DISTRIBUTED CLUSTERING BASED ON REPRESENTATIVES EVOLVED BY CROWDING DIFFERENTIAL EVOLUTION

Daniela Zaharie

Department of Computer Science, West University of Timişoara
bv. V. Pârvan, no. 4, 300223 Timişoara, ROMANIA
tel.:+40-256-592157, fax.: +40-256-592319, e-mail:dzaharie@info.uvt.ro

Abstract. *By interpreting clusters as regions with high density of data, the clustering process can be formulated as a multi-modal optimization problem. In a previous work we applied a simple crowding-based extension of the differential evolution algorithm in order to solve the multi-modal optimization problem associated to a clustering task. The role of differential evolution was to evolve some clusters representatives consisting of center vectors and scale parameters. The aim of this work is to analyze how can be used these representatives in the context of distributed clustering.*

Keywords: *differential evolution, crowding, distributed clustering, representatives, density functions.*

1. Introduction. Data clustering aims to extract useful knowledge by identifying natural groups of similar data such that the data in the same group (cluster) are sufficiently similar while data in different groups are sufficiently dissimilar. The clustering problem can be interpreted as an optimization problem in at least two ways. The first one is that of searching for a set of clusters (a data partition) which optimizes some clustering goodness criteria. The simplest approach here is that of maximizing the intra-cluster similarity while minimizing the inter-cluster similarity. Unfortunately such an approach could lead to degenerate partitions (e.g. each data defines a cluster) so constraints concerning the number of clusters or the minimal number of elements in each cluster should be introduced. These constraints increase the difficulty of the optimization problem and the clustering algorithm should find a trade-off between different criteria. Therefore approaches based on multi-objective optimization have been recently proposed [11],[3].

The second optimization approach in clustering is based on the idea that clusters are high-density regions and identifying them means finding local maxima of a density function. This approach appears in density-based clustering and two of the most representative algorithms of this type are DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [1] and DENCLUE (Density Clustering) [5]. An advantage of density-based algorithms is the fact that they are robust enough to identify clusters in noisy data. The main disadvantage of these approaches is that they are difficult be implemented.

DBSCAN does not use an explicit density function being based on concepts like dense neighborhoods (used to identify the so-called core points) and density based connectivity (which allows identifying arbitrary shaped clusters). The results produced by DBSCAN are highly influenced by the neighborhood radius and the density threshold (minimum number of data in a dense neighborhood).

DENCLUE uses a measure of density constructed based on some influence functions which measure the influence that data have in their neighborhood. A common influence function is the gaussian one, i.e. the influence a data x has on a data y is expressed as $f(x,y) = \exp(-d(x,y)^2/(2\sigma^2))$ where $d(x,y)$ is a dissimilarity measure. The density in x can then be defined as $D(x) = \sum_{y \in \mathcal{N}_x} f(x,y)$, $\mathcal{N}_x$ denoting a neighborhood of x. The clusters are identified by the so-called density attractors defined as local maxima of the density function, $D(x)$. For a given data the corresponding attractor is determined by a local optimization based on gradient information. In order to identify arbitrarily shaped clusters the concept of high density path, similar with density based connectivity in DBSCAN, is used. The decision if the density is high or low is based on a threshold ξ. This threshold is used also to separate the data from noise and as long as its value is adequately chosen it leads to a successful cleaning of data. The ability of DENCLUE to identify the true clusters is highly dependent on the parameters σ and ξ.

The optimization problem related to a clustering task is a difficult one. In order to overcome this difficulty many researchers tried to use evolutionary algorithms [2], [6]. Most of the evolutionary approaches address the clustering problem by searching in an evolutionary manner for a data partition which optimizes a clustering goodness criterion. In this case each element of the population will encode a partitioning, e.g. a mapping between data and clusters identifiers. A similar representation is used also in the case when multiple criteria are taken into account and the problem is solved by using a multi-objective evolutionary algorithm [3].

The evolutionary approaches of density-based clustering are fewer [14], [19]. In this case the aim of the evolutionary algorithm is to find the local maxima of the density function, thus each element of the population will be a candidate for a local maxima. The local maxima identified by the evolutionary process defines some representatives of clusters and from these representatives both the number and the content of clusters can be obtained. The unsupervised niche clustering (UNC) algorithm [14] evolves a population of cluster

centers by using a genetic algorithm combined with a deterministic crowding mechanism [13] in order to allow the identification of all local maxima. UNC supposes that the data in a cluster are generated by a normal distribution (the mean of the normal distribution corresponds with the center of the cluster and the covariance matrix contains information about the scale and the orientation of the cluster). Thus each cluster is identified by a center and some scale and orientation parameters. The centers are determined by a genetic algorithm while the other parameters are estimated by using the current values of the centers.

The approach we introduced in [19] (CDE-clustering) starts from the same underlying idea as UNC but there are some differences: (i) UNC uses a gray coding genetic algorithm while CDE-clustering is based on a Differential Evolution algorithm combined with a crowding mechanism [16]; (ii) in UNC each cluster is described by an arbitrary hyper-ellipsoid described by a center and set of scale and orientation parameters while in CDE-clustering a cluster can be described by a set of normally oriented hyper-ellipsoids (each hyper-ellipsoid is described by a center and set of scale parameters); (iii)UNC evolves only the cluster centers while in CDE-clustering both the centers and the hyper-ellipsoids scales are evolved.

However in both approaches the result of the clustering process is a set of representatives which can be used in order to obtain the data partition, usually by applying a nearest neighbor classification with respect to the representatives. On the other hand extracting representatives from data plays an important role in distributed clustering. Distributed data mining aims to extract knowledge from data distributed at different sites without unifying all data at a central site. In the last years different approaches to cluster distributed data have been proposed [7], [8], [9]. The basic idea of all these approaches is to construct local models by applying a local clustering algorithm to each site, to collect at a central site all these local models into a global one which is then sent to local sites in order to classify the data at their place. The local and global models usually consist of descriptors (representatives) of clusters.

The aim of this paper is to analyze how can be extended the CDE-clustering approach proposed in [19] in order to deal with distributed data.

Details on the structure of CDE-clustering are presented in the next section while the particularities of distributed clustering are analyzed in section 3. Section 4 contains results obtained for some test data and section 5 concludes the work.

2. CDE-clustering.

The CDE-clustering is a density based algorithm which exploits the formulation of the clustering problem as a multi-modal optimization one. It uses the sequential differential evolution combined with a crowding mechanism proposed in [16]. The differential evolution is applied here to identify representatives of clusters consisting of clusters centers and scale parameters. A different application of differential evolution in clustering is that presented in [10] where the elements of the population correspond to different data partitions.

In the following we shall describe the multi-modal optimization problem associated to the clustering problem and the main elements of CDE-clustering in order to identify the particularities which allow extending this approach to distributed clustering.

2.1. Formulating the clustering problem as a multi-modal optimization problem.

The key element in the multi-modal approach in clustering is constructing a function whose local maxima corresponds to dense regions allowing the identification of clusters representatives. Let $X = \{x_1, \ldots, x_N\}$ be a set of n-dimensional data, $x_i = (x_i^1, \ldots, x_i^n)$. A natural density function, $f : \mathbb{R}^n \to \mathbb{R}$, is a sum of gaussians:

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{s} \exp\left(-\frac{1}{2} d^2(x, x_i)\right) \tag{1}$$

where $s > 0$ is a normalization parameter end d is a distance function. Different hypotheses concerning the clusters shapes lead to different distances and normalization parameters, s. The simplest case is when we suppose that all clusters are spherical of radius σ. This case corresponds to the euclidean distance:

$$d_\sigma^2(x, y) = \frac{1}{\sigma^2} \sum_{j=1}^{n} (x^j - y^j)^2, \qquad s = \sigma^{n/2} \tag{2}$$

A similar density function, based on only one parameter, σ, is used in DENCLUE [5].

When the clusters are considered to be hyper-ellipsoids with the same orientation as the axes and having the radii $\sigma^1, \ldots, \sigma^n$ then d is a particular case of the Mahalanobis distance:

$$d_{\sigma^1 \ldots \sigma^n}^2(x, y) = \sum_{j=1}^{n} \frac{(x^j - y^j)^2}{(\sigma^j)^2}, \quad s = (\sigma^1 \cdots \sigma^n)^{1/2} \tag{3}$$

This is the distance used in CDE-clustering. Each cluster is described through a set of representatives and each representative is a pair (c, σ) where $c \in \mathbb{R}^n$ is a vector in the data domain and $\sigma \in \mathbb{R}_+^n$ is the vector containing the scale parameters.

The general case corresponds to hyper-ellipsoids of arbitrary orientation. In this case the scale and the orientation are described by a covariance matrix, Σ, and d is the general Mahalanobis distance:

$$d_\Sigma^2(x,y) = (x-y)^T\Sigma^{-1}(x-y), \qquad s = (\det\Sigma)^{1/2} \tag{4}$$

This is the variant used in UNC, where each cluster is described by only one representative (c, Σ).

Depending on the density function type, a cluster of center c can be described by (c, σ), $(c, \sigma^1, \ldots, \sigma^n)$ or (c, Σ). If the clusters to be detected are not necessarily spherical then the first descriptor is not able to capture the cluster structure. The most flexible variant is the last one because it can be used to describe arbitrary oriented hyper-ellipsoids. However when n is large the number of parameters to be estimated $(n(n+1)/2)$ is also large.

The variant chosen for CDE-clustering is a compromise solution which is based on describing clusters by using multiple normally oriented hyper-ellipsoids instead of using one arbitrary oriented hyper-ellipsoid. The reason of this compromise is that the scale parameters of these normally oriented hyper-ellipsoids are easier to be estimated than a covariance matrix Σ.

With the above notations the clustering problem can be formulated as follows: find a set of clusters representatives, $\{(c_k, \sigma_k)\}_{k=\overline{1,K}}$, $c_k \in \mathbb{R}^n, \sigma_k \in D(\sigma) = [\sigma^1_{min}, \sigma^1_{max}] \times \ldots \times [\sigma^n_{min}, \sigma^n_{max}]$ which approximate the local maxima of the density function. An important role in the optimization process is played by the constraints imposed on the scale parameters σ_k. Without introducing constraints on the values of the scales the natural tendency will be to identify a single large cluster centered almost in the center of the region to which the data belong. The most difficult problem is to find the appropriate range for the scale parameters. This depends on the size of clusters, information which we do not have apriori. This problem is similar with that of choosing an appropriate common σ (as in the case of DENCLUE) but here the scales values will be further adjusted.

Supposing that the data belong to a domain $D(x) = [x^1_{min}, x^1_{max}] \times \ldots \times [x^n_{min}, x^n_{max}]$ the range for a scale parameter, σ^i will be $[\epsilon, (x^i_{max} - x^i_{min})/\beta]$ with $\epsilon > 0$ the lower bound for the scales range and $\beta > 1$ a parameter controlling the upper bound.

The clustering problem is thus equivalent with that of finding all local maxima $(c, \sigma) \in \mathbb{R}^n \times \mathbb{R}^n$ of the function

$$g(c,\sigma) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\sqrt{\sigma^1 \ldots \sigma^n}} \exp\left(-\frac{1}{2} d^2_{\sigma^1 \ldots \sigma^n}(c, x_i)\right) \tag{5}$$

subjected to the constraints $c \in D(x)$ and $\sigma \in D(\sigma)$.

In order to solve this multi-modal optimization problem we used a crowding-based variant of differential evolution [16] which proved to be effective in identifying multiple optima.

2.2. Crowding differential evolution. Differential evolution (DE) has been proposed in [15] as an heuristic, inspired by simplex methods, able to efficiently solve difficult optimization problems on continuous domains. Its particularity consists in the search operator based on an internal perturbation scheme not on an external one as is usual in classical mutation operators. To summarize the particularities of the DE algorithm let us consider a simple maximization problem of a function $f : D \subset \mathbb{R}^n \to \mathbb{R}$: find $x^* \in D \subset \mathbb{R}^n$ such that $f(x^*) \geq f(x)$ for all $x \in D$. Let us denote by $X = (x_1, \ldots, x_m)$ the current generation and by y_i the offspring corresponding to x_i.

Different schemes of constructing $y_i = (y_i^1, \ldots, y_i^n)$ starting from the elements of X have been proposed. The most frequently used is:

$$y_i^j = \begin{cases} x^j_{r_3} + F \cdot (x^j_{r_1} - x^j_{r_2}), & \text{with probability } p_c \\ x_i^j, & \text{with probability } 1 - p_c \end{cases} \tag{6}$$

for $j \in \{1, \ldots, n\}$. In eq.(6) r_1, r_2, r_3 are distinct indices randomly selected from $\{1, \ldots, m\}$, $F \in (0, 2)$ is a parameter which controls the magnitude of the perturbation and $p_c \in [0, 1]$ is a probability value which controls the ratio of new components in the offspring. The selection step of classical DE is a simple one: an offspring, y_i, replaces its parent, x_i, only if it is better $(f(y_i) \geq f(x_i))$.

Recently, some DE variants for multi-modal optimization problems have been proposed [4],[16], [18]. The simplest of them is the crowding-based differential evolution (CDE) proposed in [16]. The basic idea of (CDE) is to modify the classical sequential DE by introducing a crowding mechanism in the selection step. The unique difference between CDE and the classical sequential differential evolution is only in the selection step where the new generated element, y_i doesn't replace x_i but the element in the population which is most similar to y_i. This means that CDE is based on a global crowding mechanism which implies the computation at each step of the distances between the new generated offspring and all the population elements. This global computation is a significant disadvantage only when working with large populations. The effect of this crowding mechanism is that a fixed-size population concentrates on the local maxima of the objective function. In order to extract the approximation of local maxima a post-processing step should be applied. This post-processing step can be based either on computing distances between population elements or on an heuristic valley detection [17].

Algorithm 1 CDE-clustering

```
initialize the population
repeat
  for all i ∈ {1,...,m} do
    construct a new c_i by using eq. 6
    construct a new σ_i by using the absolute value of the right member in eq. 6
    find the element (c, σ) most similar to (c_i, σ_i)
    if g(c_i, σ_i) ≥ g(c, σ) then
      replaces (c, σ) with (c_i, σ_i)
    end if
  end for
until a stopping condition is satisfied
{Postprocessing stage:}
Collect representatives {ρ_1,...,ρ_K} from the population and label them
Refine the representatives labels
Classify the data
Eliminate small clusters and reclassify the data
```

2.3. Description of CDE-clustering. We consider a population of m elements. The population size should be at least as large as the number of clusters we expect to detect. Each element of the population is a pair (c, σ), $c \in D(x)$, $\sigma \in D(\sigma)$. The centers are initialized with elements randomly selected from the set of data while the scales are randomly selected in $D(\sigma)$. Both the centers, c_i, and the scales, σ_i, are evolved based on the DE rule (6). However, since the scale parameters should be positive when generating new values for them, the relation $x^j_{r_3} + F \cdot (x^j_{r_1} - x^j_{r_2})$ is replaced with its absolute value. Another difference from the original crowding-based differential evolution is the distance used in the crowding mechanism: instead of the euclidean distance we used the particular case of the Mahalanobis distance (3) with respect to the scale parameters of the trial element. A new generated element is accepted only if it satisfies the constraints.

The general structure of the algorithm is presented in algorithm 1. After applying the CDE to the population, the clusters representatives are determined by post-processing the obtained population.

Representatives collecting. The first postprocessing step aims to extract some representatives from the population. Each representative, ρ, has three components: the center, $c(\rho)$, the scale parameters, $\sigma(\rho)$, and the label $L(\rho)$. The basic idea of this step is to iteratively construct a set of labelled representatives, $R = \{\rho_1, \ldots, \rho_K\}$ starting from the empty set. The center and the scale of the first representative are initialized with the first element of the population and its label is set to 1. Then for each element, $x = (c(x), \sigma(x))$, of the population starting with the second one the nearest representative, $\rho^* = (c(\rho^*), \sigma(\rho^*), L(\rho^*))$, is determined. This representative is determined by using the Mahalanobis distance between $c(x)$ and $c(\rho)$ with respect to the parameters $\sigma(\rho)$. Considering the distances $d_1 = d_{\sigma(\rho^*)}(x, \rho)$ and $d_2 = d_{\sigma(x)}(x, \rho)$ the following situations are analyzed:

(i) If $d_1 \leq \delta_1$ or $d_2 \leq \delta_1$ then the representative ρ^* should be modified to include the information given by x: $c(\rho^*) = (c(\rho) + c(x))/2$ and $\sigma^j(\rho^*) = \max\{\sigma^j(\rho^*), \sigma^j(x)\}$ for all $j \in \{1, \ldots, n\}$. The label of ρ^* remains unchanged.

(ii) If $d_1 > \delta_2$ and $d_2 > \delta_2$ then a new representative is generated. Its center and its scale parameters are those of x and its label is a new one, obtained by incrementing the largest existing label (this means initiating a new cluster).

(iii) In all the other cases a new representative is generated having the same center and parameters as x but the same label as ρ^* (this would be a representative associated to an existing cluster).

The parameters δ_1 and δ_2 are some thresholds which defines the influence area of a center and satisfy $\delta_1 < \delta_2$. Values which proved to give good results in experiments are $\delta_1 = 1$ and $\delta_2 = \sqrt{3}$.

Labels refinement. Due to the iterative manner of constructing the set of representatives it is possible that representatives which should describe the same cluster be differently labelled. To avoid such situations the representatives set is repeatedly scanned and for each pair (ρ_i, ρ_k) for which $d_{\sigma(\rho_i)}(c(\rho_i), c(\rho_k)) \leq \delta_2$ or $d_{\sigma(\rho_k)}(c(\rho_i), c(\rho_k)) \leq \delta_2$ and $L(\rho_i) \neq L(\rho_k)$ the label of the worse representative (with respect to the fitness function) is replaced with the label of the better one. This iterative process should continue until no such pairs are found. However it is possible that this never happens. Let us consider, for instance, three representatives, ρ_1,ρ_2 and ρ_3 satisfying $d_{\sigma(\rho_2)}(\rho_1, \rho_2) < \delta_2$, $d_{\sigma(\rho_2)}(\rho_3, \rho_2) < \delta_2$, $L(\rho_1) \neq L(\rho_3)$ and $g(c(\rho_1), \sigma(\rho_1)) > g(c(\rho_2), \sigma(\rho_2))$,$g(c(\rho_3), \sigma(\rho_3)) > g(c(\rho_2), \sigma(\rho_2))$. In this situation when the pair (ρ_1, ρ_2) is analyzed, the label of ρ_2 is changed with that of ρ_1 and when the pair (ρ_2, ρ_3) is analyzed the label of ρ_2 is changed with the label of ρ_3 and so on. To avoid such situations, we stopped the iterative process after a maximal number of cycles (equal, for instance, with the current number of representatives).

Data classification. For each data instance the nearest representative is determined and its label will be given

to the data. However if the distance between the data and the nearest representative is larger than a given value, δ_3 (e.g. $\delta_3 = \sqrt{5}$), then the data is considered to be noise (it does not belong to a cluster).

Small cluster deletion. The final step is a refining one used to eliminate some small clusters. The data belonging to clusters which have a number of elements which is less than a given threshold (e.g. $N/(4K)$, K being the number of identified clusters) are reassigned to other clusters based on the same idea of the nearest representative.

3. Distributed CDE-clustering.

The aim of distributed clustering is to identify clusters in data stored at different sites without unifying them at a central site. Thus we have to process an ensemble of data subsets such that we do not have access simultaneously to all data. In order to ensure a low level of communication between sites and a high level of security the size of data transferred between sites should be as small as possible. This motivates the fact that the most distributed variants of clustering algorithms are based on the idea of applying a local clustering algorithm at each site, constructing models for the obtained local data partitions, combining these local models into a global one and obtaining the final clustering at each site by using the global model.

The differences between the existing distributed clustering algorithms are related mainly with the structure of the models representing the information about the local / global data partitions which should be transferred between the data sites.

For instance in the distributed variant of K-means proposed in [8] the following communication steps are executed: (i) samples of data selected from each site are sent to the central site; (ii) statistical information obtained by applying K-means at the central site to the received sampled data are sent back to the local sites; (iii) new statistical information collected at the local sites by applying local K-means are sent to the central site. This communication process is iterated until the local center are stabilized.

Another distributed approach is that based on DBSCAN [7]. The main idea here is to exploit the so-called core points which could be considered as representatives of dense regions. Since the number of these core points could be large, especially for regions with high density, instead of using them as local representatives a subset of them, called complete set of specific core points, is used. Based on all local sets of specific core points a global model is constructed at the central site. The global model involves also some range values allowing the estimation of the clusters extension. This global model is sent to the local sites where is used to cluster each data subset.

The approach which we propose is based on a natural extension of CDE-clustering (see algorithm 2). As was presented in the previous section CDE-clustering leads to some representatives of clusters, each cluster being described by a set of such representatives which give information about its position and its extension. These representatives seems to be good candidates for local models. On the other hand since crowding DE does not need large populations to identify the local maxima in order to increase the accuracy of the local model the entire population (whose size is not related with the number of data to be processed) could be considered as a local model (in this case the step 3 in algorithm 2 is not executed). We analyzed both these variants and a comparative analysis is presented in the next section (see table 1).

Algorithm 2 Distributed CDE-clustering

1: **for all** data subsets **do**
2: evolve the population $\{(c_1, \sigma_1), \ldots, (c_m, \sigma_m)\}$ by applying crowding DE
3: construct the local representatives
4: **end for**
5: collect the local models
6: extract the global representatives from the local models
7: classify the local subsets based on the global representatives

Starting from the local models the global model is constructed by applying the representatives collecting and labels refining steps from CDE-clustering either to the set of local representatives or to the set of local populations. Once the global representatives are constructed they are sent to local sites where are used to classify the data. A final refinement of clusters (similar with step 15 in Algorithm 1) needs a supplementary communication step between local sites in order to transfer the number of elements in each cluster. The implementation presented in this work does not use such a final refinement of clusters.

By transferring the local models to the central site only at the end of the evolutionary process the local representatives are determined independently at each site. Sometimes a more accurate global clustering could be obtained if some communication between partial local models is allowed during the evolutionary process. This means in fact to collect periodically (after a given number of generations, called an epoch) the local models and to construct a partial global model which is sent back to local sites in order to influence the adjustment of the local models. This adjustment is ensured by including into the objective function the influence of the local

models corresponding to other sites through the partial global model. More specifically the objective function is in this case:

$$g(c,\sigma) = \frac{1}{2N}\sum_{i=1}^{N}\frac{1}{\sqrt{\sigma^1\ldots\sigma^n}}\exp\left(-\frac{d^2_{\sigma^1\ldots\sigma^n}(c,x_i)}{2}\right) + \frac{1}{4K}\sum_{i=1}^{K}\left(\frac{1}{\sqrt{\sigma^1\ldots\sigma^n}}\exp\left(-\frac{d^2_{\sigma^1\ldots\sigma^n}(c,c_i)}{2}\right) + \frac{1}{\sqrt{\sigma^1_i\ldots\sigma^n_i}}\exp\left(-\frac{d^2_{\sigma^1_i\ldots\sigma^n_i}(c,c_i)}{2}\right)\right) \quad (7)$$

where $\{(c_1,\sigma_1),\ldots,(c_K,\sigma_K)\}$ are the global representatives constructed at the previous epoch. Splitting the evolutionary process in epochs and introducing intermediary communication steps increases the computational cost of distributed CDE clustering but can improve the accuracy of the final clustering.

In order to estimate the complexity order of distributed CDE-clustering let us consider the case of clustering N data (n dimensional vectors) distributed in s subsets of N/s elements. For each subset o population of m elements ($2n$ dimensional vectors) is evolved for g generations. The most costly operations in the CDE stage are the computation of the objective function ($O(g\cdot m\cdot n\cdot N/s\cdot s) = O(g\cdot m\cdot n\cdot N)$) and the computation of distances in the crowding mechanism ($O(g\cdot s\cdot m^2\cdot n)$). In the postprocessing stage the costly operations are: representative extraction, labels refinement and final classification of data. Let us denote with $K_1,\ldots,K_s$ the number of representatives for each data subset and with K the number of the representatives in the global model.

If the local models consist of the entire local populations then the representatives are extracted only at the global level and the complexity order of this process is $O(s\cdot m\cdot K)$ for representative collecting and $O(n\cdot K^3)$ for labels refinement (this step is based on iterating the refinement process for K times).

When the local models consists of local representatives there are two stages for representative extraction: a local and a global one. The local step for each subset belongs to $O(m\cdot n\cdot K_i + n\cdot K_i^3)$ while the global set belongs to $O(K\cdot n\sum_{i=1}^{s}K_i + nK^3)$.

In both cases the final classification step belongs to $O(K\cdot n\cdot N)$. What is important is the fact that the algorithm is linear with respect do the size of data set. On the other hand the current implementation variant is quadratic with respect to the population size (which is a drawback when working with high-dimensional data which ask for larger populations) and cubic with respect to the number of representatives (this could be critical when many clusters should be detected). However, since $K < m \ll N$ the computational cost is not unacceptable. Moreover the distributed approach does not increase significantly the computational cost of the non distributed variant. On the contrary, in the case of parallel implementations savings in the clustering time could be obtained.

4. Experiments on some synthetic data. The main aim of the experiments is to analyze the effectiveness of the distributed CDE-clustering. With this aim we divided some sets of data in subsets and compared the results obtained by applying the distributed approach with those obtained by applying the CDE-clustering algorithm on the entire set of data. We also analyzed the difference between the variant when the entire local population is sent to the central site and the variant when the local model consists only of the representatives extracted from the local population.

Test data. In the experiments we used sets of up to 5000 synthetic data generated based on normal distributions with different covariance matrices (see fig. 1(a),(b),(c) and fig. 5 (a), (b), (c)). The test data are bi-dimensional in order to allow a visual inspection of results. They have been normalized by dividing each attribute to the difference between the largest and the smallest value corresponding to that attribute. The subsets to be placed at local sites have been generated either in a random manner (for each data in the global set the destination subset has been randomly selected - fig. 5) or in a controlled one in order to model situations when all elements of some clusters are at the same site while elements of other clusters are distributed at different sites (see figs. 1, 3).

Parameters. Even if differential evolution is sensitive to the choice of its control parameters (p_c and F), preliminary tests suggested that no significant differences are obtained by changing p_c and F. Thus in all tests we used the values $p_c = 0.9$ and $F = 0.5$. Concerning the population size and the number of generations most tests have been executed for $m = 60$ and 100 generations. In the case of processing s data subsets the local evolutionary process was based on populations of m/s elements. Concerning the population size, we have to mention that it is not directly related with the size of the data set but with the number of clusters we expect to find. The parameters involved in the post-processing steps had the following values: $\delta_1 = 1$, $\delta_2 \in \{\sqrt{2},\sqrt{3}\}$ and $\delta_3 = \sqrt{5}$.

Quality measures. In order to compare two clusterings we computed the success ratio and we estimated the clustering error. The success ratio expresses the number of cases when the correct number of clusters has been detected. Since for the test data the real classification is known we used the following measure for the clustering error [12]:

$$Err = \frac{2}{N(N-1)} \sum_{i<j} \epsilon_{ij} \tag{8}$$

with

$$\epsilon_{ij} = \begin{cases} 0 & \text{if class}(x_i) = \text{class}(x_j) \text{ and } L(x_i) = L(x_j) \\ & \text{class}(x_i) \neq \text{class}(x_j) \text{ and } L(x_i) \neq L(x_j) \\ 1 & \text{otherwise} \end{cases}$$

Err expresses the ratio of data pairs which are not classified by the algorithm as is expected (data belonging to the same class should receive the same label while data belonging to different classes should receive different labels).

Results. Table 1 presents results obtained for data distributed in 2 to 6 subsets both by the variant which send the entire local populations to the central site and by the variant working with local models based on representatives. The first variant is slightly superior with respect to the second one. The values of success ratio decrease when the number of subsets increases but the classification error (averaged only for successful cases) remains almost at the same level as in the case of grouped data (first row in table). Besides the number of detected clusters also the number of representatives found by the algorithm is reported. This value is about twice as the number of clusters meaning that each clusters is described, in the average, by two representatives. All values are averages obtained for 30 independent runs of the algorithm. Final clustering results are illustrated in fig. 5. These results suggest that the distributed approach has an acceptable accuracy.

Subsets	Success ratio	Number of representatives	Number of clusters	Classification error
Local models: entire local populations				
1	24/30	17.43	7.93 ± 0.442	0.12842 ± 0.0012
2	24/30	17.16	8.03 ± 0.546	0.12809 ± 0.0010
3	22/30	17.96	8.10 ± 0.597	0.12786 ± 0.0015
4	21/30	16.73	8.26 ± 0.573	0.12710 ± 0.0026
5	17/30	17.46	8.40 ± 0.757	0.12718 ± 0.0018
6	16/30	19.53	8.66 ± 0.942	0.12532 ± 0.0033
Local models: representatives extracted from local populations				
1	24/30	17.43	7.93 ± 0.442	0.12842 ± 0.0012
2	22/30	15.46	8.10 ± 0.597	0.12831 ± 0.0021
3	21/30	15.06	8.16 ± 0.597	0.12674 ± 0.0012
4	16/30	16.33	8.73 ± 1.062	0.12717 ± 0.0009
5	17/30	17.93	8.43 ± 0.843	0.12666 ± 0.0024
6	17/30	17.76	8.73 ± 1.093	0.12608 ± 0.0022

Table 1: Influence of the number of subsets and of the type of the local models on the quality of the clustering

The worsening of the success ratio when the number of subsets increases can be explained by the sensitivity of the behavior of CDE-clustering on the value of β which controls the upper bound of scales parameter and which should be related to the clusters extension. We used the same value ($\beta = 17$) in all situations. Small values of β leads to large upper bounds for scale parameters which lead to a small number of clusters while large values of β lead to a large number of clusters. The influence of β on the clustering results is illustrated in fig. 4.

Sometimes, constructing the global model only at the end of the evolutionary process does not ensure a good final clustering. Table 2 presents results obtained by collecting the local models and constructing the global model during the evolutionary process (at the end of each epoch consisting of a given number of generations). The data belongs to two clusters and are divided in two subsets such that each subset contains one half of each cluster. The correct number of final clusters should be 2. As appears in Table 2 transferring the partial local models and a modified objective function (see eq. 7) improves the quality of the clustering (when the local models are collected only at the end of the evolutionary process in only one case out of ten the algorithm found the right number of clusters, while when the partial local models are collected every ten generations in all situations the algorithm identified the right number of clusters). Illustrations of these results are also presented in fig. 3.

In figs. 1, 2 and 3 besides the structure of clusters (illustrated with different colors) also the representatives are figured: the centers by red points and the scale parameters by ellipses having radii equal to the scale parameters multiplied by $\delta_2 = \sqrt{3}$. Ellipses with the same color correspond to the same cluster. Thus, one can visualize the cluster descriptors. In images containing the clustering results, the black points correspond to data considered as noise. Analyzing these points one can see that data classified as noise by the local clustering

Gen.×epochs	Number of cases			Number of clusters
	2 clusters	3 clusters	4 clusters	
100× 1	1	7	2	3.1 ± 0.538
50× 2	4	6	0	2.6 ± 0.489
25× 4	8	2	0	2.2 ± 0.4
10× 10	10	0	0	2 ± 0

Table 2: Influence of the communication of partial models on the ability of finding the correct number of clusters. Test data: two discs divided in halves (see figure 3)

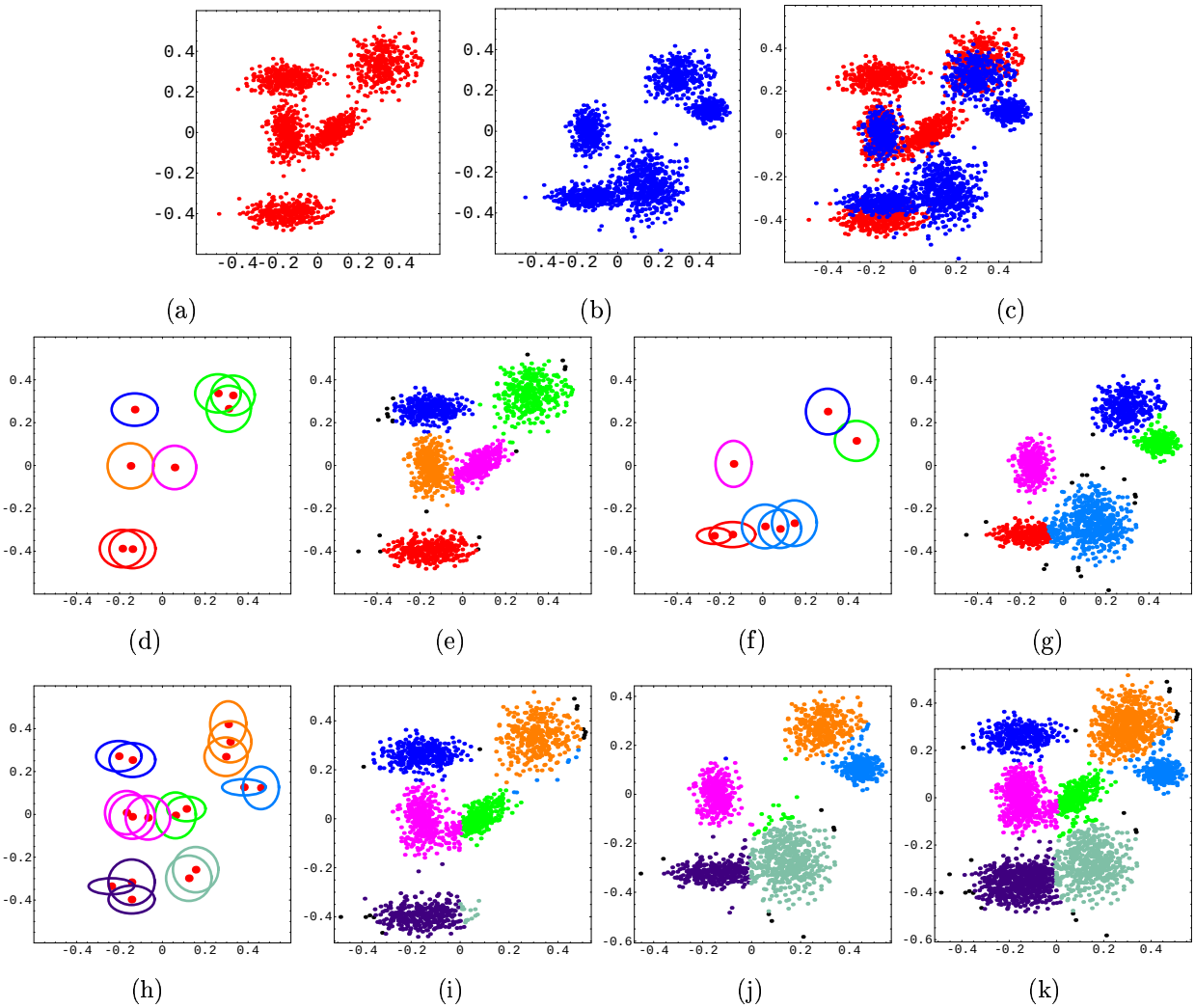

Figure 1: Results of distributed CDE-clustering for two data subsets. The local models consist of the entire local populations. (a) Data in the first subset; (b) Data in the second subset; (c) The entire set of data; (d),(f) Representatives for the first and second subset, respectively (each representative is marked by its center and an ellipse corresponding to the determined scale parameters multiplied by δ_2); representatives belonging to the same cluster have the same color; (e),(g) Clusters obtained by independent clustering of the subsets; (h) Global representatives obtained by combining the results for each subset; (i),(j) Clusters in data subsets obtained by using the global representatives; (k) The final clustering obtained by combining the partial clusterings on subsets.

process are in fact data inside the clusters in the case when a global model is used. This makes the difference between completely independent clustering at each site and a distributed approach based on a communication between sites.

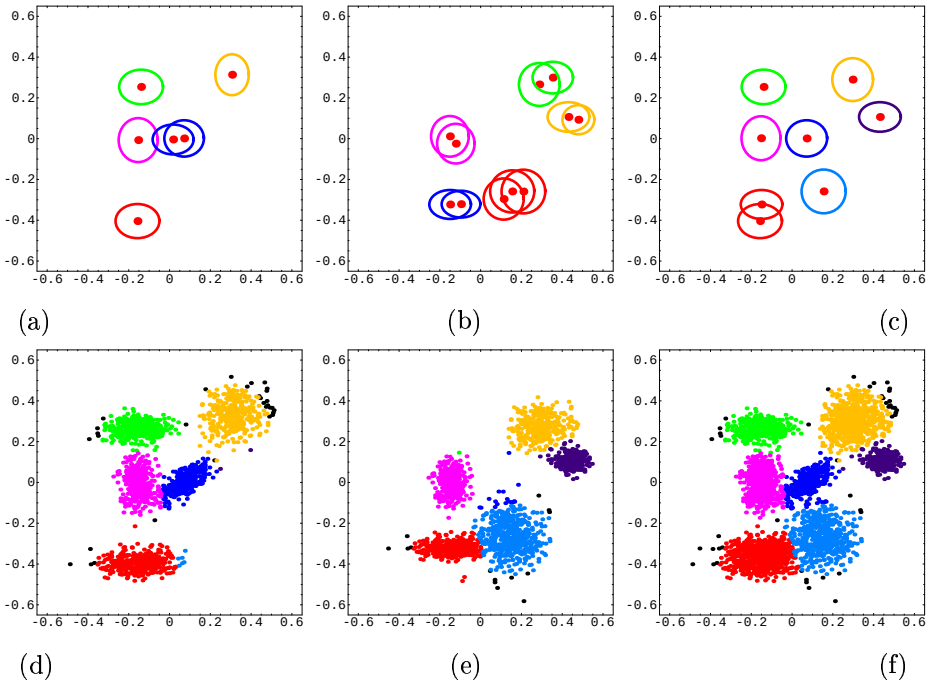

Figure 2: Results of distributed CDE-clustering for two data subsets. The local models consist of representatives obtained by postprocessing the local population. (a) Local model for the first subset; (b) Local model for the second subset; (c) Global model; (d) Classification of the data in the first subset (by using the global model); (d) Classification of the data in the second subset (by using the global model); (e) Global classification.

5. Conclusions and open problems. Due to the fact that CDE-clustering naturally produces a set of clusters representatives, it is a good candidate for distributed clustering. Extending it for distributed sources of data involves only to generate the global model starting from the local models by applying the representatives collection and labels refinement steps from Algorithm 1. Thus the algorithm complexity is not significantly increased.

The main problem of the distributed version of CDE-clustering is the same as in the case of the non-distributed one: choosing values for β, δ_1, δ_2 and δ_3. In the case of distributed CDE-clustering different values of these parameters can be used for the algorithm applied for each data subset. Unfortunately this means choosing many parameters, a difficult task in the absence of some prior information about the clusters structure. Finding right values for the parameters (especially for β) is still an open problem. Another open problem is to analyze the scalability of the algorithm with respect to the size and the number of data subsets.

References

[1] M. Ester, H.P. Kriegel, J. Sander, X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Proc. of the 2nd Intern. Conf. on Knowledge Discovery and Data Mining*, AAAI Press, 226-231, 1996.

[2] A.A. Freitas. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer-Verlag, 2002.

[3] J. Handl, J. Knowles. Evolutionary Multiobjective Clustering. X. Yao et al (Eds.), *Proc. of PPSN VIII, LNCS 3242*, 1081-1091, 2004.

[4] Z. V. Hendershot; A Differential Evolution Algorithm for Automatically Discovering Multiple Global Optima in Multidimensional, Discontinuous Spaces, *Proc. of MAICS 2004, Fifteenth Midwest Artificial Intelligence and Cognitive Sciences Conference*, Chicago IL, April 16-18, pp. 92-97, 2004.

[5] A. Hinneburg, D.A. Keim. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. *Proc. of 4rd Intern. Conf. on Knowledge Discovery and Data Mining*, AAAI Press, 58-65, 1998.

[6] A.K. Jain, M.N. Murty, P.J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3), 1999.

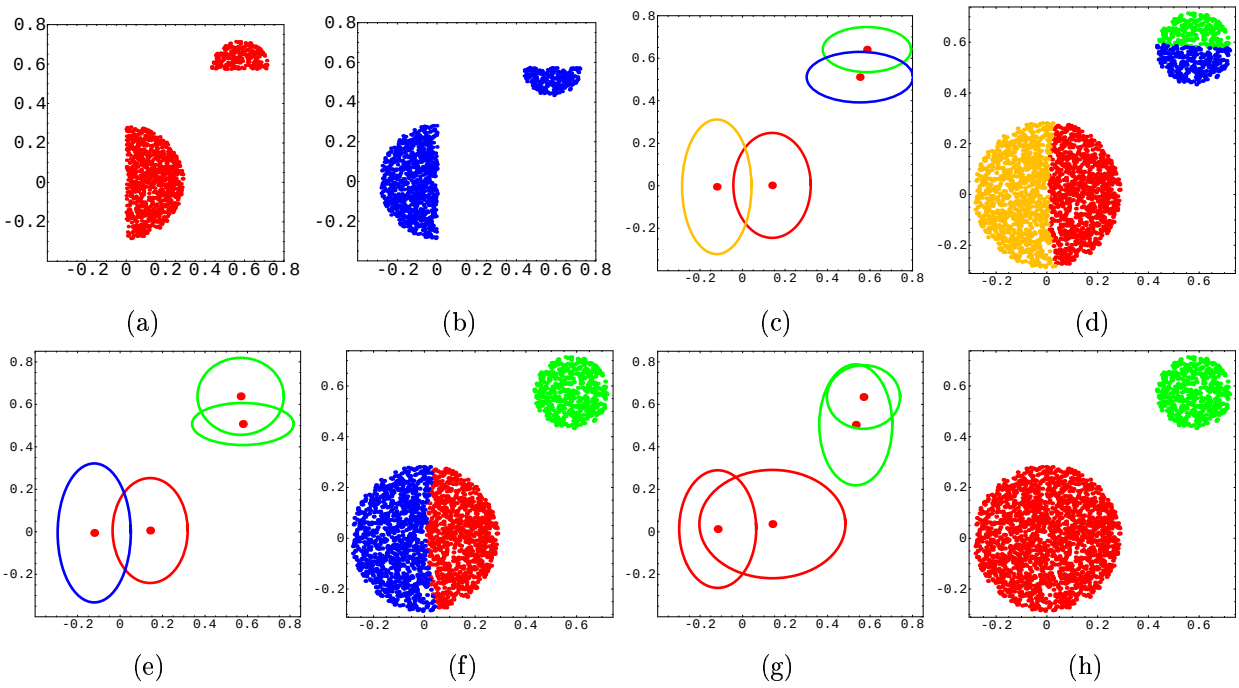

Figure 3: Illustration of the influence of the use of intermediary global models (a),(b) The initial data distributed in two subsets; (c),(d) Results obtained in one epoch of 100 generations; (e),(f) Results obtained in 2 epochs of 50 generations; (g),(h) Results obtained in 4 epochs of 25 generations.

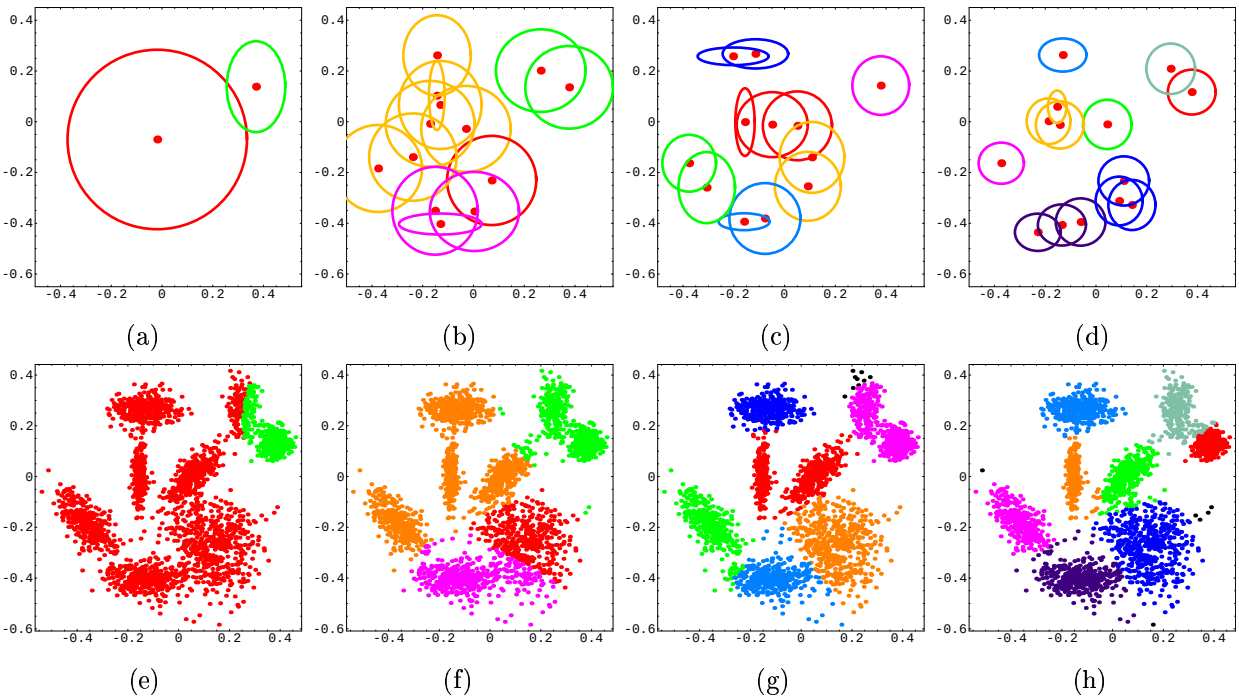

Figure 4: Influence of β on the clustering results (the ellipses radii are obtained by multiplying the evolved scale parameters by $\sqrt{2}$)(a),(e) $\beta = 4$; (b),(f) $\beta = 8$; (c),(g) $\beta = 10$; (d),(h) $\beta = 14$

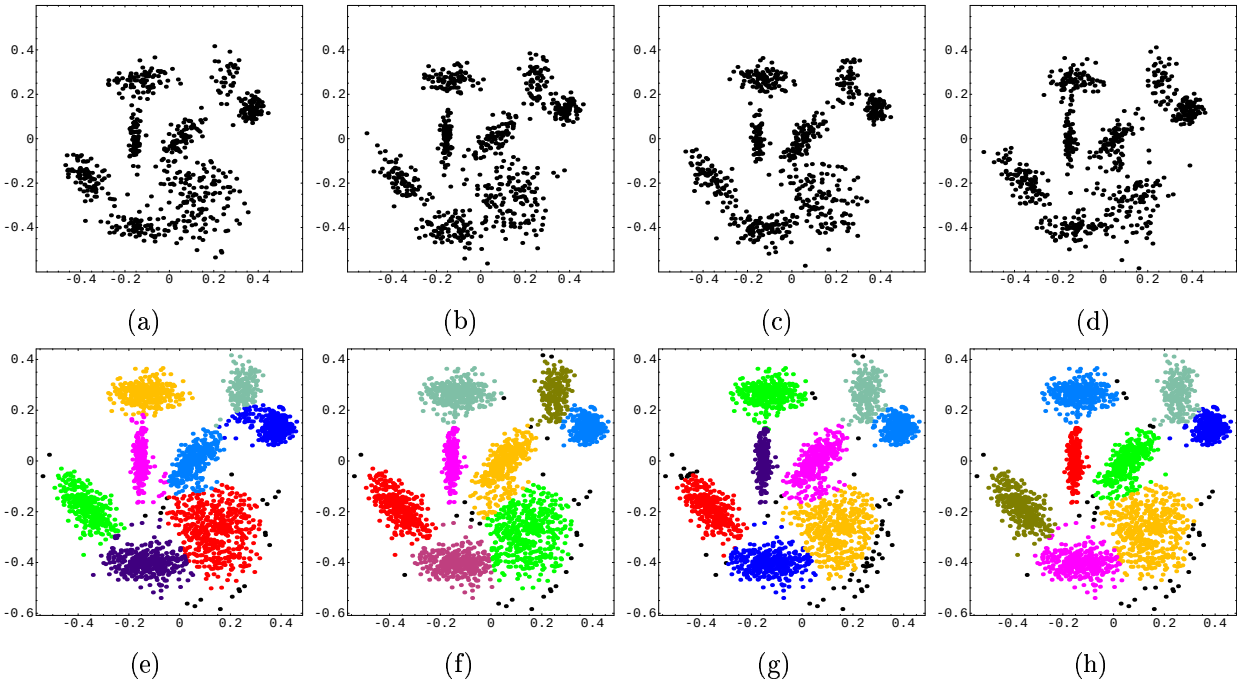

Figure 5: (a),(b),(c),(d) Data randomly distributed in four subsets. (e), (f), (g), (h) Results of clustering for different number of subsets: (e) 2 subsets; (f) 4 subsets; (g) 5 subsets; (h) 6 subsets

[7] E. Januzaj, H.P. Kriegel, M. Pfeifle. DBDC: Density Based Distributed Cluster. in E. Bertino et al., EDBT 2004, LNCS 2992, pp. 88-105, 2004.

[8] R. Jin, A. Goswami, G. Agrawal. Fast and Exact Out-of-Core and Distributed K-Means Clustering, to appear in Knowledge and Information System (online version: http://www.cs.kent.edu/ jin/Papers/kais.pdf).

[9] H.P. Kriegel, A. Pryakhin, M. Schubert. Effective and Efficient Distributed Model-based Clustering, Proc. of 5th IEEE International Conference on Data Mining, Houston, pp. 258-265, 2005.

[10] T. Krink, S. Paterlini. Differential Evolution and Particle Swarm Optimization in Partitional Clustering. *Technical Report no. 446*, Dept. of Political Economics, Modena, Italy, 2003.

[11] M.H.C Law, A.P. Topchy, A.K. Jain. Multiobjective Data Clustering, *Computer Vision and Pattern Recognition*, 2, 424-430, 2004.

[12] N. Labroche, N. Monmarché, G. Venturini. A New Clustering Algorithm Based on the Chemical Recognition System of Ants, in Harmelen, F. van (Ed.) *Proc. of the 15th European Conference on Artificial Intelligence*, Lyon, France, 345-349, 2002.

[13] S.W. Mahfoud; Crowding and Preselection Revisited, in R. Männer and B. Manderick (eds.),*Parallel Problem Solving from Nature*,Elsevier, 2, 27-36, 1993.

[14] O. Nasraoui, E. Leon, R. Krishnapuram. Unsupervised Niche Clustering: Discovering an Unknown Number of Clusters in Noisy Data Sets, in A. Ghosh and L. C. Jain (Eds.), *Evolutionary Computing in Data Mining*, Springer Verlag, 2005.

[15] R. Storn, K. Price. Differential Evolution; A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Technical Report TR-95-012*, ICSI, 1995.

[16] R. Thomsen. Multimodal Optimization Using Crowding-Based Differential Evolution. *Proc. of the IEEE Congress on Evolutionary Computation*, 2, 1382-1389, 2004.

[17] K. Ursem. Multinational Evolutionary Algorithms, in *Proc. of the IEEE Congress of Evolutionary Computation*, 3, pp. 1633-1640, 1999.

[18] D. Zaharie. A Multipopulation Differential Evolution Algorithm for Multimodal Optimization, in R. Matoušek, and P. Ošmera (eds.), *Proc. of Mendel'04*, Brno, June 16-18, 17-22, 2004.

[19] D. Zaharie. Density Based Clustering with Crowding Differential Evolution, in *Proc. of 7th Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, Timisoara, 25-29 sept. 2005, IEEE Computer Science Press, pp. 343-350, 2005.