A new topological clustering algorithm for interval data

Guénaël Cabanes, Younès Bennani, Renaud Destenay, André Hardy

A new topological clustering algorithm for interval data

Guénaël Cabanes[a,*], Younès Bennani[a], Renaud Destenay[b], André Hardy[b]

[a] *LIPN-CNRS, UMR 7030*
Université Paris 13, 99 Avenue J-B. Clément, 93430 Villetaneuse, France
[b] *Department of Mathematics. FUNDP*
University of Namur, 8 Rempart de la Vierge. B 5000 Namur, Belgium

Abstract

Clustering is a very powerful tool for automatic detection of relevant sub-groups in unlabeled data sets. In this paper we focus on interval data: i.e. where the objects are defined as hyper-rectangles. We propose here a new clustering algorithm for interval data, based on the learning of a Self Organizing Map. The major advantage of our approach is that the number of clusters to find is determined automatically; no a priori hypothesis for the number of clusters is required. Experimental results confirm the effectiveness of the proposed algorithm when applied to interval data.

Keywords: Interval data, Clustering, Self-Organizing Map

1. Introduction

Unsupervised classification, or clustering, is a very powerful tool for automatic detection of relevant sub-groups (or clusters) in unlabeled data sets, when one does not have prior knowledge about the underlying structure of these data. Patterns in the same cluster should be similar to each other, while patterns in different clusters should not (internal homogeneity and external separation). Clustering plays an indispensable role for understanding various phenomena described by data sets and is considered as one of the most challenging tasks

*Corresponding author
Email address: guenael.cabanes@lipn.univ-paris13.fr (Guénaël Cabanes)

in unsupervised learning. Various approaches have been proposed to solve the problem [**?** **?** **?** **?** **?**].

However, most clustering algorithms are defined to deal with vectorial data in $\mathbb{R}^d$. This kind of representation is frequently used to analyze data from physical measurements, counts or indices, but there are many other kinds of information that can not be described with vectors. This is the case of complex data described for example with a text, a picture or a hierarchical structure. In this paper we focus on interval data (also known as symbolic interval data). In a vectorial space, interval data are defined by hyper-rectangles. A given data x is thus defined as a closed and bounded interval in $\mathbb{R}^d$, characterized by two vectors, the lower bound ($x_l = [x_{l1}, \ldots, x_{ld}]$) and the upper bound ($x_u = [x_{u1}, \ldots, x_{ud}]$), such that $\forall j \in [1, \ldots, d], x_{lj} \leqslant x_{uj}$. Intervals are often used to model quantities which vary between two bounds, upper and lower, without further assumptions on the distribution between these bounds [**?** **?** **?** **?**].Several clustering methods are available for interval variables. For example, [**?**] presented an iterative relocation algorithm to partition a set of symbolic objects into classes so as to minimize the sum of the description potential of the classes. [**?**] proposed partitioning clustering methods for interval data based on city-block distances. SCLUST [**?**] is a partitioning clustering method and a symbolic extension of the well-known Dynamical Clustering method [**?**]. DIV [**?**] is a symbolic hierarchical monothetic divisive clustering procedure based on the extension of the within class sum-of-squares criterion. SCLASS [**?**] and SPART [**?**] are symbolic hierarchical monothetic divisive methods based on the generalized Hypervolumes clustering criterion. Hardy [**?**] developed a module called SHICLUST containing the symbolic extensions of four well-known classic hierarchical clustering methods: the single linkage, complete linkage, centroid and Ward methods. The corresponding aggregation indices used the L_1, L_2, Hausdorff and De Carvalho [**?**] dissimilarity measures [**?** **?**]. The hierarchical component of Hipyr [**?**] also contains extensions of the four classic hierarchical clustering methods. Other clustering methods for interval data can be found in [**?** **?** **?** **?** **?** **?**].

We present here a new clustering algorithm for interval data, based on the learning of a Self Organizing Map (SOM) [?]. This unsupervised learning algorithm is a popular nonlinear technique for dimensionality reduction and data visualization, with a very low computational cost. It can be seen as a K-means algorithm with topological constraints, usually with a better overall clustering performance [?]. Bock [? ?] proposed a visualization of symbolic interval data by constructing a SOM. In the SODAS software [?], such a map is constructed in the SYKSOM module. SYKSOM assumes a data table of n items that are described by p interval-type variables. The n items are first clustered into a smaller number of mini-clusters (reduction step), and these mini-clusters are then assigned to the vertices of a fixed, prespecified rectangular lattice $\mathcal{L}$ of points in the plane such that similar clusters (in the original data space) are represented by neighboring vertices in the lattice $\mathcal{L}$. Other papers concerning SOM algorithms for interval-valued data can be found in the literature [? ? ? ? ? ?]. For example, [?] uses a distance based on Hadamard product and [?] proposes a fuzzy representation based on Gowda and Diday's dissimilarity measure [?]. All these algorithms can be seen as vector quantization and visualization tools for symbolic interval data, and cannot be used directly to obtain a clustering of the data.

The proposed algorithm is a two-level clustering method for interval data. The key idea of the two-level clustering approach based on SOM is to combine the dimension reduction and the fast learning capabilities of SOM in the first level to construct a new reduced space, then to apply a clustering method in this new space to produce a final set of clusters in the second level (see [? ?] for examples with vectorial data). The two-level methods are known to reduce the computational time and allow a visual interpretation of the clustering results [?]. In particular, the use of SOM+K-means or SOM+Hierarchical clustering gives better results than the use of K-means or a Hierarchical clustering alone [? ?]. The major advantage of the new algorithm in comparison to existing methods is that the number of clusters to find is detected automatically, i.e., no a priori hypothesis for the number of clusters is required. This problem,

also known as the model selection problem, is one of the most challenging in clustering. Indeed, the existing clustering algorithms for interval data need to have the number of clusters as a user-given parameter [**?** **?** **?** **?**], which is usually very difficult to determine a priori.

The remainder of this paper is organized as follows. Section 2 presents an adaptation of SOM allowing an automatic two-level clustering. Section 3 describes the new algorithm for interval data. In section 4 we present the experimental protocol and results are shown in section 5. In section **??**, we compare the new algorithm with existing methods on artificial and real datasets. Conclusions are given in section **??**.

2. Simultaneous Two-Level clustering of Self-Organizing Map

Kohonen's Self-Organizing Map (SOM) can be described as a competitive unsupervised learning neural network [**?**]. When an observation is recognized, the activation of an output cell - competition layer - inhibits the activation of other neurons and reinforces itself. It is said that it follows the so called "Winner Takes All" rule. Actually, neurons are specialized in the recognition of one kind of observation. A SOM often consists of a two-dimensional map of neurons which are connected to n inputs according to n weight connections $w^j = (w_1^j, ..., w_d^j)$ and to their neighbors with topological links. A training set is used to organize these maps under topological constraints of the input space. Thus, a mapping between the input space and the network space is constructed; two close observations in the input space would activate two close units of the SOM. An optimal spatial organization is determined by the SOM from the input data, and when the dimension of the input space is lower than three, both the position of weight vectors and direct neighborhood relations between cells can be represented visually. Thus, a visual inspection of the map provides qualitative information about the map and the choice of its architecture. The winner neuron updates its prototype vector, making it more sensitive for later presentation of that type of input. This allows different cells to be trained for different types of

data. To achieve a topological mapping, the neighbors of the winner neuron can adjust their prototype vector towards the input vector as well, but at a lesser degree, depending on how far away they are from the winner. Usually a radial symmetric Gaussian neighborhood function K_{ij}, between two neurons i and j, is used for this purpose.

The key idea of the two-level clustering approach based on SOM is to combine the dimension reduction and the fast learning capabilities of SOM in the first level to construct a new reduced vector space, and to apply another clustering method in this new space to produce a final set of clusters in the second level. Although the two-level methods are more interesting than the traditional approaches (in particular by reducing the computational time and by allowing a visual interpretation of the partition result [**?** **?** **?** **?**]), the data segmentation obtained from the SOM is not optimal, since part of the information is lost during the first stage (dimension reduction). Moreover, this separation in two stages is not suited for a dynamic (incremental) segmentation of data which move in time, in spite of important needs for analysis tools for this type of data. The S2L-SOM algorithm (Simultaneous Two-Levels -SOM, [**?**]) has been proposed to overcome these problems by simultaneous performing learning and clustering of the SOM from data information.

2.1. The S2L-SOM algorithm

In the S2L-SOM algorithm, it is proposed to associate to each neighborhood connection a real value ν_{ij} which indicates the relevance of the connected neurons i and j. This value is representative of the data distribution between i and j, and can be viewed as the number of data having i and j as the two best representatives neurons. Given the organization constraint of the SOM, both closest prototypes of each data must be connected by a topological connection. This connection "will be rewarded" by an increase of its value, whereas all other connections from the winner neuron "are punished" by a reduction of their values. The values of ν will be used to create sets of connected prototypes; each set not connected to the others is representative of one cluster. Thus, at the end

of the training, a set of inter-connected prototypes will be an artificial image of a relevant sub-group of the whole data set.

Connectionist learning is often presented as a minimization of a cost function. In our case, it will be carried out by the minimization of the distance between the input samples and the map prototypes, weighted by a neighborhood function K_{ij}. To do that, we use a gradient algorithm [**?**]. The cost function to be minimized is defined by:

$$\tilde{R}(w) = \frac{1}{N} \sum_{k=1}^{N} \sum_{j=1}^{M} K_{j,u^*(x^{(k)})} \parallel w^j - x^{(k)} \parallel^2 \tag{1}$$

where N represents the number of learning samples, M the number of neurons in the map, $u^*(x^{(k)})$ is the neuron having the closest weight vector to the input pattern $x^{(k)}$, and K_{ij} is a positive symmetric kernel function: the neighborhood function [**?**]. The relative importance of a neuron i compared to a neuron j is weighted by the value of the kernel function K_{ij} which can be defined as:

$$K_{ij} = \frac{1}{\lambda(t)} \times e^{-\frac{d_1^2(i,j)}{\lambda^2(t)}} \tag{2}$$

where $\lambda(t)$ is the temperature function modeling the topological neighborhood extent, defined as:

$$\lambda(t) = \lambda_i (\frac{\lambda_f}{\lambda_i})^{\frac{t}{t_{max}}} \tag{3}$$

where λ_i and λ_f are the initial and the final temperature respectively. t_{max} is the maximum number allocated to the time (number of iterations for the x learning sample). $d_1(i,j)$ is the Manhattan distance defined between two neurons i and j on the map grid, with coordinates (k,m) and (r,s) respectively:

$$d_1(i,j) = \parallel r - k \parallel + \parallel s - m \parallel . \tag{4}$$

The S2L-SOM training process is highly similar to the Competitive Hebbian Learning (CHL) approach [**?**]. The difference lies in that the CHL method does not change reference vectors at all (which could be interpreted as having a zero learning rate), it only generates a number of neighborhood edges between the units of the SOM without values. In the S2L-SOM we associate a real value

to each neighborhood connection, which indicates the relevance of the connected couple for the local neuron prototype vector. The value of this connection is adapted during the learning process. It was proved by Martinetz [**?**] that the so generated graph is optimally topology-preserving in a very general sense. In particular each edge of this graph belongs to the Delaunay triangulation [**?**] corresponding to the given set of reference vectors. The S2L-SOM learning algorithm proceeds essentially in three phases:

1. **Initialization Step** :
   - Define the topology of the SOM.
   - Initialize the prototypes w^j.
   - Initialize to 0 connection values ν_{ij} between each pair of prototypes i and j.
2. **Competition Step** :
   - Present a data $x^{(k)}$ randomly chosen.
   - Among the M prototypes, choose $u^*(x^{(k)})$ and $u^{**}(x^{(k)})$ the two best representatives for this data according to a distance metric d (usually the Euclidean distance):

$$u^*(x^{(k)}) = \underset{1\leq i\leq M}{Argmin}\, d(x^{(k)}, w^i)$$
$$u^{**}(x^{(k)}) = \underset{1\leq i\leq M,\ i\neq u^*}{Argmin}\ d(x^{(k)}, w^i).$$

   - Increase the connection value between $u^*(x^{(k)})$ and $u^{**}(x^{(k)})$, according to the learning rate $\varepsilon(t)$, a decreasing function of time between $[0, 1]$, proportional to $1/t$ (see [**?**]). Decrease the values of other connections involving $u^*(x^{(k)})$:

$$\nu_{u^*u^{**}}(t) = \nu_{u^*u^{**}}(t-1) - \varepsilon(t)r(t)\left(\nu_{u^*u^{**}}(t-1) - 1\right)$$
$$\nu_{u^*i}(t) = \nu_{u^*i}(t-1) - \varepsilon(t)r(t)\left(\nu_{u^*i}(t-1)\right)\ \forall i \in \text{ neighbors of } u^*$$

with :

$$r(t) = \frac{1}{1 + e^{-\left(\frac{t}{t_{max}}\right)}}.$$

3. **Adaptation step** :

   - Update prototypes w^j:

$$w^j(t) = w^j(t-1) - \varepsilon(t) K_{ju^*(x^{(k)})}(w^j(t-1) - x^{(k)}).$$

4. **Repeat step 2 and 3** until $t = t_{tmax}$.
5. **Final clustering** : Determine $P = \{C_i\}_{i=1,\ldots,L}$, the set of L groups of connected prototypes such that $\nu > 0$. Each data $x^{(k)}$ belongs to the cluster of $u^*(x^{(k)})$. The final prototype clustering provides a disjoint data clustering, as each data point is represented by an unique prototype u^*.

3. A new two-level clustering algorithm for interval data

In this paper we propose an extension of our previous algorithm S2L-SOM to interval data. The main idea is to define a prototype w^j of the SOM as an interval, i.e. a pair of vectors: lower and upper bounds.

$$w^j = [w_l^j, w_u^j] \in \mathbb{I} = \{[a, b] | a, b \in \mathbb{R}^d, a_i \leqslant b_i, i = 1, \ldots, d\} \tag{5}$$

During the learning of the SOM, the two bounds of each prototype will be updated to improve data representation (see Figure 2 for an example).

The algorithm works as follows:

Let $X = \{x^{(i)}\}_{i=1\ldots N}$ be a set of interval data where $x^{(i)} = [x_l^{(i)}, x_u^{(i)}]$, $d(x, y)$ is a distance between two intervals (see section 4.1) and t_{max} the maximum number of iterations.

1. **Initialization Step** :

   - As in 2.1, but the prototypes are now intervals: $w^j = [w_l^j, w_u^j]$.

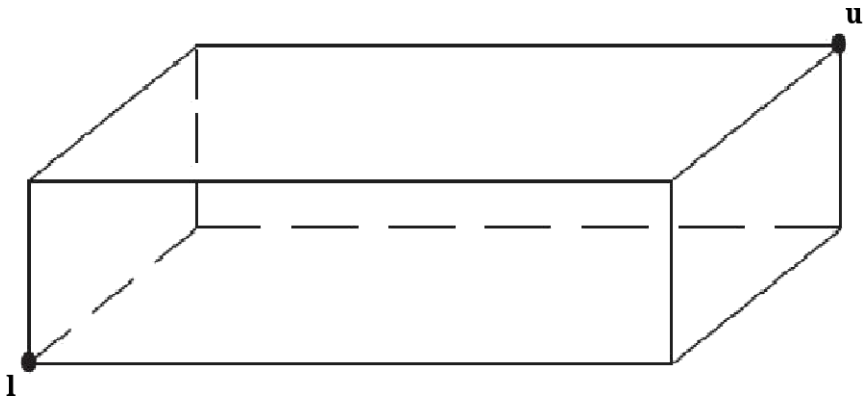


Figure 1: lower (l) and upper (u) bounds of an interval data in 3 dimensions.

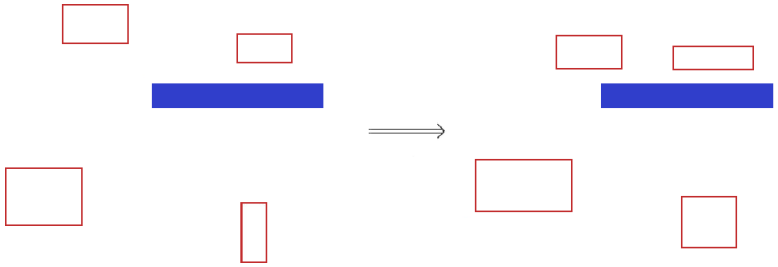


Figure 2: Example of prototypes update (red intervals) when a data (in plain blue) is presented.

2. **Competition Step** :

   - As in 2.1, using an adapted distance metric d (see section 4.1) instead of the Euclidean distance.

3. **Adaptation step** :

   - Update prototype bounds w^j:

$$w_l^j(t) = w_l^j(t-1) - \varepsilon(t) K_{ju^*(x^{(k)})}(w_l^j(t-1) - x_l^{(k)})$$

$$w_u^j(t) = w_u^j(t-1) - \varepsilon(t) K_{ju^*(x^{(k)})}(w_u^j(t-1) - x_u^{(k)}).$$

4. **Repeat step 2 and 3** until $t = t_{max}$.

5. **Final clustering** : Determine $P = \{C_i\}_{i=1,...,L}$, the set of L groups of connected prototypes such as $\nu > 0$. Each data $x^{(k)}$ belongs to the cluster of $u^*(x^{(k)})$.

In this study we used the default parameters of the SOM Toolbox [**?**] for the learning of the SOM and we use $t_{max} = max(N, 50 \times M)$ as in [**?**]. One can note that the lower and upper bounds of the prototypes are always updated toward respectively a lower and upper bound of a data point. In that case, no inversion is possible and the prototype's upper bound is always greater or equal to the lower bound. At the end of the clustering process, a cluster is a set of prototypes which are linked together by neighborhood connections with positive values. Thus, the number of clusters is determined automatically.

4. Experimental Protocol

We propose to test the effectiveness of the new algorithm for three types of distances between interval data and three types of initialization of the prototypes.

4.1. Distances measures

Many distance measures have been defined to compare intervals [**?**]. For this work, we have selected three distances that require little computational power: the distance to vertices and the Hausdorff-type L1 and L2 distances (noted L1 and L2 in the paper). Let d be the dimension of the data representation space (i.e. the number of interval variables). The distance to vertices $d_S(x, x')$ is proportional to the sum of the Euclidean distances between the two upper bounds and the two lower bounds:

$$d_S(x, x') = 2^{d-1} \left(\| x_l - x'_l \|^2 + \| x_u - x'_u \|^2 \right).$$

L1 distance is defined by:

$$d_{L1}(x, x') = \sum_{j=1}^{d} max\{|x_{lj} - x'_{lj}|, |x_{uj} - x'_{uj}|\}.$$

L2 distance is defined by:

$$d_{L2}(x, x') = \sqrt{\sum_{j=1}^{d} [max\{|x_{lj} - x'_{lj}|, |x_{uj} - x'_{uj}|\}]^2}.$$

Figure 3 shows an example of measure of distance L1 and L2.

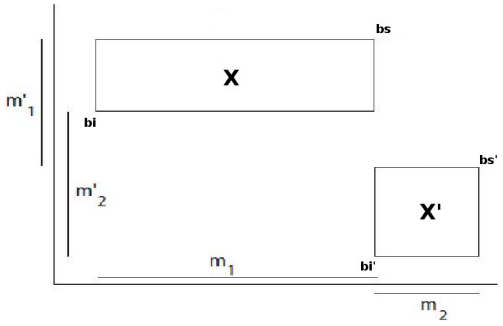


Figure 3: Example of measure of distance L1 and L2. Here $d_{L1}(x, x') = m_1 + m'_2$ and $d_{L2}(x, x') = \sqrt{{m_1}^2 + {m'_2}^2}$.

4.2. Initialization of the prototypes

Three types of initialization have been tested. The "data-based" initialization randomly selects M data points without repetition as initial prototypes.

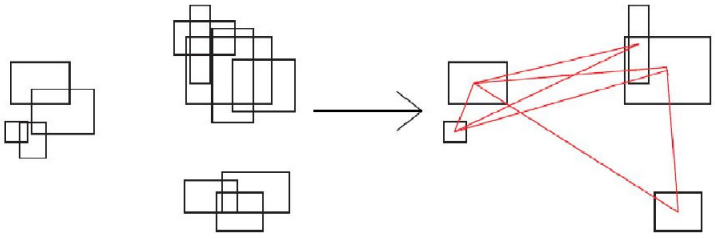


Figure 4: Example of "data-based" initialization.

"Points" initialization first determines the smallest hyper-rectangle enclosing all the data points. Then M points are selected randomly in this hyper-rectangle. It may be noted that a point is a hyper-rectangle having the lower and upper bounds equal.

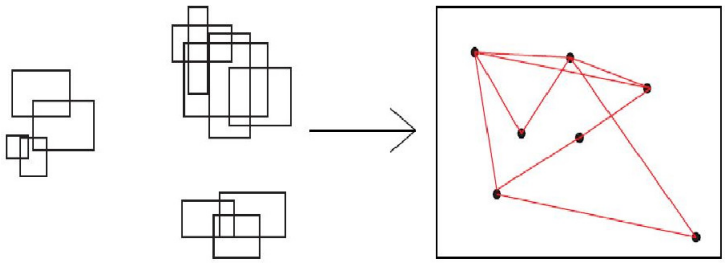


Figure 5: Example of "Points" initialization.

The last initialization, "Linear", also uses points as initial prototypes. To determine their positions, a Principal Component Analysis (PCA) [**?**] is performed from the position of the centers of the data hyper-rectangles. Prototypes are placed linearly on the plane defined by the two first components (see Figure 6). The main difference with the previous two initializations is the fact that the map generated in this way is already organized.

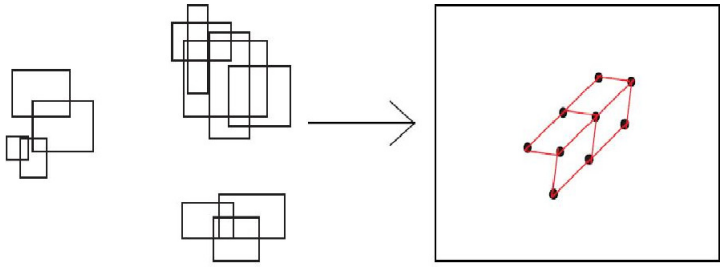


Figure 6: Example of "linear" initialization. On the right one can see that the map is already organized.

4.3. Experimental datasets

In this paper the quality of clustering was evaluated using two external criteria (Jaccard index [**?**] and Corrected Rand index [**?**]); both are frequently used [**?**]. These indices are used if data-independent labels (categories) are available, to evaluate how well a given cluster solution corresponds to these external labels. The main difference between Jaccard and Rand is that the Jaccard index computes the proportion of pairs of objects where both objects

Database Name	Number of data points	Number of clusters	Number of variables	Clusters properties
2Dim	200	2	2	Convex clusters, well separated
3Dim	400	4	3	Convex clusters, well separated
5Dim	400	4	5	Convex clusters, well separated, different sizes
Sun	195	5	2	Convex clusters, well separated, different shapes
Hooks	60	6	2	Convex clusters, touching intervals
Cross	60	3	2	Convex clusters, overlapping intervals
Target	250	2	2	Non Convex clusters, non linear separations

Table 1: Description of the seven artificial datasets

belong to the same label and to the same cluster, whereas the Rand index also takes into account the number of pairs of objects where both objects belong to different labels and to different clusters. Jaccard index gives values between 0 (no match) and 1 (perfect match). Corrected Rand index gives values between -1 and 1. We launched the algorithm 100 times on each database, then we computed the mean and the standard deviation values of the indices to check the validity of the results.

To test the new algorithm, we created seven artificial datasets of various difficulty (Table 1).

"2Dim" consists of two groups of 200 data points each, separated linearly in two dimensions. Each group is in a square and the number of data in each group is the same. "3Dim" consists of four groups of equal size arranged on the vertices of a tetrahedron in a three-dimensional space. In "5Dim", the groups are of various shapes and sizes in a space with five dimensions.

The data "Sun" consists of five classes oriented in different ways. This set contains 195 data points in two dimensions. The data set "Hooks" consists of

six groups of intervals which are in contact with each other. "Cross" consists of three intersecting groups, with a group of squares around the center, a group of elongated vertical intervals and the last interval elongated horizontally. Finally, "Target" consists of squares of equal sizes, divided into two groups, one forming a ring and the second at the center of this ring.

5. Results

5.1. Quality

All the results for the three distances and the three initializations are summarized in Tables 2 and **??**.

Jaccard	Linear			Data-based			Points		
	Vertexes	L1	L2	Vertexes	L1	L2	Vertexes	L1	L2
Sun	0.990 (0.016)	0.998 (0.005)	0.989 (0.019)	0.971 (0.038)	0.977 (0.030)	0.972 (0.038)	0.979 (0.026)	0.972 (0.046)	0.974 (0.038)
Hooks	1.000 (0.000)	1.000 (0.000)	1.000 (0.000)	0.909 (0.140)	0.884 (0.126)	0.931 (0.108)	0.709 (0.166)	0.705 (0.164)	0.693 (0.159)
Cross	0.986 (0.056)	0.989 (0.041)	0.972 (0.075)	0.966 (0.067)	0.973 (0.060)	0.971 (0.062)	0.988 (0.061)	0.989 (0.059)	0.977 (0.110)
Target	0.988 (0.074)	0.990 (0.060)	0.987 (0.067)	0.663 (0.227)	0.628 (0.227)	0.669 (0.230)	0.641 (0.222)	0.591 (0.204)	0.664 (0.248)
2Dim	0.971 (0.068)	0.935 (0.091)	0.913 (0.125)	0.622 (0.187)	0.783 (0.168)	0.672 (0.102)	0.996 (0.013)	0.909 (0.100)	0.940 (0.122)
3Dim	0.994 (0.012)	0.981 (0.075)	0.994 (0.011)	0.993 (0.018)	0.979 (0.066)	0.992 (0.017)	0.991 (0.036)	0.975 (0.070)	0.992 (0.017)
5Dim	0.990 (0.015)	0.993 (0.011)	0.994 (0.012)	0.954 (0.086)	0.950 (0.087)	0.952 (0.087)	0.982 (0.045)	0.967 (0.064)	0.971 (0.064)
Total	**0.988 (0.034)**	**0.984 (0.040)**	**0.978 (0.044)**	**0.868 (0.109)**	**0.882 (0.109)**	**0.880 (0.092)**	**0.898 (0.081)**	**0.873 (0.101)**	**0.887 (0.108)**

Table 2: Clustering quality using the Jaccard index for each database with different initialization and different measures of distances between intervals: mean (standard deviation). The Total row is the global mean quality.

Rand	Linear			Data-based			Points		
	Vertexes	L1	L2	Vertexes	L1	L2	Vertexes	L1	L2
Sun	0.997 (0.004)	0.999 (0.001)	0.997 (0.005)	0.992 (0.010)	0.995 (0.007)	0.993 (0.010)	0.995 (0.006)	0.993 (0.013)	0.994 (0.010)
Hooks	1.000 (0.000)	1.000 (0.000)	1.000 (0.000)	0.980 (0.037)	0.977 (0.026)	0.987 (0.022)	0.926 (0.056)	0.926 (0.056)	0.920 (0.065)
Cross	0.994 (0.025)	0.996 (0.013)	0.989 (0.035)	0.989 (0.021)	0.991 (0.019)	0.991 (0.020)	0.994 (0.032)	0.995 (0.032)	0.982 (0.098)
Target	0.993 (0.044)	0.992 (0.054)	0.992 (0.040)	0.790 (0.148)	0.764 (0.154)	0.798 (0.144)	0.774 (0.146)	0.721 (0.159)	0.794 (0.155)
2Dim	0.986 (0.033)	0.968 (0.045)	0.958 (0.061)	0.815 (0.092)	0.894 (0.083)	0.839 (0.050)	0.998 (0.006)	0.955 (0.049)	0.971 (0.060)
3Dim	0.999 (0.003)	0.993 (0.028)	0.999 (0.003)	0.998 (0.004)	0.993 (0.024)	0.998 (0.004)	0.997 (0.013)	0.992 (0.025)	0.998 (0.004)
5Dim	0.997 (0.004)	0.998 (0.003)	0.998 (0.003)	0.988 (0.022)	0.987 (0.023)	0.988 (0.023)	0.995 (0.012)	0.991 (0.016)	0.992 (0.017)
Total	**0.995 (0.016)**	**0.993 (0.020)**	**0.990 (0.021)**	**0.936 (0.048)**	**0.943 (0.048)**	**0.942 (0.039)**	**0.954 (0.039)**	**0.939 (0.050)**	**0.950 (0.058)**

Table 3: Clustering quality using the Rand index for each database with different initialization and different measures of distances between intervals: mean (standard deviation). The Total row is the global mean quality.

The values of the external criteria show that, for these data, the algorithm is able to find without error the correct number of groups and to produce a good segmentation of the data set. Tables 2 and **??** and Figure **??** show that on average the used distance measure does not significantly influence the quality of the results, even if the distance to vertices seems to be slightly better on these

datasets. However, the initializations chosen can affect the clustering quality. In particular, the "Linear" initialization gives better results on average than the others (Figure **??**), especially for the datasets "2Dim", "Hook" and "Target". The main explanation is that the linear initialization avoids a "twist" of the map that may appear with other initializations, increasing the topological error and thus decreasing the quality of the clustering.

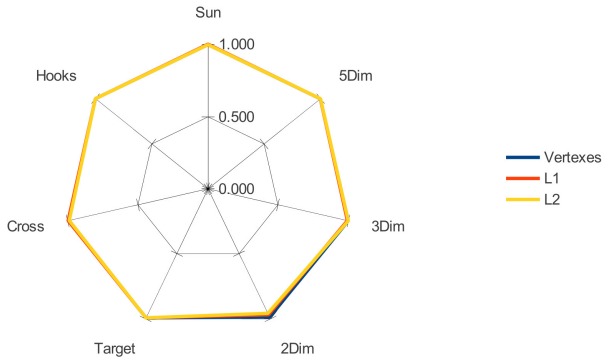


Figure 7: Clustering quality using the Rand index for each database with different measures of distances between intervals, using a linear initialization. The index values are plotted along the axis for each database.

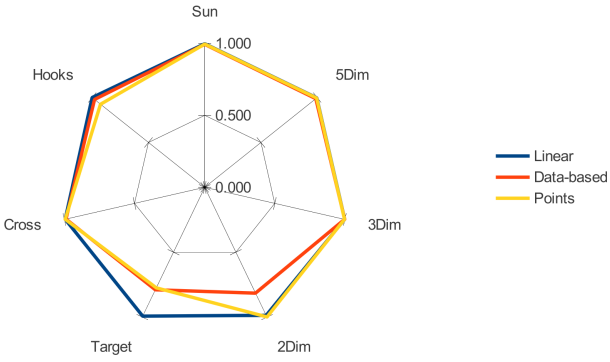


Figure 8: Clustering quality using the Rand index for each database with different prototypes initializations, using the distance to vertices. The index values are plotted along the axis for each database.

The visualization of the clusters obtained with a linear initialization and using the distance to vertices confirms the quality of the clustering algorithm

adapted to interval data (Figures **??** to **??**). In particular, the algorithm is able to detect clusters of different orientation and shape, potentially non-convex ("Target"), even if they are in contact ("Hooks", "Cross"). Unlike all K-means based algorithms (e.g. SCLUST), this algorithm can deal with non hyper-spherical clusters, non convex clusters and non linear separation between clusters. It is also the only algorithm of this type that doesn't need the final number of clusters to be chosen as a parameter, as it is detected automatically during the learning.

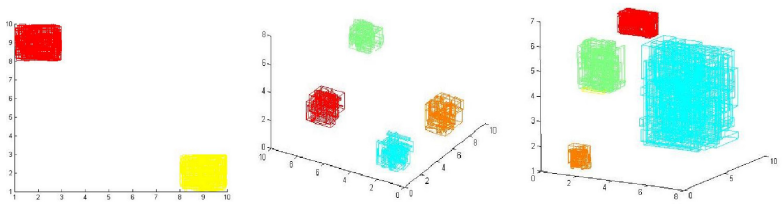


Figure 9: Clusters obtained for data "2Dim", "3Dim" and "5Dim" (first 3 dimensions).

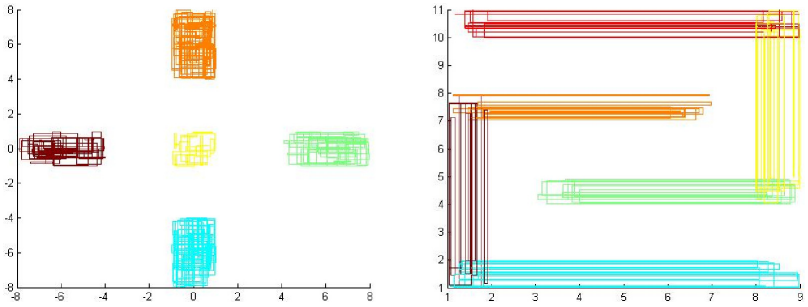


Figure 10: Clusters obtained for data "Sun" and "Hooks".

5.2. Visualization

The results are also confirmed by visual inspection. Indeed, the SOM-based clustering algorithm is a powerful tool for visualization of the obtained segmentation in two dimensions. Clusters are easily and clearly identifiable, as well as regions without data (unconnected neurons). As one can notice from figures **??**, the results obtained by the new algorithm can be visualized in two dimensions

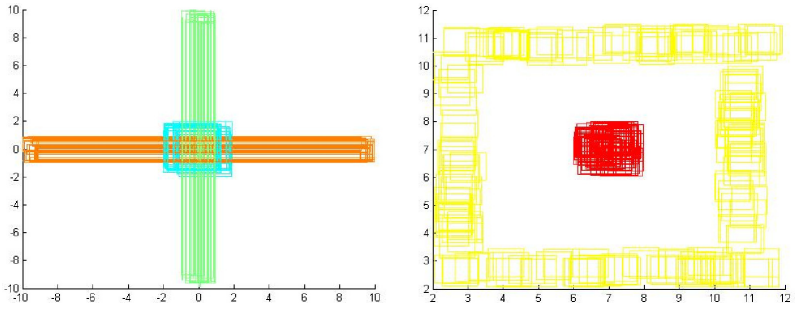


Figure 11: Clusters obtained for data “Cross” and “Target”.

using the mapping obtained with SOM. This allows visualizing the clustering of databases having more than two dimensions, as the “5Dim” database (Fig. **??**).

In Figure **??**, each hexagon represents a prototype of the SOM together with its associated data. Hexagons showing the same color are in the same cluster. Dark blue hexagons are not part of any cluster.

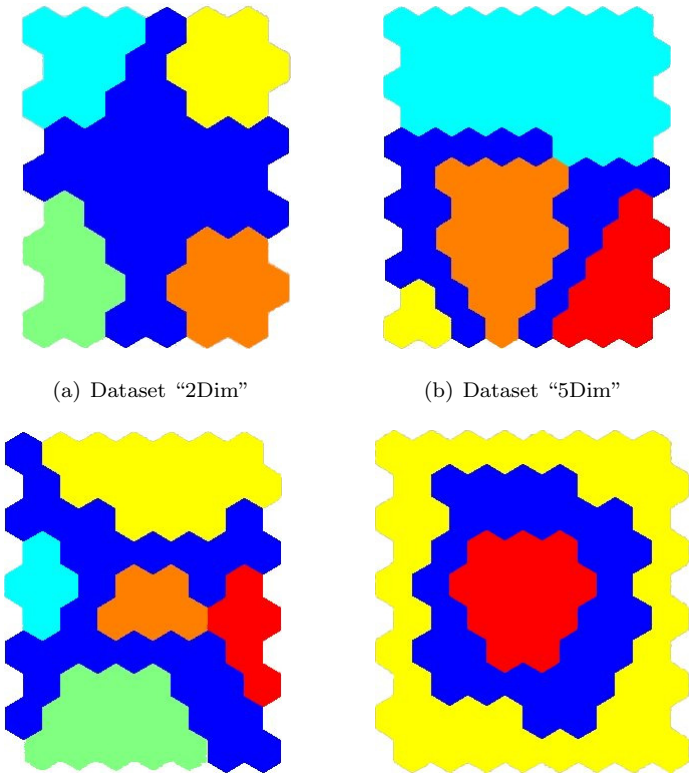
6. Comparison on artificial and real datasets

In the previous section, we showed the effectiveness of the proposed algorithm to solve various clustering problems. The capability to automatically detect the number of clusters is an undeniable advantage. However, it is important to test its performances in comparison to the existing interval clustering algorithms.

6.1. Artificial datasets

In this section, we compared the quality of the clustering and the processing speed of our algorithm in comparison to six other methods described in [**?**]: DIV, SCLUST and the four hierarchical methods contained in SHICLUST (a symbolic extension of single linkage, complete linkage, centroid and Ward).

We applied each algorithm 20 times on each dataset and kept the best result according to an external criterion as in [**?**]. We chose the Jaccard index and the Corrected Rand index as quality criteria. We used a linear initialization and



(a) Dataset “2Dim” (b) Dataset “5Dim”

(c) Dataset “Sun” (d) Dataset “Target”

Figure 12: Segmentation visualization for data “2Dim”, “5Dim”, “Sun” and “Target”.

the distance to vertices for our algorithm. Whereas our method automatically detects the number of clusters, we gave the true number of cluster as a parameter to the other algorithms. The results are shown in Tables **??** and **??**.

Jaccard	Proposed	DIV	SCLUST	SINGLE	CENTROID	WARD	COMPLETE
2Dim	1.00	1.00	1.00	1.00	1.00	1.00	1.00
3Dim	1.00	1.00	1.00	1.00	1.00	1.00	1.00
5Dim	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Cross	1.00	0.29	1.00	1.00	1.00	1.00	1.00
Hook	1.00	1.00	0.67	1.00	1.00	1.00	1.00
Sun	1.00	0.70	0.70	1.00	1.00	1.00	1.00
Target	1.00	0.41	0.41	1.00	0.42	0.43	0.42

Table 4: Clustering quality using the Jaccard index for each database using different clustering algorithms. The index range is [0, 1], with 1 denoting a perfect match between the obtained and the expected solutions.

Rand	Proposed	DIV	SCLUST	SINGLE	CENTROID	WARD	COMPLETE
2Dim	1.00	1.00	1.00	1.00	1.00	1.00	1.00
3Dim	1.00	1.00	1.00	1.00	1.00	1.00	1.00
5Dim	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Cross	1.00	0.62	1.00	1.00	1.00	1.00	1.00
Hook	1.00	1.00	0.93	1.00	1.00	1.00	1.00
Sun	1.00	0.92	0.92	1.00	1.00	1.00	1.00
Target	1.00	0.51	0.51	1.00	0.50	0.55	0.53

Table 5: Clustering quality using the Rand index for each database using different clustering algorithms. The index range is [0, 1], with 1 denoting a perfect match between the obtained and the expected solutions.

Our algorithm gives the expected clustering for all these datasets. We can see that DIV and SCLUST methods perform well when the clusters are convex and well separated, but fail otherwise. The quality of the hierarchical methods is much better for these datasets. In particular, the single linkage method also gives the expected clustering for all these datasets. Other methods fail to detect the non-convex and non linearly separable clusters in the "Target" dataset.

Regarding the exponential increase of the size of the databases in recent years [**?**], the algorithms' complexity becomes a substantial issue for most real applications. It is expected for data-mining tools to have at most a linear complexity in the number of data points, i.e. that the processing time increases linearly with the number of data. Figure **??** shows the result of the processing time of our algorithm, SCLUST and SHICLUST. The four methods in SHICLUST have the same processing time. DIV presents a complexity similar to SCLUST and is not shown in the figure for clarity. Our algorithm was tested from a Matlab script, the others are part of the SODAS software.

From these results, we see that DIV, SCLUST and our algorithm clearly have a linear complexity in the size of the dataset, whereas SHICLUST present an exponential complexity. Our algorithm seems slightly slower than SCLUST, but the SODAS software is compiled and therefore more optimized than a Matlab script.

These experiments show that the quality of our algorithm is better than the DIV and SCLUST methods and similar to SHICLUST. However, the proposed algorithm reaches this quality in a linear complexity, which is not the case of

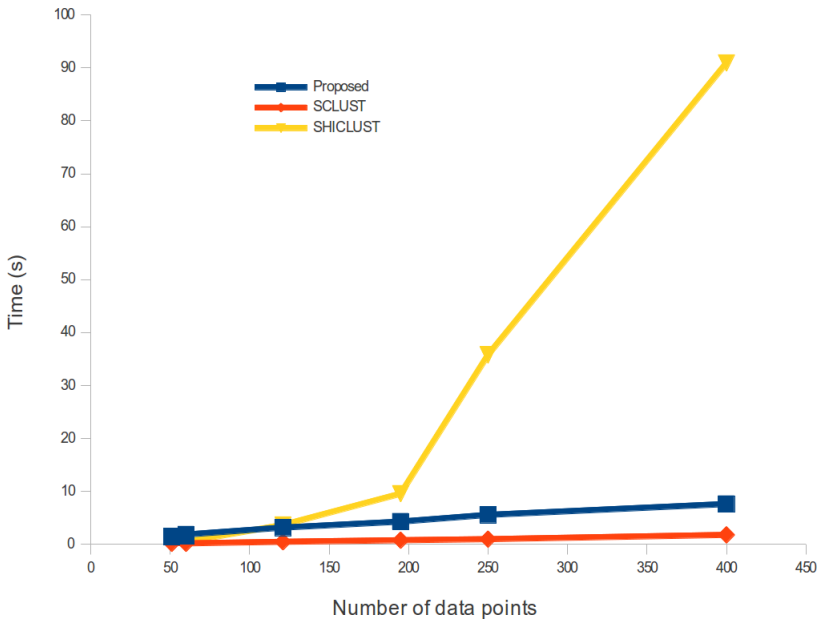


Figure 13: Processing time of the three methods depending on the number of data points.

SHICLUST. We therefore propose a method capable of excellent results in a low processing time, outperforming the existing algorithms in this trade-off.

6.2. Real dataset

In order to test our algorithm on a real dataset and to compare its efficiency in comparison with other clustering methods, we performed a clustering of a well-studied database ("Long-Term Instrumental Climatic Data Bases of the People's Republic of China" [1]). This dataset is derived from instrument measurements at 60 meteorological stations and consists of monthly intervals of temperatures, minimum and maximum, during one year (1988), for a total of 12 interval variables.

In [**?**], several interval clustering algorithms were used to analyze the climatic dataset, in particular SCLUST and the four methods contained in SHICLUST. All these algorithms were applied many times to the dataset, with a

[1] `http://rda.ucar.edu/datasets/ds578.5/`

	SYKSOM	SYKCLUST	Our Algorithm
Jaccard	0.17	0.83	0.85
Rand	0.55	0.90	0.92
Number of clusters	12	2	2*

Table 6: Quality of the clusterings obtained with three methods. *: our algorithm detects this number automatically, whereas for the other methods the number of clusters has been user-defined.

range of values for the choice of the number of clusters varying from 2 to 10. All the obtained clustering results have been compared using internal quality indices (i.e.: Calinski and Harabasz, Duda-Hart and Beale indices [**?**]). Based on these indices, the "best" clustering results were defined. In particular, the best number of clusters is 2 for this dataset. We used this clustering as a reference in this paper, as we don't have any a priori clustering for these data.

We compared our algorithm with a SOM-based method for interval data, SYKSOM, and an adaptation of SYKSOM, called SYKCLUST, allowing a final clustering of the prototypes using SCLUST [**?**]. We used a linear initialization and the distance to vertices for our algorithm. The chosen topology is a 3x4 map with hexagonal neighborhood for the three methods. We computed the Jaccard and the Corrected Rand index for the clustering of the three methods in comparison to the reference clustering (Table **??**).

It is clear that SYKSOM cannot approach the reference clustering, as the number of obtained clusters is the number of chosen prototypes (i.e. 12 in that case). The SYKCLUST extension uses SCLUST on the prototypes to reduce the number of clusters. However, we have to define this number before the learning. Here we chose the expected value (2 clusters) and we obtained a similar clustering to the reference, with a Jaccard and a Rand index above 80%. Finally, the quality of our algorithm is very good, equivalent to SYKCLUST for these indices, but the most important aspect is that the right number of clusters has been found automatically, this property being valuable for the analysis of real datasets when we do not have knowledge about the clustering structure.

7. Conclusion

The results confirm the effectiveness of the proposed algorithm to deal with interval data. The main advantages of this method are: the number of clusters is automatically determined, the algorithm is able to classify groups of non-convex shape; the process is reliable and fast. Indeed, the complexity is linear in the number of data. In the case of interval data, we note that the algorithm discriminates perfectly groups of intervals of different shapes, even if there is contact between groups of intervals and even in the case where the centers of the intervals are the same. These properties are very significant for this type of data. Moreover, the algorithm takes advantage of all the visualization tools developed for SOM-based methods.

The tests show that the algorithm is relatively insensitive to the distance used to compare data and prototypes. On the contrary, the initialization of the prototypes is important for the quality of the final result; a linear initialization gives better results (it is usually true in SOM in general [**?**]).

Once a distance and a prototype are defined, the adaptation of SOM to any type of data seems perfectly appropriate in order to obtain a classification of the data. We plan to work on new adaptations for the analysis of other types of complex data (such as multi-valued, modal or structured data). Furthermore, in the case of mixed data clustering (real-valued + interval-valued), we plan to propose an extension of our method by considering an objective function composed of several terms: one term for each data type. Each term of this composite objective function uses an appropriate distance for each type of the data involved. The optimization of the objective function can be done in a global manner. Finally, we wish to use the Shrinkage-Divergence Proximity (SDP) redesign distance framework [**?**] to redefine a meaningful distance function in order to adapt our clustering algorithm to high-dimensional spaces.

References

[1] M. Avriel, Nonlinear Programming: Analysis and Methods, Dover Publishing, 2003.

[2] L. Billard, E. Diday, Symbolic Data Analysis: Conceptual Statistics and Data Mining, John Wiley and Sons, 2006.

[3] H.-H. Bock, E. Diday (Eds.), Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, Springer Verlag, Heidelberg, 2000.

[4] H.-H. Bock, Clustering algorithms and Kohonen maps for symbolic data, Journal of the Japanese Society of Computational Statistics, 15, 2002, pp. 1–13.

[5] H.-H. Bock, Visualizing symbolic data by Kohonen maps, in: E. Diday, M. Noirhomme-Fraiture (Eds.), Symbolic Data Analysis and the SODAS Software, Wiley, 2008, pp. 205–234.

[6] E. L. J. Bohez, Two-level cluster analysis based on fractal dimension and Iterated Function Systems (IFS) for speech signal recognition, IEEE Asia-Pacific Conference on Circuits and Systems, 1998, pp. 291–294.

[7] P. Brito, Hierarchical and pyramidal clustering for symbolic data, Journal of the Japanese Society of Computational Statistics, 2002, pp. 231–244.

[8] G. Cabanes, Y. Bennani, A simultaneous two-level clustering algorithm for automatic model selection, in: Proceedings of the International Conference on Machine Learning and Applications (ICMLA'07), 2007, pp. 316–321.

[9] G. Cabanes, Y. Bennani, D. Fresneau, Enriched topological learning for cluster detection and visualization, Neural Networks, 32(1), 2012, pp. 186–195.

[10] M. Chavent, H.-H. Bock, Clustering Methods for Symbolic Objects - Criterion-Based Divisive Clustering for Symbolic Data, in: H.-H. Bock,

E. Diday (Eds.), Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, Springer Verlag, Heidelberg, 2000, pp. 299–311.

[11] M. Chavent, F.A.T. de Carvalho, Y. Lechevallier, R. Verde, New clustering methods for interval data, Computational Statistics, 21(23), 2006, pp. 211–230.

[12] Y. Chen, B. Qin, T. Liu, Y. Liu, S. Li, The Comparison of SOM and K-means for Text Clustering, Computer and Information Science, 3(2), 2010, pp. 268–274.

[13] F.A.T. de Carvalho, Proximity coefficients between Boolean symbolic objects, in: E. Diday et al. (Eds), New Approaches in Classification and Data Analysis, Series: Studies in Classification, Data Analysis, and Knowledge Organization, 5, Springer Verlag, Berlin, 1994, pp. 387–394.

[14] F.A.T. de Carvalho, R.M.C.R. De Souza, M. Chavent, Y. Lechevallier, Adaptative Hausdorff Distances and Dynamic Clustering of Symbolic Interval Data, Pattern Recognition Letters, 27(3), 2006, pp. 167–179.

[15] F.A.T. de Carvalho, P. Brito, H.-H. Bock, Dynamic clustering for interval data based on L2 distance, Computational Statistics, 21(2), 2006, pp. 231-250.

[16] A.B. dos S. Dantas, F.A.T. de Carvalho, Adaptive Batch SOM for Multiple Dissimilarity Data Tables, in: Proc. 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI),2011, pp. 575 – 578.

[17] B. Delaunay, Sur la sphère vide, Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvennykh Nauk, 7, 1934, pp. 793–800.

[18] R.M.C.R. De Souza, F.A.T. de Carvalho, Clustering of interval data based on city-block distances, Pattern Recognition Letters, 25(3), 2004, pp. 353 – 365.

[19] E. Diday, Clustering analysis, in: K.S. Fu (Ed.), Digital Pattern Recognition, Springer Verlag, Berlin, 1976, pp. 47–94.

[20] E. Diday, F. Esposito, An introduction to symbolic data analysis and the SODAS software, Intelligent Data Analysis, 7, 2003, pp. 193–218.

[21] E. Diday, M. Noirhomme-Fraiture (Eds.), Symbolic Data Analysis and the SODAS Software, Wiley, 2008.

[22] P. D'Urso, L.D. Giovanni, Midpoint radius self-organizing maps for interval-valued data with telecommunications application, Applied Soft Computing, 11, 2011, pp. 3877-3886.

[23] F. Esposito, D. Malerba, A. Appice, Dissimilarity and matching, in: E. Diday, M. Noirhomme-Fraiture (Eds.), Symbolic Data Analysis and the SODAS Software, Wiley, 2008, pp. 123–148.

[24] B. Everitt, S. Landau, M. Leese, Cluster Analysis, Arnold, 2001.

[25] A.D. Gordon, Classification, 2nd Edition, Chapman and Hall/CRC, 1999.

[26] A.D. Gordon, An iteractive relocation algorithm for classifying symbolic data, in: W. E. A. Gaul (Ed.), Data analysis: scientific modeling and practical application, Springer Verlag, 2000, pp. 17 – 23.

[27] K.C. Gowda, E. Diday, Symbolic clustering using a new dissimilarity measure. Pattern Recognition, 24(6), 1991, pp. 567 – 578.

[28] C. Hajjar, H. Hamdan, Self-organizing map based on L2 distance for interval-valued data, in: Proc. 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), 2011, pp. 317 – 322.

[29] C. Hajjar, H. Hamdan, Self-organizing map based on Hausdorff distance for interval-valued data, in: Proc. IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2011, pp. 1747 – 1752.

[30] M. Halkidi, Y. Batistakis, M. Vazirgiannis, On Clustering Validation Techniques, Journal of Intelligent Information Systems 17(2-3), 2001, pp. 107–145.

[31] A. Hardy, Les méthodes de classification et de détermination du nombre de classes : du classique au symbolique, in: M. Chavent, O. Dordan, C. Lacomblez, M. Langlais, B. Patouille (Eds.), Comptes-rendus des 11èmes Rencontres de la Société Francophone de Classification, 2004, pp. 48–55.

[32] A. Hardy, Validation of clustering structure: determination of the number of clusters, in: E. Diday, M. Noirhomme-Fraiture (Eds.), Symbolic Data Analysis and the SODAS Software, Wiley, 2008, pp. 235–262.

[33] A. Hardy, N. Kasoro Mulenda, Une nouvelle méthode de classification pour des données intervalles, Mathématiques et Sciences Humaines, 187, 2009, pp. 79–91.

[34] A. Hardy, J. Baune, SYKSOM, méthode de représentation et de classification de données symboliques basée sur les cartes de Kohonen, Technical Report, University of Namur, Namur, Belgium, 2010.

[35] J.A. Hartigan, Clustering Algorithms, Wiley, 1975.

[36] C.-M. Hsu, M.-S. Chen, On the Design and Applicability of Distance Functions in High-Dimensional Data Space, IEEE Transactions on Knowledge and Data Engineering, 21(4), 2009, pp. 523–536.

[37] L. Hubert, P. Arabie, Comparing Partitions, Journal of Classification, 2, 1985, pp. 193–218.

[38] M. F. Hussin, M. S. Kamel, M. H. Nagi, An efficient two-level SOMART document clustering through dimensionality reduction, in: ICONIP, 2004, pp. 158–165.

[39] A. Irpino, R. Verde, Dynamic clustering of interval data using a Wasserstein-based distance, Pattern Recognition Letters, 29, 2008, pp. 1648–1658.

[40] A. K. Jain, R. C. Dubes, Algorithms for clustering data, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[41] L. Kaufman, P.J. Rousseeuw, Finding groups in data, Wiley, New York, 1990.

[42] T. Kohonen, Self-Organizing Maps, Springer-Verlag, Berlin, 2001.

[43] E. E. Korkmaz, A two-level clustering method using linear linkage encoding, in: International Conference on Parallel Problem Solving From Nature, Lecture Notes in Computer Science, Vol. 4193, Springer-Verlag, 2006, pp. 681–690.

[44] P. Lyman and H. R. Varian, How Much Information, 2003. Retrieved from http://www.sims.berkeley.edu/how-much-info-2003.

[45] D. Malerba, F. Esposito, V. Giovale, V. Tamma, Comparing dissimilarity measures for symbolic data analysis, in: Proceedings of Techniques and Technologies for Statistics - Exchange of Technology and Know-How, 1, 2001, pp. 473–481.

[46] T. Martinetz, Competitive Hebbian Learning rule forms perfectly topology preserving maps, in: S. Gielen, B. Kappen (Eds.), Proceedings of the International Conference on Artificial Neural Networks (ICANN'93), Springer, Heidelberg, 1993, pp. 427–434.

[47] L.D.S. Pacifico, F.A.T. de Carvalho, A batch self-organizing maps algorithm based on adaptive distances, in: Proc. of the 2011 International Joint Conference on Neural Networks (IJCNN), 2011, pp. 2297 – 2304.

[48] J.-Y. Pirçon, Le clustering et les processus de Poisson pour de nouvelles méthodes monothétiques, Ph.D. thesis, University of Namur, Belgium, 2004.

[49] A. Ultsch, Clustering with SOM: U*C, in: Proceedings of the Workshop on Self-Organizing Maps, 2005, pp. 75–82.

[50] R. Verde, F.A.T. de Carvalho, Y. Lechevallier, A dynamical clustering algorithm for multi-nominal data., in: H. Kiers et al. (Ed.), Data analysis, classification, and related methods., Berlin: Springer, 2000, pp. 387–393.

[51] J. Vesanto, Neural network tool for data mining: SOM Toolbox, 2000.

[52] M.-S. Yang et al., Self-organizing map for symbolic data, Fuzzy Sets and Systems, 203, 2012, pp. 49–73.