

A Simple Clustering Algorithm Based on Weighted Expected Distances^{*}

Ana Maria A. C. Rocha (✉)¹[0000–0001–8679–2886],
M. Fernanda P. Costa²[0000–0001–6235–286X], and Edite M. G. P.
Fernandes¹[0000–0003–0722–9361]

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

Abstract. This paper contains a proposal to assign points to clusters, represented by their centers, based on weighted expected distances in a cluster analysis context. The proposed clustering algorithm has mechanisms to create new clusters, to merge two nearby clusters and remove very small clusters, and to identify points ‘noise’ when they are beyond a reasonable neighborhood of a center or belong to a cluster with very few points. The presented clustering algorithm is evaluated using four randomly generated and two well-known data sets. The obtained clustering is compared to other clustering algorithms through the visualization of the clustering, the value of the DB validity measure and the value of the sum of within-cluster distances. The preliminary comparison of results shows that the proposed clustering algorithm is very efficient and effective.

Keywords: Clustering Analysis · Partitioning algorithms · Weighted Distance

1 Introduction

Clustering is an unsupervised machine learning task that is considered one of the most important data analysis techniques in data mining. In a clustering problem, unlabeled data objects are to be partitioned into a certain number of groups (also called clusters) based on their attribute values. The objective is that objects in a cluster are more similar to each other than to objects in another cluster [1,2,3,4]. In geometrical terms, the objects can be viewed as points in a a -dimensional space, where a is the number of attributes. Clustering partitions these points into groups, where points in a group are located near one another in space.

^{*} This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the R&D Unit Project Scope UIDB/00319/2020.

There are a variety of categories of clustering algorithms. The most traditional include the clustering algorithms based on partition, the algorithms based on hierarchy, algorithms based on fuzzy theory, algorithms based on distribution, those based on density, the ones based on graph theory, based on grid, on fractal theory and based on model. The basic and core ideas of a variety of commonly used clustering algorithms, a comprehensive comparison and an analysis of their advantages and disadvantages are summarized in an interesting review of clustering algorithms [5]. The most used are the partitioning clustering methods, being the K-means clustering the most popular [6]. K-means clustering (and other K-means combinations) subdivide the data set into K clusters, where K is specified in advance. Each cluster is represented by the centroid (mean) of the data points belonging to that cluster and the clustering is based on the minimization of the overall sum of the squared errors between the points and their corresponding cluster center. While partition clustering constructs various partitions of the data set and uses some criteria to evaluate them, hierarchical clustering creates a hierarchy of clusters by combining the data points into clusters, and after that these clusters are combined into larger clusters and so on. It does not require the number of clusters in advance. However, once a merge or a split decision have been made, pure hierarchical clustering does not make adjustments. To overcome this limitation, hierarchical clustering can be integrated with other techniques for multiple phase clustering. Linkage algorithms are agglomerative hierarchical methods that considers merging clusters based on the distance between clusters. For example, the single-link is a type of linkage algorithm that merges the two clusters with the smallest minimum pairwise distance [7]. K-means clustering easily fails when the cluster forms depart from the hyper spherical shape. On the other hand, the single-linkage clustering method is affected by the presence of outliers and the differences in the density of clusters. However, it is not sensitive to shape and size of clusters. Model-based clustering assumes that the data set comes from a distribution that is a mixture of two or more clusters [8]. Unlike K-means, the model-based clustering uses a soft assignment, where each point has a probability of belonging to each cluster. Since clustering can be seen as an optimization problem, well-known optimization algorithms may be applied in the cluster analysis, and a variety of them have been combined with the K-means clustering, e.g. [9,10,11].

Unlike K-means and pure hierarchical clustering, the herein proposed partitioning clustering algorithm has mechanisms that dynamically adjusts the number of clusters. The mechanisms are able to add, remove and merge clusters depending on some pre-defined conditions. Although for the initialization of the algorithm, an initial number of clusters to start the points assigning process is required, the proposed clustering algorithm has mechanisms to create new clusters. Points that were not assigned to a cluster (using a weighted expected distance of a particular center to the data points) are gathered in a new cluster. The clustering is also able to merge two nearby clusters and remove the clusters that have a very few points. Although these mechanisms are in a certain way similar to others in the literature, this paper gives a relevant contribution

to the object similarity issue. Similarity is usually tackled indirectly by using a distance measure to quantify the degree of dissimilarity among objects, in a way that more similar objects have lower dissimilarity values. A probabilistic approach is proposed to define the weighted expected distances from the cluster centers to the data points. These weighted distances rely on the average variability of the distances of the points to each center with respect to their estimated expected distances. The larger the variability, the smaller the weighted distance. Therefore, weighted expected distances are used to assign points to clusters (that are represented by their centers).

The paper is organized as follows. Section 2 briefly describes the ideas of a cluster analysis and Sect. 3 presents the details of the proposed weighted distances clustering algorithm. In Sect. 4, the results of clustering four sets of data points with two attributes, one set with four attributes, and another with thirteen attributes are shown. Finally, Sect. 5 contains the conclusions of this work.

2 Cluster Analysis

Assume that n objects, each with a attributes, are given. These objects can also be represented by a data matrix $\mathbf{X}$ with n vectors/points of dimension a . Each element $X_{i,j}$ corresponds to the j th attribute of the i th point. Thus, given $\mathbf{X}$, a partitioning clustering algorithm tries to find a partition $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$ of K clusters (or groups), in a way that similarity of the points in the same cluster is maximum and points from different clusters differ as much as possible. It is required that the partition satisfies three conditions:

1. each cluster should have at least one point, i.e., $|C_k| \neq 0$, $k = 1, \dots, K$;
2. a point should not belong to two different clusters, i.e., $C_k \cap C_l = \emptyset$, for $k, l = 1, \dots, K$, $k \neq l$;
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 points 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.

3 Weighted Distances Clustering Algorithm

Distance based clustering is a very popular technique to partition data points into clusters, and can be used in a large variety of applications. In this section, a probabilistic approach is described that relies on the weighted distances (WD) of each data point X_i , $i = 1, \dots, n$ relative to the cluster centers m_k , $k = 1, \dots, K$. These WD are used to assign the data points to clusters.

3.1 Weighted Expected Distances

To compute the WD between the data point X_i and center m_k , we resort to the *distance vectors* DV_k , $k = 1, \dots, K$, and to the variations VAR_k , $k = 1, \dots, K$, the variability in cluster center m_k relative to all data points. Thus, let $DV_k = (DV_{k,1}, DV_{k,2}, \dots, DV_{k,n})$ be a n -dimensional distance vector that contains the distances of data points $X_1, X_2, \dots, X_n$ (a -dimensional vectors) to the cluster center $m_k \in \mathbb{R}^a$, where $DV_{k,i} = \|X_i - m_k\|_2$ ($i = 1, \dots, n$). The componentwise total of the *distance vectors*, relative to a particular data point X_i , is given by:

$$T_i = \sum_{k=1}^K DV_{k,i}.$$

The probability of m_k being surrounded and having a good centrality relative to all points in the data set is given by

$$p_k = \frac{\sum_{i=1}^n DV_{k,i}}{\sum_{i=1}^n T_i}. \quad (1)$$

Therefore, the expected value for the distance vector component i (relative to point X_i) of the vector DV_k is $E[DV_{k,i}] = p_k T_i$. Then, the WD is defined for each component i of the vector DV_k , $WD(k, i)$, using

$$WD(k, i) = DV_{k,i} / VAR_k \quad (2)$$

(corresponding to the weighted distance assigned to data point X_i by the cluster center m_k), where the variation VAR_k is the average amount of variability of the distances of the points to the center m_k , relative to the expected distances:

$$VAR_k = \left(\frac{1}{n} \sum_{i=1}^n (DV_{k,i} - E[DV_{k,i}])^2 \right)^{1/2}.$$

The larger the VAR_k , the lower the weighted distances $WD(k, i)$, $i = 1, \dots, n$ of the points relative to that center m_k , when compared to the other centers. A larger VAR_k means that in average the difference between the estimated expected distances and the real distances, to that center m_k , is larger than to the other centers.

3.2 The WD-based Algorithm

Algorithm 1 presents the main steps of the proposed weighted expected distances clustering algorithm. For initialization, a problem dependent number of clusters is provided, e.g. K , so that the algorithm randomly selects K points from the data set to initialize the cluster centers. To determine which data points are assigned to a cluster, the algorithm uses the WD as a fitness function. Each point X_i has a WD relative to a cluster center m_k , the above defined $WD(k, i)$.

The lower the better. The minimization of $WD(k, i)$ relative to k takes care of the local property of the algorithm, assigning X_i to cluster C_k .

Algorithm 1 Clustering Algorithm

Require: data set $\mathbf{X} \equiv X_{i,j}$, $i = 1, \dots, n$, $j = 1, \dots, a$, $It_{\max}$, $N_{\min}$

- 1: Set initial K ; $It = 1$
 - 2: Randomly select a set of K points from the data set $\mathbf{X}$ to initialize the K cluster centers m_k , $k = 1, \dots, K$,
 - 3: **repeat**
 - 4: Based on m_k , $k = 1, \dots, K$, assign the data points to clusters and eventually add a new cluster, using Algorithm 2
 - 5: Compute η_D using (4)
 - 6: Based on η_D , eventually remove clusters by merging two at a time, and remove clusters with less than $N_{\min}$ points, using Algorithm 3
 - 7: Set $\tilde{m}_k \leftarrow m_k$, $k = 1, \dots, K$
 - 8: Compute improved values for the cluster centers, m_k , $k = 1, \dots, K$ using Algorithm 4
 - 9: Set $It = It + 1$
 - 10: **until** $It > It_{\max}$ **or** $\max_{k=1, \dots, K} \|\tilde{m}_k - m_k\|_2 \leq 1e - 3$
 - 11: **return** K^* , m_k , $k = 1, \dots, K^*$ and optimal $\mathcal{C}^* = \{C_1^*, C_2^*, \dots, C_{K^*}^*\}$.
-

Thus, to assign a point X_i to a cluster, the algorithm identifies the center m_k that has the lowest weighted distance value to that point, and if $WD(k, i)$ is inferior to the quantity $1.5\eta_k$, where η_k is a threshold value for the center m_k , the point is assigned to that cluster. On the other hand, if $WD(k, i)$ is between $1.5\eta_k$ and $2.5\eta_k$ a new cluster C_{K+1} is created and the point is assigned to C_{K+1} ; otherwise, the $WD(k, i)$ value exceeds $2.5\eta_k$ and the point is considered ‘noise’. Thus, all points that have a minimum weighted distance to a specific center m_k between $1.5\eta_k$ and $2.5\eta_k$ are assigned to the same cluster C_{K+1} , and if the minimum weighted distances exceed $2.5\eta_k$, the points are ‘noise’. The goal of the threshold η_k is to define a bound to border the proximity to a cluster center. The limit $1.5\eta_k$ defines the region of a good neighborhood, and beyond $2.5\eta_k$ the region of ‘noise’ is defined. Similarity in the weighted distances and proximity to a cluster center m_k is decided using the threshold η_k . For each cluster k , the η_k is found by using the average of the $WD(k, \cdot)$ of the points which are in the neighborhood of m_k and have magnitudes similar to each other. The similarity between the magnitudes of the WD is measured by sorting their values and checking if the consecutive differences remain lower than a threshold Δ_{WD} . This value depends on the number of data points and the median of the matrix WD values (see Algorithm 2 for the details):

$$\Delta_{WD} = \frac{n-1}{n} \text{median}[WD(k, i)]_{k=1, \dots, K, i=1, \dots, n}. \quad (3)$$

During the iterative process, clusters that are close 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 defined as a function of the search region of the data points and also depends on the current number of clusters and number of attributes of the data set:

$$\eta_D = \frac{\min_{j=1,\dots,a} \{\max_i X_{i,j} - \min_i X_{i,j}\}}{K(a-1)}. \quad (4)$$

Furthermore, all clusters that have a relatively small number of points, i.e., all clusters that verify $|C_k| < N_{\min}$, where $N_{\min} = \min\{50, \max\{2, \lceil 0.05n \rceil\}\}$, are removed, their centers are deleted, and the points are coined as ‘noise’ (for the current iteration), see Algorithm 3.

Algorithm 2 Assigning Points Algorithm

Require: m_k for $k = 1, \dots, K$, data set $X_{i,j}$, $i = 1, \dots, n$, $j = 1, \dots, a$

- 1: Compute each component i of the vector DV_k , $i = 1, \dots, n$ ($k = 1, \dots, K$)
- 2: Compute matrix of WD as shown in (2)
- 3: Compute Δ_{WD} using (3)
- 4: **for** $k = 1, \dots, K$ **do**
- 5: $WD_{\text{sort}}(k) \leftarrow \text{sort}(WD(k, :))$
- 6: **end for**
- 7: **for** $k = 1, \dots, K$ **do**
- 8: **for** $i = 1, \dots, n - 1$ **do**
- 9: **if** $(WD_{\text{sort}}(k, i + 1) - WD_{\text{sort}}(k, i)) > \Delta_{WD}$ **then**
- 10: break
- 11: **end if**
- 12: **end for**
- 13: Compute $\eta_k = \frac{\sum_{l=1}^i WD_{\text{sort}}(k, l)}{i}$
- 14: **end for**
- 15: **for** $i = 1, \dots, n$ **do**
- 16: Set $\underline{k} = \arg \min_{k=1,\dots,K} WD(k, i)$
- 17: **if** $WD(\underline{k}, i) < 1.5\eta_{\underline{k}}$ **then**
- 18: Assign point X_i to the cluster $C_{\underline{k}}$
- 19: **else**
- 20: **if** $WD(\underline{k}, i) < 2.5\eta_{\underline{k}}$ **then**
- 21: Assign point X_i to the new cluster C_{K+1}
- 22: **else**
- 23: Define point X_i as ‘noise’ (class X_{noise})
- 24: **end if**
- 25: **end if**
- 26: **end for**
- 27: Compute center m_{K+1} as the centroid of all points assigned to C_{K+1}
- 28: Update K
- 29: **return** K , m_k , C_k , for $k = 1, \dots, K$, and X_{noise} .

Summarizing, to achieve the optimal number of clusters, the Algorithm 1 proceeds iteratively by:

- adding a new cluster whose center is represented by the centroid of all points that were not assigned to any cluster according to the rules previously defined;
- defining points ‘noise’ if they are not in an appropriate neighborhood of the center that is closest to those points, or if they belong to a cluster with very few points;
- merging clusters that are sufficiently close to each other, as well as removing clusters with very few points.

On the other hand, the position of each cluster center can be iteratively improved, by using the centroid of the data points that have been assigned to that cluster and the previous cluster center [12,13], as shown in Algorithm 4.

Algorithm 3 Merging and Removing Clusters Algorithm

Require: $K, m_k, C_k, k = 1, \dots, K, \eta_D, N_{\min}, X_{noise}$

```

1: for  $k = 1$  to  $K - 1$  do
2:   for  $l = k + 1$  to  $K$  do
3:     Compute  $D_{k,l} = \|m_k - m_l\|_2$ 
4:     if  $D_{k,l} \leq \eta_D$  then
5:       Compute  $m_k = \frac{1}{|C_k| + |C_l|}(|C_k|m_k + |C_l|m_l)$ 
6:       Assign all points  $X \in C_l$  to the cluster  $C_k$ 
7:       Delete  $m_l$  (and remove  $C_l$ )
8:       Update  $K$ 
9:     end if
10:  end for
11: end for
12: for  $k = 1, \dots, K$  do
13:   if  $|C_k| < N_{\min}$  then
14:     Define all  $X \in C_k$  as points ‘noise’ (add to class  $X_{noise}$ )
15:     Delete  $m_k$  (remove  $C_k$ )
16:     Update  $K$ 
17:   end if
18: end for
19: return  $K, m_k, C_k$  for  $k = 1, \dots, K$ , and  $X_{noise}$ .
```

Algorithm 4 Update Cluster Centers Algorithm

Require: $m_k, C_k, k = 1, \dots, K$,

1: Update all cluster centers:

$$m_k = \frac{1}{|C_k| + 1} \left(\sum_{X_l \in C_k} X_l + m_k \right) \quad \text{for } k = 1, \dots, K$$

2: **return** $m_k, k = 1, \dots, K$

4 Computational Results

In this section, some preliminary results are shown. Four sets of data points with two attributes, 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. The algorithms are coded in MATLAB®. The performance of the Algorithm 1, based on Algorithm 2, Algorithm 3 and Algorithm 4, depends on some parameter values that are problem dependent and dynamically defined: initial $K = \min\{10, \max\{2, \lceil 0.01n \rceil\}\}$, $N_{\min} = \min\{50, \max\{2, \lceil 0.05n \rceil\}\}$, $\Delta_{WD} > 0$, $\eta_k > 0, k = 1, \dots, K$, $\eta_D > 0$. The clustering results were obtained for $It_{\max} = 5$, except for ‘Iris’ and ‘Wine’ where $It_{\max} = 20$ was used.

4.1 Generated Data Sets

The generator code for the first four data sets is the following:

Problem 1. ‘mydata’³ with 300 data points and two attributes.

Problem 2. 2000 data points and two attributes.

```
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. 200 data points and two attributes.

```
X = [randn(100,2)*0.75+ones(100,2); randn(100,2)*0.5-ones(100,2)];
```

Problem 4. 300 data points and two attributes.

```
mu1 = [1 2]; sigma1 = [3 .2; .2 2];
mu2 = [-1 -2]; sigma2 = [2 0; 0 1];
X = [mvnrnd(mu1,sigma1,200); mvnrnd(mu2,sigma2,100)];
```

In Figs. 1 - 4, the results obtained for Problems 1 - 4 are depicted:

- the plot (a) is the result of assigning the points (applying Algorithm 2) to the clusters based on the initialized centers (randomly selected K points of the data set);
- the plot (a) may contain a point, represented by the symbol ‘ $\diamond$ ’, which identifies the position of the center of the cluster that is added in Algorithm 2 with the points that were not assigned (according to the previously referred rule) to a cluster;
- plots (b) and (c) are obtained after the application of Algorithm 4 to update the centers that result from the clustering obtained by Algorithm 2 and Algorithm 3, at an iteration;

³ available at Mostapha Kalami Heris, Evolutionary Data Clustering in MATLAB (URL: <https://yarpiz.com/64/ypml101-evolutionary-clustering>), Yarpiz, 2015.

- the final clustering is visualized in plot (d);
- plots (a), (b), (c) or (d) may contain points ‘noise’ identified by the symbol ‘×’;
- plot (e) shows the values of the Davies-Bouldin (DB) index, at each iteration, obtained after assigning the points to clusters (according to the previously referred rule) and after the update of the cluster centers;
- plot (f) contains the K-means clustering when $K = K^*$ is provided to the algorithm.

The code that implements the K-means clustering [14] is 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 DB index [15]. The DB index aims to evaluate intra-cluster similarity and inter-cluster differences by computing

$$DB = \frac{1}{K} \sum_{k=1}^K \max_{l=1, \dots, K, l \neq k} \left\{ \frac{S_k + S_l}{d_{k,l}} \right\} \quad (5)$$

where S_k (resp. S_l) represents the average of all the distances between the center m_k (resp. m_l) and the points in cluster C_k (resp. C_l) and $d_{k,l}$ is the distance between m_k and m_l . The smallest DB index indicates a valid optimal partition.

The computation of the DB index assumes that a clustering has been done. The values presented in plots (a) (on the x-axis), (b), (c) and (d) (on the y-axis), and (e) are obtained out of the herein proposed clustering (recall Algorithm 2, Algorithm 3 and Algorithm 4). The values of the ‘DB index(m,X)’ and ‘CS index(m,X)’ (on the y-axis) of plot (a) come from assigning the points to clusters based on the usual minimum distances of points to centers. The term CS index refers to the CS validity measure and is a function of the ratio of the sum of within-cluster scatter to between-cluster separation [13]. The quantity ‘Sum of Within-Cluster Distance (WCD)’ (in plots (b), (c) and (d) (on the x-axis)) was obtained out of our proposed clustering process.

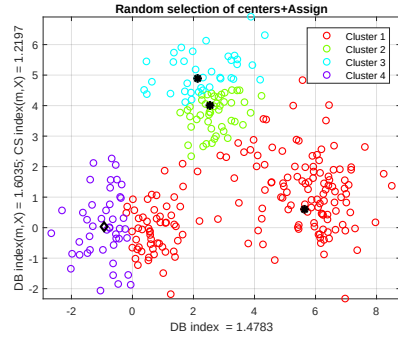
In conclusion, the proposed clustering is effective, very efficient and robust. As it can be seen, the DB index values that result from our clustering are slightly lower than (or equal to) those registered by the K-means clustering.

4.2 ‘Iris’ and ‘Wine’ data sets

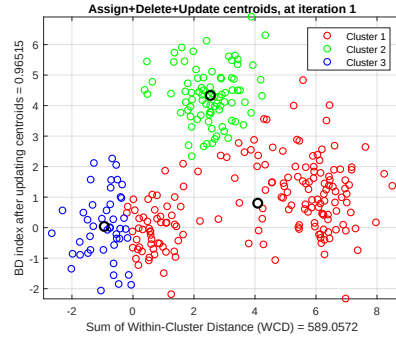
The results of our clustering algorithm, when solving Problems 5 and 6 are now shown. To compare the performance, our algorithm was run 30 times for each data set.

Problem 5. ‘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) [16].

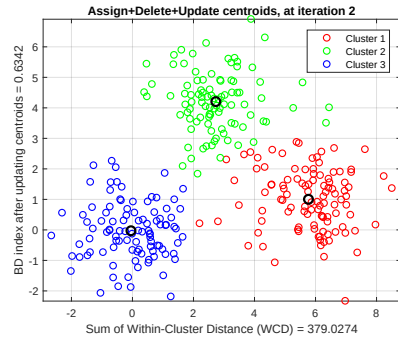
Problem 6. ‘Wine’ with 178 data points. It contains chemical analysis of 178 wines derived from 3 different regions, with 13 attributes [16].



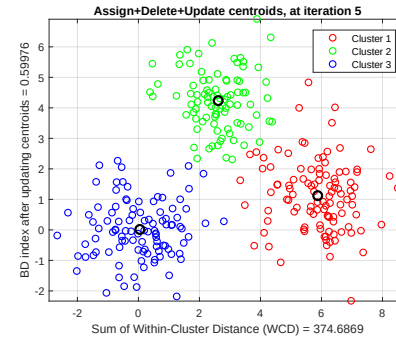
(a) Assign after initialization



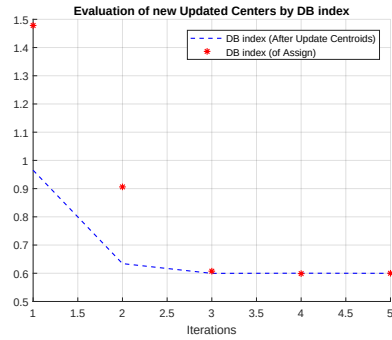
(b) 1st. iteration



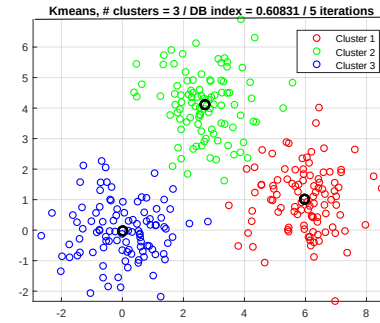
(c) 2nd. iteration



(d) final clustering (5th iteration)

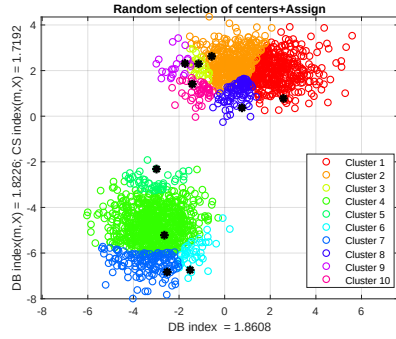


(e) DB index plot

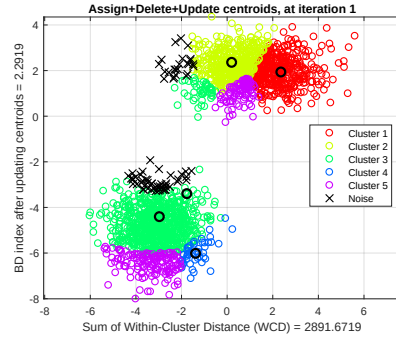


(f) K-means with $K = 3$

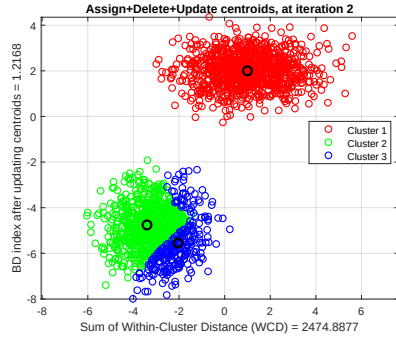
Fig. 1. Clustering process of Problem 1



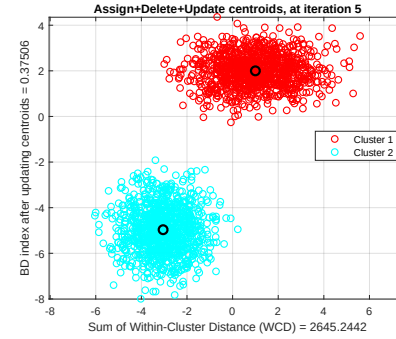
(a) Assign after initialization



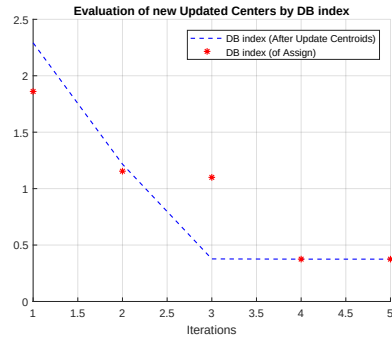
(b) 1st. iteration



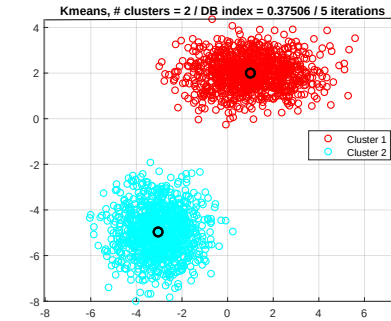
(c) 2nd. iteration



(d) final clustering (5th iteration)

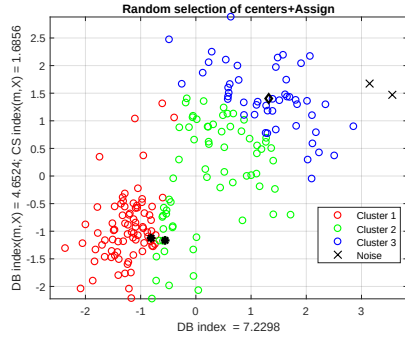


(e) DB index plot

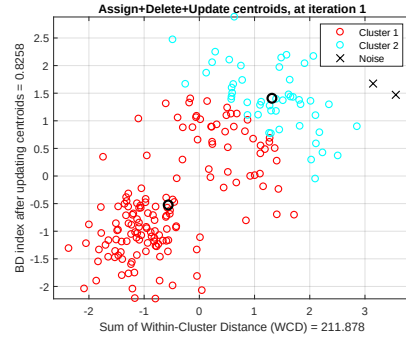


(f) K-means with $K = 2$

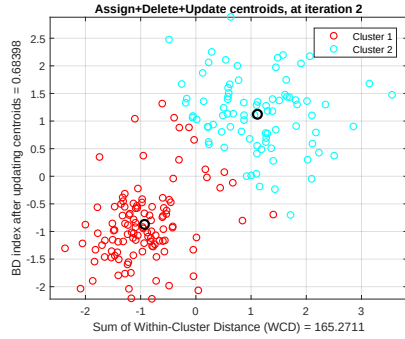
Fig. 2. Clustering process of Problem 2



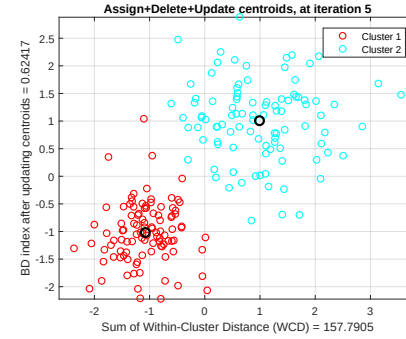
(a) Assign after initialization



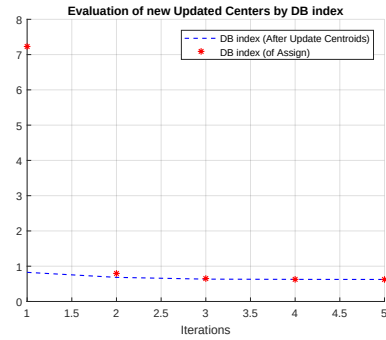
(b) 1st. iteration



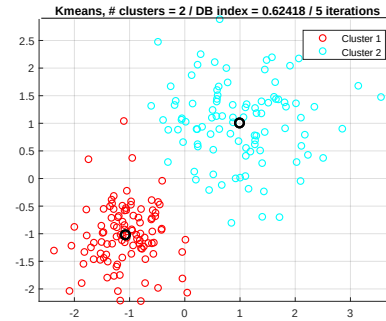
(c) 2nd. iteration



(d) final clustering (5th iteration)

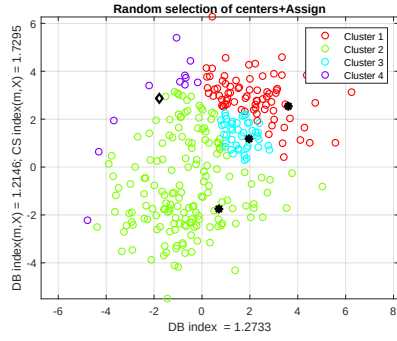


(e) DB index plot

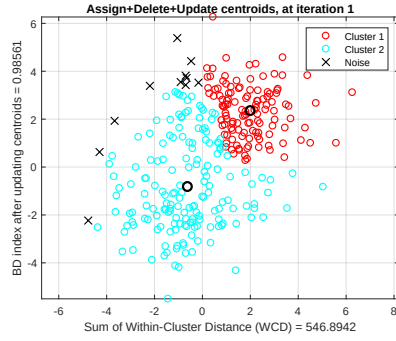


(f) K-means with $K = 2$

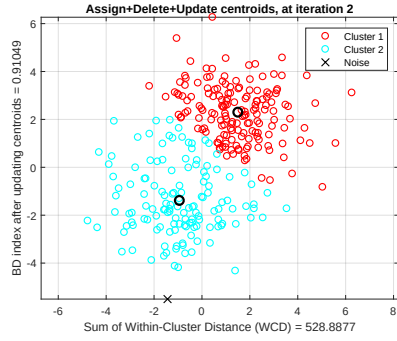
Fig. 3. Clustering process of Problem 3



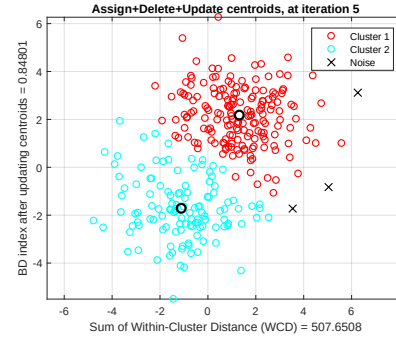
(a) Assign after initialization



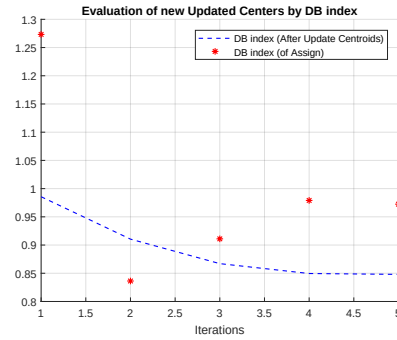
(b) 1st. iteration



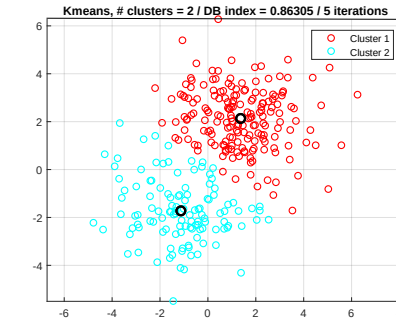
(c) 2nd. iteration



(d) final clustering (5th iteration)



(e) DB index plot



(f) K-means with $K = 2$

Fig. 4. Clustering process of Problem 4

The results are compared to those of [17], that uses a particle swarm optimization (PSO) approach to the clustering, see Table 1. When solving the ‘Iris’ problem, our algorithm finds the 3 clusters in 77% of the runs (23 successful runs out of 30). When solving the problem ‘Wine’, 27 out of 30 runs identified the 3 clusters.

Table 1 shows the final value of a fitness function, known as *Sum of Within-Cluster Distance* (WCD) (the best, the average (avg.), the worst and the standard deviation (St.Dev.) values over the successful runs). The third column of the table has the WCD that results from our clustering, i.e., after assigning the points to clusters (according to the above described rules based on the WD – Algorithm 2), merging and removing clusters if appropriate (Algorithm 3) and updating the centers (Algorithm 4), ‘WCD_{WD}’. The number of points ‘noise’ at the final clustering (of the best run, i.e., the run with the minimum value of ‘WCD_{WD}’), ‘# noise’, the number of iterations, *It*, the time in seconds, ‘time’, and the percentage of successful runs, ‘suc.’, are also shown in the table. The column identified with ‘WCD’ contains values of WCD (at the final clustering) using **X** and the obtained final optimal cluster centers to assign the points to the clusters/centers based on the minimum usual distances from each point to the centers.

For the same optimal clustering, the ‘WCD’ has always a lower value than ‘WCD_{WD}’. When comparing the ‘WCD’ of our clustering with ‘WCD’ registered in [17], our obtained values are higher, in particular the average and the worst values, except the best of the results ‘WCD’ for problem ‘Wine’. The variability of the results registered in [17] and [18] (see also Table 2) are rather small but this is due to the type of clustering based on the PSO algorithm. The clustering algorithm K-NM-PSO [18] is a combination of K-means, the local optimization method Nelder-Mead (NM) and the PSO. In contrast, our algorithm finds the optimal clustering much faster than the PSO clustering approach in [17]. This is a point in favor of our algorithm, an essential requirement when solving large data sets.

The results of our algorithm shown in Table 2 (third column identified with ‘WCD_D’) are based on the designed and previously described Algorithms 1, 2, 3 and 4, but using the usual distances between points and centers for the assigning process (over the iterative process). These values are slightly better than those obtained by the K-means (available in [18]), but not as interesting as those obtained by the K-NM-PSO (reported in [18]).

5 Conclusions

A probabilistic approach to define the weighted expected distances from the cluster centers to the data points is proposed. These weighted distances are used to assign points to clusters, represented by their centers. The new proposed clustering algorithm relies on mechanisms to create new clusters. Points that are not assigned to a cluster, based on the weighted expected distances, are gathered in a new cluster. The clustering is also able to merge two nearby clusters and

Table 1. Clustering results for Problems 5 and 6

problem		Algorithm 1					results in [17]	
		‘WCD _{WD} ’	# noise	<i>It</i>	time	suc.	‘WCD’	time
‘Iris’	best	107.89	0	5	0.051		100.07	97.22 0.343
	avg.	108.21		8.7	0.101	77%	100.17	97.22 0.359
	worst	109.13		12	0.156		100.47	97.22 0.375
	St.Dev.	0.56					0.18	
‘Wine’	best	17621.45	0	6	0.167		16330.19	16530.54 2.922
	avg.	19138.86		9.1	0.335	90%	17621.12	16530.54 2.944
	worst	20637.04		13	1.132		19274.45	16530.54 3.000
	St.Dev.	1249.68					1204.04	

Table 2. Results comparison for Problems 5 and 6

problem		Algorithm 1					K-means	K-NM-PSO
		‘WCD _D ’	# noise	<i>It</i>	time	suc.	‘WCD’	‘WCD’
‘Iris’	best	97.20	0	4	0.096		97.33	96.66
	avg.	97.21		8.7	0.204	63%	106.05	96.67
	St.Dev.	0.01					14.11	0.008
‘Wine’	best	16555.68	0	6	0.159		16555.68	16292.00
	avg.	17031.79		10.3	0.319	90%	18061.00	16293.00
	St.Dev.	822.08					793.21	0.46

remove the clusters that have very few points. Points are also identified as ‘noise’ when they are beyond a limit of a reasonable neighbourhood (as far as weighted distances are concerned) and when they belong to a cluster with very few points. The preliminary clustering results, and the comparisons with the K-means and other K-means combinations with stochastic algorithms, show that the proposed algorithm is very efficient and effective.

In the future, data sets with non-convex clusters, clusters with different shapes, in particular those with non-spherical blobs, 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 large data sets with clusters of different shapes and non-convex.

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

References

1. Jain A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. *ACM Comput. Surv.* **31**(3), 264–323 (1999)

2. Greenlaw, R., Kantabutra, S.: Survey of clustering: algorithms and applications. *Int. J. Inf. Retr. Res.* **3**(2), 29 pages (2013)
3. Ezugwu, A.E.: Nature-inspired metaheuristics techniques for automatic clustering: a survey and performance study. *SN Applied Sciences* **2**, 273–329 (2020)
4. Mohammed, J.Z., Meira Jr., W.: *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*, 2nd Edition, Cambridge University Press, (2020)
5. Xu, D., Tian, Y.: A comprehensive survey of clustering algorithms. *Ann. Data Sci.* **2**(2), 165–193 (2015)
6. 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)
7. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*. Cambridge University Press (2008)
8. Fraley, C., Raftery, A.E.: Model-based clustering, discriminant analysis and density estimation. *J. Am. Stat. Assoc.* **97**(458), 611–631 (2002)
9. Kwedlo, W.: A clustering method combining differential evolution with K-means algorithm. *Pattern Recognit. Lett.* **32**, 1613–1621 (2011)
10. Patel, K.G.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)
11. He, Z., Yu, C.: Clustering stability-based evolutionary K-means. *Soft Computing* **23**, 305–321 (2019)
12. Sarkar, M., Yegnanarayana, B., Khemani, D.: A clustering algorithm using evolutionary programming-based approach. *Pattern Recognit. Lett.* **18**, 975–986 (1997)
13. Chou, C.-H., Su, M.-C., Lai, E.: A new cluster validity measure and its application to image compression. *Pattern Anal. Applic.* **7**: 205–220 (2004)
14. Asvadi, A.: K-means Clustering Code. Department of ECE, SPR Lab., Babol (Noshirvani) University of Technology, <http://www.a-asvadi.ir/> (2013)
15. Davies, D.L., Bouldin, D.W.: A cluster separation measure. *IEEE Trans. Pattern Anal. Mach. Intell.* vol. PAMI-1(2), 224–227 (1979)
16. 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)
17. Cura, T.: A particle swarm optimization approach to clustering. *Expert Syst. Appl.* **39**, 1582–1588 (2012)
18. Kao, Y.-T., Zahara, E., Kao, I.-W.: A hybridized approach to data clustering. *Expert Syst. Appl.* **34**, 1754–1762 (2008)