A Clustering Algorithm Based on Fitness Probability Scores for Cluster Centers Optimization*

M. Fernanda P. Costa (✉)[1][0000−0001−6235−286X], Ana Maria A. C. Rocha[2][0000−0001−8679−2886], and Edite M. G. P. Fernandes[2][0000−0003−0722−9361]

[1] Centre of Mathematics,
University of Minho, Campus de Gualtar, 4710-057 Braga, Portugal
mfc@math.uminho.pt
[2] ALGORITMI Center,
University of Minho, Campus de Gualtar, 4710-057 Braga, Portugal
{arocha,emgpf}@dps.uminho.pt

Abstract. In the present paper, we propose an iterative clustering approach that sequentially applies five processes, namely: the assign, delete, split, delete and optimization. It is based on the fitness probability scores of the cluster centers to identify the least fitted centers to undergo an optimization process, aiming to improve the centers from one iteration to another. Moreover, the parameters of the algorithm for the delete, split and optimization processes are dynamically tuned as problem dependent functions. The presented clustering algorithm is evaluated using four data sets, two randomly generated and two well-known sets. The obtained clustering algorithm is compared with other clustering algorithms through the visualization of the clustering, the value of a validity measure and the value of the objective function of the optimization process. The comparison of results shows that the proposed clustering algorithm is effective and robust.

Keywords: Clustering Analysis · Fitness Probability Score · Differential Evolution

1 Introduction

Clustering is an unsupervised machine learning task and consists of grouping a set of data points in a way that similarity of the elements in a group – also called cluster – is maximized, whereas similarity of elements in two different groups, is minimized. Clustering methods can be categorized as partitioning, hierarchical, fuzzy, density-based or model-based methods. The most popular are the partitioning and hierarchical clustering. There are different types of partitioning

* This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00013/2020 and UIDP/00013/2020 of CMAT-UM.

clustering methods being the K-means clustering the most popular [1]. These methods subdivide the data set into K clusters, where K should be specified *a priori*. Each cluster is represented by the centroid (mean) of the data points belonging to that cluster. Hierarchical methods do not require the *a priori* specification of the number of clusters and the result of the clustering can be easily visualized in a tree-based representation of the data, known as a dendrogram. Partitioning and hierarchical clustering are suitable to find spherical-shaped clusters or convex clusters, i.e., they work rather well when clusters are compact and separated. On the other hand, when non-convex clusters and outliers (or noises) are present in the data set, they inaccurately identify clusters. Density-based clustering algorithms are the most appropriate for this type of data. DBSCAN is a popular density-based clustering technique that was introduced in [2]. It can find out clusters of different shapes and sizes from data containing noise and outliers. Alternatively, with clustering algorithms that cannot separate clusters that are non-linearly separable in the input space, the use of a kernel function tackles the problem. The idea is concerned with (before clustering) mapping the points to a higher-dimensional feature space (the kernel space) using a nonlinear function. Then, the kernel-based clustering method partitions the data points that are linearly separable in the new space [3].

Applications of clustering are varied and emerge in the field of data mining [4], in bioinformatics [5], pattern recognition [6], image processing, to name a few.

Since clustering can be seen as an optimization problem, well-known optimization algorithms, in particular metaheuristics, may be applied in clustering analysis. Varied contributions have been made in this area [7,8,9,10,11,12,13]. In general, metaheuristics have been combined with the K-means clustering, e.g., [9,11,14] However, contrary to the K-means, and other K-means combinations with metaheuristics, that require knowledge of the number of clusters in advance, it is possible to design an algorithm that dynamically adds, deletes and merges clusters based on a fitness function to evaluate the goodness of the clustering result.

The clustering algorithm presented in this paper resorts to this type of mechanisms to try to find the optimal (or near-optimal) clustering. Thus, the main contributions of this article include the definition of the algorithm parameters that are dynamically computed and depend on the characteristics of the data set. Furthermore, at each iteration of the algorithm, not all cluster centers but only a few of them are identified as the least fitted centers, and undergo an optimization process. The proposed clustering algorithm has mechanisms to merge two nearby clusters, delete the smallest cluster, split the largest cluster, and optimize specifically selected cluster centers iteratively. Although these mechanisms are similar to others in the literature, this article greatly contributes to this field of cluster analysis by proposing the definition of the parameter values - target values and thresholds - that are problem dependent and dynamically computed.

The paper is organized as follows. Section 2 describes the proposed clustering algorithm and shows details concerning the problem dependent parameters. In

Sect. 3, the results relative to two sets of data points with two attributes and two sets with four and thirteen attributes respectively are shown. Finally, Sect. 4 contains the conclusions of this work.

2 Clustering Algorithm

Let a set of n patterns or data points, each with a attributes, be given. These patterns can also be represented by a data matrix $\mathbf{X}$ with n vectors of dimension a. Each element X_{ij} corresponds to the jth attribute of the ith pattern/point. Thus, given $\mathbf{X}$, a partitioning clustering algorithm tries to find a partition $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$ of K clusters (or groups), in a way that similarity of the patterns in the same cluster is maximum and patterns from different clusters differ as much as possible. The partition must satisfy three properties:

1. each cluster should have at least one point, i.e., $|C_k| \neq 0$, $k = 1, \ldots, K$;
2. a point should not belong to two different clusters, i.e., $C_i \bigcap C_j = \emptyset$, for $i, j = 1, \ldots, K$, $i \neq j$;
3. each point should belong to a cluster, i.e., $\sum_{k=1}^{K} |C_k| = n$;

where $|C_k|$ is the number of points in cluster C_k. Since there are a number of ways to partition the patterns and maintain these properties, a fitness function should be provided so that the adequacy of the partitioning is evaluated. Therefore, the clustering problem could be stated as finding an optimal solution, i.e., partition $\mathcal{C}^*$, that gives the optimal (or near-optimal) adequacy, when compared to all the other feasible solutions.

2.1 Clustering Algorithm with Centers Optimization

The partitioning clustering algorithm herein proposed does not require *a priori* specification of the number of clusters. Although for the initialization, a number of clusters must be specified, K, and the corresponding centers (that represent the clusters), $m_1, m_2, \ldots, m_K$ randomly selected (or generated), the algorithm iteratively adds, merges and deletes centers (removing the correspondent clusters) according to some problem dependent rules that vary dynamically as the iterative process progresses.

In Algorithm 1, the five main steps of the proposed clustering algorithm are shown - from line 6 to line 10. This is an iterative process that automatically finds the optimal clustering.

Briefly, after a set of cluster centers being randomly generated in the region of the data points, each point is assigned to a cluster based on the minimum distance of that point to all the centers. Then, clusters may be merged and deleted if the smallest distance between the cluster centers is below a threshold and the number of points in a cluster is considered very small relatively to the number of data points in the set. Furthermore, the cluster with maximum hypervolume may be split into two new clusters if its hypervolume exceeds a target value. The algorithm checks again if clusters may be merged or deleted.

Finally, a set of the current cluster centers are identified as the least fitted centers, according to their fitness probability scores - further ahead described - and undergo an optimization process.

Algorithm 1 Clustering Algorithm

Require: a number of attributes, n number of data points, $\mathbf{X} = (X_{i,j}), i = 1, \ldots, n$, $j = 1, \ldots, a$ data set; $It_{\max}$

1: Set $K = \max\{2, \lceil 0.01n \rceil\}$; $N_{\min} = \max\{2, \lceil 0.05n \rceil\}$; $It = 1$
2: Compute $\underline{X}_j = \min_{i=1,\ldots,n} X_{i,j}$ and $\overline{X}_j = \max_{i=1,\ldots,n} X_{i,j}$ for $j = 1, \ldots, a$;
3: Compute $A_{\min} = \min_{j=1,\ldots,a}(\overline{X}_j - \underline{X}_j)$;
4: Randomly generate a set of cluster centers m_k, $k = 1, \ldots, K$ using

$$m_{k,j} = \underline{X}_j + rand(\overline{X}_j - \underline{X}_j) \quad \text{for } j = 1, \ldots, a$$

where *rand* is a uniformly distributed number in $[0, 1]$.
5: **repeat**
6: Assign points to current cluster centers $m_k, k = 1, \ldots, K$ using Algorithm 2
7: Merge clusters that have the two closest centers and remove the cluster with fewer points, using Algorithm 3, $\eta_D = \frac{1}{K} A_{\min}$ and $N_{\min}$
8: Add one cluster by splitting the cluster with maximum hypervolume using Algorithm 4
9: Merge clusters that have the two closest centers and remove the cluster with fewer points, using Algorithm 3, $\eta_D = \frac{2}{K} A_{\min}$ and $N_{\min}$
10: Compute improved positions for the least fitted cluster centers (maintaining the fitter ones as constant) by optimizing the fitness function (WCD in (6)), using Algorithm 5
11: Set $It = It + 1$
12: **until** $It > It_{\max}$ or cluster centers do not move
13: **return** K^*, $m_k, k = 1, \ldots, K^*$ and $\mathcal{C}^* = \{C_1^*, \ldots, C_K^*\}$.

To find which cluster to assign a point X_i $(i = 1, \ldots, n)$, the simplest idea is to find the closest distance from that point to a center m_k $(k = 1, \ldots, K)$, i.e., if the index of the closest center is k_i, then

$$d_{i,k_i} = \min_{k=1,\ldots,K} \|X_i - m_k\|_2. \tag{1}$$

Algorithm 2 presents the main steps of this idea. If no points have been assigned to a cluster, the center will be deleted.

During the iterative process, the two closest clusters to each other, measured by the distance between their centers, may be merged if the distance between their centers is below a threshold, herein denoted as η_D. This parameter value is dynamically defined as a function of the search region of the data points and also depends on the current number of clusters. Furthermore, if the cluster with the lowest number of points is considered very small relatively to the number of data points in the set, i.e., if $|C_{k_i}| < N_{\min}$ where $|C_{k_i}| = \min_k |C_k|$ and

$N_{\min} = \max\{2, \lceil 0.05n \rceil\}$ is the threshold, the points are coined as 'noise', the cluster is removed and the center is deleted, see Algorithm 3.

Algorithm 2 Assigning Algorithm

Require: K, cluster centers m_k, $k = 1, \ldots, K$ and the data set $\mathbf{X}$
Set $C_k = \emptyset, k = 1, \ldots, K$
Compute $d_{i,k} = \|X_i - m_k\|_2$ from a data point X_i, $i = 1, \ldots, n$ to cluster center m_k, $k = 1, \ldots, K$
for $i = 1$ to n **do**
    Identify $\min_k d_{i,k}$ and the index $k_i \in \{1, 2, \ldots, K\}$ of closest center
    Assign point X_i to cluster C_{k_i}
end for
for $k = 1$ to K **do**
    if $|C_k| = 0$ **then**
        Delete m_k (and remove C_k)
    end if
end for
Update K
return m_k and C_k, $k = 1, \ldots, K$

Algorithm 3 Deleting Algorithm

Require: K, m_k, C_k for $k = 1, \ldots, K$, $\eta_D > 0$, $N_{\min}$
Compute $D_{i,j} = \|m_i - m_j\|_2, i = 1, \ldots, K-1$, $j = i+1, \ldots, K$
Identify $D_{i_m,j_m} = \min_{i,j} D_{i,j}$ (indices $i_m, j_m \in \{1, 2, \ldots, K\}$)
if $D_{i_m,j_m} \leq \eta_D$ **then**
    Replace center m_{i_m} by $\frac{1}{|C_{i_m}|+|C_{j_m}|}(|C_{i_m}|m_{i_m} + |C_{j_m}|m_{j_m})$
    Assign points in C_{j_m} to cluster C_{i_m}
    Delete m_{j_m} (and remove C_{j_m})
    Update K
end if
Identify $|C_{k_i}| = \min_k |C_k|$ (index $k_i \in \{1, 2, \ldots, K\}$)
if $|C_{k_i}| < N_{\min}$ **then**
    Define all points in C_{k_i} as 'noise/outlier'
    Delete m_{k_i} (and remove C_{k_i})
    Update K
end if
return m_k and C_k, $k = 1, \ldots, K$.

One cluster may be added (one at a time) by splitting the cluster with maximum hypervolume $\mathcal{V}_k$ [7], where

$$\mathcal{V}_k = \left(\det \left(\frac{1}{|C_k|} \sum_{X_i \in C_k} (X_i - m_k)(X_i - m_k)^T \right) \right)^{1/2}. \tag{2}$$

However, our strategy is to allow the cluster with maximum volume to be split only if its volume is not smaller than a target value η_V. The value of this parameter is found to be problem dependent and is given by

$$\eta_V = \frac{1}{|C_k|}\left(\sum_{k=1}^{K}\mathcal{V}_k + \frac{a}{a-1}\sum_{k=1}^{K} Z_k\right) \text{ with } Z_k = \frac{1}{a}\sum_{j=1}^{a}\sigma_{k,j}^2 \tag{3}$$

and the vector $\sigma_k \in \mathbb{R}^a$ contains the deviations of the vectors directed from m_k to every point $X_i \in C_k$. These deviations are computed componentwise as follows [15]:

$$\sigma_{k,j} = \frac{1}{|C_k|}\left(\sum_{X_i \in C_k}(X_{i,j} - m_{k,j})^2\right)^{1/2} \text{ for } j = 1, \ldots, a. \tag{4}$$

After the cluster to be split has been identified, e.g. the cluster C_{k_i}, center m_{k_i} is modified and a new center, m_{K+1}, is created out of m_{k_i}. Let σ_{k_i,j_M} be the largest component of the vector σ_{k_i} in (4). The only component modified in the center m_{k_i} is j_M; similarly, the component of the new center m_{K+1} that differs from m_{k_i} is m_{K+1,j_M}:

$$m_{k_i,j_M} = m_{k_i,j_M} + 1.5\sigma_{k_i,j_M} \text{ and } m_{K+1,j_M} = m_{k_i,j_M} - 1.5\sigma_{k_i,j_M}.$$

Algorithm 4 describes the main steps of the splitting process.

Algorithm 4 Splitting Algorithm

Require: K, m_k, C_k, for $k = 1, \ldots, K$
1: **for** $k = 1$ to K **do**
2: Compute $\mathcal{V}_k$ using (2)
3: Compute the vector σ_k using (4)
4: **end for**
5: Compute η_V using (3)
6: Identify the cluster, C_{k_i}, with maximum volume $\mathcal{V}_{k_i}$, (index $k_i \in \{1, 2, \ldots, K\}$)
7: **if** $\mathcal{V}_{k_i} \geq \eta_V$ **then**
8: Modify center m_{k_i} and create the new center m_{K+1} as previously described
9: Assign the points in cluster C_{k_i} to the new centers m_{k_i} and m_{K+1} using Algorithm 2
10: Update K;
11: **end if**
12: **return** m_k and C_k, $k = 1, \ldots, K$.

2.2 Fitness Probability Scores

In line 10 of Algorithm 1, improved cluster centers are obtained using an optimization method. Evolutionary algorithms and metaheuristics have been suggested for similar purposes. Unlike other proposals [7,13,16], among others, our

algorithm does not optimize all the current cluster centers. Only a few are selected for optimization. This way a reduction in computational effort is notable.

To choose the centers to be optimized, a fitness probability score ('FPscore') of the center is used. This 'FPscore' is a measurement of the cluster vicinity and fitness relative to the data points that have been assigned to that cluster. Thus, clusters with lower 'FPscore' are considered to have least fitted centers and undergo an optimization process; whereas the clusters with higher 'FPscore' have fitter centers and need not to be optimized. The 'FPscore' are computed as follows. Based on the current K cluster centers $m_1, m_2, \ldots, m_K$, the *fitness function*

$$\mathcal{F}_k = \frac{1}{|C_k|} \sum_{X_i \in C_k} \|X_i - m_k\|_2 \quad \text{with} \quad S = \sum_{k=1}^{K} \mathcal{F}_k$$

is used to define the 'FPscore', as follows:

$$FPscore_k = \frac{1}{K-1} \frac{S - \mathcal{F}_k}{S}. \tag{5}$$

We note that

- the sum of the $FPscore$ of the cluster centers is equal to one;
- a cluster center with a lower fitness $\mathcal{F}_k$ value is awarded a higher $FPscore$; thus allocating to a fitter cluster a *higher probability of selection* to be maintained to the next iteration;
- a cluster center with a larger $\mathcal{F}_k$ value is awarded a lower $FPscore$; thus allocating to a least fitted cluster center a *smaller probability of selection* so that the center position (relative to the data points that have been assigned to it) should be improved during optimization.

The idea is to maintain for the next iteration cluster centers that have a $FPscore$ higher than p_{FP}, set as the average of the $FPscore$ of the current cluster centers, and optimize the remaining cluster centers using an optimization method, e.g., the differential evolution (DE) algorithm, using the objective function known as 'sum of within-cluster distances'

$$WCD = \sum_{i=1}^{n} d_{i,k_i} \tag{6}$$

where the index k_i represents the index of cluster center closest to data point X_i, as described in (1). This optimization process is described in Algorithm 5.

2.3 Problem dependent parameters

In this subsection, we summarize the paper contributions concerning with the definition of problem dependent parameters for the proposed clustering algorithm:

- number of centers for initialization, $K = \max\{2, \lceil 0.01n \rceil\}$;

– minimum number of points in a cluster, $N_{\min} = \max\{2, \lceil 0.05n \rceil\}$ (for Algorithm 3);
– a threshold value for the smallest distance between centers, η_D - minimum amplitude (relative to the attributes) divided by current number of clusters (for Algorithm 3);
– a target value for the cluster with maximum hypervolume, η_V, as defined in (3) (for Algorithm 4);
– p_{FP} - average value of the $FPscore$ of the current cluster centers - defined in (5) (for Algorithm 5).

Algorithm 5 Optimization Algorithm

Require: m_k, C_k for $k = 1, \ldots, K$;
1: Compute $\mathcal{F}_k$ and $FPscore_k$ for $k = 1, \ldots, K$ using (5)
2: Compute $p_{FP} = \frac{1}{K} \sum_{k=1}^{K} FPscore_k$
3: Select the fitter cluster centers by checking if $FPscore_k \geq p_{FP}$, $k = 1, \ldots, K$ and save the indices in $\mathcal{K}$
4: Compute new cluster centers m_k^*, $k = 1, \ldots, K$, $k \notin \mathcal{K}$, using the centers $m_k, k \in \mathcal{K}$ as constant values,

$$\min_{m_k, k=1,\ldots,K,\, k \notin \mathcal{K}} WCD$$

5: Set $m_k \leftarrow m_k^*$
6: **return** m_k, $k = 1, \ldots, K$.

3 Computational Results

In this preliminary study, the algorithms are coded in MATLAB®. Two sets of data points with two attributes (see Subsect. 3.1 below), one set with four attributes (known as 'Iris') and one set with thirteen attributes (known as 'Wine') are used to compute and visualize the partitioning clustering.

In line 10 of Algorithm 5, the DE algorithm is used. This algorithm is run for 5 iterations and the size of population is 20 (and 50 for 'Iris' and 'Wine' problems). The other DE parameters are $\beta_m = 0.2$, $\beta_M = 0.8$ (bounds of the scaling factor), $p_{CR} = 0.2$ (crossover probability).

Some comparisons are included to evaluate the goodness of our clustering. The DBSCAN clustering algorithm [2] as well as the K-means clustering are used. DBSCAN is a density-based spatial clustering algorithm and depends on two parameters: ϵ (a numeric scalar that defines a neighborhood search radius around each point) and $MinPts$ (a positive integer the gives the minimum number of neighbors required for a core point). The code[3] has been used in our comparisons [17]. Our experience solving a large variety of data sets seem to show that

[3] https://www.mathworks.com/matlabcentral/fileexchange/52905-dbscan-clustering-algorithm

DBSCAN clustering is very sensitive to variations on the parameters, in particular ϵ. The code that implements the K-means clustering [18] is also used for comparative purposes, although with K-means the number of clusters K had to be specified in advance. The performances of the tested clustering algorithms are measured in terms of a cluster validity measure, the Davies-Bouldin (DB) index [19]. The DB index aims to evaluate intra-cluster similarity and inter-cluster differences by computing

$$DB = \frac{1}{K}\sum_{i=1}^{K} \max_{j=1,\ldots,K,\, j\neq i} \left\{ \frac{S_i + S_j}{d_{i,j}} \right\} \tag{7}$$

where S_i (resp. S_j) represents the average of all the distances between the center m_i (resp. m_j) and the points in cluster C_i (resp. C_j) and $d_{i,j}$ is the distance between m_i and m_j. The smallest DB index indicates a valid optimal partition. We note that when the number of clusters is one, the DB index is 0.

3.1 Data Sets

The first two problems have data randomly generated from particular distributions, have two attributes and have been drawn from MATLAB online manual. The last two problems are well-known in the literature.

Problem 1. 600 data points with $a = 2$.

```
mu1 = [2 2]; sigma1 = [0.9 -0.0255; -0.0255 0.9]; mu2 = [5 5];
sigma2 = [0.5 0 ; 0 0.3]; mu3 = [-2, -2]; sigma3 = [1 0 ; 0 0.9];
X = [mvnrnd(mu1,sigma1,200); mvnrnd(mu2,sigma2,200);
mvnrnd(mu3,sigma3,200)];
```

Problem 2. 2000 data points with $a = 2$.

```
mu1 = [1 2]; Sigma1 = [2 0; 0 0.5]; mu2 = [-3 -5]; Sigma2 = [1 0; 0 1];
X = [mvnrnd(mu1,Sigma1,1000); mvnrnd(mu2,Sigma2,1000)];
```

Problem 3. 'Iris' with 150 data points. It contains three categories (types of iris plant) with 4 attributes (sepal length, sepal width, petal length and petal width) [20].

Problem 4. 'Wine' with 178 data points. It contains chemical analysis of 178 wines derived from 3 different regions, with 13 attributes [20].

3.2 Results

Plots to visualize the clustering are included in Figs. 1 - 2. The results for Problems 3 and 4 are shown in Tables 1 and 2.

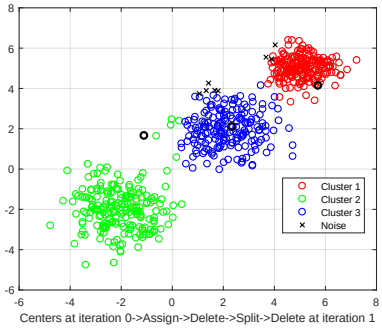
We show in Fig. 1 six plots. The first 3 plots correspond to the clustering that is obtained with our algorithm. Case (a) shows the clustering obtained

after the assign, delete, split and delete processes on the initial centers randomly generated. Case (b) shows the centers obtained by the first call to DE, where only the center of cluster 2 (green in the plot) has been optimized (thus, it is the least fitted cluster center among the three). From iteration 2 to iteration 3, only the cluster center of cluster 1 (red in the plot) is optimized - the other two are then considered as having a good fitter according to the corresponding clusters. Plots (d) and (e) display the clustering obtained by DBSCAN using $\epsilon = 0.5$ and $\epsilon = 0.75$ respectively. As it can be seen, DBSCAN with $\epsilon = 0.5$ considers a large number of points in the data set as 'noise'. Finally, plot (f) shows the clustering obtained with K-means when $K = 3$ is provided to the algorithm.

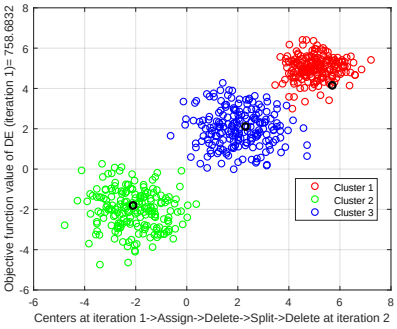
The six plots in Fig. 2 correspond to the clustering process applied to Problem 2. The visualized 3 centers in plot (a) have been randomly generated in the space of the points (during initialization) and were the only ones preserved after the assign, delete, split and delete processes. Comparing plots (a) and (b), it is possible to conclude that the 3 centers were selected to be optimized by the optimization algorithm. Based on their new positions, and the processes assign, delete, split and delete, one of the centers has been deleted. The iterative process ended at the 6th iteration with two cluster, and comparing with plot (b), the center of cluster 2 (the light blue one) has not been moved. For comparative purposes, we include plots (d) and (e) with the clustering obtained by DBSCAN, for $\epsilon = 0.5$ and $\epsilon = 1$ respectively. We note that clustering in (d) contains a large number of points coined as 'noise' which makes the DB index to be reduced. The plot (f) contains the result of K-means clustering when $K = 2$ is provided.

The results of our clustering algorithm, when solving Problems 3 and 4, are compared with those of [10] (that uses a particle swarm optimization (PSO) approach to the clustering) and [12] (an enhanced genetic algorithm (EGA)), see Table 1. To compare the performance, our algorithm was run 30 times for each data set. When solving the 'Iris' problem, our algorithm finds the 3 cluster in 100% of the runs (30 successful runs out of 30). When solving the problem 'Wine', 22 out of 30 runs identified 3 clusters. In the table, we show the optimal objective function value WCD (the best, the average (avg.) and the worst values over the successful runs). Although our values of WCD are slightly higher than those registered in [10] and [12], the WCD of the best runs are of the same order of magnitude than their competitors. We also note that the variation between the best, avg. and worst WCD values is larger than the variations reported in the papers in comparison. Since the basic DE algorithm is applied to optimize the cluster centers during the iterative automatic clustering, whereas the PSO and EGA algorithms (in [10] and [12]) have been specifically designed and modified to integrate the clustering process (with assign, remove and split processes) into the heuristic algorithm, explains the differences in objective function variations.

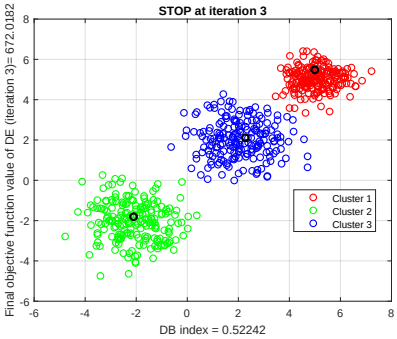
Unlike the PSO approach in [10], our clustering methodology does not seem to be affected by the dimension of the data set $\mathbf{X}$ as far as computational time is concerned (see 'time' - in seconds - in Table 1). Table 2 shows the cluster centers obtained in our best clustering.



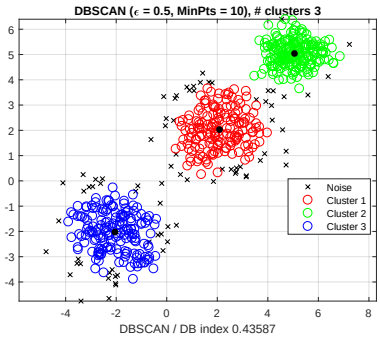
(a) Algorithm 1 at iteration 1



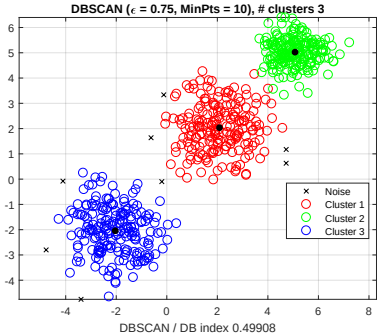
(b) Algorithm 1 at iteration 2



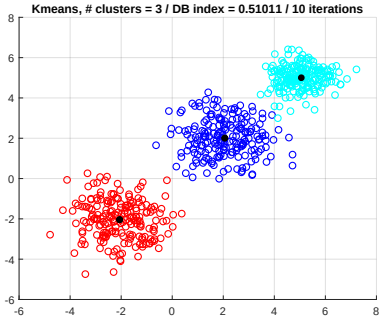
(c) Algorithm 1 results



(d) DBSCAN $\epsilon = 0.5$

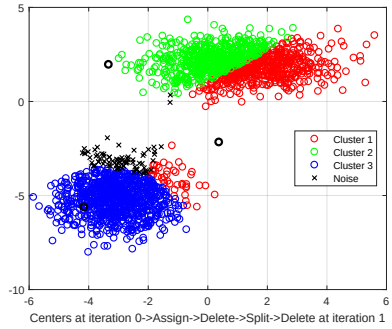


(e) DBSCAN $\epsilon = 0.75$

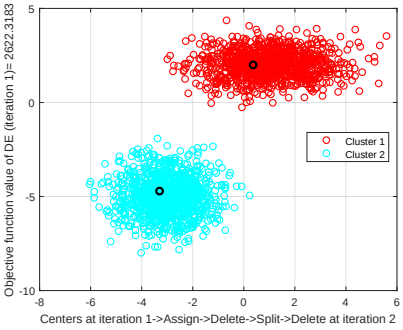


(f) K-means $K = 3$

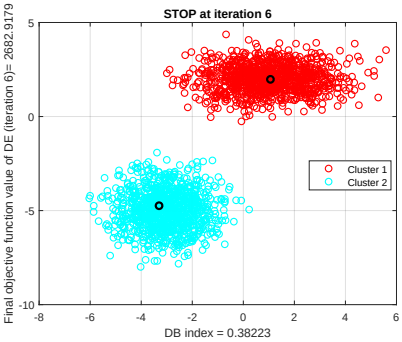
Fig. 1. Visualization of the results for Problem 1



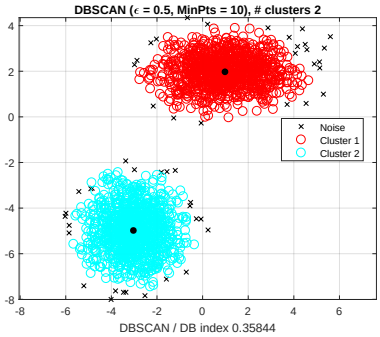
(a) Algorithm 1 at iteration 1



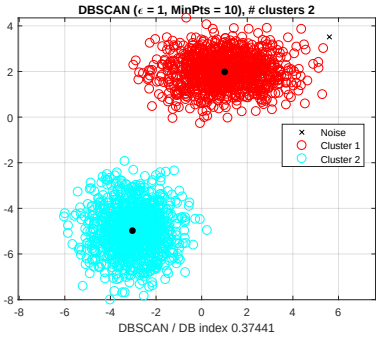
(b) Algorithm 1 at iteration 2



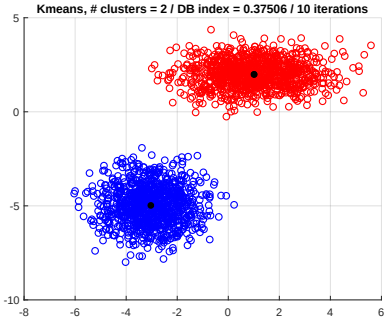
(c) Algorithm 1 results



(d) DBSCAN $\epsilon = 0.5$



(e) DBSCAN $\epsilon = 1$



(f) K-means $K = 2$

Fig. 2. Visualization of the results for Problem 2

From the reported experiments, we may conclude that the clustering strategy herein proposed presents satisfactory results, very similar to those in comparison, is effective and robust. In most of the tested runs, the number of expected/optimal clusters is found after a steady number of iterations and time.

Table 1. Clustering results for Problems 3 and 4

problem		Algorithm 1				results in [10]		results in [12]
		WCD	It	time	suc.	WCD	time	WCD
'Iris'	best	97.4149	4	1.107		97.22	0.343	97.1170
	avg.	98.0501	4.2	1.252	100%	97.22	0.359	97.0395
	worst	99.3268	6	1.758		97.22	0.375	97.3259
'Wine'	best	16540.69	5	1.063		16530.54	2.922	16499.32
	avg.	17013.92	8.0	1.941	73%	16530.54	2.944	16527.50
	worst	17998.60	10	2.752		16530.54	3.000	16555.68

Table 2. Centers of best clustering obtained by Algorithm 1, for Problems 3 and 4

'Iris'			'Wine'		
Center 1	Center 2	Center 3	Center 1	Center 2	Center 3
6.70740e+00	4.97307e+00	5.90988e+00	1.37146e+01	1.34037e+01	1.25181e+01
3.05717e+00	3.40055e+00	2.88684e+00	4.22842e+00	2.72431e+00	1.59380e+00
5.62126e+00	1.43060e+00	4.32993e+00	2.39641e+00	1.36000e+00	2.73048e+00
2.12030e+00	2.57315e-01	1.36495e+00	1.55840e+01	2.07645e+01	1.74152e+01
			7.97346e+01	1.07011e+02	1.08444e+02
			1.08178e+00	1.90686e+00	9.80000e-01
			2.55062e+00	2.27351e+00	2.64061e+00
			6.07698e-01	3.08006e-01	4.28489e-01
			8.66250e-01	2.13267e+00	8.28290e-01
			1.07218e+01	7.35849e+00	7.65873e+00
			1.46198e+00	1.28529e+00	9.60483e-01
			1.43378e+00	2.10639e+00	2.55457e+00
			4.54128e+02	6.85721e+02	1.14131e+03

4 Conclusions

The preliminary experiments, carried out with the proposed methodology to the clustering process of a set of given patterns, show that the clustering algorithm based on the fitness probability scores to select the cluster centers that should undergo an optimization process is effective and robust. The clustering methodology is an iterative process and relies on five sequentially applied main

processes, namely: i) assign points of the data set to the current cluster centers; ii) delete a cluster center if the number of points of the correspondent cluster is very small and merge two clusters if their distance is below a threshold; iii) split the cluster with maximum hypervolume if its value exceeds a target; iv) delete a center and merge two clusters if it appropriate; and v) optimize a specifically selected set of the cluster centers using an optimization algorithm, e.g., the DE algorithm.

The tested problems have compact and well separated clusters, except two of the clusters in the 'Wine' problem, but in the future, patterns with non-convex clusters, clusters with different shapes and sizes will be addressed. Our proposal is to integrate a kernel function into our clustering approach.

We aim to further investigate the dependence of the parameter values of the algorithm on the number of attributes, in particular, when highly dimensional data sets should be partitioned. The set of tested problems will also be enlarged to include patterns with larger number of clusters than those in analysis, patterns with a large number of attributes, and patterns of different shapes and non-convex.

Acknowledgments. The authors wish to thank two anonymous referees for their comments and suggestions to improve the paper.

References

1. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In L.M. Le Cam, J. Neyman (Eds.), Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, vol. 1, 281–297 (1967)
2. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. KKD-96 Proceedings 226–231. AAAI Press (1996)
3. Mohammed, J.Z., Meira Jr., W.: Data Mining and Machine Learning: Fundamental Concepts and Algorithms, 2nd Edition, Cambridge University Press, (2020)
4. Mirkin, B.: Clustering For Data Mining: A Data Recovery Approach. Computer Science and Data Analysis Series, Chapman & Hall/CRC (2005)
5. Higham, D.J., Kalna, G., Kibble, M.: Spectral clustering and its use in bioinformatics. J. Comput. Appl. Math. **204**, 25–37 (2007)
6. Haraty, R.A., Dimishkich, M., Masud, M.: An enhanced k-means clustering algorithm for pattern discovery in healthcare data. Int. J. Distrib. Sens. Netw. **2015**, Article ID 615740, 11 pages (2015)
7. Sarkar, M., Yegnanarayana, B., Khemani, D.: A clustering algorithm using evolutionary programming-based approach. Pattern Recognit. Lett. **18**, 975–986 (1997)
8. Das, S., Abraham, A., Konar, A.: Automatic clustering using an improved differential evalution algorithm. IEEE Trans. Syst. Man Cyber. Syst., **38**, 218–237 (2008)
9. Kwedlo, W.: A clustering method combining differential evolution with K-means algorithm. Pattern Recognit. Lett. **32**, 1613–1621 (2011)

10. Cura, T.: A particle swarm optimization approach to clustering. Expert Syst. Appl. **39**, 1582–1588 (2012)
11. Patel, KG.K., Dabhi, V.K., Prajapati, H.B.: Clustering using a combination of particle swarm optimization and K-means. J. Intell. Syst. **26**(3), 457–469 (2017)
12. El-Shorbagy, M.A., Ayoub, A.Y., Mousa, A.A., El-Desoky, I.M.: An enhanced genetic algoritm with new mutation for cluster analysis. Computational Statistics **34**, 1355–1392 (2019)
13. Ezugwu, A.E.-S., Agbaje, M.B., Aljojo, N., Els, R., Chiroma, H., Elaziz, M.A.: A comparative performance of hybrid firefly algorithms for automatic data clustering. IEEE Access 8, 121089–121118 (2020)
14. He, Z., Yu, C.: Clustering stability-based evolutionary K-means. Soft Computing **23**, 305–321 (2019)
15. Memarsadeghi, N., Mount, D.M., Netanyahu, N.S., Le Moigne, J.: A fast implementation of the ISODATA clustering algorithm. Int. J. Computat. Geom. Appl. **17**(1), 71–103 (2007)
16. Prabha, K.A., Visalakshi, N.K.: Improved particle swarm optimization based K-means clustering. International Conference on Intelligent Computing Applications, IEEE Publisher CPS 59–63, DOI 10.1109(ICICA.2014.21 (2014)
17. Mostapha Kalami Heris, Evolutionary Data Clustering in MATLAB (URL: https://yarpiz.com/64/ypml101-evolutionary-clustering), Yarpiz (2015)
18. Asvadi, A.: K-means Clustering Code. Department of ECE, SPR Lab., Babol (Noshirvani) University of Technology, http://www.a-asvadi.ir/ (2013)
19. Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. vol. PAMI-1(2), 224–227 (1979)
20. Dua, D., Graff, C.: UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science (2019)