17

The Search for Irregularly Shaped Clusters in Data Mining

Angel Kuri-Morales[1] and Edwyn Aldana-Bobadilla[2]
[1]*Instituto Tecnológico Autónomo de México*
[2]*Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas,UNAM*
México

1. Introduction

One of the basic endeavours in Data Mining is the unsupervised process of determining which objects in the database do share interesting properties. The solution of this problem is usually denoted as "clustering", i.e. the identification of sets in the database which may be grouped in accordance with an appropriate measure of likelihood. Clustering can be considered the most important unsupervised learning problem. As every other problem of this kind, it deals with finding a structure in a collection of unlabeled data. A loose definition of clustering could be "the process of organizing objects into groups whose members are similar in some way". A cluster is therefore a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters. In clustering the more traditional algorithms are based on similarity criteria which depend on a metric. This fact imposes basic constraints on the shape of the clusters found. These shapes are hyperspherical in the metric's space due to the fact that each element in a cluster lies within a radial distance relative to a given center. Working with metric spaces works fine when the hyperspheres are well behaved. However, there are cases where there is an unavoidable degree of confusion because the hyperspheres do overlap to a certain extent. It has been shown (Haykin,1994) that iff the elements of the database exhibit a Gaussian behavior a Bayesian classifier will minimize the degree of confusion, even when classification errors are unavoidable. Several clustering methods have been proposed. Most exhibit the limitation discussed above. In this work, however, we discuss three alternatives of clustering algorithms which do not depend on simple distance metrics and, therefore, allow us to find clusters with more complex shapes in n-dimensional space.

a. Clustering based on optimization of validity indices.
b. Clustering based on optimization of entropy
c. Clustering based on optimization of membership.

This chapter begins with an account of the principles of the traditional clustering methods. From these it is evident that irregularly shaped clusters are difficult to deal with. With this limitation in mind we propose some non-traditional approaches which have shown to be effective in our search for possibly non-hyperspherical locus. The clustering process is a highly non-linear and usually non-convex optimization problem which disallows the use of traditional optimization techniques. For this reason we discuss some optimization techniques pointing to Genetic Algorithms as a good alternative for our purposes.

2. Clustering

Our problem will be focused under the assumption that the data under consideration correspond to numerical variables. This will not be always the case and several other methodologies have been devised to allow clustering with non-numerical (or categorical) variables.
Research on analyzing categorical data has received significant attention see (Agresti,2002), (Chandola et al.,2009),(Chang & Ding,2005) and (Gibson,2000). However, many traditional techniques associated to the exploration of data sets assume the attributes have continuous data (covariance, density functions, Principal Component's Analysis, etc.). In order to use these techniques, the categorical attributes have to be discarded, although they are loaded with valuable information. Investigation under way (Kuri & Garcia,2010) will allow us to apply the methods to be discussed in what follows even in the presence of categorical variables. Suffice it to say, however, that we shall not consider these special cases and, rather,focus on numerically expressible attributes of the data sets.

2.1 Definition

Clustering, as stated, is an unsupervised process that allows the partition of a data set X in k groups or clusters in accordance with a similarity criterion. Generally all elements of data set X belong to a *metric space* D, where there exist relationships between them that are expressible in terms of a distance. In the great majority of clustering methods, the similarity between two or more elements is defined as a measure of its proximity whose value is a consequence of such distance. The elements are rendered similar if its proximity is less than a certain threshold. This threshold represents the radius that defines the bound of a cluster as a n-dimensional spherical hull.

2.2 Metrics

Formally a metric is a function that defines the distance between elements of a set. The set with a metric is called a metric space.

Definition 1 *A metric on a set X is a function* $d : X \times X \to R$ *where R is the set of the real numbers. For all* x, y, z *in X this function must satisfy the following conditions*

1. $d(x,y) \geq 0$
2. $d(x,y) = 0 \quad if \quad and \quad only \quad if \quad x = y$
3. $d(x,y) = d(y,x)$
4. $d(x,z) \leq d(x,y) + d(y,z)$ (triangle inequality)

In a clustering process a popular set of metrics is one of the Minkowsky family (Cha,2008). Its value is defined by the following equation.

$$d_{mk}(P,Q) = \sqrt[\alpha]{\sum_{i=1}^{n} |P_i - Q_i|^{\alpha}} \tag{1}$$

where P and Q are two vectors in an n-dimensional space. The α value is an integer number that represents one particular metric ($\alpha = 2$ corresponds to the Euclidian Distance). However, this distance is sometimes not an appropriate measure for our purpose. For this reason sometimes the clustering methods use statistical metrics such

as Mahalanobis'(Mahalanobis,1936), Bhattacharyya's (Bhattacharyya,1943) or Hellinger's (Pollard,2002), (Yang et al.,2000). These metrics statistically determine the similarity of the probability distribution between random variables P and Q.

2.3 Methods

Among the many approaches to clustering we wish to mention some of the better known ones. We are careful to mention at least one example of every considered approach, although this account is, by no means, exhaustive. In what follows we call "traditional" those methods which emphasize the use of a metric to determine a set of centroids. A centroid is, intuitively, the point around which all the elements of the cluster do group. This is the reason why metric-based methods yield hyperspherical clusters.

2.3.1 Traditional methods

As already mentioned, traditional methods typically yield well defined centroids. The algorithms supporting the method offer a wide variety. Some use concrete logic, others fuzzy logic; some approach the problem intuitively (even in a simplistic fashion) as is the case of hierarchical clustering. Others, on the other hand, approach the clustering problem as a (non-obvious) problem of competition (such as the ones stemming from Kohonen's approach).

2.3.1.1 K-means

The main idea is to define k centroids, one for each cluster. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed. At this point we need to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. This algorithm aims at minimizing an objective function, in this case a squared error function. The objective function is:

$$J = \sum_{i=1}^{n} \sum_{j=1}^{k} \left\| x_i - c_j \right\|^2 \tag{2}$$

where $\left\| x_i - c_j \right\|^2$ is s a chosen distance between a data point x_i and and the cluster center c_i. (For details see (MacQueen,1967))

2.3.1.2 Fuzzy C-means

This is a method which allows one object in the data set to belong to two or more clusters. It is based on minimization of the following objective function:

$$J = \sum_{i=1}^{n} \sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2 \qquad 1 \leq m < \infty \tag{3}$$

where m is any real number greater than 1, u_{ij} is the degree of membership of x_i in j, x_i is the $i-th$ of d-dimensional measured data, c_j is the d-dimension center of the cluster, and $\left\| x_i - c_j \right\|$ is any norm expressing the distance between any measured data and the center.

Fuzzy partitioning is carried out through an iterative optimization of the above equation by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \frac{\|x_i - c_j\|}{\|x_i - c_k\|}^{2/(m-1)}} \tag{4}$$

$$c_j = \frac{\sum_{i=1}^{n} u_{ij}^m x_i}{\sum_{i=1}^{n} u_{ij}^m} \tag{5}$$

the iterations will stop when

$$max_{ij} = \left\{ \left| u_{ij}^{k+1} - u_{ij}^{k} \right| \right\} < \epsilon \tag{6}$$

where $0 \leq \epsilon \leq 1$ and k is the iteration step. The reader can find more details in (Dunn,1973).

2.3.1.3 Hierarchical clustering

Given a set of N items to be clustered, and a distance matrix, the basic process is:

1. Start by assigning each item to a cluster, so that if you have N items, you now have N clusters, each containing just one item. Let the distances between the clusters be the same as the distances between the items they contain.
2. Find the closest pair of clusters and merge them into a single cluster, so that now you have one cluster less.
3. Compute distances between the new cluster and each of the old clusters.
4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

Of course there is no point in having all the N items grouped in a single cluster but, once you have gotten the complete hierarchical tree, for k clusters simply cut the $k-1$ longest links. Representative methods in this category are BIRCH (Zhang,1996), CURE and ROCK (Guha,1998).

2.3.1.4 Self organizing maps

These kind of neural networks basically allows the mapping of a set of vectors in n-dimensional space into a smaller set in (typically) 2-dimensional space. The idea is to assign every vector in the original n-space an XY coordinate in a way such that neighboring elements in XY plane (the so-called neurons) correspond to neighboring elements in the original n-dimensional space. This yields a map in XY where physically close sets of neurons define clusters in n dimensional space. The procedure for the training algorithm is:

1. Initialize
   - Define a grid on a Cartesian plane XY.
   - Every pair of coordinates in $\mathbb{N}$ is associated to a neuron.
   - The number of neurons is given by $N = max(X_i) \times max(Y_j)$
   - A weight vector $w_{ij} \in \mathbb{R}$ is associated to every neuron.
   - A set V, consisting of all elements of the data set, is defined. We denote the cardinality of V with J. That is $J = |V|$.
   - Define a maximum number of epochs E. An epoch consists of the (typically) random traversal of all J vectors.

- All elements of the weight vectors are assigned random numbers.
- A maximum number of epochs, T, is defined.
- A learning rate $\eta[e(t)]$ in $\mathbb{R}$ is defined, where $e(t)$ is the t-*th* epoch.
- A feedback function of neuron i to the winning neuron k in epoch t is defined, where $r_{ik}[e(t)] \in \mathbb{R}$

2. $i = 1$
3. An input vector V_j is randomly selected (without replacement).
4. A winning neuron k is determined as the neuron that has the minimum distance $d_k = min_i \|V - w_i\|$ to the input vector V_j.
5. The weight vectors are iteratively adapted according to the following function: $w_{ij}(t) = w_{ij}(t-1) + \eta[e(t)] * r_{ik}[e(t)] * (V_j - w_{ij}(t-1))$
6. If $i = J$

   $t \leftarrow t + 1$

   If $t > T$ end algorithm.

   Update $\eta[e(t)]$

   Update $r_{ik}[e(t)]$

   Go to step 2

   else

   $i = i + 1$

   Go to step 3.

   endif

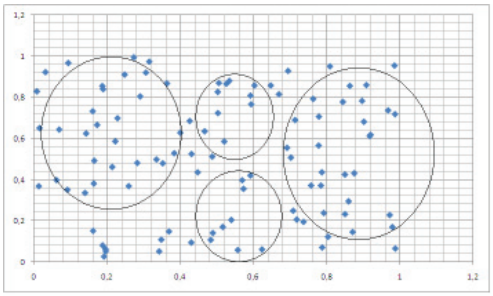
Upon ending, the algorithm will have placed similar neurons in neighboring areas of the plane. That is, it will have organized (hence the name SOM) similar neurons to lie physically close to each other in XY. A cluster will, therefore, consist of those objects in the data set whose coordinates in the original space of attributes are pointed to by neighboring neurons in the Cartesian plane. A complete description of this method is found in (Kohonen,1997)

2.3.2 Limitations of traditional methods

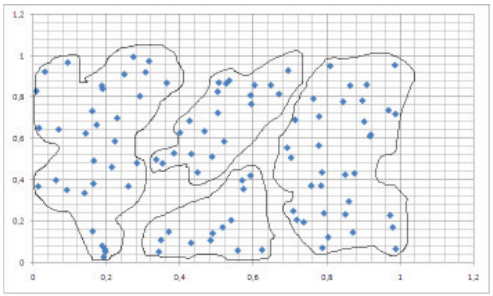
In general a good clustering method must:

- Be able to handle multidimensional data sets.
- Be independent of the application domain.
- Have a reduced number of parameters.
- Be able to display computational efficiency.
- Be able to yield irregularly shaped clusters.

The last point is the most difficult to achieve for the following reason. If we assume that all elements of data set X belong to a *metric space* D, then there exist relationships between them that are expressible in terms of a distance. The elements are similar if its proximity is less than a certain threshold. This threshold represents the radius that defines the bound of a cluster as a n-dimensional spherical hull. This is illustrated in Fig. 1(a) (for $n = 2$) where the bound clusters are convex shapes. However an ideal case would allow us to obtain arbitrary shapes for the clusters that adequately encompass the data. Fig. 1(b) illustrates this fact.



(a) Circular shapes



(b) Arbitrary shapes

Fig. 1. Differents ways of clustering for the same dataset

For example, assume we have a problem in 2-dimensional space (say $X - Y$) and that we define a metric which assigns greater merit to the coordinates in the Y axis than those in the X axis. To make this hypothesis definite, suppose that the values of the X-coordinates are k times as important as those of the Y-coordinates. We define a distance. Then a set of two hypothetical clusters would look like the ones illustrated in Fig. 2(a).
In this figure the coordinates (x_i, y_i) of point i are set in units of the same relative size. However, if we were to change the X axis so that every coordinate x_i is replaced by x_i' where $x_i' = kx_i$ then we would now see the same two clusters as shown in Fig. 2(b). Clearly, now the clusters exhibit a circular (hypersphere, for $n = 2$) shape. A similar experiment may be performed for any metric space. Working with metric spaces works fine when the hyperspheres are well behaved. This is illustrated in Fig. 3. Here (and in the following 2 figures) the elements of the database are assumed to lie on the surface of the sphere.
However, there are cases such as the one illustrated in Fig. 4(a) where there is an unavoidable degree of confusion.
In this case classification errors are unavoidable. An even more critical case is as the one shown in Fig. 4(b), where a metric algorithm will be unable to distinguish the clusters unless the width of the surface of the spheres is specified. A further example of the problems we may face is illustrated in Fig. 5, where highly irregular but, however, fairly "evident" clusters are shown. None of these would be found by a clustering algorithm based on a metric space.

2.4 Non-traditional methods

In this work we discuss three alternatives of clustering algorithms which do not depend on simple distance metrics and, therefore, allow us to find clusters with more complex shapes in n-dimensional space.

a. Clustering based on optimization of quality indices.

b. Clustering based on optimization of entropy

c. Clustering based on optimization of membership.

2.4.1 Clustering based on optimization of quality indices.

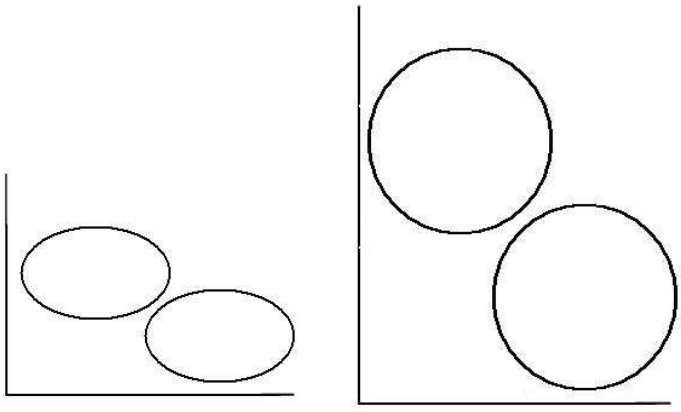
In this case, the clustering algorithm is based on the identification of those elements of the database which optimize some quality criterion. In the usual process, the clusters are found via some arbitrary clustering algorithm and then assigned a quality grade according to a certain quality index. An interesting alternative is to find the elements of the clusters from the indices directly. The problem with this approach is that the purported indices are usually expressed mathematically with some complex function which is not prone to optimization by any of the traditional methods. Hence, we analyze clusters resulting from the optimization of a given quality index with genetic algorithms (GA). In particular, we point out that any elitist GA has been proved to find a global optimum if given enough time (Rudolph,1994). Furthermore, we know that not all GAs are equally efficient and, further, that a variation called Vasconcelos GA (VGA) is best among a family of GAs (Kuri,2002). We point out that there have many attempts to prove that a specific quality index is relatively better than others (Kovacs,2006). We have selected some of the more interesting ones for our analysis. Applying VGA we have found clusters which depend on one of the following quality indices.

– Dunn and Dunn-like validity indices

– Davies-Bouldin validity index

– SD validity index

– Variance of the nearest neighbor (VNN)



(a) Clusters in X-Y (b) Clusters in X-kY

Fig. 2. Hypothetical clustering

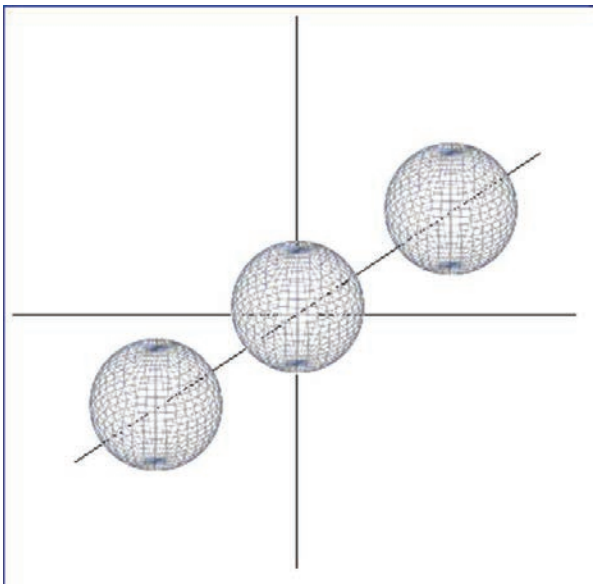


Fig. 3. Disjoint spherical clusters

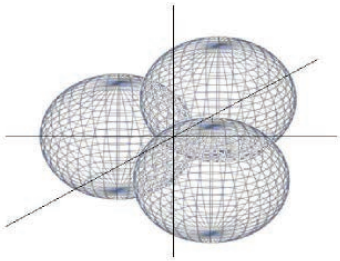
None of the resulting clusters has an immediate spherical shape unless the coordinates of the space of solution are translated into the indices: a rather involved and sometimes downright impossible task.

2.4.2 Clustering based on optimization of entropy

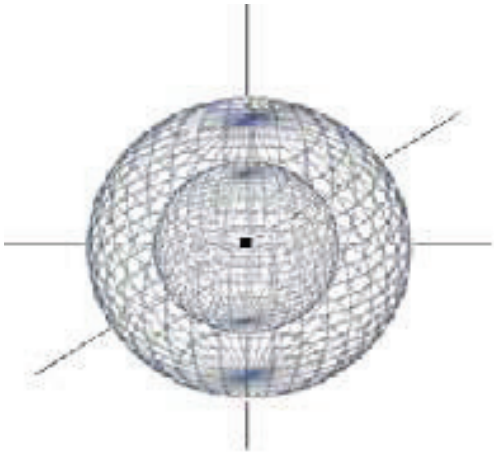
In broad terms, the information of a data set may be expressed as the expected information for all the symbols in such set. From Shannon's information theory (Shannon,1949) we may write the entropy of a source S as:

$$H(S) = \sum_{i=1}^{n} p(s_i) I(s_i) = -\sum_{i=1}^{n} p(s_i) log_2(p(s_i)) \tag{7}$$

where $p(s_i)$ is the probability that symbol s_i is output by the source; $I(s_i)$ is the information present in symbol i in bits:$I(s_i) = -log_2 p(s_i)$. There are several possible approaches to clustering by considering the information in the data rather than a distance between their elements (COOLCAT). To solve these problems we apply a rugged genetic algorithm. In order to test the efficiency of our proposal we artificially created several sets of data with known properties in a tridimensional space. By applying this algorithm [called the Fixed Grid Evolutionary Entropic Algorithm (FGEEA) (Kuri & Aldana,2010)] we were able to find highly irregular clusters that traditional algorithms cannot. Some previous work is based on algorithms relying on similar approaches (such as ENCLUS'(Cheng,1999) and CLIQUE's (Agrawal et al.,1998)). The differences between such approaches and FGEEA will also be discussed.



(a) Overlapping clusters



(b) Concentric clusters

Fig. 4. Clusters with different overlapping degree

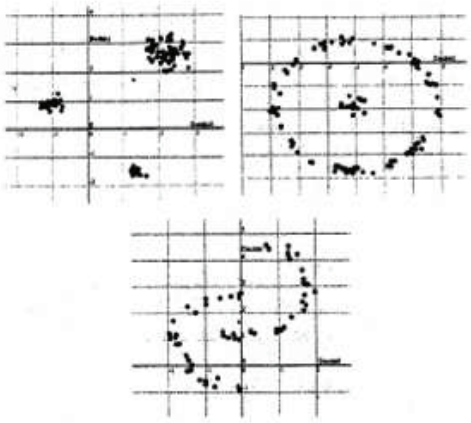


Fig. 5. Irregularly shaped clusters

2.5 Clustering based on optimization of membership

A final approach to be discussed is one in which the elements of a cluster are determined by finding a set of metric locus which encompass the elements of the database; one set per cluster. To make the idea definite a formula originally developed by Johan Gielis (Gielis,2003) (which has been called the "superformula") is examined. It allows the generation of n-dimensional bodies of arbitrary shape by modifying certain parameters. This approach allows us to represent a cluster as the set of data contained "inside" a given body without resorting to a distance (Kuri & Aldana,2008). Therefore, we replace the idea of nearness by one of membership. Gielis superformula (GSF) generalizes the equation of a hyper-ellipse by introducing some parameters which increase the degrees of freedom of the resulting figures when geometrically interpreted. It is given by:

$$r(\rho) = \left[\left| \frac{cos(\frac{m\rho}{4})}{a} \right|^{n_2} + \left| \frac{sin(\frac{m\rho}{4})}{b} \right|^{n_3} \right]^{n_1} \tag{8}$$

Where r is the radius and ρ is the angle. In Fig. 6 we show some forms generated from $a = b = 1$ and several values assigned to m, n_1, n_2, n_3. It is possible to generalize the formula to three or more dimensions. In Fig. 7 we show some forms obtained for a 3-dimensional space.
The parameters of the superformula are encoded in the chromosome of the genetic algorithm (GA) (one set per cluster). The GA maximizes the number of elements enclosed in arbitrary figures defined by Gielis' formula. The variables to optimize are m, $n1$, $n2$, $n3$, a, b for each of the clusters. Variables cx, cy, cz which correspond to the centers of the clusters are also introduced. These variables place the locus in a definite point in space. Therefore, the encoding of one cluster for the GA is as shown in Fig. 8.
The full chromosome, assuming there are $four$ clusters, is shown in Fig. 9.
Clearly, this approach is not hampered by distance considerations. A point to be made is that the generalization of Gielis' formula has not been achieved. However, the same approach may be attempted in a way that yields arbitrary shapes in N-space. This is still a matter of open research.

3. Clustering as an optimization problem

The problem of finding an appropriate set of centroids may be seen, in general, as an optimization problem. This is clear from equations (2), (3), (4) and (1) (in the case of Euclidean based SOMs). In all of these cases we wish to minimize a distance. In fact, the method is defined by the minimizing algorithm. Were we able to find an optimization algorithm suitable to all the cases, then the difference between the various methods would almost fundamentally rely on the definition of the distance. Several such general methods exist. Before describing one of particular interest for our purpose we make a brief mention of classical optimization schemes and their inherent limitations.

3.1 Classical optimization

In classical optimization the search for the solution usually depends on iterating along the direction of the negative steepest slope of the function under study. When there is a single variable the slope is found by differentiating with respect to the independent variable. If there are more variables then gradient based methods are used.

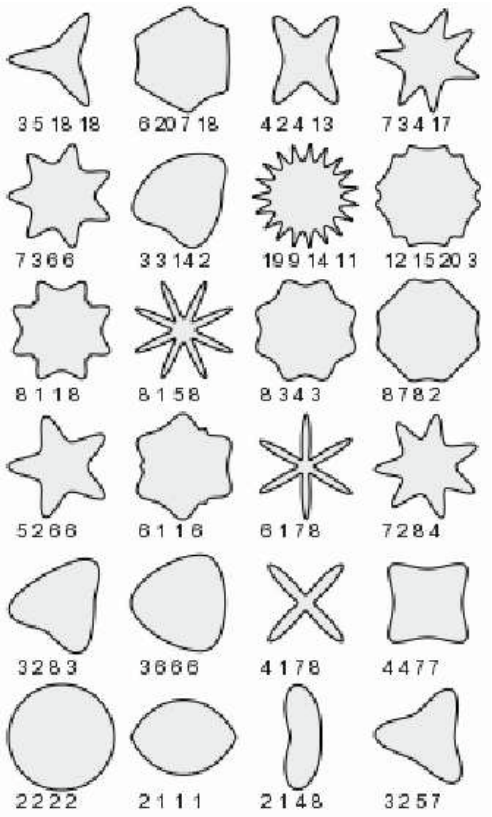


Fig. 6. Shapes in 2D generated by the Gielis equation

3.1.1 Convexity

In order for these methods to work it is stipulated that the function be convex. A real-valued function $f(x)$ defined on an interval is called convex if for any two points x_1 and x_2 in its domain X and any $t \in [0,1]$,

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2) \tag{9}$$

This implies that when moving on the optimization path, no point will map out of the space of the function. In other words, and this is the point we want to stress, in classical optimization the function has to be well defined and convex. If either of these two conditions is not met, then the process collapses and we are unable to tackle the problem.

3.1.2 Differentiability

Clearly, if the purportedly optimizable function is not amenable to differentiation, then the problem (which, precisely, depends on the existence of the gradient) is intractable. This too, constitutes a serious limitation. Many functions exhibit discontinuities and/or poles and none of these functions is optimizable (in general) via the classical optimization methods. During the last few decades, given that the computational costs have decreased dramatically, there

has been a tendency to consider computationally intensive methods. Such methods were not practical before the advent of low cost computers. Among the many methods which have recently arisen, we may mention: tabu search, simulated annealing, ant colony optimization, particle swarm optimization and evolutionary computation. Some of the variations of evolutionary computation are: evolutionary strategies, evolutionary programming, genetic programming and genetic algorithms. GAs have been extensively used in otherwise difficult or intractable optimization problems. In our case, GAs turn out to be the key to generalize most of the clustering techniques to be discussed in what follows. For this reason we make a brief digression and discuss the basic tenets of GAs.

3.2 Genetic algorithms

Genetic Algorithms (an interesting introduction to GAs and other evolutionary algorithms may be found in (Bäck,1996)) are optimization algorithms which are frequently cited as "partially simulating the process of natural evolution". Although this a suggestive analogy behind which, indeed, lies the original motivation for their inception, it is better to understand them as a kind of algorithms which take advantage of the implicit (indeed, unavoidable) granularity of the search space which is induced by the use of the finite binary representation in a digital computer.

In such finite space, numbers originally conceived as existing in R^n actually map into B^m space. Thereafter it is simple to establish that a genetic algorithmic process is a finite Markov chain (MC) whose states are the populations arising from the so called genetic operators: (typically) selection, crossover and mutation (Rudolph,1994). As such they display all of the properties of a MC. From this fact one may prove the following mathematical properties of a GA:

(1) The results of the evolutionary process are independent of the initial population and

(2) A GA preserving the best individual arising during the process will converge to the global optimum (albeit the convergence process is not bounded in time).

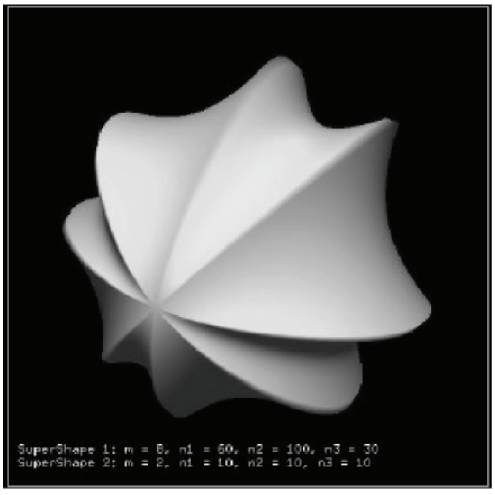


Fig. 7. Shapes in 3D generated by the Gielis equation

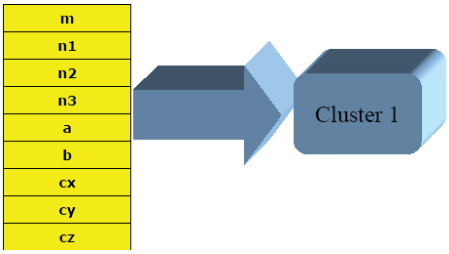


Fig. 8. Part of the chromosome encoding a cluster.

Their most outstanding feature is that, as opposed to other more traditional optimization techniques, the GA iterates simultaneously over several possible solutions. Then, other plausible solutions are obtained by combining (crossing over) the codes of these solutions to obtain hopefully better ones. The solution space (SS) is, therefore, traversed stochastically searching for increasingly better plausible solutions. In order to guarantee that the SS will be globally explored some bits of the encoded solution are randomly selected and changed (a process called mutation). The main concern of GA-practitioners (given the fact that well designed GAs, in general, will find the best solution) is to make the convergence as efficient

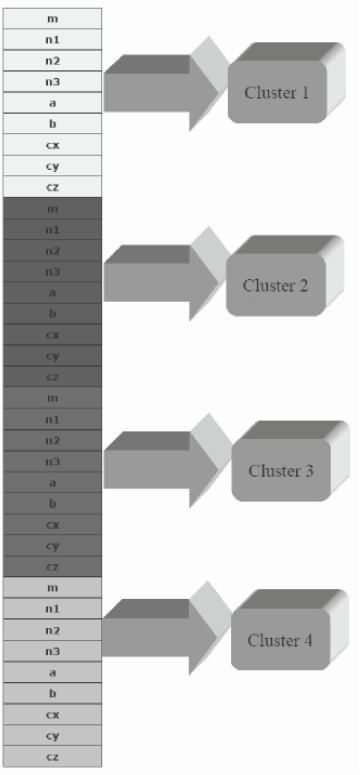


Fig. 9. Full chromosome

Algorithm	Relative Performance	Number of Optimized Functions
VGA	1.000	2,736
EGA	1.039	2,484
TGA	1.233	2,628
SGA	1.236	2,772
CGA	1.267	3,132
RHC	3.830	3,600

Table 1. Relative Performance of Different Breeds of Genetic Algorithms

as possible. The work of Forrest et al. has determined the characteristics of the so-called Idealized GA (IGA) which is impervious to GA-hard problems (Forrest,1993).

3.2.1 Vasconcelos' genetic algorithms

The implementation of the IGA is unattainable in practice. However, a practical approximation called the Vasconcelos' GA (VGA) has been repeatedly tested and proven to be highly efficient (Kuri,2002). The VGA, therefore, turns out to be an optimization algorithm of broad scope of application and demonstrably high efficiency. A statistical analysis was done by minimizing a large number of functions and comparing the relative performance of six optimization methods[1] of which five are GAs. The ratio of every GA's absolute minimum (with probability $p = 0.95$) relative to the best GA's absolute minimum may be found in Table 1 under the column "Relative Performance". The number of functions which were minimized to guarantee the mentioned confidence level is shown under "Number of Optimized Functions". It may be seen that VGA, in this study, was the best of all the analyzed variations. Interestingly the CGA (the classical or "canonical" genetic algorithm) comes at the bottom of the list with the exception of the random mutation hill climber (RHC) which is not an evolutionary algorithm. According to these results, the minima found with VGA are, in the worst case, more than 25% better than those found with the CGA. Due to its tested efficiency, we now describe in more detail VGA.

As opposed to the CGA, VGA selects the candidate individuals deterministically picking the two extreme (ordered according to their respective fitness) performers of the generation for crossover. This would seem to fragrantly violate the survival-of-the-fittest strategy behind evolutionary processes since the genes of the more apt individuals are mixed with those of the least apt ones. However, VGA also retains the best n individuals out of the $2n$ previous ones. The net effect of this dual strategy is to give variety to the genetic pool (the lack of which is a cause for slow convergence) while still retaining a high degree of elitism. This sort of elitism, of course, guarantees that the best solutions are not lost. On the other hand, the admixture of apparently counterpointed plausible solutions is aimed at avoiding the proliferation of similar genes in the pool. In nature as well as in GAs variety is needed in order to ensure the efficient exploration of the space of solutions[2]. As stated before, all elitist GAs will eventually converge to a global optimum. The VGA does so in less generations. Alternatively we may say that VGA will outperform other GAs given the same number of generations. Besides, it is easier to program because we need not to simulate a probabilistic process. Finally, VGA is impervious

[1] VGA: Vasconcelos' GA; EGA: Eclectic GA; TGA: Elitist GA; SGA: Statistical GA; CGA: Canonical (or Simple) GA; RMH: Random Mutation Hill Climber.

[2] The Latin American philosopher José Vasconcelos proposed that the admixture of all races would eventually give rise to a better one he called the "cosmic" race; hence the algorithm's name.

Algorithm 1 Vasconcelos Genetic Algorithm (VGA)

Require: p_c, p_m, n, G {Crossover probability, mutation probability, number of individuals and number of generations}
Ensure: After G iterations the best individual is the best solution.

```
population ⇐ generatePopulation(n) {Generate random population of n individuals}
l ⇐ getIndividualLength() {Determine the genome length}
bitsToMutate ⇐ n * l * p_m {Calculate the number of bits that will be mutated}
for all individual ∈ population do
    calculateFitness()
end for
population ⇐ orderByFitness(population) {Order population by fitness value}
numIteration ⇐ 0
repeat
    for i = 1 to n/2 do
        individual_1 ⇐ selectIndividual(i)
        individual_2 ⇐ selectIndividual(n − i + 1)
        if p_c ≥ random() then
            crossover(individual_1, individual_2)
        end if
        newPopulation ⇐ newPopulation ∪ {individual_1, individual_2}
    end for
    population ⇐ population ∪ newPopulation {Merge new individuals and previous population}
    for i = 1 to bitsToMutate do
        indexIndividual ⇐ selectIndividualToMutate()
        mutate(indexIndividual)
    end for
    for all individual ∈ population do
        calculateFitness()
    end for
    population ⇐ orderByFitness(population) {Order population by fitness value}
    population ⇐ retainTopN(population) {Retain the best n individuals and discard the worst n individuals}
    numIteration ⇐ numIteration + 1
until numIteration < G
```

to negative fitness's values. We, thus, have a tool which allows us to identify the best values for a set of predefined metrics possibly reflecting complementary goals. For these reasons we use in our work VGA as the optimization method. In what follows we explain our proposal based in the concepts mentioned above

3.2.2 Application of genetic algorithms to clustering

We are now in the position of suggesting an immediate application of a general optimization method to achieve non-traditional clustering. The clear advantage of the utilization of this tool is that we are in the position of selecting arbitrary metrics as a vehicle for clustering. As mentioned above, some of the traditional clustering methods rely on a specific optimization algorithm to operate. A case in point is Kohonen's algorithm which aims at finding a set of centroids (one per neuron) such that similar neurons (and the corresponding centroids) share neighboring positions in the typical 2D space. This is done by a set of consecutive contests between the neurons in the network whereupon the victorious neuron acts as an attractor for the rest of the neurons. Since different neurons are apt to emerge as winners in every iteration, this brilliant approach resembles a competition whose tide comes and goes and ends up with a self-organized configuration. A similar purpose may be attempted by simply specifying a function which minimizes the distance between the neurons in 2D space while, simultaneously, finding a cluster (as per Euclidean distance, for example) for every neuron. When the SOM algorithm was developed, GAs had not yet consolidated as general optimization techniques. As far as we know this approach has not been attempted and here we simply wish to stress the generality of the genetic optimization.

4. Optimization in non-traditional methods

4.1 The data sets

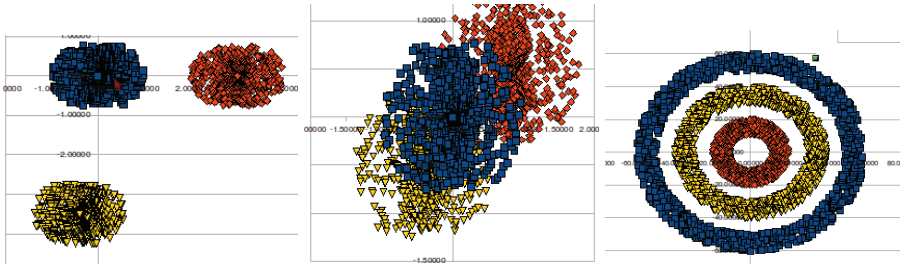
In order to illustrate the performance of non-traditional clustering methods, three types of data sets are analyzed in this work. We shall call them "A", "B" and "C" respectively. Every set is composed of vectors (in a $3D$ space) that belong to three different spheres which we call sphere 1, 2 and 3 respectively. There are $10{,}000$ vectors in each one of the spheres. They were generated from.

$$x = x_0 + r\sin\theta\,\cos\varphi \tag{10}$$

$$y = y_0 + r\sin\theta\,\sin\varphi \tag{11}$$

$$z = z_0 + r\cos\theta \tag{12}$$

from uniformly distributed values for $r \in [0,1)$, $(0 \leq \varphi \leq 2\pi$ and $0 \leq \theta \leq \pi)$. For set A the three centers of the spheres were chosen so that the spheres would not intersect (see Fig.3). In set B, the chosen centers yield partially overlapping data (see Fig. 4(a)). Finally, in set C, the spheres shared a common center (see Fig.4(b)). However, in the last set for sphere 1 $r \in [0,1)$; for sphere 2 $r \in [0,0.666)$; for sphere 3 $r \in [0,0.333)$. In this case, then, spheres 1, 2 and 3 share the same space where the density of 2 is larger than that of 1 and the density of 3 is larger than the other two. Our intent is to choose vectors in set A, B and C whose distribution is not uniform but Gaussian. To achieve this, we determined to divide the space of probabilities of a Gaussian curve in 20 equally spaced intervals. The area under the curve for a normal distribution with $\mu = 0$ and $\sigma = 1$ between -4 and +4 is very closely equal to one. Therefore, it is easy to see that 5%, of the observations will be between -4 and -1.654; 5%, will be between -1.654 and -1.280, etc. The required normal behavior may be approximated by selecting 50 of the uniformly distributed values from the interval $[-4,-1{,}654)$; another 50 from the



(a) Disjoint data set (b) Overlapped data set (c) Overlapped data set

Fig. 10. Spatial distribution of the test data set

interval $[-1.654, -1.280)$, etc. In all we will end up with 1000 vectors for every sphere. These vectors will now be very closely Gaussian. When data is normally distributed, a Bayesian classifier is optimal. The behavior of one such classifier will serve as a base point. To stress: when the distribution of the data set to classify is Gaussian, a Bayesian classifier yields the best theoretical results (by minimizing the probability of classification error independently of the degree of overlap between the distributions of the clusters) (Haykin,1994). Hence, we resorted to Gaussian distributed data in order to establish a behavior relative to the best theoretical one when measuring the performance of non-traditional methods. Our claim is that, if the methods perform satisfactorily when faced with Gaussian data, they will also perform reasonably well when faced with other possible distributions. That is, we wish to show that the results obtained with non-traditional methods are close to those obtained with a Bayesian classifier for the same data set. This would mean that these results correspond to an efficient algorithm. The data sets are ilustrated in Fig. 10.

4.2 Validity index based

There are some methods that consist in finding certain models for clusters and attempting to optimize the fit between the data and the model. There are many ways to attempt the suggested modeling. In particular, one may try to optimize a validity index. Validity indices are used to compare the performance of the clustering results obtained through some given method. If we work "backwards" and optimize the purported index, we should come up with a "good" clustering. The following indices are used with this purpose.

4.2.1 Dunn and Dunn-like validity indices

A cluster validity index for clustering proposed in (Dunn,1973), attempts to identify "compact and well separated clusters". The index is defined by following equation for a specific number of clusters.

$$D_{nc} = min_{nc}\left\{min_{j=i+1,\ldots,nc}\left\{\frac{d(c_i,c_j)}{max_{k=1,\ldots,nc}diam(c_k)}\right\}\right\} \tag{13}$$

where nc is the number of clusters, $d(c_i,c_j)$ is the dissimilarity function between two clusters c_i and cj defined as

$$d(c_i,c_j) = min_{x\in c_i, y\in c_j} d(x,y) \tag{14}$$

and $diam(c_k)$ is the cluster diameter which may be considered as a measure of dispersion of the clusters. The diameter of a cluster C can be defined as follows:

$$diam(C) = max_{x,y\in C}\{d(x,y)\} \tag{15}$$

If the data set contains well-separated clusters, the distance between clusters is usually large and the diameter of the clusters is expected to be small. Therefore a large value is indicative of a better clustering result.

4.2.2 Davies-Bouldin validity index

A similarity measure R_{ij} between the clusters C_i and C_j is defined based on a measure of dispersion of a cluster s_i and a dissimilarity measure between two clusters d_{ij}. The Rij index is defined to satisfy the following conditions (DaviesBouldin,2009):

1. $R_{ij} \geq 0$
2. $R_{ij} = R_{ji}$
3. if $s_i = 0$ and $s_j = 0$ then $R_{ij} = 0$
4. if $s_j > s_k$ and $d_{ij} = d_{ik}$ then $R_{ij} > R_{ik}$
5. if $s_j = s_k$ and $d_{ij} < d_{ik}$ then $R_{ij} > R_{ik}$

These conditions imply that R_{ij} is nonnegative and symmetric. The usual definition of similarity measure is:

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{16}$$

$$d_{ij} = d(v_i, v_j) \tag{17}$$

$$s_i = \frac{1}{\|c_i\|} \sum_{x \in c_i} d(x, v_i) \tag{18}$$

The Davies-Bouldin index measures the average of similarity between each cluster an its most similar one (Kovacs,2006) The value of the index is calculated as follows:

$$DB = \frac{1}{nc} \sum_{i=1}^{nc} R_i \tag{19}$$

The lower value of this index means better clustering result due to the clusters have to be compact and separated.

4.2.3 SD validity index

The SD validity index is based on the concepts of the average scattering for clusters and total separation between clusters (Halkidi et al.,2001). The average scattering is defined as:

$$Scat(nc) = \frac{1}{nc} \sum_{i=1}^{nc} \frac{\|\sigma(v_i)\|}{\|\sigma(X)\|} \tag{20}$$

The total separation between clusters is given by following equation:

$$Dis(nc) = \frac{D_{max}}{D_{min}} \sum_{k=1}^{nc} \left[\sum_{z=1}^{nc} \|v_k - v_z\| \right]^{-1} \tag{21}$$

where $D_{max} = max(\|vi - vj\|)\ \forall i, j \in \{1,2,3,\ldots,nc\}$ is the maximum distance between cluster centers. The $D_{min} = min(\|vi - vj\|)\ \forall i, j \in \{1,2,3,\ldots,nc\}$ is the minimum distance between cluster centers. Now, we can define a validity index based on equations above, as follows

$$SD(nc) = \alpha Scat(nc) + Dis(nc) \tag{22}$$

where α is a weighting factor equal to $Dis(c_{max})$ where c_{max} is the maximum number of input clusters. Lower value of SD index means better clustering results.

4.2.4 Variance of the nearest neighbor (VNN)

This validity index is based on the study of the local environment of a data elements. Formally the deviation of the nearest neighbor distances is determined in every cluster. It is defined as follows:

$$d_{min}(x_i) = min_{y \in C_i}\{d(x_i, y)\} \tag{23}$$

$$d_{min}(C_i) = \frac{\sum_{x_i \in C_i} d_{min}(x_i)}{\|C_i\|} \tag{24}$$

$$V(C_i) = \frac{1}{\|C_i\|^{-1}} \sum_{x_i \in C_i} d_{min}(x_i) - d_{min}(C_i) \tag{25}$$

Based on above equations it is posible to define the validity index called Variance of the Nearest Neighbor Distance (VNND) (Kovacs,2006)

$$VNND = \sum_{i=1}^{nc} V(C_i) \tag{26}$$

This index measures the homogeneity of the clusters. Lower index value means more homogenous clustering. The principal advantage is that this index does not use global references points to calculate its value. The VNND index is based on local information of each point. Therefore, it can measure arbitrary shaped clusters.

4.2.5 Validity index optimization (VIO)

Clearly, the problem of obtaining the best value of a Validity Index is an optimization problem. To this purpose we use a VGA. The individuals of the algorithm have been encoded as follows:

1. The length of the genome is equal to $nc * N$, where nc is the number of clusters and N is the number of dimensions of the data set.
2. To the gene i is assigned a value in $\mathbb{R}$ that represents one coordinate of the center of the $k - th$ cluster.
3. The center of the $k - th$ cluster is given by a sequence of adjacent genes. The length of this sequence is N.

Fig. 11 exemplifies a genome for $nc = 3$ and $N = 3$. The fitness function is given by a particular Validity Index, so that the best individual will be one whose centers values allow to obtain the optimal value of the index. The following tests were performed:

1. VGA was run 20 times (with different seeds of the pseudo random number generator) with three disjoint clusters. The same data set was tested with the Bayesian Classifier. The results obtained are shown in Table 2.

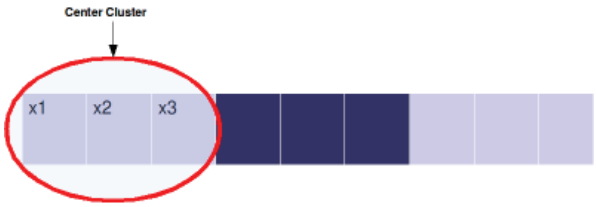


Fig. 11. Genome of the individual ($nc = 3$ and $N = 3$)

Index	Type of Optimization	Average Effectiveness
Dunn and Dunn	Maximize	99.00
SD Validity Index	Minimize	99.00
Davies-Boulding	Minimize	99.00
Nearest Neighbor Distance	Minimize	99.00
Bayesian Classifier Efectiveness		99.00

Table 2. Results obtained with VIO for different index and disjoint clusters

2. VGA was run 20 times (with different seeds of the pseudo random number generator) with three overlapping clusters. The same data set was tested with the Bayesian Classifier. The results obtained are shown in Table 3.
3. VGA was run 20 times (with different seeds of the pseudo random number generator) with three concentric clusters. The same data set was tested with the Bayesian Classifier. The results obtained are shown in Table 4.

The following results (Table 2) correspond to the data set with disjoint clusters. We can see that all methods show similar trends. We believe that this fact is due to the spatial distribution of the clusters. In Table 3 and Table 4 are shown the results obtained with the data set with overlapping clusters. We can see that the effectiveness is reduced substantially. However the SD Validity index yields better results with respect to other indices and close results relative to a Bayesian Classifier.
The results obtained show that the best index is SD for the data set used. We can see that the proposed method yields ”good” results relative to a Bayesian classifier.

4.3 Entropy based

The Entropic Evolutionary Clustering is a method based on the measurement of the information contained in data set D. This approach is based on maximizing density in terms of the entropy of the area of the space that represents a cluster.

4.3.1 ENCLUS

Cheng et al (Cheng,1999), developed an algorithm called ENCLUS (Entropic Clustering) in which they link the entropy with two concepts that the authors call *coverage* and *density*. These are determined by the space's segmentation. Segmentation is made iteratively. Thereafter, several conditions have to be satisfied for every iteration of the algorithm. The space segmentation is a partition on non-overlapping rectangular units based on CLIQUE (Clustering in Quest) algorithm where a unit is dense if the fraction of the elements contained

Index	Type of Optimization	Average Effectiveness	Ratio
Dunn and Dunn	Maximize	67.00	0.77
SD Validity Index	Minimize	71.00	0.82
Davies-Boulding	Minimize	43.00	0.49
Nearest Neighbor Distance	Minimize	67.00	0.77
Bayesian Classifier Efectiveness		87.00	

Table 3. Results obtained with VIO for different index and overlapping clusters

Index	Type of Optimization	Average Effectiveness	Ratio
Dunn and Dunn	Maximize	43.00	0.63
SD Validity Index	Minimize	59.00	0.87
Davies-Boulding	Minimize	41.00	0.60
Nearest Neighbor Distance	Minimize	57.00	0.84
Bayesian Classifier Efectiveness		68.00	

Table 4. Results obtained with VIO for different index and concentric clusters

in the unit is greater than a certain threshold. A cluster is the maximum set of connected dense units.

4.3.2 COOLCAT

Another similar work is the so-called COOLCAT algorithm (Barbara,2002) which also approaches the clustering problem on entropic considerations but is mainly focused on categorical sets of data.

4.3.3 Fixed grid evolutionary entropic algorithm (FGEEA)

The Fixed Grid Evolutionary Entropic Algorithm is a clustering method based on the measurement of the information contained in data set D. This is based on the assumption that the areas of space with more information represent a cluster. This fact is illustrated in Fig. 12. The steps of the algorithm are:

(1) The metric space D is transformed to a metric space D' where its elements are partitions of the space D such that each contains zero or more elements of D. (see example in Fig. 13)

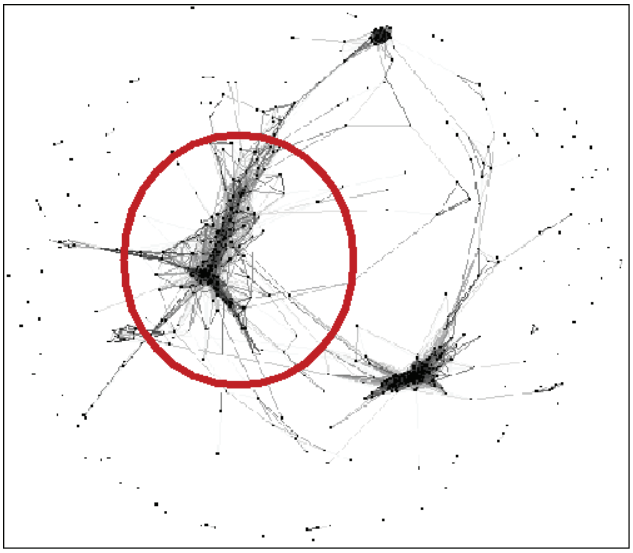


Fig. 12. Example of a high density area (with less information

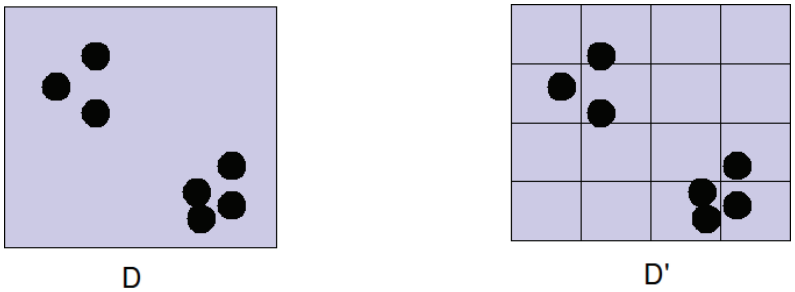


Fig. 13. Transformation of D to D'. The elements on D' are represented as cells

(2) The amount of information of an element(cell) of D' is given by the number of elements (of the original space D) that belong to it. Here each partition or cell of D' is equivalent to a symbol s and the space D' is equivalent to the source S.

(3) In general for a n-dimensional space D, its transformation D' is denominated Hypercube Wrapper (HW). In Fig. 14 is shown a HW in 3D.

(4) Each partition or cell of HW is called a "voxel".

(5) The entropy of HW is:

$$H(HW) = \sum_{i=1}^{n} p(s_i)I(s_i) = -\sum_{i=1}^{n} p(s_i)log_2(p(s_i)) \tag{27}$$

where s_i is a voxel and $HW = S$.

(6) Our hypothesis is that areas with high density have minimum entropy with respect to areas with low density. Therefore the areas with minimum entropy correspond to a cluster. To determine the entropy of a cluster we introduce a concept we call intracluster entropy, defined as:

$$H(c_i) = \sum p(s_j)log_2(s_j) \qquad \forall s_j \in c_i \tag{28}$$

Where $H(c_i)$ is the intracluster entropy of $i-th$ cluster. In order to determine that s_j belongs to c_i we use a genetic algorithm, as discussed in what follows.

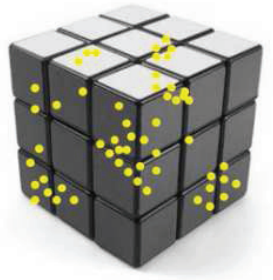


Fig. 14. Hypercubic Wrapper in a tri-dimensional space where the points represent elements of the data set and the subdivisions are voxels

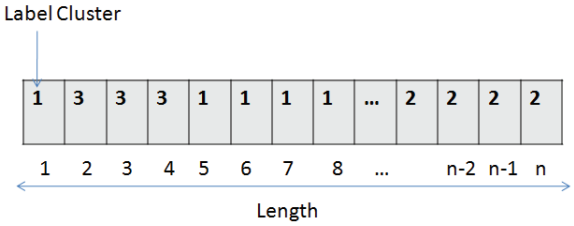


Fig. 15. Genome of the individual (k=3)

4.3.3.1 Application of VGA

Our aim is to find the areas of the subspace HW where the entropy is minimal. We find groups of voxels such that each group has minimal entropy (intracluster entropy). Clearly this is an optimization problem which we tackle with VGA. The individuals of the algorithm have been encoded as follows:

a. The length of the genome is equal to $|HW|$. It is composed by all symbols (or voxels).

b. Each gene is assigned a label that represents the cluster to which it belongs.

c. It has a sequential index. Such index will allow mapping all symbols to subspace HW. Fig. 15 exemplifies a genome for $k = 3$.

Now, we define the fitness function as:

$$f(individual_i) = min \sum_{j=0}^{k} H(c_j) \qquad for \qquad i \leq N \tag{29}$$

Subject to:

$$\sum_{j=0}^{k} H(c_j) \geq H(S) \tag{30}$$

$$\left| \sum_{j=0}^{k} H(c_j) - H(S) \right| > \Delta_1 \tag{31}$$

Where N is the size of the population and Δ_1 is a parameter that represents a threshold of the difference between the sum of intracluster entropies and the entropy of source S. Additionally we have introduced a constraint called "intracluster density" defined as:

$$dc_i \leq \epsilon \tag{32}$$

where ϵ is the threshold density. One last constraint is the intracluster density (dc_i). It is the number of elements of data set X which belong to the symbols of $i - th$ cluster:

$$dc_i = \frac{\alpha}{\beta} \tag{33}$$

where α is the number the elements that belong to the data set X and β is the number of symbols within cluster i. This constraint ensures that entropy is minimal within any given cluster. The algorithm yields a best individual which represents a set of clusters of symbols that are map into sets of voxels in HW, as shown in Fig. 16

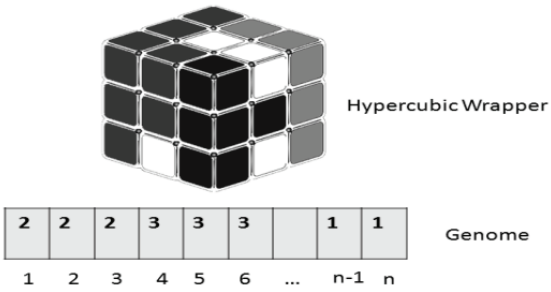


Fig. 16. Possible clustering delivered by VGA. Different intensities in the cube represent a different cluster. (Empty voxels are white).

4.3.3.2 Experimental results

Our algorithm was tested with three synthetic data sets that were described above. The values of the parameters of FGEEA are given in Table 5. This values were determined experimentally. The VGA was run 20 times (with different seeds of the pseudo random number generator) per data set. The same data sets was tested with other algorithms such as K-Means, Kohonen Maps and Fuzzy C-Means. As in the VIO method all tests include a test with the Bayesian Classifier. The results obtained with disjoint clusters are shown in Table 6. This allows us to see that the results of FGEEA are similar to those given by some alternative algorithms. The high effectiveness in all cases is due to the spatial distribution of data set.
The results obtained with overlapping clusters are shown in Table 7 where we can see that the effectiveness decreases significantly in general. However FGEEA showed better results than traditional methods and close results to Bayesian Classifier.
The results obtained in the two last cases (overlapping and concentric clusters) are due to the fact that it is not possible to find a simple separable boundary. Therefore, the boundary decision is unclear and the vast majority of the clustering methods yield poor solutions. The closeness of the results obtained so far (from VIO and FGEEA) relative to a Bayesian Classifier, tells us that both of these approaches are quite efficient. In future works we will report on experiments encompassing a wider range of data sets. We expect them to further support our hypothesis.

4.4 Membership based

As stated, when we approach the clustering problem as the search for the elements in a cluster which belong to a locus we are actually looking for membership functions. There is one

Parameter	Value
N (Number of Individuals)	500
G(Generations)	1000
p_m (Mutation probability)	0.001
p_c(Crossover Probability)	0.99
ϵ	5
δ_1	$3.5 \leq 3.6$

Table 5. Parameters test

Algorithm	Average Effectiveness
FGEEA	0.98
K-Means	0.99
Kohonen Maps	0.99
Fuzzy C-Means	0.98
Bayesian Classifier Efectiveness	0.99

Table 6. Results obtained with disjoint clusters data set

membership function for every cluster and we wish to find:

a. The position of the locus in N-space and

b. The precise definition of parameters the function

The initial work of Gielis had to do with the generalization of the formula of an ellipsoid. By enriching the set of defining parameters, he was able to find complex and somewhat irregular bodies in 2D and 3D. Our aim is to take advantage of this approach when trying to encompass the elements of the data set in a way such that no elements are left out of the "bodies" while, simultaneously, the density of the clusters is appropriate. In this sense, every membership function is determined by a core family of functions. Every core defines a different approach to the problem. We say that any plausible approach represents a kernel function. Gielis' formula represents only one of these kernels. Other kernels are possible, for which see section 4.4.2

4.4.1 Clustering with an n-dimensional extension of Gielis superformula

It is possible to apply Gielis' "superformula" to tackle the problem of clustering in an n-Dimensional space. The formula was originally developed to tackle data in $2D$. Later it was extended to $3D$. According to Gielis, it is possible to extend the formula to 3, 4, or n dimensions, by means of spherical product of superformulas. However, the problem of finding these products is non-trivial and difficult to undertake as n grows.

4.4.1.1 Gielis' formula

Gielis superformula (GSF) generalizes the equation of a hyper-ellipse by introducing some parameters which increase the degrees of freedom of the resulting figures when geometrically interpreted, as remarked when discussing equation (8). By using this approach we were able to find the parameters and positions of the locus (i.e the membership functions) in practical 3D problems. The solution we describe assumes that the number of clusters is already known. To illustrate our method we consistently assumed, without loss of generality, that there are 4 clusters and we are working on 3D. Firstly, we generate 4 n-dimensional object (in what

Algorithm	Average Effectiveness	Ratio
FGEEA	0.72	0.83
K-Means	0.51	0.59
Kohonen Maps	0.66	0.76
Fuzzy C-Means	0.15	0.17
Bayesian Classifier Efectiveness	0.87	

Table 7. Results obtained with overlapping clusters data set

Algorithm	Average Effectiveness	Ratio
FGEEA	0.53	0.78
K-Means	0.36	0.53
Kohonen Maps	0.47	0.69
Fuzzy C-Means	0.23	0.34
Bayesian Classifier Efectiveness	0.68	

Table 8. Results obtained with concentric clusters data set

follows NDO). In this case each NDO corresponds to a $3D$ cluster. These are shown in Fig. 17 For every cluster we calculated 2,500 coordinates. Therefore we got a data set ($3D$ coordinates) of size 10,000. Hence, there are 10,000 elements that we know, a priori, belong to the 4 clusters defined by the NDOs. The goal was to use this set as an input for VGA in a way such that we may verify that the values of the parameters in GSF correspond to those of Fig. 17

4.4.1.2 Encoding the problem for the genetic algorithm

The variables to optimize are m, n_1, n_2, n_3, a, b for each of the clusters. The encoding of one cluster for VGA was discussed in section 2.5. We have introduced variables cx, cy, cz which correspond to the centers of the clusters. For illustration purposes we found the initial coordinates for these centers by applying a fuzzy-c means algorithm. This allows us to compare the results obtained from VGA and those used to define the clusters we used. Fig. 18 shows the $3D$ positions of the centers. The encoding chromosome was shown in Fig. 8 and Fig. 9

4.4.1.3 Fitness function

The fitness for the individuals in the population is given by the correct membership assignment of every vector in the data set to the NDOs given the parameters encoded in the individual. For instance, in a set of size N assumed to lie within 4 clusters a "good" individual is one in which the parameters in GSF yield 4 NDOs which include all N vectors. Hence, the fitness of an individual is given by the number of vectors lying within the clusters encoded in the individual's chromosome. In Fig. 19(a) we may see an individual whose genome corresponds to 4 NDOs which include 5 vectors out of 10. Hence, its fitness is 5. It may happen that an individual includes all vectors in a single NDO as shown in Fig. 19(b). In such case, even if the inclusion is total, the individual is less than optimal and this representation must be penalized. To this effect we introduced an index given by the ratio of the number of clusters induced by the individual and the target number of clusters. In the example these quotient would evaluate to 1/4 or 0.25. If we now multiply this index times the included number of vectors the individual induces we get which is the proposed fitness function. The following equation generalizes this criterion.

$$f(individual_i) = N * \frac{nc_{vga}}{nc} \tag{34}$$

where nc_{vga} is the number of clusters induced by an individual (or induced by VGA) and nc is the desired number of clusters.

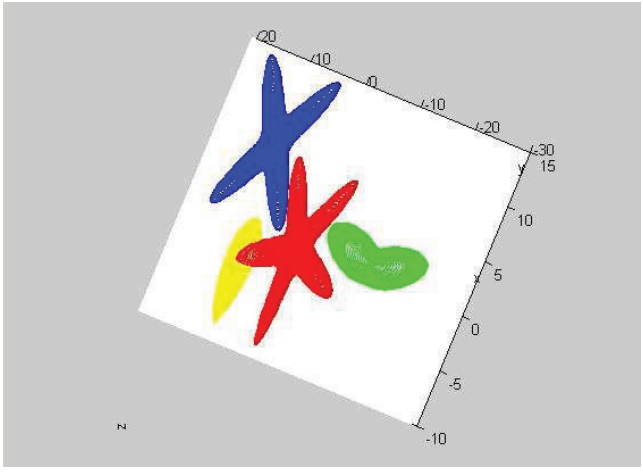
4.4.1.4 Cluster membership for an NDO

A very basic subproblem one has to cope with is how to determine whether a given vector lies inside a given NDO. To this effect we first generate a grid over the surface of the NDO.

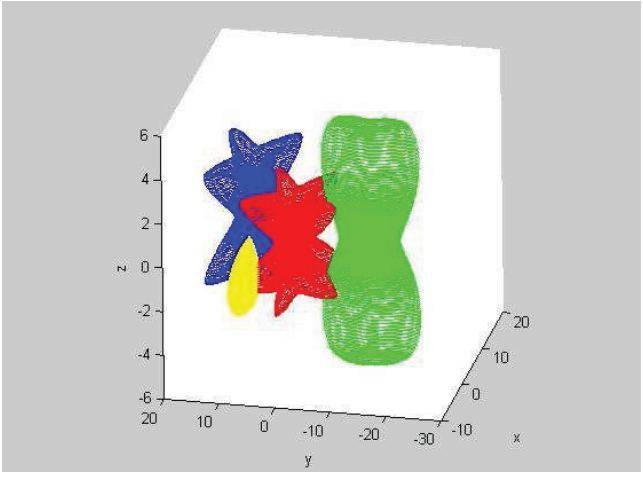
We applied a triangulation algorithm called "ear clipping" which, for every 3 non-consecutive points, generates a triangle belonging to the grid. In the end the algorithm yields a number of triangles leading to a body's characterization as shown in Fig. 20. To test whether a point lies within the NDO we make, for every triangle, a projection which has an angle of spread. This defines an orthogonal cone on the grid. One then tests algebraically whether the point under analysis is within the cone.

4.4.1.5 Experimental results

Here the data set differs the previous methods. A synthetic data set (in *tri*-dimensional space) was generated through the 'superformula'. The data set is composed by four clusters (NDOs)



(a) NDOs projected into (x,y)



(b) NDOs represented in (x,y,z)

Fig. 17. Two views of the NDOs

Algorithm	Average Effectiveness
Gielis-VGA	0.97
K-Means	0.89
Kohonen Maps	0.91
Fuzzy C-Means	0.96
Bayesian Classifier Efectiveness	0.98

Table 9. Results obtained with clusters data set generated through the Superformula

with some overlapping degree. Our goal was to find through method described the "Gielis Bodies" that encompass the data set. Thus, this bodies should be similar to the NDO generated a priori to construct the data set. The VGA was executed 20 times with a sample of size 10,000 whose memberships are known for 4 clusters. The results gotten so far are preliminary (see Table 9). It is necessary to confirm the effectiveness of the method with data sets as those used in the methods described above. Although we have achieved reasonable clustering for synthetic data where traditional clustering techniques perform poorly, several issues remain to be solved.

4.4.1.6 Considerations

The method is unique in the sense that it is the only one in which there are strictly no metric considerations involved. Therefore, it the only one guaranteed (in principle) to find an acceptable solution to problems such as the one corresponding to data set "C". However, several issues require attention in order for the method to attain practical generality.

a. The fitness function has to be refined. It is not clear whether a simple strategy as outlined above will be adequate in most cases.

b. The computation involved in the determination of the membership of the vectors to an NDO has to be improved, given the computationally intensive nature of the evolutionary algorithm.

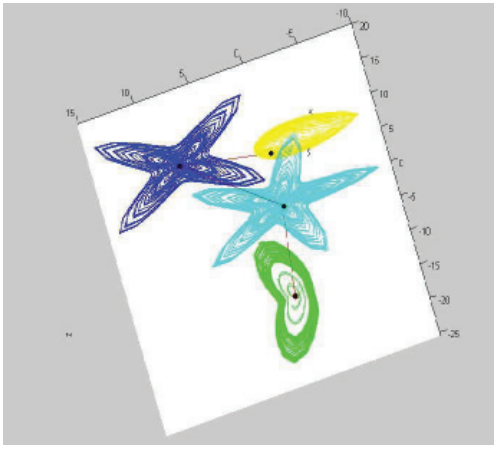
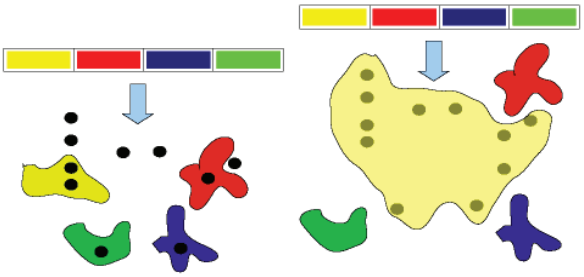


Fig. 18. Centers of the NDO's projected in *bi*-dimensional plane.



(a) Individual whose genome achieves partial inclusion for a data set

(b) Full inclusion in a single set

Fig. 19. Clustering proposal by the different chromosomes

c. The best spread angle has to be determined a priori. We would like to relieve the user from this task.

d. As the number of clusters increases so does the number of parameters. This leads to an even higher computational cost. Some initial attempts at parallelizing the algorithms involved are under way (Guadarrama,2010). Fortunately, this fact is relatively natural because GAs are, by their nature, easily parallelizable. When working in highly parallel computers, therefore, the problem becomes more easily tractable. And, since the technological tendency is to increase the processors both in computer clusters as in the individual chips, this approach seems promising.

e. Most importantly, Gielis' formula, as it stands, is extremely difficult to generalize to n-dimensional spaces. In data mining, clearly, higher dimensional spaces are most common. And these spaces are not amenable to the application of the method as of today. Nonetheless, there are many applications (for example, in robotics, medical applications and image processing, in general) which may greatly benefit form this approach.

All in all we know, from initial experiments, that this method will perform adequately where others simply will not do because of conceptual constraints, as stressed in the introduction. In the past it has been customary to verify the goodness of the clusters gotten by a given method through tests which emphasize one or several criteria. In this method the goodness of the cluster is a necessary condition for the clusters to be found. Therefore, there is no need

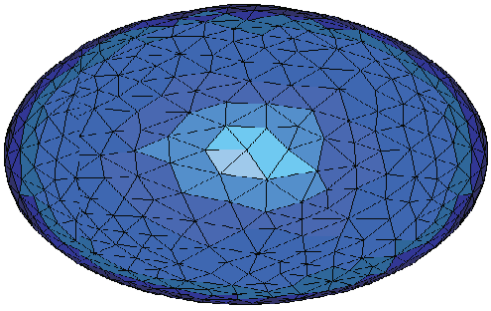


Fig. 20. Surface grid on a 3D NDO

for "outside" validity measures. In fact, the method may be seen as a dynamic measure of such validity, which is another interesting issue. That is, the validity is defined as the one maximizing the set of vectors lying in the NDOs.

4.4.2 Alternative methods

Since Gielis' formula is an extension of a hyper-ellipse, it displays some of the limitations of the resulting locus. In other words, there is no absolute freedom of choice of the possible forms of the locus involved. It is relatively simple to envision alternative choices for the generating kernels. For example, one might try rational functions (Hazewinkel,2001), in general; Padé approximants (Baker & Graves,1996); radial basis functions (Buhmann,2003) and combinations of these. The application of the mathematical formulations of different kernels allows to generalize to n-dimensional spaces. And this generalization is relatively simple. Furthermore, it allows us to find richer sets of n-dimensional locus (or, equivalently, richer sets of membership functions). The promise of this approach is well worth exploring

5. Conclusions

We have analyzed three ways to avoid the limitations of hyper-spherical clusters. We have shown that there is interesting experimental evidence that these methods yield satisfactory approximations. This is true even in the case where the original groups are definitely non-spherical. Of the three approaches, the optimization of validity indices may be considered a reasoned alternative to the usual practice of defining a metric; then testing the results casuistically to determine their usefulness. The density based approach, although (strictly speaking) is based on a distance, relies on the idea of defining an n-dimensional mesh a priori and then determining the amount of information. It differs from other density based methods in that the characteristics of the mesh are a parameter, rather than the result of the search. Finally, the methods based on membership functions seem to hold the greater promise. They also present the greatest challenge on two accounts. First, the computational cost is very high. Second, the systematic generalization to N-space is problematic. In all three cases, in general, the methods are guaranteed to yield better results than typical clustering algorithms because they will uncover the patterns underlying the elements of a cluster regardless of its shape.

6. References

[Agrawal et al.,1998] Agrawal, Rakesh, Gehrke, Johannes, Gunopulos, Dimitrios, and Raghavan, Prabhakar: Automatic subspace clustering of high dimensional data for data mining applications, SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data, ACM, 94–105, 1998

[Agresti,2002] Agresti A.: Categorical Data Analysis. Wiley Series in Probability and Statistics. Wiley-Interscience, 2nd edition, 2002.

[Bhattacharyya,1943] Bhattacharyya A.: On a measure of divergence between two statistical populations defined by their probability distributions, Math. Soc 35, volume 35, 99–109, 1943

[Bäck,1996] Bäck, Th.: Evolutionary Algorithms in Theory and Practice, Oxford University Press, 1996

[Baker & Graves,1996] Baker , G. A., Jr. and Graves-Morris, P.: Padé Approximants. Cambridge U.P., 1996.

[Barbara,2002] Barbara, Daniel, Li, Yi, and Couto, Julia: COOLCAT: an entropy-based algorithm for categorical clustering, CIKM, ACM, 582–589, 2002

[Buhmann,2003] Buhmann, Martin D.: Radial Basis Functions: Theory and Implementations, Cambridge University Press,2003

[Cha,2008] Cha, Sung-Hyuk: Taxonomy of nominal type histogram distance measures, MATH'08: Proceedings of the American Conference on Applied Mathematics, World Scientific and Engineering Academy and Society (WSEAS), 325–330, 2008

[Chandola et al.,2009] Chandola V., Boriah S., and Kumar V.: A framework for exploring categorical data. In SDM, pages 185-196, 2009.

[Chang & Ding,2005] Chang C. and Ding Z.: Categorical data visualization and clustering using subjective factors. Data Knowl. Eng., 53(3):243-262, 2005.

[Cheng,1999] Cheng, Chun Hung, Fu, Ada Wai-Chee, and Zhang, Yi: Entropy-based Subspace Clustering for Mining Numerical Data, KDD, 84–93, 1999

[DaviesBouldin,2009] Davies, David L., and Bouldin, Donald W.: A Cluster Separation Measure, Pattern Analysis and Machine Intelligence, IEEE Transactions on, Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-1(2), volume PAMI-1, 224–227, January 2009

[Dunn,1973] Dunn, J. C.: A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters, Journal of Cybernetics 3, volume 3, 32–57, 1973

[Ester,1996] Ester, Martin, Kriegel, Hans-peter, Jörg, S, and Xu, Xiaowei: A density-based algorithm for discovering clusters in large spatial databases with noise, , AAAI Press, 226–231, 1996

[Forrest,1993] Forrest, and Mitchell: What Makes a Problem Hard for a Genetic Algorithm? Some Anomalous Results and Their Explanation, MACHLEARN: Machine Learning 13, volume 13, 1993

[Gibson,2000] Gibson D., Kleinberg J., and Raghavan P.: Clustering categorical data: an approach based on dynamical systems. The VLDB Journal,8(3-4):222-236, 2000.

[Gielis,2003] Gielis, Johan: A generic geometric transformation that unifies a wide range of natral and abstract shapes, American Journal of Botany 90(3), volume 90, 333–338, 2003

[Guadarrama,2010] Guadarrama, C.:Data Compression through Pattern Recognition, Master's Thesis, IIMAS UNAM, 2010, to be published.

[Guha,1998] Guha, Sudipto, Rastogi, Rajeev, and Shim, Kyuseok: CURE: an efficient clustering algorithm for large databases, SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data, ACM, 73–84, 1998

[Halkidi et al.,2001] Halkidi, Maria, Batistakis, Yannis, and Vazirgiannis, Michalis: On Clustering Validation Techniques, J. Intell. Inf. Syst. 17(2-3), volume 17, 107–145, 2001

[Hazewinkel,2001] Hazewinkel, Michiel: Rational function, Encyclopaedia of Mathematics, Springer, 2001

[Haykin,1994] Haykin, Simon: Neural networks: A comprehensive foundation, MacMillan, 1994

[Hinneburg et al,1998] Hinneburg, Alexander, Hinneburg, Er, and Keim, Daniel A.: An Efficient Approach to Clustering in Large Multimedia Databases with Noise, , AAAI Press, 58–65, 1998

[Kohonen,1997] Kohonen, Teuvo: Self-organizing maps, Springer-Verlag New York, Inc., 1997

[Kovacs,2006] Kovacs, Ferenc, and Ivancsy, Renata: A novel cluster validity index: variance of the nearest neighbor distance., WSEAS Transactions on Computers, volume 3,

477–483, March 2006

[Kuri & Aldana,2008] Kuri-Morales, Angel, and Bobadilla, Edwin Aldana: Clustering with an N-dimensional extension of Gielis superformula, AIKED'08: Proceedings of the 7th WSEAS International Conference on Artificial intelligence, knowledge engineering and data bases, World Scientific and Engineering Academy and Society (WSEAS), 343–350, 2008

[Kuri,2002] Kuri-Morales, Angel Fernando: A Methodology for the Statistical Characterization of Genetic Algorithms, MICAI, volume 2313, Springer, 79–88, Eds: Coello, Carlos A. Coello, de Albornoz, Alvaro, Sucar, Luis Enrique, and Battistutti, Osvaldo Cairó, 2002

[Kuri & Aldana,2010] Kuri-Morales, Ángel Fernando, and Aldana-Bobadilla, Edwin: Finding Irregularly Shaped Clusters Based on Entropy, ICDM, volume 6171, Springer, 57–70, Eds: Perner, Petra, 2010

[Kuri & Garcia,2010] Kuri-Morales, Ángel Fernando, and Garcia-Garcia, Javier: Encoding Categorical Variables for Unsupervised Clustering in Large Databases, 2010, to be published.

[Yang et al.,2000] Le Cam, Lucien, and Lo Yang, Grace: Asymptotics in Statistics Some Basic Concepts , volume XIII, 2nd edition, Springer Series in Statistics, 2000

[Li,1999] Li, Xin, Mak, Man-Wai, and Li, Chi-Kwong: Determining the Optimal Number of Clusters by an Extended RPCL Algorithm, JACIII 3(6), volume 3, 467–473, 1999

[Mahalanobis,1936] Mahalanobis, P. C: On the generalised distance in statistics, In Proceedings National Institute of Science, World Scientific and Engineering Academy and Society (WSEAS), 49–55, 1936

[MacQueen,1967] McQueen, J. B.: Some Methods of Classification and Analysis of Multivariate Observations, Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability, 281–297, Eds: Cam, L. M. Le, and Neyman, J., 1967, first publication about K-Means algorithm?

[Ng,1994] Ng, Raymond T., and Han, Jiawei: Efficient and Effective Clustering Methods for Spatial Data Mining, Department of Computer Science, University of British Columbia No. TR-94-13, May 1994.

[Pollard,2002] Pollard, David: A User's Guide to Measure Theoretic Probability, Cambridge University Press, Cambridge, 2002

[Rudolph,1994] Rudolph, G.: Convergence Analysis of Canonical Genetic Algorithms, IEEE Transactions on Neural Networks 5(1), volume 5, 96–101, January 1994

[Shannon,1949] Shannon, C. E., and Weaver, W.: The Mathematical Theory of Communication, Scientific American, July 1949

[Sheikholeslami et al.,1998] Sheikholeslami, Gholamhosein, Chatterjee, Surojit, and Zhang, Aidong: WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases, VLDB, Morgan Kaufmann, 428–439, Eds: Gupta, Ashish, Shmueli, Oded, and Widom, Jennifer, 1998

[Wang,2000] Wang, W., Yang, J., and Muntz, R. R.: An Approach to Active Spatial Data Mining Based on Statistical Information, IEEE Transactions on Knowledge and Data Engineering 12(5), volume 12, 715–728, 2000

[Zhang,1996] Zhang, T., Ramakrishnan, R., and Livny, M.: BIRCH: an efficient data clustering method for very large databases, Proceedings of ACM-SIGMOD International Conference of Management of Data, 103–114, June 1996