lett.*, **2004**, *25*(11), pp 1263–1271.
86. M. Yang, H. Chiang, C. Chen, and C. Lai. *Fuzzy Sets Syst.*, **2008**, *159*(4), pp 390–405.
87. J. Deng, J. Hu, H. Chi, and W. J. An Improved Fuzzy Clustering Method for Text Mining, in *Second International Conference on Networks Security, Wireless Communications and Trusted Computing;* **2010**, pp 65–69.
88. P. D'Urso, D. Di Lallo, and E. Maharaj. *Soft Comput.*, **2013**, *17*, pp 83–131.
89. P. D'Urso, C. Cappelli, D. Di Lallo, and R. Massari. *Physica A*, **2013**, *392*, pp 2114–2129.
90. E. Maharaj, D. P., and D. Galagedera. *J. Classif.*, **2010**, *27*, pp 231–275.
91. P. D'Urso and E. Maharaj. *Fuzzy Sets Syst.*, **2012**, *193*, pp 33–61.
92. E. Maharaj and P. D'Urso. *Inf. Sci.*, **2011** *181*, pp 1187–1211.
93. R. Coppi and P. D'Urso. *Comput. Stat. Data Anal.*, **2006**, *50*(6), pp 1452–1477.
94. R. Coppi, P. D'Urso, and P. Giordani. Springer Berlin Heidelberg, 2004, pp 463–470.
95. R. Coppi, P. D'Urso, and P. Giordani. In *Modern Information Processing*, Bouchon-Meunier, B., Coletti, G., and Yager, R. R., Eds.; Elsevier Science: Amsterdam, 2006, pp 195–206.
96. P. D'Urso, E. Maharaj, and A. Alonso. *Fuzzy Sets Syst.*, **2017**, *318*, pp 56–79.
97. P. D'Urso, L. De Giovanni, and R. Massari. . *Fuzzy Sets Syst.*, **2016**, *305*, pp 1–28.
98. P. D'Urso, L. De Giovanni, and R. Massari. *Chemom. Intell. Lab. Syst.*, **2015**, *141*(C), pp 107–124.
99. J. Vilar, B. Lafuente-Rego, and P. D'Urso. *Fuzzy Sets Syst.*, **2017** *340*, pp 38–72.
100. J. Caiado, E. Maharaj, and P. D'Urso. In *Handbook of Cluster Analysis* C. Hennig, M. Meila, F. Murtagh, and R. Rocci, Eds, Chapman and Hall, 2015, page 241–263.
101. R. Coppi, P. D'Urso, and P. Giordani. *J. Classif*, **2010**, *27* pp 54–88.
102. A. Di Nola, V. Loia, and A. Stain. Genetic Spatial Based Clustering, in *The Ninth IEEE International Conference on Fuzzy Systems, 2000.*; **2000**, pp 953–956.
103. Y. A. Tolias and S. M. Panas. *IEEE Trans. Syst. Man Cybern. A Syst. Hum.*, **1998**, *28*(3), pp 359–369.
104. Y. A. Tolias and S. M. Panas. *IEEE Signal Process. Lett.*, **1998**, *5*(10), pp 245–247.
105. D. L. Pham and J. L. Prince. *IEEE Trans. Med. Imaging*, **1999**, *18*(9), pp 737–752.
106. A. W. C. Liew, S. H. Leung, and W. H. Lau. *IEE Proc. Vis. Image Signal Process.*, **2000**, *147*(2), pp 185–192.
107. A. W. C. Liew, S. H. Leung, and W. H. Lau. *IEEE Trans. Fuzzy Syst.*, **2003**, *11*(4), pp 542–549.
108. D. L. Pham. *Comput. Vis. Image Unders*, **2001**, *84*(2), pp 285–297.
109. A. W. C. Liew and H. Yan. *IEEE Trans. Med. Imaging*, **2003**, *22*(9), pp 1063–1075.
110. L. Cinque, G. Foresti, and L. Lombardi. *Pattern Recognit.*, **2004**, *37*, pp 1797–1807.
111. K. Chuang, H.-L. Tzeng, S. Chen, J. Wu, and T.-J. Chen. *Comput. Med. Imaging.*, **2006**, *30*, pp 9–15.
112. M. Disegna, , P. D'Urso, and F. Durante. *Spat. Stat.*, **2017**, *21*, pp 209–225.
113. M. Sato and Y. Sato. *Int. J. Uncertain. Fuzziness Knowl. Based Syst.*, **1994** *02*(02), pp 127–142.
114. M. Sato, Y. Sato, and L. C. Jain. *Fuzzy Clustering Models and Applications.*. Physica-Verlag, 1997.
115. P. D'Urso. *Int. J. Uncertain. Fuzziness Knowl-Based Syst.*, **2004**, *12*(03), pp 287–326.
116. P. D'Urso. *IEEE Trans. Fuzzy Syst.*, **2005**, *13*(5), pp 583–604.
117. P. D'Urso and M. Massari. *Fuzzy Sets Syst.*, **2013**, *215*, pp 29–54.
118. S. Tokushige, H. Yadohisa, and K. Inada. *Comput. Stat.*, **2007** *22*(1), 1–16.
119. J. Liu. *Pattern Recognit.*, **2010**, *43*, 1334–1345.
120. M. Yang and P. J.A. *Fuzzy Sets Syst.*, **1997**, *91*(3), pp 319–326.
121. S. Chatzis. *Expert Syst. Appl.*, **2011**, *38*(7), pp 8684–8689.
122. J. Ji, C. Zhou, T. Bai, J. Zhao, and Z. Wang. *Adv. Inf. Sci. Serv. Sci.*, **2012**, *4*(7), pp 256–264.
123. J. Ji, W. Pang, C. Zhou, X. Han, and Z. Wang. *Knowl. Based Syst.*, **2012**, *30*(C), pp 129–135.
124. R. J. Hathaway and J. C. Bezdek. *Trans. Syst. Man Cybern. B*, **2001**, *31*(5), pp 735–744.
125. R. J. Hathaway and J. C. Bezdek. *Pattern Recognit. Lett.*, **2002**, *23*(1–3), pp 151–160.
126. D.-Q. Zhang and S.-C. Chen. *Neural Process. Lett.*, **2003**, *18*(3), pp 155–162.
127. H. Timm, C. Borgelt, C. Döring, and R. Kruse. *Fuzzy Sets Syst.*, **2004**, *147*(1), pp 3–16.
128. K. Honda and H. Ichihashi. *IEEE Trans. Fuzzy Syst.*, **2004**, *12*(2), pp 183–193.
129. T. Havens, J. Bezdek, C. Leckie, L. Hall, and M. Palaniswami. *IEEE Trans. Fuzzy Syst*, **2012**, *20*(6), pp 1130–1146.
130. P. Huber. *Massive Data Sets: Proceedings of a Workshop*. National Academies Press: Washington DC, 1997, pp 169–184.
131. R. J. Hathaway and J. C. Bezdek. *Comput. Stat. Data Anal.*, **2006**, *51*(1), pp 215–234.

132. P. Hore, L. O. Hall, and D. B. Goldgof. Single Pass Fuzzy c-Means, in *2007 IEEE International Fuzzy Systems Conference;* 2007, pp 1–7.
133. P. Hore, L. O. Hall, D. B. Goldgof, Y. Gu, A. A. Maudsley, and A. Darkazanli. *J. Signal Process. Syst.*, **2009**, *54*(1–3), pp 183–203.
134. S. Eschrich, J. Ke, L. O. Hall, and D. B. Goldgof. *Trans. Fuzzy Syst.*, **2003**, *11*(2), pp 262–270.
135. R. Chitta, R. Jin, T. Havens, and A. Jain. Approximate Kernel k-Means: Solution to Large Scale Kernel Clustering, in *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining;* 2011, pp 895–903.
136. R. Chitta, R. Jin, T. C. Havens, and A. K. Jain. Approximate Kernel k-Means: Solution to Large Scale Kernel Clustering, in *Proc. of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining;* **2011**, pp 895–903.
137. B. U. Shankar and N. Pal. FFCM: An effective approach for large data sets, in *Proc. of the Third International Conference on Fuzzy Logic, Neural Nets and Soft Computing*, Iizuka, Japan; August 1994, pp 331–332.
138. T. W. Cheng, D. B. Goldgof, and L. O. Hall. Fast Clustering with Application to Fuzzy Rule Generation in *Proc. of 1995 IEEE International Conference on Fuzzy Systems;* vol. 4, 1995, pp 2289–2295.
139. J. F. Kolen and T. Hutcheson. *IEEE Trans. Fuzzy Syst.*, **2002**, *10*(2), pp 263–267.
140. R. Cannon, J. Dave, and J. Bezdek. *IEEE Trans. Pattern Anal. Mach. Intell.*, **1986**, PAMI-*8*(2), pp 248–255.
141. L. Liao and T. Lin. A Fast Spatial Constrained Fuzzy Kernel Clustering Algorithm for MRI Brain Image Segmentation, in *2007 International Conference on Wavelet Analysis and Pattern Recognition*, vol. 1; 2007, pp 82–87.
142. W. Pedrycz. *IEEE Trans. Syst. Man Cybern. B (Cybern.)*, **1998**, *28*(1), pp 103–109.
143. P. Lingras and C. West. *J. Intell. Inf. Syst.*, **2004**, *23*(1), pp 5–16.
144. W.-L. Hung, J.-S. Lee, and C.-D. Fuh. *Int. J. Uncertain. Fuzziness Knowl. Based Syst.*, **2004**, *12*, pp 513–530.
145. T. Denoeux and M. H. Masson. *IEEE Trans. Syst. Man Cybern B (Cybern.)*, **2004**, *34*(1), pp 95–109.
146. J. Zhou, C. Hung, X. Wang, and S. Chen. Fuzzy clustering based on credibility measure, in *Proc. of the Sixth International Conference on Information and Management Sciences*; Lhasa, China, 2007, pp 404–411.
147. C. Hwang and F. C.-H. Rhee. *IEEE Trans. Fuzzy Syst.*, **2007**, *15*(1), pp 107–120.
148. J. Shan, H. D. Cheng, and Y. Wang. *Med. Phys*, **2012**, *39*(9), pp 5669–5682.
149. N. Chen, Z. Xu, and M. Xia. *Appl. Math*, **2014**, *29*(1), 1–17.
150. L. Silva, R. Moura, A. Canuto, R. Santiago, and B. Bedregal. New Ways to Calculate Centers for Interval Data in Fuzzy Clustering Algorithms, in *2014 IEEE Conference on Norbert Wiener in the 21st Century (21CW)*; 2014, pp 1–6.
151. L. H. Son. *Expert Syst. Appl.*, **2015**, *42*(1), pp 51–66.
152. P. D'Urso. *Inf. Sci*, **2010**, *400*(C), pp 30–62.
153. J. Banfield and A. Raftery. *Biometrics*, **1993**, *49*(3), pp 803–821.
154. C. Fraley and A. E. Raftery. *Comput. J.*, **1988**, *41*(8), pp 578–588.
155. H. Bock. *Comput. Stat. Data Anal.*, **1996**, *23*(1), pp 5–28.
156. H. Bock. *Bull. Int. Stat. Inst.*, **1998**, *57*, pp 603–606.
157. G. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. John Wiley & Sons, Inc.: New York, 1992.
158. G. J. McLachlan and K. Basford. *Mixture Models: Inference and Applications to Clustering*. John Wiley & Sons, Ltd: Chichester, 1999.
159. C. Fraley and A. E. Raftery. *J. Am. Stat. Assoc.*, **2002**, *97*(458), pp 611–631.
160. G. McLachlan and S. Rathnayake. In *Handbook of Cluster Analysis*, Hennig, C., Meila, M., Murtagh, F., and Rocci R., Eds.; Chapman and Hall, 2015, pp 145–172.
161. M. Alfó and S. Viviani. In *Handbook of Cluster Analysis*, Hennig, C., Meila, M., Murtagh, F., and Rocci, R., Eds.; Chapman and Hall, 2015; pp 217–240.
162. V. Mountcastle. *J. Neurophysiol.*, **1957** *20*(4), pp 408–434.
163. S. Grossberg. *Biol. Cybern.*, **1976**, *21*(3), pp 145–159.
164. G. Milligan and M. Cooper. *Biol. Cybern.*, **1975**, *19*, pp 1–18.
165. C. v. d. Malsburg. *Kybernetiky*, **1973**, *14*(1), pp 85–100.
166. S.-I. Amari. *Bull. Math. Biol.*, **1980**, *42*(3), pp 339–364.
167. M. Oja, S. Kaski, and T. Kohonen. *Neural Comput. Surv.*, **2003**, *3*, pp 1–156.
168. T. Kohonen. *Self-Organization and Associative Memory* 3rd ed. Springer: Berlin, 1989.
169. S. Kaski. In *Encyclopedia of Machine Learning and Data Mining*, Springer, 2017; pp 1129–1132.
170. E. Oja and S. Kaski. *Kohonen Maps*. Elsevier Science B.V.: Amsterdam, 1999.
171. T. Kohonen. *Proc. IEEE*, **1990**, *78*(9), pp 1464–1480.
172. H. Bauer, M. Herrmann, and T. Villmann. *Neural Netw.*, **1999**, *12*(4), pp 659–676.
173. P. Conti and L. De Giovanni. On the Mathematical Treatment of Self Organization: Extension of Some Classical Results, in *International Conference on Artificial Neural Networks;* **1991**, pp 1809–1812.
174. H. Ritter and K. Schulten. *Biol. Cybern.*, **1988**, *60*(1), pp 59–71.
175. H. Ritter and K. Schulten. Kohonen's Self-Organizing Maps: Exploring their Computational Capabilities, in *Proc. of IEEE International Conference on Neural Networks*; 1988, pp 109–116.
176. H. Ritter and K. Schulten. *Biol. Cybern.*, **1986**, *54*, pp 99–106.
177. J. C. Fort. *Neural Netw*, **2006**, *19*(6), pp 812–816.
178. M. Budinich and J. Taylor. *Neural Comput.*, **1995**, *7*(2), pp 284–289.
179. H. Ritter. *IEEE Trans. Neural Netw.*, **1991** *2*(1), pp 173–175.
180. E. Erwin, K. Obermayer, and K. Schulten. *Biol. Cybern.*, **1992**, *67*(1), pp 47–55.
181. S. Kaski, T. Honkela, K. Lagus, and T. Kohonen. *Neurocomputing*, **1998**, *21*(1-3), pp 101–117.
182. J. Kangas, T. Kohonen, and J. Laaksonen. *IEEE Trans. Neural Netw.*, **1990**, *1*(1), pp 93–99.
183. E. Jang, C. Fyfe, and H. Ko. *Bregman Divergences and the Self-Organising Map.* Springer: Berlin Heidelberg, 2008; pp 452–458.
184. T. Villmann and S. Haase. *Neural Comput.*, **2011**, *23*(5), pp 1343–1392.
185. H. Yin and N. Allinson. *IEEE Trans. Neural Netw*, **2001**, *12*(2), pp 405–411.
186. D. MacDonald and C. Fyfe. The Kernel Self-Organising Map, in *Proceedings of the Fourth International Conference on*

Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000, vol. 1; 2000, pp 317–320.

187. A. Rauber, D. Merkl, and M. Dittenbach. *IEEE Trans. Neural Netw.*, **2002**, *13*(6), pp 1331–1341.
188. P. D'Urso and L. De Giovanni. *Appl. Soft Comput.*, **2011**, *11*(5), pp 3877–3886.
189. S. Kaski, J. Kangas, and T. Kohonen. *Neural Comput. Surv.*, **1998**, *1*, pp 102–350.
190. M. Polla, T. Honkela, and T. Kohonen. Bibliography of Self-Organizing Map (SOM) Papers: 2002–2005 addendum. Technical report, Helsinki University of Technology, TKK Reports in Information and Computer Science, TKK-ICS-R23, 2009.
191. P. D'Urso, L. De Giovanni, and R. Massari. *Fuzzy Sets Syst.*, **2014**, *237*, pp 63–89.
192. P. D'Urso, L. De Giovanni, E. Maharaj, and R. Massari. *J. Chemom.*, **2014**, *28*(1), pp 28–51.
193. P. D'Urso and L. De Giovanni. *Neurocomputing*, **2008**, *71*(13-15), pp 2880–2892.

PIERPAOLO D'URSO
Sapienza University of Rome, Rome, Italy

LIVIA DE GIOVANNI
LUISS Guido Carli University, Rome, Italy