Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Towards Parameter-Free Clustering for Real-World Data

Jian Hou[a,*], Huaqiang Yuan[a], Marcello Pelillo[b,c]

[a] School of Computer Science and Technology, Dongguan University of Technology, Dongguan 523808, China
[b] DAIS, Ca' Foscari University, Venice 30172, Italy
[c] European Centre for Living Technology, Ca' Foscari University, Venice 30123, Italy

ARTICLE INFO

Article history:
Received 27 December 2021
Revised 24 July 2022
Accepted 20 September 2022
Available online 22 September 2022

Keywords:
Clustering
Real-world data
Dominant set
Density peak

ABSTRACT

While many clustering algorithms have been published, existing algorithms are often afflicted by some problems in processing real-world data. We present an algorithm to deal with two of these problems in this paper. First, the majority of clustering algorithms depend on one or more parameters. Second, some algorithms are not suitable for clusters of Gaussian distribution, whereas clusters of many real datasets are of Gaussian distribution approximately. Our algorithm generates clusters sequentially, and each cluster is obtained by expanding an initial cluster. The initial cluster is extracted with the dominant set algorithm, and we study the correlation between the pairwise data similarity matrix and clustering result to determine the involved scaling parameter adaptively. In expanding the initial cluster, we improve the density peak algorithm so that the expansion will not cross the boundary between two clusters, and the involved density parameter has little influence on clustering results. In our algorithm, the cluster expansion enables our algorithm to work well with clusters of Gaussian distribution, and two involved parameters can be fixed or determined adaptively. Our algorithm goes a step forward in parameter-free clustering for real-world data, and it is shown to perform better than or comparably to some commonly used algorithms with parameters in experiments with synthetic datasets composed of Gaussian clusters and real datasets.

1. Introduction

Many clustering approaches have been published and some important algorithms include k-means, normalized cut (NCut) [1], DBSCAN [2] and mean shift [3], etc. Traditional clustering algorithms include partition-based [4], density-based [5], distribution-based [6] and hierarchical clustering [7]. In recent developments, spectral clustering [8], affinity propagation [9], multi-view clustering [10], ensemble clustering [11], subspace clustering [12] and some variants [13] have attracted much attention. While many algorithms partition the data points to obtain clusters, the dominant set (DSet) algorithm [14] designs a non-parametric measure to evaluate internal coherency and treats an internally coherent subset of data points as a cluster. Given the pairwise similarity matrix, this algorithm detects clusters sequentially. The density peak algorithm (DP) [15] identifies cluster centers by detecting local density peaks, and the members of each cluster are then determined using the density relationship among data points. It is shown to be promising in experiments, and the related works include [16]. In addition to the hard clustering algorithms which force one data point to belong to one single cluster, fuzzy clustering has also received much interests and many algorithms are published in, e.g., [17].

With a vast amount of clustering algorithms available, the simple k-means algorithm is still one of the most widely used approaches in processing real-world data. This implies that existing clustering algorithms are faced with some common challenges in real-world data clustering. We deal with two of these challenges in this paper. First, many algorithms depend on user-specified parameters, especially the number of clusters. The examples include k-means, spectral clustering and distribution based clustering, etc. Although there are already some parameter-free algorithms [18] and methods to estimate the number of clusters [19,20], our experiments in Section 4 show that it is still quite difficult to estimate the parameters accurately for real datasets. Second, some algorithms are not suitable for clustering data of Gaussian distribution. Different from synthetic data which can be designed to be of any distribution, in many cases data points of the same type from real-world problems follow the Gaussian distribution approximately [21]. If we treat the data points of the same type as a cluster, then the clusters in many real datasets follow the Gaussian distributions approximately. We will show

[*] Corresponding author.:
E-mail address: houjian@dgut.edu.cn (J. Hou).

https://doi.org/10.1016/j.patcog.2022.109062
0031-3203/© 2022 Elsevier Ltd. All rights reserved.

Table 1
Major symbols used in this paper.

symbol	meaning
$S \in R^{n\times dim}$	the set of n data of *dim*-dimension for clustering
$A = \{a_{ij}\}$	$n \times n$ pairwise similarity matrix
a_{ij}	similarity between i and j
$A' = \{a'_{ij}\}$	pairwise similarity matrix after histogram equalization
a'_{ij}	similarity between i and j after histogram equalization
D	a subset of S
$X = \{x_i\}$	weights of data points, used in DSet [14]
ρ_i	local density of i
δ_i	distance from i to the nearest larger-density neighbor
d_c	radius of neighborhood in density calculation, a parameter of DP [15]
d_{ij}	distance between i and j
$\overline{d}$	mean of pairwise distances
σ	scaling parameter in similarity calculation
ζ	standard deviation of pairwise similarity values
ϵ	percentage of data points in density calculation, a parameter of our algorithm
Eps	radius of neighborhood, a parameter of DBSCAN [2]
$MinPts$	number of data points in the Eps neighborhood, a parameter of DBSCAN [2]

in Section 3 that DBSCAN may not work well with this type of datasets.

In this paper we present a sequential clustering algorithm to deal with these two problems, and each cluster is obtained by expanding an initial cluster. The initial cluster is extracted based on the DSet algorithm, which requires the pairwise data similarity matrix as the only input. While a scaling parameter σ is involved in building the similarity matrix, we present a method to determine σ adaptively. The initial cluster extracted this way is usually a dense subset of a real cluster, and we expand it to obtain the final cluster with an improved DP algorithm. The original DP algorithm is suitable for clusters of Gaussian distribution, and we improve it so that the expansion will not cross the boundary between clusters. The expansion step involves a density parameter, which has little influence on clustering results in our experiments. In summary, our algorithm is able to generate good results with real datasets (composed of approximate Gaussian clusters), and two parameters can be fixed or determined adaptively.

Our work is motivated by the DSets-DBSCAN algorithm in [22], which is designed to accomplish parameter-free clustering by combining DSet and DBSCAN algorithms. Compared with [22], this paper has the following contributions. In the first step, we present a method to obtain the similarity matrix required by DSet clustering, thereby avoiding the problem caused by histogram equalization in the DSets-DBSCAN algorithm. In the second step, we present a DP-based cluster expansion method to replace the DBSCAN-based expansion in DSets-DBSCAN, thereby making our algorithm work well with real datasets composed of approximate Gaussian clusters. Based on these two contributions, in experiments our algorithm outperforms DSets-DBSCAN on most real datasets in clustering accuracy, and consumes less running time than the latter on all datasets.

2. Related Works

Our algorithm is closely related to the DSet, DP and DSets-DBSCAN algorithms. For completeness, in this part we introduce these three algorithms in brief and discuss their properties as a preparation of our approach. Before introducing the three algorithms, we firstly provide in Table 1 the major symbols used in this paper.

2.1. DSet

We define S as the set of n data points for clustering, $A = \{a_{ij}\}$ as the $n \times n$ pairwise similarity matrix, a_{ij} as the similarity between i and j, and D as a subset of S. With $i \in D$ and $j \notin D$, we then define

$$\phi_D(i,j) = a_{ij} - \frac{1}{|D|}\sum_{k\in D} a_{ik}, \tag{1}$$

where $|D|$ is the number of data points in D, and

$$w_D(i) = \begin{cases} 1, & \text{if} \quad |D| = 1, \\ \sum_{j\in D\setminus\{i\}} \phi_{D\setminus\{i\}}(j,i) w_{D\setminus\{i\}}(j), & \text{otherwise.} \end{cases} \tag{2}$$

Here $w_D(i)$ measures the relationship between two similarities, i.e., the average similarity between i and other members in D, and the average of pairwise similarities of the data points in D excluding i. Roughly speaking, $w_D(i) > 0$ means a close connection between i and the other members in D, whereas $w_D(i) < 0$ indicates the reverse.

With $W(D) = \sum_{i\in D} w_D(i)$, we call a subset D a dominant set if $W(T) > 0$ holds for any non-empty $T \subseteq D$ and

1. $w_D(i) > 0$, for all $i \in D$,
2. $w_{D\bigcup\{i\}}(i) < 0$, for all $i \notin D$.

In this definition, the first condition means that each member has a close connection with all the others in the same dominant set, indicating a large internal coherency. On the contrary, the second condition implies that no outside data point has a close connection with the data points inside a dominant set, indicating a large external incoherency. Based on these two properties, we treat dominant sets as clusters.

With the DSet algorithm, one can extract a cluster with the replicator dynamics or other dynamics [23]. With $X = \{x_i\}$, $i = 1, \cdots, n$ denoting the weights of n data points, these weights are calculated with the replicator dynamics as

$$x_i^{(t+1)} = x_i^{(t)} \frac{(Ax^{(t)})_i}{x^{(t)\prime}Ax^{(t)}}. \tag{3}$$

At convergence, the data points with weights above a threshold (0.0001 in this paper) form a cluster (dominant set), i.e.,

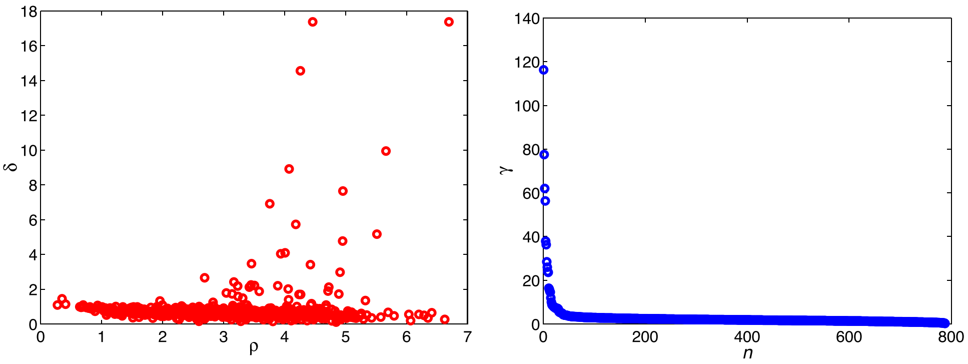
$$D = \{j\}, \quad s.t. \quad j \in S \quad and \quad x_j > 0.0001. \tag{4}$$

After one cluster is extracted, the next one is obtained in the remaining data points, and the clustering process is accomplished sequentially.

An open source library of the DSet algorithm is provided in [24], and some further works on DSet include [25,26]. By a reverse application of the dominant set concept, a divergent subset algorithm was proposed for remote sensing in [27]. This algorithm solves the long-standing problem of determining the number of endmembers, indicating a breakthrough in hyperspectral image processing. As dominant set has a large requirement of the internal similarity, the DSet algorithm tends to extract small clusters of large density from the densest area of a dataset. This property will be used in our algorithm to provide the initial clusters for subsequent expansion.

2.2. DP

The DP algorithm detects cluster centers by making use of the properties of local density peaks, and determines the membership of other data points with density relationship between data points. Local density reflects the data distribution in the neighborhood of a data point. In [15], the cutoff kernel defines local density as the number of neighbors in a neighborhood of radius d_c. Here d_c is a



(a) $\rho - \delta$ decision graph

(b) γ decision graph

Fig. 1. The $\rho - \delta$ decision graph and γ decision graph used in the DP algorithm. In the γ decision graph, the γ values of the data points are sorted in the decreasing order.

parameter in local density calculation. With the Gaussian kernel, the local density ρ_i of a data point i is calculated as

$$\rho_i = \sum_{j \in S, j \neq i} exp(-\frac{d_{ij}^2}{d_c^2}), \qquad (5)$$

where d_{ij} denotes the distance between i and j. With the local density available, a data point is called a density peak if it has larger local density than neighboring data points.

The distance δ_i to the nearest larger-density neighbor is designed as

$$\delta_i = \min_{j \in S, \rho_j > \rho_i} d_{ij}. \qquad (6)$$

As cluster centers of the DP algorithm are selected from local density peaks, they have large densities ρ and are surrounded by smaller-density data points. This means that they are distant from the nearest larger-density data points, indicating large δ. Noticing that non-center data points usually cannot have large ρ and large δ simultaneously, the DP algorithm uses this difference in cluster center identification. Specifically, it is proposed in [15] to identify cluster centers by a $\rho - \delta$ decision graph or γ decision graph with $\gamma = \rho\delta$. An example of the two decision graphs with Aggregation dataset is shown in Figure 1. In Figure 1(a) the data points with both large ρ and large δ are selected as cluster centers, and in Figure 1(b) the data points with large γ are selected as cluster centers. In the next step the other data points are clustered with the relative density relationship of data points. Specifically, each data point is assigned to be in the same cluster as the nearest larger-density neighbor.

One important contribution of the DP algorithm is the clustering method of non-center data points, i.e., one data point is assigned the label of the nearest larger-density data point. Therefore in DP clustering, the data points with large densities are included into clusters in the first place, followed by those with small densities. It can be observed that this algorithm generates clusters following the mode of cluster expansion, which starts from large-density areas and ends at small-density ones. This property makes DP especially suitable for clustering the data of Gaussian distribution, and therefore we will use this algorithm to expand the initial clusters generated by the DSet algorithm.

2.3. DSets-DBSCAN

In [22], DSets-DBSCAN is presented as a parameter-free clustering algorithm. Similar to the DSet algorithm, this algorithm extracts clusters sequentially. In extracting each cluster, an initial cluster is firstly extracted with DSet, and then a cluster expansion is accomplished based on the DBSCAN algorithm.

In extracting the initial clusters, the DSet algorithm uses the pairwise similarity matrix as the only input. With d_{ij} denoting the distance between i and j, a commonly used similarity measure is $a_{ij} = exp(-d_{ij}/\sigma)$, where σ is a scaling parameter. Evidently large σ's result in large similarity values, and then large initial clusters. An illustration is in Figure 2, where $\overline{d}$ denotes the mean of pairwise distances. In order to obtain desired initial clusters without parameter input, the DSets-DBSCAN algorithm uses histogram equalization [28] to transform the similarity matrix. As an image enhancement method, histogram equalization flattens the histogram of gray levels and increases the contrast of gray levels. By treating the original pairwise similarity matrix $A = \{a_{ij}\}$ as an image, we can also transform A to a new pairwise similarity matrix $A' = \{a'_{ij}\}$.

Specifically, we build a histogram $H = \{h_i\}$, $i = 1, \cdots, N_b$, of all similarity values, with N_b denoting the number of bins in the histogram. If a_{ij} lies in the $k-th$ bin, then

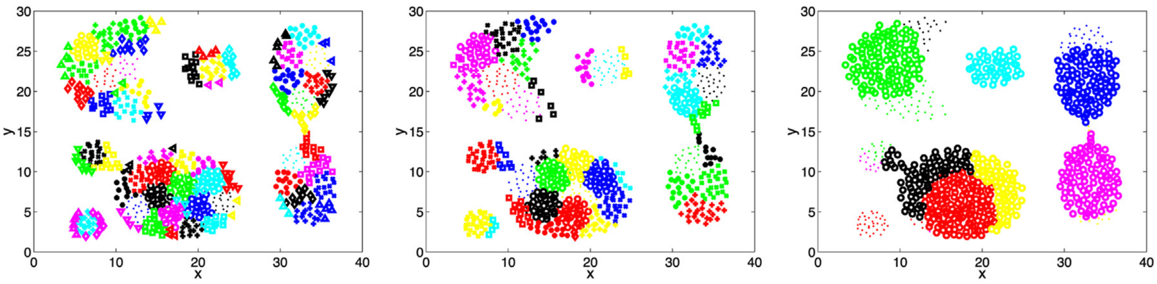
$$a'_{ij} = \sum_{m=1}^{k} \frac{h_m}{n^2}. \qquad (7)$$

As all the similarity values in the $k-th$ bin share the same new value, the calculations in Eq. (7) are conducted N_b times. As histogram equalization determines new similarities by sorting the original ones, we obtain a uniform distribution of new similarities, and new similarity matrices are invariant to σ's, as shown in Figure 3. Therefore the fist step does not involve any parameter. Since large and small similarities are distributed evenly, the generated clusters won't be too large or too small, suitable to be used as the initial clusters.

In the second step, the initial cluster is expanded based on the DBSCAN algorithm. The DBSCAN-based expansion treats the the data points in the initial cluster as core points, and then those data points in the neighborhood of core points are included into the cluster. The DBSCAN algorithm involves two parameters *MinPts* and *Eps*. In DSets-DBSCAN, *Eps* is determined based on *MinPts* as follows. For each data point i in the initial cluster, we calculate the distance between i and its *MinPts*-th nearest neighbor in the initial cluster. Then *Eps* is set as the maximum of these distances, which indicates the minimum density in the initial cluster. The only parameter in the second step is *MinPts*, which is selected to be 4 empirically in [22].

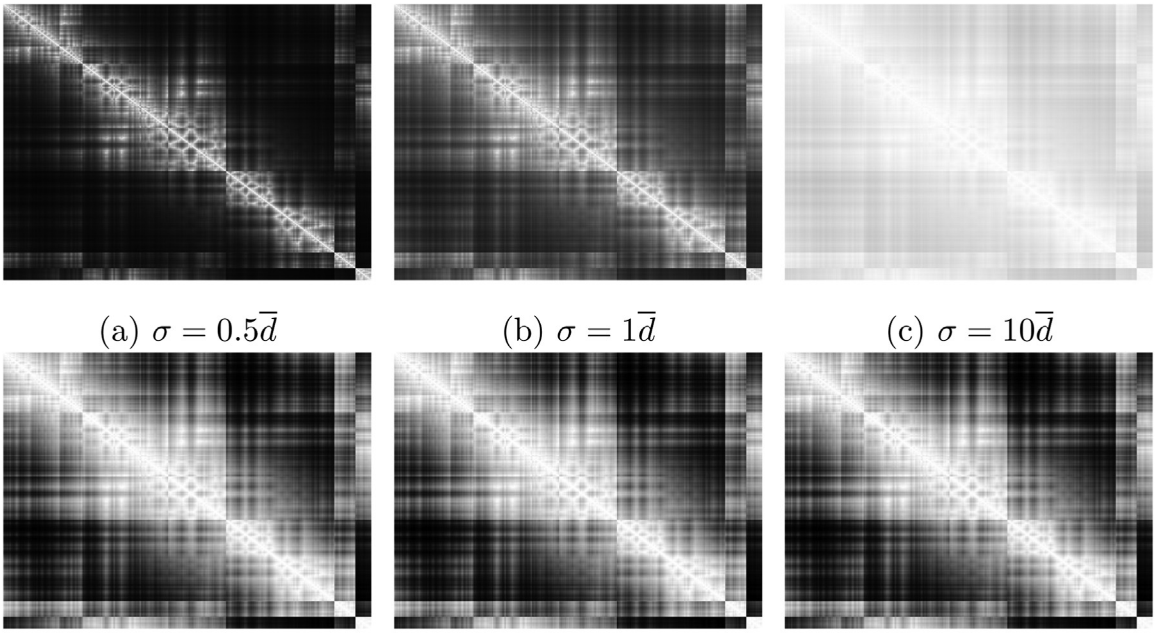
3. Our Approach

The parameter-free DSets-DBSCAN algorithm obtains clusters sequentially, and each cluster is obtained by firstly extracting an initial cluster with the DSet algorithm and then expanding the initial cluster with the DBSCAN algorithm. Our further work on this



(a) $\sigma = 0.5\overline{d}$ (b) $\sigma = 1\overline{d}$ (c) $\sigma = 10\overline{d}$

Fig. 2. Clustering results of DSet on Aggregation with different σ's.



(a) $\sigma = 0.5\overline{d}$ (b) $\sigma = 1\overline{d}$ (c) $\sigma = 10\overline{d}$

(d) $\sigma = 0.5\overline{d}$ (e) $\sigma = 1\overline{d}$ (f) $\sigma = 10\overline{d}$

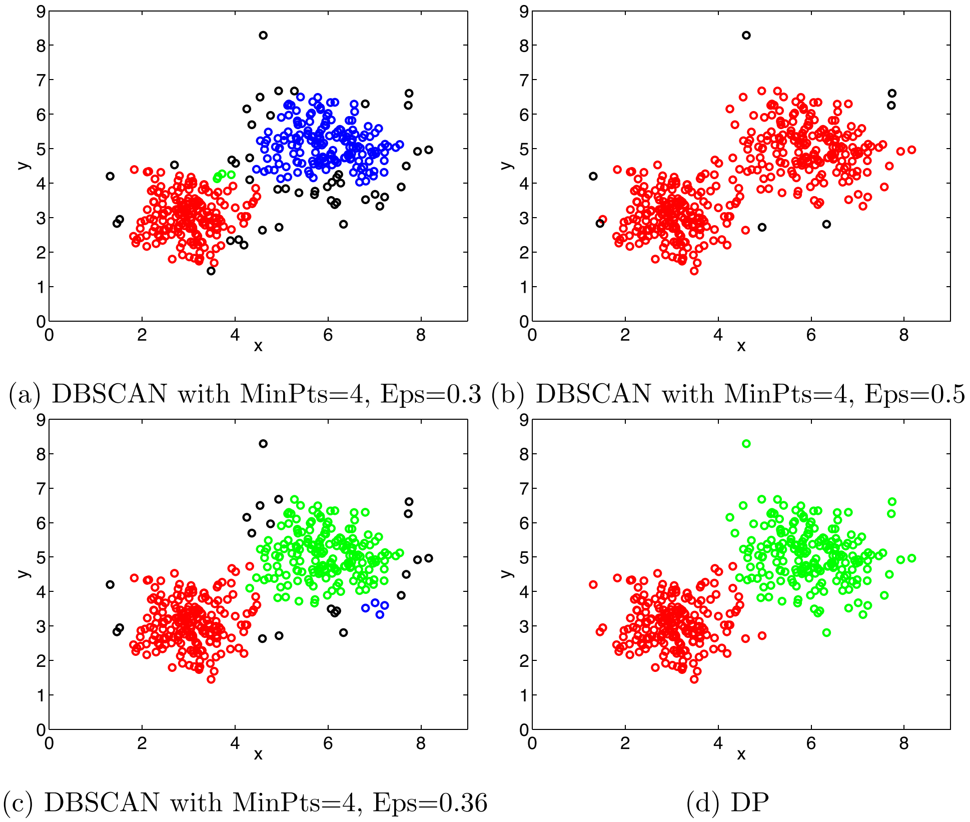
Fig. 3. Demonstration of similarity matrices before and after histogram equalization transformation. The top and bottom rows show the original and transformed similarity matrices, respectively. The similarity values are scaled to [0,255] to be displayed in the figure.

algorithm shows that it has some problems in dealing with real-world data. First, in extracting the initial cluster, the histogram equalization transformation is computationally expensive. Second, in expanding the initial cluster, the DBSCAN-based expansion may not work well with clusters of (approximate) Gaussian distribution. In order to solve these two problems, we present an adaptive method to calculate the similarity matrix in the first step, and a DP-based expansion method in the second step. The details of these two methods are presented below in details.

3.1. Initial Cluster Extraction

As we intend to expand the initial cluster to obtain the final one, the initial cluster should be a subset of a real cluster, indicating the small size of the initial cluster. Meanwhile, the initial cluster cannot be too small, since it should have sufficient information which can be used in subsequent cluster expansion. In addition, this step should have small or no dependence on parameters. In order to satisfy these conditions, the DSets-DBSCAN algorithm transforms the similarity matrix by histogram equalization, and then do DSet clustering to generate the initial cluster. However, a similarity matrix is composed of $n \times n$ similarity values, and transforming the similarity matrix by histogram equalization results in a large computation burden. This problem calls for a more efficient method to calculate the similarity matrix.

Histogram equalization changes the contrast of similarity values and therefore influences the sizes of initial clusters. In other words, there seems to be some correlation between the contrast of similarity values and the initial cluster sizes. As the contrast of similarity values can be measured by the standard deviation, we conduct experiments on 20 synthetic datasets composed of Gaussian clusters and 20 real datasets to study the influence of similarity matrices on clustering results. As a result, we find that our algorithm generates best or near-best results on the majority of datasets when the standard deviation ζ of the similarity values is about 0.1. This result means at $\zeta = 0.1$, the contrast of similarity values results in initial clusters of appropriate sizes, which are suitable for cluster expansion. The details of this part of experiments are presented in Section 4. Noticing that the increase of σ reduces the standard deviation of the similarity matrix, we present an adaptive method to determine σ and calculate the similarity matrix. Starting from $\sigma = \overline{d}$, we calculate the similarity matrix and the corresponding standard deviation ζ. If $\zeta > 0.1$, we increase σ



(a) DBSCAN with MinPts=4, Eps=0.3 (b) DBSCAN with MinPts=4, Eps=0.5

(c) DBSCAN with MinPts=4, Eps=0.36

(d) DP

Fig. 4. DBSCAN and DP clustering results on a dataset with clusters of Gaussian distribution. Dark circles denote unclustered data, and circles of other colors denote data in different obtained clusters.

by $\bar{d}$ and calculate the new similarity matrix and standard deviation, until the standard deviation is less than 0.1. This step typically requires to calculate the $n \times n$ similarity matrix and standard deviation for several times, and the time complexity is $O(n^2)$.

In addition, we show that the initial cluster from the DSet algorithm is a large-density subset in a real cluster. We calculate the weights of data points with Eq. (3), and then the data points with weights greater than a threshold form a cluster (dominant set). All the data points are assigned the same initial weights at the beginning. In the iteration, the data points with large average similarities with others receive larger weights gradually, whereas those not similar with others are assigned smaller and smaller weights. At convergence data points with large weights constitute a dominant set. Here we observe that data points in a dominant set are with larger densities than the neighboring outside data points. In other words, the initial cluster is a large-density subset in a real cluster. This property is shown to be useful in the subsequent DP-based cluster expansion.

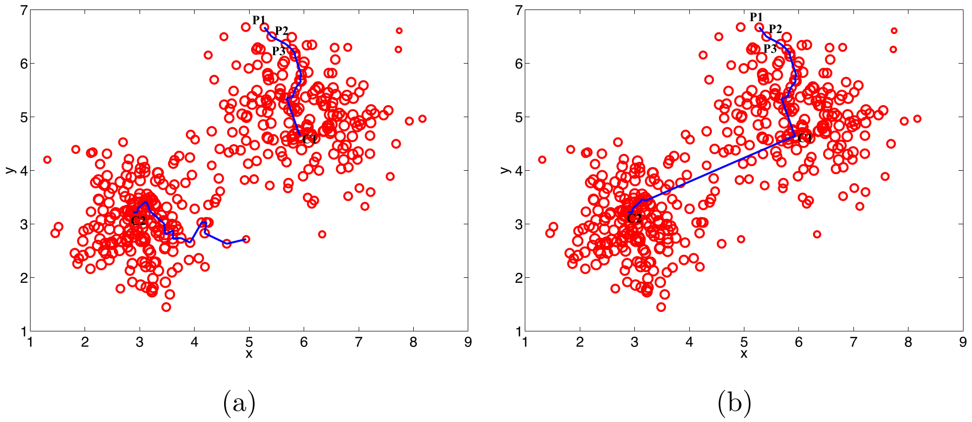
3.2. Cluster Expansion

In DSets-DBSCAN, the cluster expansion is accomplished with the DBSCAN algorithm. We have found that this expansion method may not be suitable for clusters of Gaussian distribution. The DBSCAN algorithm determines cluster members based on a density threshold, but selecting the appropriate density parameters is often difficult with datasets of non-uniform distribution. In a cluster of Gaussian distribution, the densities of data points near the border are evidently smaller than those of data points in the central area (Figure 4). In this case, a large density threshold may prevent some data points near the border from being included into this kind of clusters (Figure 4(a)). However, if we reduce the density threshold, two neighboring clusters may be merged into one, as shown in Figure 4(b). We fix $MinPts = 4$ (following [2]) and do a grid search of the parameter Eps in the range [0.1,1], and obtain the best result at $Eps = 0.36$, as shown in Figure 4(c). The result in Figure 4(c) is better than those in Figure 4(a) and Figure 4(b), but there are still some data points in the border which are left unclustered. This observation shows that the DBSCAN-based expansion method may not suitable for real-world data composed of approximate Gaussian clusters.

Considering the difficulty of DBSCAN-based expansion in processing clusters of Gaussian distribution, we propose to expand the initial cluster based on the DP algorithm. In Section 2.2 we find that the DP algorithm generates clusters in the mode of cluster expansion, starting from large-density cluster centers and ending at small-density data points. Meanwhile, in a cluster of Gaussian distribution, data points in the center have larger densities than those in the border. This observation seems to indicate that the DP algorithm is suitable for clusters of Gaussian distribution.

In Figure 5(a), a border data point P1 is connected to its nearest larger-density neighbor P2, and P2 is connected to its nearest larger-density neighbor P3. Repeating this process, P1 is finally linked to the cluster center C1, which has the largest density in the right cluster. Based on the DP algorithm, P1 is in the same cluster as P2, and P2 is in the same cluster as P3. Therefore P1, P2, P3, $\cdots$, are all in the cluster which has C1 as the cluster center. In this sense, we can say that P1 is *attracted* to C1 and included into the right cluster.

Now we observe that the DP algorithm has no requirement on the density of the inside data points, and each border data point of small density is attracted to a cluster center and included into the corresponding cluster, thereby solving the problem of DBSCAN shown in Figure 4(a). Meanwhile, if the two cluster centers (density peaks) C1 and C2 are identified correctly in Figure 5(a), the



(a)

(b)

Fig. 5. Demonstration of the clustering process of the DP algorithm. Circles denote data points, and circle sizes are proportional to data densities. One data point and its nearest neighbor with larger density are connected by a blue line.

data points surrounding C1 will be attracted to C1 and included into the right cluster, and those surrounding C2 be included into the left cluster. In this way, the two clusters will not be merged into one, and the problem shown in Figure 4(b) is solved. After the two problems shown in Figure 4(a) and Figure 4(b) are solved, the clustering result with DP is shown in Figure 4(d), where border data points are clustered, and two clusters are not merged into one.

The DP algorithm is able to solve the problem of DBSCAN-based expansion and generate the results in 4(d), on condition that the two cluster centers are identified correctly. However, it is difficult to determine the thresholds of ρ and δ based only on the $\rho - \delta$ decision graph, due to the ambiguity between large and small values of ρ and δ. We cannot even determine the number of clusters from Figure 1(a). The γ decision graph in Figure 1(b) reduces the difficulty to one threshold, but it is still not easy to find out the right number of clusters, i.e., 7 clusters in this example. In fact, many existing algorithms based on DP require to specify the number of clusters. However, even if the number of clusters is given, it is still possible that one cluster has multiple data points being identified as cluster centers, whereas another cluster has none [29].

In order to avoid the difficulty in identifying cluster centers, we choose to identify large-density areas instead. This is based on the observation that data densities vary smoothly in clusters of Gaussian distribution, and a cluster center tends to lie in a large-density area. Recall that with the DSet algorithm, the initial cluster is a large-density subset of a real cluster. Therefore we can expand the initial cluster with the DP algorithm to obtain the final cluster.

In the original DP algorithm, all the cluster centers are identified firstly, and then each non-center data point is assigned the same label as the nearest larger-density neighbor. In our algorithm, the initial clusters are extracted sequentially, and each time we have only one initial cluster. If we follow the original DP algorithm and sort unclustered data points in decreasing order of local density, the large-density data points including cluster centers in other clusters may be included. For example, in Figure 5(b) the cluster center C1 of the right cluster has a smaller density than some data points in the left cluster. In expanding the left cluster, if C1 hasn't been included into other clusters, it will be included into the left cluster. This further means the whole right cluster will be merged with the left one. It is possible to set a threshold on δ to avoid this problem, but as discussed above, such a threshold is often difficult to be determined. Therefore in this paper we present a different method to solve this problem.

Instead of considering unclustered data points according to decreasing order of local density, we sort the non-center data points in the increasing order according to their distances to the initial cluster. In this way, the nearest data points to the initial cluster, but not the data points of largest densities, are considered first. For each data point i in the queue, if its nearest larger-density neighbor is in the cluster, then i is included into the cluster. Here it is worth noticing that the newly included data points have smaller and smaller densities in the expansion. Once the cluster is expanded to the density valley between two clusters, the densities of neighboring data points starts to rise, as illustrated in Figure 5. Therefore the cluster expansion cannot cross the density valley and include the data points of other clusters, and the cluster merging problem is solved. One example of the clustering process is shown in Figure 6, where the left cluster is not expanded across the boundary between two clusters.

DP-based cluster expansion requires to estimate local densities of the data points. As the pairwise similarity matrix is already available in DSet clustering, we measure the local density by the average similarity with nearest neighbors. The number of nearest neighbors is set as the percentage ϵ of the number of data points. As will be shown in the experiments, the clustering results of our algorithm is not sensitive to this parameter, and we can select a fixed ϵ for different datasets without large performance loss.

3.3. Algorithm

We present a sequential clustering algorithm to deal with clusters of Gaussian distribution. In obtaining each cluster, we firstly extract an initial cluster based on the DSet algorithm. Then a DP-based method is proposed to expand the initial cluster to the final one. Noticing that these two steps involve two parameters σ and ϵ, we present a method to determine σ adaptively, and will show that ϵ has little influence on clustering results in a large range. The sequential clustering mode determines the number of clusters automatically, and the DP-based expansion method enables us to include border data points of small density while avoiding merging different clusters. In summary, our algorithm does not require to specify parameters, and is able to generate good results for datasets composed of approximate Gaussian clusters. Since the clusters in many real-world datasets are of Gaussian distribution approximately [21], our algorithm goes a step forward in achieving parameter-free clustering of real-world data. The whole clustering process is summarized in in Algorithms 1 to 2.

Our algorithm uses the Euclidean distance to calculate data similarity and density. With high-dimensional data, Euclidean distance is often not able to measure the data relationship accurately, thereby degrading the performance of our algorithm. Based on this observation, and motivated by the DenMune algorithm [30], we reduce the high-dimensional data to 2-D using the t-sne algorithm

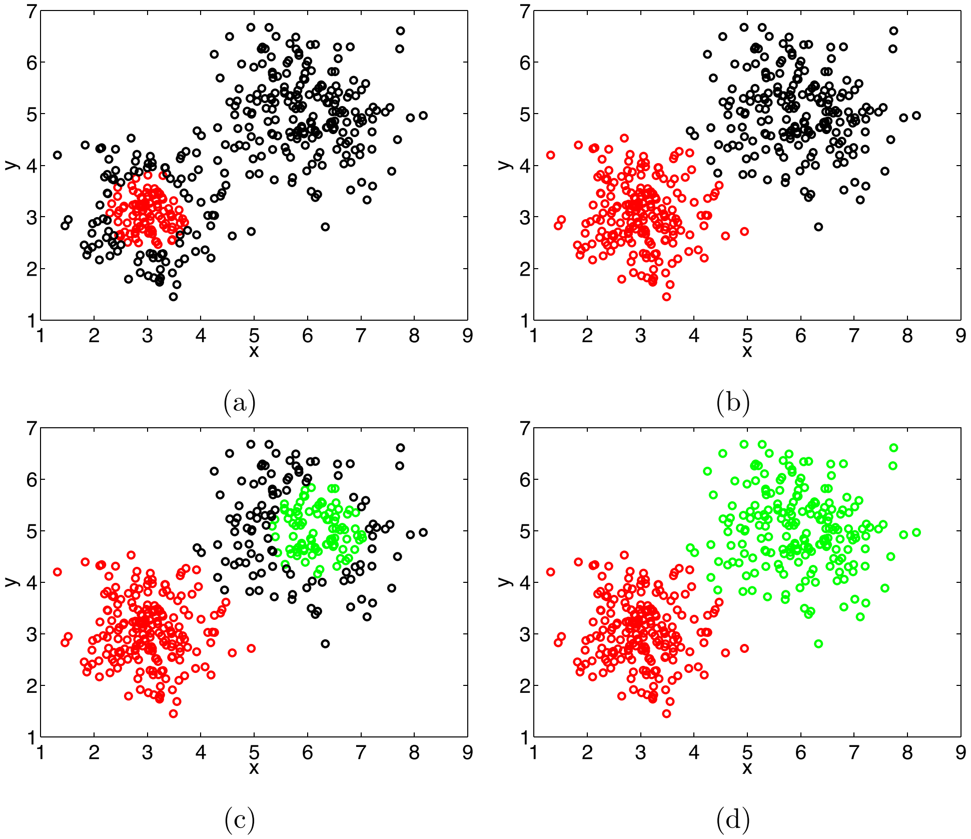


Fig. 6. Demonstration of the expansion process. From (a) to (d), we extract the first initial cluster, the first final cluster, the second initial cluster, and the second final cluster, respectively.

in the first step, and then do clustering with our algorithm. Similar to the results in [30], our experiments show that the results with the reduced 2-D data are better than those with the original high-dimensional data. The detailed comparisons are skipped for space reason.

4. Experiments

Experiments are conducted to test the performance of our algorithm. Firstly, we test the influence of the parameters σ and ϵ on clustering results. After that, our algorithm is compared with the DSets-DBSCAN algorithm. Finally, some other commonly used algorithms of different types are added into comparison.

The datasets used in experiments include 20 synthetic datasets and 20 real datasets. As our algorithm is designed for clusters of (approximate) Gaussian distribution, all the clusters of the 20 synthetic datasets are of Gaussian distribution. In synthetic datasets, D31, R15, Unbalance, S1, S2, A1, A2, A3, Dim032, Dim064, Dim128, Dim256, Dim512 and Dim1024 are taken from the clustering basic benchmark[1], and Varydensity is taken from the ELKI project[2]. The remaining 5 synthetic datasets Spread1, Spread2, Spread3, Spread4 and Spread5 of varying spreads and sizes are generated following [19]. Specifically, these 5 datasets cover different dimensions, different number of clusters and different sizes, and the clusters of Gaussian distribution are generated with different standard deviations. The 20 real datasets are taken from UCI machine learning repository. The features of these datasets are summarized in Table 2, where NP denotes the number of data points, ND represents the number of dimensions of the data, and NC stands for the number of clusters in the dataset.

4.1. Influence of parameters

Our algorithm involves two parameters, namely σ in extracting initial clusters and ϵ in expanding initial clusters. Motivated by histogram equalization which influences the contrast of similarity values, we propose to determine σ adaptively based on the standard deviation of similarity values. With $\sigma = \overline{d}, 2\overline{d}, \cdots, 20\overline{d}$, we calculate the similarity matrices and corresponding standard deviations, and then obtain the clustering results. The clustering results evaluated with NMI (normalized mutual information) with respect to standard deviation ζ are shown in Figure 7. As to another parameter ϵ, we test different values of ϵ and find that this parameter has little influence on the determination of ζ. In Figure 7 we present the results with $\epsilon = 2$ and $\epsilon = 3.6$.

We discuss the results in Figure 7 as follows. First, on the synthetic datasets, the best results can be obtained around $\zeta = 0.1$, and both larger and smaller ζ may result in significant performance loss. This observation is applicable to the cases with both $\epsilon = 2$ and $\epsilon = 3.6$. Second, on the real datasets, the variance of clustering results with respect to ζ is relatively small, and the best or near-best results of the datasets can also be obtained at $\zeta = 0.1$. This observation also holds with both $\epsilon = 2$ and $\epsilon = 3.6$. Based on this observation, we present the method to determine σ adaptively in Section 3.1.

To evaluate the influence of another parameter ϵ on the clustering results, we set $\zeta = 0.1$ and test the algorithm with ϵ ranging from 0.5 to 5 in the step of 0.1, and report the results on synthetic and real datasets in Figure 8.

[1] http://cs.joensuu.fi/sipu/datasets/
[2] https://elki-project.github.io/datasets/

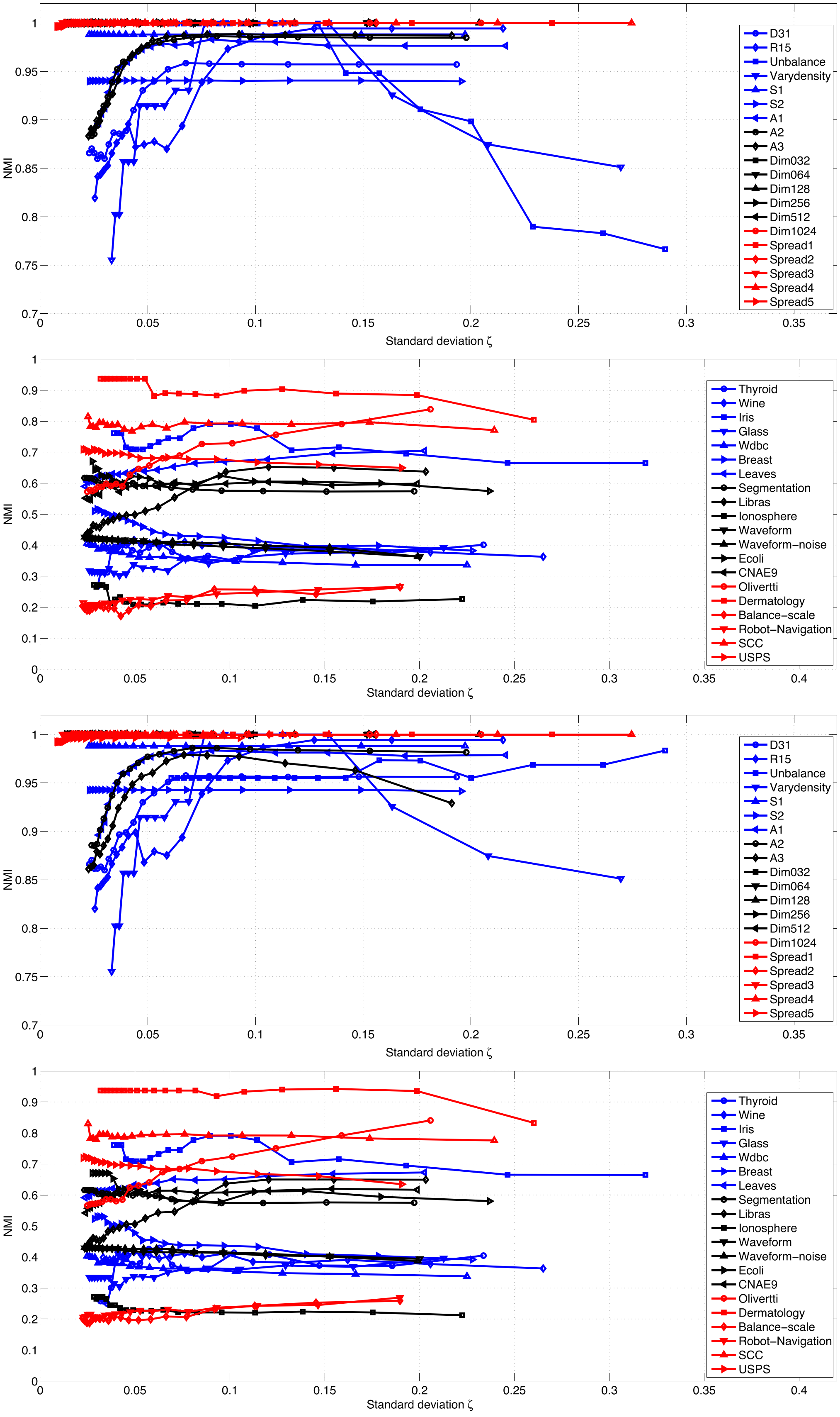
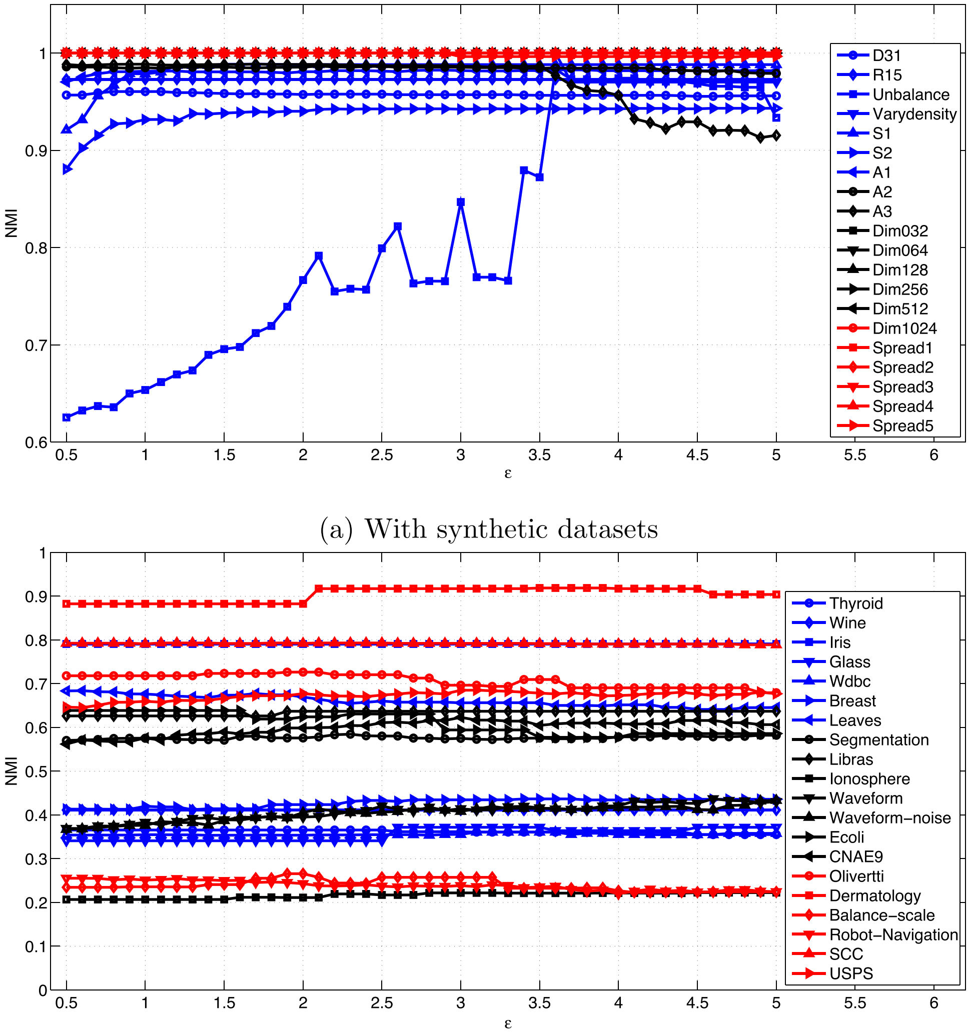


Fig. 7. The relationship between clustering results and standard deviations of similarity matrices. The top two are obtained with $\epsilon = 2$, and bottom two with $\epsilon = 3.6$.

Table 2
Characteristics of datasets.

dataset	NP	ND	NC	dataset	NP	ND	NC
D31	3100	2	31	Thyroid	215	5	3
R15	600	2	15	Wine	178	13	3
Unbalance	6500	2	8	Iris	150	4	3
Varydensity	150	2	3	Glass	214	9	6
S1	5000	2	15	Wdbc	569	30	2
S2	5000	2	15	Breast	699	9	2
A1	3000	2	20	Leaves	1600	64	100
A2	5250	2	35	Segmentation	2310	19	7
A3	7500	2	50	Libras	360	90	15
Dim032	1024	32	16	Ionosphere	351	34	2
Dim064	1024	64	16	Waveform	5000	21	3
Dim128	1024	128	16	Waveform-noise	5000	40	3
Dim256	1024	256	16	Ecoli	336	7	8
Dim512	1024	512	16	CNAE9	1080	856	9
Dim1024	1024	1024	16	Olivertti	400	92×112	40
Spread1	1000	2	10	Dermatology	366	33	6
Spread2	2000	10	20	Balance-scale	625	4	3
Spread3	3500	20	35	Robot-Navigation	5456	24	4
Spread4	200	35	2	SCC	600	60	6
Spread5	5000	50	50	USPS	11000	256	10



(a) With synthetic datasets

(b) With real datasets

Fig. 8. Clustering results with different ϵ.

We observe in Figure 8 that the parameter ϵ exerts little influence on the clustering results on all the real datasets and most of the synthetic datasets. Several exceptions include the S1, S2, A3 and Unbalance datasets. On the S1 and S2 datasets, too small ϵ causes an evident performance loss. On the A3 dataset, too large ϵ (>3.6) degrades the clustering result evidently. The influence of ϵ on the clustering result on Unbalance dataset is a little complex. With the increase of ϵ, the clustering result firstly improves, then experiences some fluctuations, and then reaches the peak at $\epsilon = 3.6$, and finally becomes stable. These observations propel us to select $\epsilon = 3.6$ for our algorithm.

While our algorithm involves two parameters ζ and ϵ, Figure 7 shows that $\zeta = 0.1$ generates the best or near-best results for most datasets and Figure 8 indicates that our algorithm is not sensitive to ϵ for most datasets. Based on these observations, our algorithm with $\zeta = 0.1$ and $\epsilon = 3.6$ can be regarded as a parameter-free algorithm. In the following our experiments are conducted with this parameter-free version of our algorithm.

4.2. Comparison with DSets-DBSCAN

In the DSets-DBSCAN algorithm, the parameter Eps is then calculated based on $MinPts$ and initial clusters (see details in [22]). Therefore the DSets-DBSCAN algorithm has one single parameter $MinPts$, which is set as 4 in [22]. In our experiment, we test $MinPts$ from 2 to 10 and find that DSets-DBSCAN generates the best average result with $MinPts = 4$. With this parameter, we compare DSets-DBSCAN with our algorithm and report their clustering results and running time in Figure 9 and Figure 10.

We discuss the results in Figure 9 as follows. First, our algorithm performs better than or comparably to DSets-DBSCAN on all the 20 synthetic datasets. This indicates that our algorithm is effective in dealing with clusters of Gaussian distribution. Second, our algorithms performs better than or comparably to DSets-DBSCAN on 17 out of 20 real datasets, and is outperformed by the latter evidently on only 3 real datsets. This comparison indicates the significant advantage of our algorithm over DSets-DBSCAN in clustering of real datasets.

In the experiments, our algorithm is outperformed by DSets-DBSCAN evidently on the Thyroid, Breast and Ionosphere datasets. As these three datasets are of high dimensions, we transform them to 2-D using t-sne method to observe their approximate data structure. Figure 11 shows that all three datasets have irregular shapes and the clusters are far from the Gaussian distribution. Consequently, our algorithm based on DP-expansion does not work well on these datasets. In contrast, DSets-DBSCAN based on a density threshold is more likely to overcome the irregular density distribution, thereby generating better results than our algorithm.

Algorithm 1 Our algorithm.

```
Input:   S   //dataset to be clustered
Output:  C = {c_i}, i = 1, ..., n   //labels of data points
A = {a_ij}, a_ij = exp(d_ij/σ), i, j = 1, 2, ..., n
c_i ← 0, i = 1, ..., n
l ← 0
while There exists i such that c_i = 0 do
    l ← l + 1
    Obtain D_0 with Eq. (3) and Eq. (4)
    for i ∈ D_0 do
        c_i ← l
    end for
    D_1 = {j}, subject to c_j = 0   //the set of unclustered data points
    D = DP(D0, D1)   //expanding initial cluster with DP
    for k ∈ D do
        c_k ← l
        a_ik ← 0, a_ki ← 0, i = 1, 2, ..., n
    end for
end while
```

Algorithm 2 The DP algorithm

```
Input:   initial cluster D_0, unclustered set of data D_1
Output:  final cluster D
for j ∈ D_1 do
    Calculate the average similarity η_j between j and those in D_0
end for
Sort the data in D_1 in the decreasing order of η_j, obtaining D_2
for j ∈ D_2 do
    if the nearest larger-density neighbor of j is in D_0 then
        D_0 ← D_0 ⋃ {j}
    end if
end for
D ← D_0
```

The comparison in Figure 9 shows that our algorithm outperforms DSets-DBSCAN on the majority of datasets. Furthermore, Figure 10 indicates that our algorithm consumes much less running time than DSets-DBSCAN on all the datasets. These comparisons show the advantage of our algorithm over DSets-DBSCAN in both clustering accuracy and computation efficiency with real datasets.

4.3. Comparison with other algorithms

We then make a comparison with several existing algorithms in experiments, including k-means, EM (Expectation Maximization), X-means [31], NCut, SPRG [32], DBSCAN, CBKM [33] and DenMune [30]. The parameter settings of these algorithms are as follows.

(1) As the k-means, EM, NCut, SPRG and CBKM algorithms requires to input the number of clusters, we feed them with the ground truth number of clusters and label these results by “-gt” in Table 3 to Table 6.

(2) We also estimate the number of clusters with the LastLeap method [19] and the eigen-gap method [20], and feed the k-means, EM and SPRG algorithms. The results obtained with the assistance of these two methods are labeled by “-LL” and “-egap”, respectively.

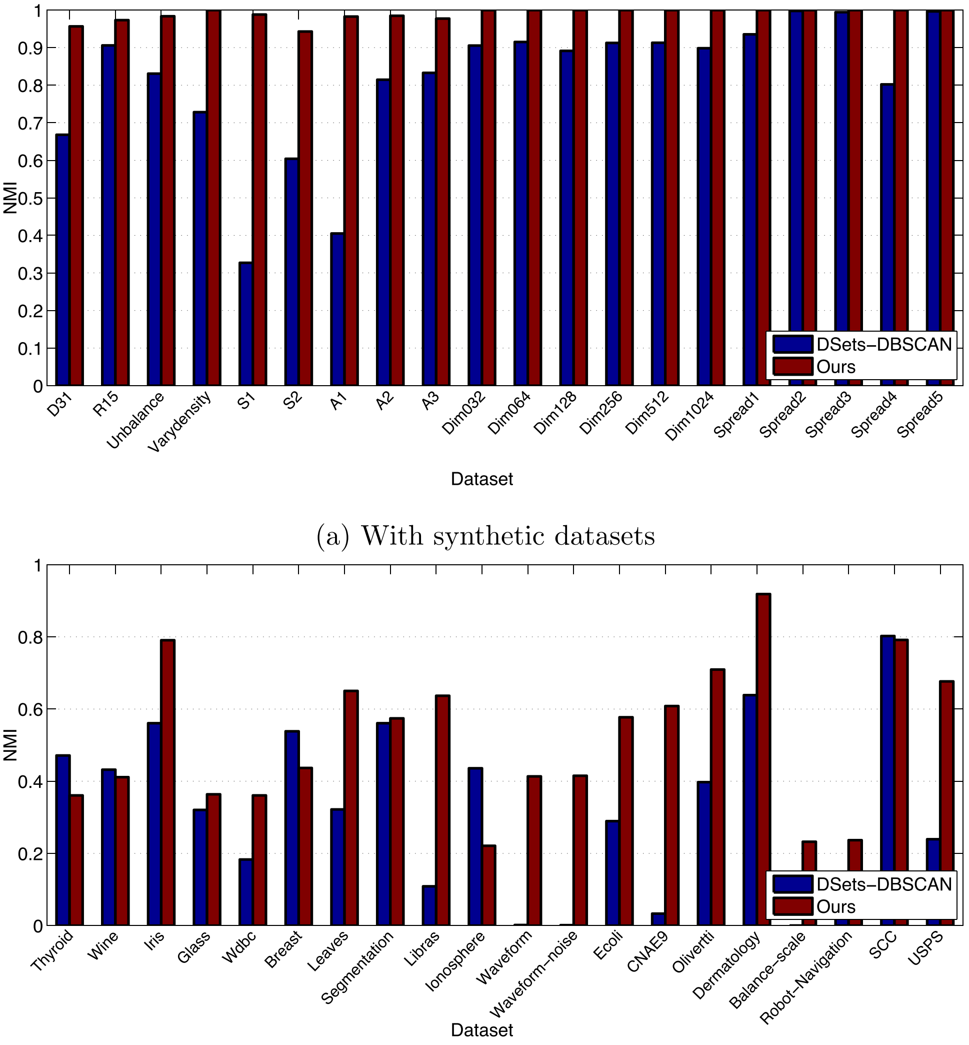
(3) The DBSCAN algorithm involves two parameters $MinPts$ and Eps, which can be determined by some heuristics [34]. We compare the heuristics proposed in [2] and [35] and select the one in [2], which performs better in our experiments. Specifically, we fix $MinPts = 4$ and determine Eps based on the 4-dist graph (see details in [2]).

(4) The DenMune algorithm has only one parameter K, which is the number of mutual nearest neighbors. Following [30], we test K from 1 to 200 for each dataset and report the best result, which is denoted by DenMune-best in Table 3 to Table 6. We also select a fixed K ($K = 37$) for all the datasets, which generates the best average result in the range from 1 to 200. The results with the fixed K are denoted by DenMune-fixed.

(5) The X-means algorithm is parameter-free, and no input is needed.

In experiments, the results of the k-means, EM, X-means, NCut and CBKM algorithms are reported as the average of 100 runs, and the results of SPRG are the average of 10 runs. The comparisons of all the algorithms are shown in Table 3 to Table 6.

The comparison results in Tables 3 to 6 are discussed as follows. First, our algorithm generates very good results on the 20 synthetic datasets. With both NMI and accuracy measures, our algo-



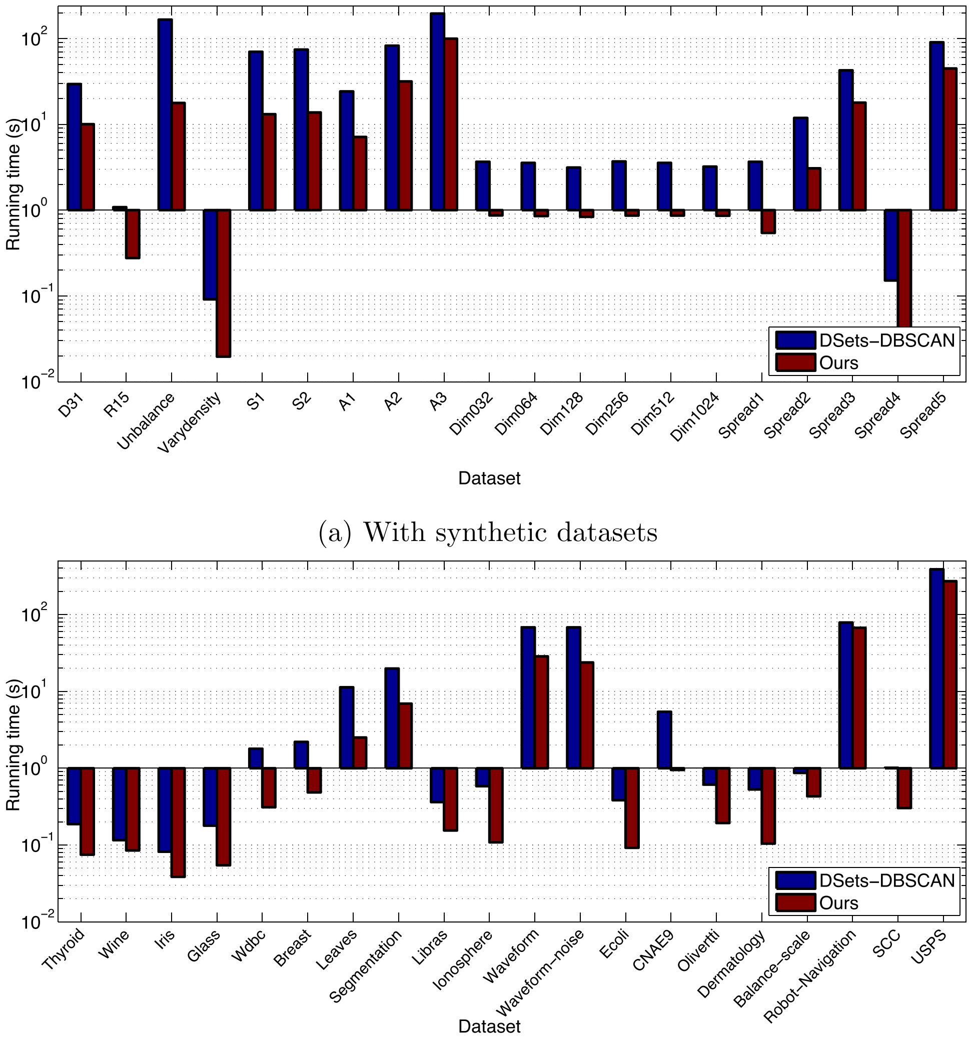
(a) With synthetic datasets

(b) with real datasets

Fig. 9. Clustering results comparison between DSet-DBSCAN and our algorithm.

Table 3
Clustering results (NMI) on 20 synthetic datasets.

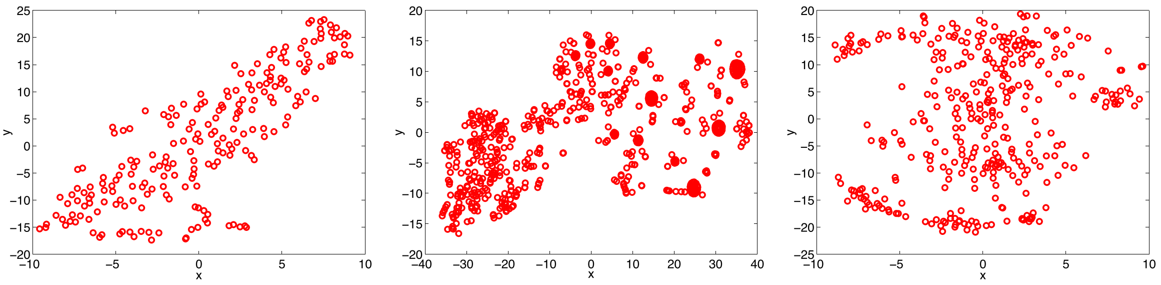
	k-means -gt	k-means -LL	k-means -egap	EM -gt	EM -LL	EM -egap	SPRG -gt	SPRG -egap	X-means	NCut -gt	DBSCAN	CBKM -gt	DenMune -best	DenMune -fixed	Ours
D31	0.92	0.92	0.67	0.93	0.93	0.67	0.90	0.67	0	0.97	0.79	0.94	0.96	0.95	0.96
R15	0.94	0.94	0.84	0.94	0.94	0.85	0.98	0.88	0	0.99	0.90	0.99	0.99	0.77	0.97
Unbalance	0.79	0.78	0.79	0.78	0.78	0.79	0.88	0.84	0.96	1.00	0	0.80	0.99	0.98	0.98
Varydensity	0.80	0.76	0.76	1.00	0.76	0.76	0.97	0.70	0.74	0.88	0.66	0.69	1.00	1.00	1.00
S1	0.94	0.94	0.60	0.94	0.95	0.60	0.95	0.55	0	0.99	0	0.95	0.99	0.97	0.99
S2	0.90	0.90	0.58	0.92	0.91	0.55	0.91	0.58	0	0.95	0	0.92	0.95	0.89	0.94
A1	0.93	0.92	0.72	0.94	0.94	0.73	0.91	0.71	0.98	0.99	0.86	0.95	0.98	0.97	0.98
A2	0.95	0.94	0.74	0.96	0.95	0.74	0.92	0.72	0.99	0.99	0.87	0.97	0.99	0.98	0.98
A3	0.95	0.95	0.77	0.95	0.95	0.77	0.95	0.76	0	0.99	0.89	0.97	0.99	0.99	0.98
Dim032	0.92	0.92	0.93	0.93	0.95	0.91	1.00	1.00	0.96	1.00	0.81	1.00	1.00	1.00	1.00
Dim064	0.93	0.93	0.92	0.92	0.94	0.94	1.00	1.00	0.97	1.00	0.82	1.00	0.99	0.99	1.00
Dim128	0.90	0.92	0.93	0.92	0.91	0.93	1.00	1.00	0.97	1.00	0.78	1.00	0.99	0.99	1.00
Dim256	0.92	0.93	0.92	0.92	0.92	0.93	1.00	1.00	0.97	1.00	0.77	1.00	1.00	1.00	1.00
Dim512	0.90	0.91	0.91	0.91	0.91	0.92	1.00	1.00	0.97	1.00	0.82	1.00	0.99	0.99	1.00
Dim1024	0.91	0.91	0.92	0.90	0.92	0.92	1.00	1.00	0.97	1.00	0.82	1.00	1.00	1.00	1.00
Spread1	0.91	0.91	0.67	0.91	0.90	0.65	0.99	0.68	0	1.00	0.91	1.00	1.00	1.00	1.00
Spread2	0.91	0.91	0.58	0.90	0.91	0.56	1.00	0.59	1.00	1.00	0.95	1.00	0.99	0.99	1.00
Spread3	0.91	0.92	0.91	0.93	0.93	0.93	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00
Spread4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.95	1.00	1.00	1.00	1.00
Spread5	0.94	0.93	0.93	0.92	0.92	0.93	1.00	1.00	0.98	1.00	0.99	1.00	0.99	0.99	1.00
Mean	0.91	0.91	0.80	0.93	0.92	0.80	0.97	0.83	0.67	0.99	0.73	0.96	0.99	0.97	0.99



(a) With synthetic datasets

(b) With real datasets

Fig. 10. Running time comparison between DSet-DBSCAN and our algorithm.



(a) Thyroid

(b) Breast

(c) Ionosphere

Fig. 11. Demonstration of three datasets transformed to 2-D using t-sne.

rithm performs comparably to DenMune-best and NCut-gt on the synthetic datasets, and outperforms all the other algorithms. This confirms that our algorithm does work for clusters of Gaussian distribution. Meanwhile, most of other algorithms also perform well on these synthetic datasets. Correspondingly, the number of clusters estimated by LL, eigen-gap and other methods are relatively accurate, as shown in Table 7. Note that in Table 7 the number of clusters estimated by X-means is the mean of 100 runs. Second, in terms of the average result on the 20 real datasets, our algorithm performs inferior to k-means-gt, EM-gt, SPRG-gt and DenMune-best, and better than all the other algorithms. The k-means-gt, EM-gt and SPRG-gt algorithms are fed with the ground truth number

Table 4
Clustering results (NMI) on 20 real datasets.

	k-means -gt	k-means -LL	k-means -egap	EM -gt	EM -LL	EM -egap	SPRG -gt	SPRG -egap	X-means	NCut -gt	DBSCAN	CBKM -gt	DenMune -best	DenMune -fixed	Ours
Thyroid	0.60	0	0.43	0.79	0	0.50	0.78	0.64	0.42	0.47	0.45	0.28	0.54	0	0.36
Wine	0.83	0.50	0.83	0.88	0.60	0.89	0.90	0.87	0.35	0.20	0.38	0.43	0.49	0.47	0.41
Iris	0.72	0.76	0.68	0.82	0.76	0.84	0.72	0.69	0.66	0.79	0.63	0.76	0.80	0.76	0.79
Glass	0.33	0.36	0.32	0.31	0.33	0.36	0.42	0.20	0.47	0.24	0.40	0.42	0.44	0.32	0.36
Wdbc	0.62	0.50	0.62	0.67	0.57	0.67	0.58	0.58	0.33	0.61	0.31	0.47	0.52	0.42	0.36
Breast	0.74	0.56	0.74	0.55	0.52	0.55	0.79	0.78	0.41	0.06	0.62	0.74	0.78	0.61	0.44
Leaves	0.72	0.28	0.49	0.72	0.31	0.51	0.72	0.50	0.63	0.56	0.27	0.67	0.71	0.54	0.65
Segmentation	0.60	0.56	0.37	0.60	0.54	0.42	0.69	0.42	0.55	0.64	0.43	0.11	0.68	0.61	0.57
Libras	0.59	0	0.30	0.58	0	0.32	0.62	0.32	0.67	0.40	0.27	0.57	0.67	0.37	0.64
Ionosphere	0.13	0	0.13	0.31	0	0.29	0.09	0.07	0.29	0.26	0.56	0.13	0.27	0.06	0.22
Waveform	0.36	0	0.36	0.51	0	0.51	0.36	0.37	0.47	0.06	0.00	0.36	0.36	0	0.41
Waveform-noise	0.36	0.46	0.37	0.46	0.37	0.46	0.36	0.36	0.49	0.00	0.00	0.36	0.32	0	0.42
Ecoli	0.59	0.61	0.62	0.60	0.53	0.48	0.55	0.59	0.60	0.54	0.23	0.62	0.69	0.66	0.58
CNAE9	0.38	0	0.15	0.45	0	0.17	0.53	0.21	0.52	0.21	0.02	0.22	0.59	0.57	0.61
Olivertti	0.83	0	0.78	0.83	0	0.80	0.88	0.81	0.87	0.46	0.38	0.83	0.88	0.29	0.71
Dermatology	0.85	0.80	0.76	0.80	0.80	0.72	0.87	0.78	0.67	0.29	0.47	0.80	0.94	0.94	0.92
Balance-scale	0.13	0	0.18	0.12	0	0.16	0.09	0.17	0	0.12	0	0.16	0.30	0	0.23
Robot-Navigation	0.11	0	0.10	0.08	0	0.08	0.13	0.12	0.20	0.03	0.18	0.11	0.29	0.24	0.24
SCC	0.74	0.70	0.60	0.74	0.71	0.46	0.74	0.60	0.71	0.59	0.79	0.74	0.86	0.85	0.79
USPS	0.45	0	0.22	0.41	0	0.20	0.51	0.24	0.57	0.21	0.32	0.45	0.81	0.78	0.68
Mean	0.53	0.30	0.45	0.56	0.30	0.47	0.57	0.47	0.49	0.33	0.34	0.46	0.60	0.42	0.52

Table 5
Clustering results (Accuracy) on 20 synthetic datasets.

	k-means -gt	k-means -LL	k-means -egap	EM -gt	EM -LL	EM -egap	SPRG -gt	SPRG -egap	X-means	NCut -gt	DBSCAN	CBKM -gt	DenMune -best	DenMune -fixed	Ours
D31	0.99	0.99	0.85	0.99	0.99	0.84	0.99	0.84	0.03	0.99	0.90	0.99	0.99	0.99	0.99
R15	0.98	0.98	0.92	0.98	0.98	0.91	0.99	0.93	0.07	0.99	0.98	0.99	0.99	0.75	0.99
Unbalance	0.88	0.87	0.89	0.86	0.87	0.87	0.94	0.90	0.99	1.00	0.29	0.77	0.99	0.99	0.99
Varydensity	0.88	0.78	0.78	1.00	0.78	0.78	0.99	0.77	0.85	0.96	0.75	0.77	1.00	1.00	1.00
S1	0.98	0.98	0.72	0.98	0.98	0.72	0.99	0.58	0.07	0.99	0.07	0.99	0.99	0.99	0.99
S2	0.98	0.98	0.72	0.98	0.98	0.66	0.98	0.70	0.07	0.99	0.07	0.98	0.99	0.97	0.99
A1	0.98	0.98	0.86	0.98	0.98	0.85	0.98	0.87	0.99	0.99	0.97	0.99	0.99	0.99	0.99
A2	0.99	0.99	0.91	0.99	0.99	0.90	0.99	0.88	0.99	0.99	0.98	0.99	0.99	0.99	0.99
A3	0.99	0.99	0.94	0.99	0.99	0.93	0.99	0.92	0.02	0.99	0.98	0.99	0.99	0.99	0.99
Dim032	0.96	0.96	0.96	0.96	0.98	0.95	1.00	1.00	0.99	1.00	0.94	1.00	1.00	1.00	1.00
Dim064	0.97	0.96	0.96	0.96	0.97	0.97	1.00	1.00	0.99	1.00	0.94	1.00	0.99	0.99	1.00
Dim128	0.94	0.95	0.96	0.96	0.95	0.96	1.00	1.00	0.99	1.00	0.92	1.00	0.99	0.99	1.00
Dim256	0.95	0.96	0.95	0.95	0.96	0.96	1.00	1.00	0.99	1.00	0.92	1.00	1.00	1.00	1.00
Dim512	0.93	0.95	0.95	0.95	0.95	0.95	1.00	1.00	0.99	1.00	0.95	1.00	0.99	0.99	1.00
Dim1024	0.95	0.94	0.95	0.94	0.95	0.95	1.00	1.00	0.99	1.00	0.95	1.00	1.00	1.00	1.00
Spread1	0.95	0.94	0.72	0.95	0.94	0.69	0.99	0.76	0.10	1.00	0.98	1.00	1.00	1.00	1.00
Spread2	0.96	0.96	0.65	0.96	0.96	0.63	1.00	0.68	1.00	1.00	0.99	1.00	0.99	0.99	1.00
Spread3	0.97	0.98	0.97	0.98	0.98	0.98	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00
Spread4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00
Spread5	0.99	0.98	0.99	0.98	0.98	0.98	1.00	1.00	0.99	1.00	0.99	1.00	1.00	1.00	1.00
Mean	0.96	0.96	0.88	0.97	0.96	0.87	0.99	0.89	0.71	0.99	0.83	0.97	0.99	0.98	0.99

of clusters. If we use the LL and eigen-gap methods to estimate the number of clusters and feed the k-means, EM and SPRG algorithms, we find that k-means-LL, k-mean-egap, EM-LL, EM-egap and SPRG-egap performs worse than our algorithm. Noticing that DBSCAN also performs unsatisfactorily on real datasets, we find that it is still quite difficult to estimate the parameters of clustering algorithms accurately for real datasets. The result of DenMune-best on each dataset are selected from 200 options of the parameter K. If we use a fixed K which generates the best average result on all datasets, we find that DenMune-fixed is outperformed by our algorithm. Compared with the synthetic datasets, Table 7 shows that the number of clusters estimated on the real datasets are less accurate, and the clustering results are therefore less satisfactory on the real datasets. In conclusion, k-means-LL, k-means-egap, EM-LL, EM-egap, SPRG-egap and DenMune-fixed can be regarded as parameter-free versions of the k-means, EM, SPRG and DenMune algorithms, and they perform inferior to our algorithm in terms of average result.

Experimental results on 20 synthetic and 20 real datasets and comparisons with some other algorithms show that our algorithm with two fixed parameters $\zeta = 0.1$ and $\epsilon = 3.6$ can be regarded as a parameter-free algorithm. However, there are still some problems to be solved. First, the parameter ϵ has a large influence on the clustering results of the Unbalance dataset. This indicates that further works are needed to deal with the clustering of unbalanced datasets. Second, while our algorithm generate (near-)best results on most of the datasets at $\zeta = 0.1$ and $\epsilon = 3.6$, the results on some high-dimension datasets are not satisfactory in comparison with other algorithms. The examples include USPS (256-D), Olivertti (10304-D) and Wdbc (30-D). Although we have used t-sne to reduce these datasets to 2-D, it is possible that the large dimension reduction results in information loss and performance

Table 6
Clustering results (Accuracy) on 20 real datasets.

	k-means	k-means	k-means	EM	EM	EM	SPRG	SPRG	X-means	NCut	DBSCAN	CBKM	DenMune	DenMune	Ours
	-gt	-LL	-egap	-gt	-LL	-egap	-gt	-egap		-gt		-gt	-best	-fixed	
Thyroid	0.82	0.53	0.69	0.93	0.53	0.72	0.92	0.79	0.50	0.74	0.76	0.62	0.81	0.53	0.59
Wine	0.93	0.68	0.93	0.96	0.71	0.96	0.96	0.95	0.67	0.57	0.61	0.70	0.68	0.67	0.72
Iris	0.86	0.78	0.81	0.88	0.78	0.90	0.86	0.85	0.78	0.90	0.77	0.88	0.89	0.78	0.88
Glass	0.68	0.71	0.55	0.67	0.70	0.60	0.73	0.56	0.75	0.65	0.59	0.60	0.76	0.56	0.72
Wdbc	0.87	0.80	0.87	0.89	0.83	0.89	0.85	0.85	0.51	0.87	0.66	0.75	0.78	0.67	0.55
Breast	0.92	0.75	0.92	0.78	0.71	0.78	0.94	0.94	0.54	0.53	0.89	0.92	0.94	0.83	0.53
Leaves	0.98	0.41	0.80	0.98	0.48	0.80	0.99	0.81	0.95	0.84	0.24	0.97	0.98	0.76	0.95
Segmentation	0.85	0.64	0.51	0.84	0.71	0.58	0.89	0.61	0.87	0.88	0.41	0.21	0.90	0.89	0.87
Libras	0.91	0.07	0.73	0.91	0.07	0.74	0.92	0.75	0.94	0.74	0.33	0.90	0.91	0.42	0.91
Ionosphere	0.59	0.54	0.59	0.70	0.54	0.68	0.56	0.54	0.52	0.64	0.80	0.59	0.58	0.50	0.54
Waveform	0.67	0.33	0.67	0.81	0.33	0.81	0.67	0.67	0.72	0.54	0.34	0.67	0.70	0.33	0.70
Waveform-noise	0.67	0.72	0.67	0.79	0.73	0.79	0.67	0.67	0.72	0.56	0.34	0.67	0.67	0.33	0.70
Ecoli	0.80	0.83	0.84	0.85	0.79	0.75	0.80	0.80	0.79	0.78	0.33	0.85	0.88	0.87	0.78
CNAE9	0.66	0.11	0.23	0.75	0.11	0.27	0.88	0.51	0.88	0.59	0.25	0.68	0.90	0.85	0.91
Olivertti	0.98	0.03	0.96	0.98	0.03	0.97	0.98	0.97	0.98	0.66	0.29	0.98	0.98	0.33	0.91
Dermatology	0.91	0.86	0.81	0.89	0.90	0.82	0.91	0.82	0.83	0.56	0.54	0.85	0.96	0.96	0.96
Balance-scale	0.59	0.43	0.61	0.58	0.43	0.60	0.57	0.62	0.43	0.59	0.43	0.61	0.57	0.43	0.60
Robot-Navigation	0.60	0.34	0.57	0.59	0.34	0.57	0.59	0.54	0.66	0.50	0.60	0.60	0.65	0.66	0.66
SCC	0.88	0.88	0.61	0.87	0.88	0.57	0.86	0.61	0.87	0.79	0.87	0.86	0.89	0.89	0.91
USPS	0.87	0.10	0.67	0.84	0.10	0.67	0.88	0.68	0.91	0.23	0.46	0.87	0.93	0.95	0.92
Mean	0.80	0.53	0.70	0.82	0.53	0.72	0.82	0.73	0.74	0.66	0.52	0.74	0.82	0.66	0.77

Table 7
Numbers of clusters estimated with different algorithms.

Algorithm	D31	R15	Unbalance	Varyden.	S1	S2	A1	A2	A3	Dim032
LL	31	15	8	2	15	15	20	35	50	16
eigen-gap	6	10	8	2	3	3	6	9	13	16
X-means	1	1	4	7.77	1	1	22.32	36.09	1	43.67
DBSCAN	16	16	1	3	1	1	20	35	50	17
DSets-DBSCAN	9	13	22	5	3	9	4	15	20	26
DenMune-best	31	15	8	3	15	15	19	34	49	16
DenMune-fixed	30	8	8	3	16	14	19	34	49	16
Ours	32	15	14	3	16	16	21	37	53	16
Ground truth	31	15	8	3	15	15	20	35	50	16
Algorithm	Dim064	Dim128	Dim256	Dim512	Dim1024	Spread1	Spread2	Spread3	Spread4	Spread5
LL	16	16	16	16	16	10	20	35	2	50
eigen-gap	16	16	16	16	16	3	3	35	2	50
X-means	44.77	46.65	45.58	46.93	45.85	1	20	35	2	45.36
DBSCAN	17	17	17	17	17	14	21	36	3	51
DSets-DBSCAN	27	23	30	29	24	17	21	34	4	49
DenMune-best	16	16	16	16	16	10	20	35	2	50
DenMune-fixed	16	16	16	16	16	10	20	35	2	50
Ours	16	16	16	16	16	10	20	35	2	50
Ground truth	16	16	16	16	16	10	20	35	2	50
Algorithm	Thyroid	Wine	Iris	Glass	Wdbc	Breast	Leaves	Segment.	Libras	Ionosphere
LL	1	2	2	8	3	4	2	3	1	1
eigen-gap	2	3	3	2	2	2	6	2	4	2
X-means	43.68	38.65	9.75	48.17	44.89	39.99	31.75	30.60	45.04	43.54
DBSCAN	2	4	3	4	4	10	8	4	4	2
DSets-DBSCAN	5	4	5	6	5	5	7	13	3	5
DenMune-best	4	2	3	9	2	2	125	7	14	7
DenMune-fixed	1	2	2	2	5	3	8	16	2	2
Ours	6	5	3	6	11	13	24	31	12	7
Ground truth	3	3	3	6	2	2	100	7	15	2
Algorithm	Waveform	Waveform-n.	Ecoli	CNAE9	Olivertti	Dermatology	Balance-scale	Robot-navi.	SCC	USPS
LL	1	7	5	1	1	5	1	1	23	1
eigen-gap	3	3	5	2	29	3	5	3	2	3
X-means	8	8	13.77	41.90	43.18	36.83	1	32	31.80	32
DBSCAN	2	2	3	2	7	3	1	101	7	48
DSets-DBSCAN	3	3	5	5	4	5	1	3	5	4
DenMune-best	35	201	3	27	50	4	286	888	4	8
DenMune-fixed	1	1	3	7	2	4	1	60	4	13
Ours	25	20	8	17	11	6	15	62	12	62
Ground truth	3	3	8	9	40	6	3	4	6	10

degradation. Finally, our algorithm works well for near-spherical clusters of (appropriate) Gaussian distributions, and may not work with special-shaped datasets, e.g., Spiral. This problem also calls for further works.

5. Conclusion

In this paper we present a clustering algorithm to deal with real-world data. In order to avoid the difficulty in determining the number of clusters, our algorithm generates clusters sequentially, and each cluster is obtained by expanding an initial cluster. In extracting the initial cluster with the dominant set algorithm, we find out the correlation between the clustering result and the standard deviation of pairwise similarity values. As a result, we present an adaptive method to determine the scaling parameter. In expanding the initial cluster, we sort unclustered data based on their distances to the initial cluster, so that border data points of small densities can be included while those of other clusters are excluded. The cluster expansion step involves a parameter in density estimation, and we show that this parameter has little influence on clustering results in a large range and therefore can be fixed for different datasets. By carefully making use of the merits of the dominant set and density peak algorithms and overcoming their deficiencies, our algorithm does not require to specify parameters and works well with clusters of Gaussian distribution. As in many cases clusters in the real-world data follow the Gaussian distribution approximately, our algorithm is a promising approach in dealing with real-world data clustering. In experiments with 20 synthetic datasets of Gaussian distribution and 20 real datasets, our algorithm performs better than the parameter-free DSets-DBSCAN algorithm in overall clustering accuracy and computation efficiency. It also performs well in comparison with some commonly used algorithms with parameter tuning. In future works, we plan to explore new methods to deal with unbalanced datasets and high-dimensional datasets.

Declaration of Competing Interest

None.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant No. 62176057.

References

[1] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern. Anal. Mach. Intell. 22 (8) (2000) 167–172.
[2] M. Ester, H.P. Kriegel, J. Sander, X.W. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: International Conference on Knowledge Discovery and Data Mining, 1996, pp. 226–231.
[3] D. Comaniciu, M. Peter, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (5) (2002) 603–619.
[4] X. Chen, W. Hong, F. Nie, J.Z. Huang, L. Shen, Enhanced balanced min cut, Int. J. Comput. Vis. 128 (7) (2020) 1982–1995.
[5] L. Bai, X. Cheng, J. Liang, H. Shen, Y. Guo, Fast density clustering strategies based on the k-means algorithm, Pattern Recognit. 71 (2017) 375–386.
[6] J. Yu, C. Chaomurilige, M.-S. Yang, On convergence and parameter selection of the EM and DA-EM algorithms for gaussian mixtures, Pattern Recognit. 77 (2018) 188–203.
[7] C. Zhong, D. Miao, P. Fránti, Minimum spanning tree based split-and-merge: a hierarchical clustering method, Inf. Sci. (Ny) 181 (16) (2011) 3397–3410.
[8] A. Ng, M. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, in: Advances in Neural Information Processing Systems, 2002, pp. 849–856.
[9] J.F. Brendan, D. Delbert, Clustering by passing messages between data points, Science 315 (2007) 972–976.
[10] C. Zhang, H. Fu, Q. Hu, X. Cao, Y. Xie, D. Tao, D. Xu, Generalized latent multi--view subspace clustering, IEEE Trans. Pattern Anal. Mach. Intell. 42 (1) (2020) 86–99.
[11] D. Huang, C.-D. Wang, J.-S. Wu, J.-H. Lai, C.-K. Kwoh, Ultra-scalable spectral clustering and ensemble clustering, IEEE Trans. Knowl. Data Eng. 32 (6) (2020) 1212–1226.
[12] L. Wang, J. Huang, M. Yin, R. Cai, Z. Hao, Block diagonal representation learning for robust subspace clustering, Inf. Sci. (Ny) 526 (2020) 54–67.
[13] Z. Yu, Z. Zhang, W. Cao, C.L.P. Chen, C. Liu, H.-S. Wong, Gan-based enhanced deep subspace clustering networks, IEEE Trans. Knowl. Data Eng. (2020) 1.
[14] M. Pavan, M. Pelillo, Dominant sets and pairwise clustering, IEEE Trans. Pattern Anal. Mach. Intell. 29 (1) (2007) 167–172.
[15] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (2014) 1492–1496.
[16] T. Qiu, C. Li, Y. Li, D-NND: A hierarchical density clustering method via nearest neighbor descent, in: International Conference on Pattern Recognition, 2018, pp. 1414–1419.
[17] A. Saha, S. Das, Axiomatic generalization of the membership degree weighting function for fuzzy c means clustering: theoretical development and convergence analysis, Inf. Sci. (Ny) 408 (2017) 129–145.
[18] B. Mukhoty, R. Gupta, K. Lakshmanan, M. Kumar, A parameter free affinity based clustering, Appl. Intell. 50 (12) (2020) 4543–6445569.
[19] A. Gupta, S. Datta, S. Das, Fast automatic estimation of the number of clusters from the minimum inter-center distance for k-means clustering, Pattern Recognit. Lett. 116 (2018) 72–79.
[20] R. Heckel, H. Bölcskei, Robust subspace clustering via thresholding, IEEE Trans. Inf. Theory 61 (11) (2015) 6320–6342.
[21] A. Lyon, Why are normal distributions normal? Brit. J. Philos. Sci. 65 (3) (2014) 621–649.
[22] J. Hou, H. Gao, X. Li, DSets-DBSCAN: A parameter-free clustering algorithm, IEEE Trans. Image Process. 25 (7) (2016) 3182–3193.
[23] S.R. Bulo, M. Pelillo, I.M. Bomze, Graph-based quadratic optimization: a fast evolutionary approach, Comput. Vis. Image Understand. 115 (7) (2011) 984–995.
[24] S. Vascon, S.R. Bulò, V. Murino, M. Pelillo, Dslib: an open source library for the dominant set clustering method, CoRR abs/2010.07906 (2020).
[25] L.T. Alemu, M. Shah, M. Pelillo, Deep constrained dominant sets for person re-identification, in: IEEE International Conference on Computer Vision, 2019, pp. 9855–9864.
[26] E. Zemene, Y.T. Tesfaye, H. Idrees, A. Prati, M. Pelillo, M. Shah, Large-scale image geo-localization using dominant sets, IEEE Trans. Pattern Anal. Mach. Intell. 41 (1) (2019) 148–161.
[27] X. Tao, T. Cui, A. Plaza, P. Ren, Simultaneously counting and extracting endmembers in a hyperspectral image based on divergent subsets, IEEE Trans. Geosci. Remote Sens. 58 (12) (2020) 8952–8966.
[28] T. Acharya, A.K. Ray, Image processing: principles and applications, Wiley-Interscience, 2005.
[29] J. Hou, A. Zhang, N. Qi, Density peak clustering based on relative density relationship, Pattern Recognit. 108 (2020) 1–16.
[30] M. Abbas, A. El-Zoghabi, A. Shoukry, Denmune: density peak based clustering using mutual nearest neighbors, Pattern Recognit. 109 (2021).
[31] D. Pelleg, A. Moore, X-means: Extending k-means with efficient estimation of the number of clusters, in: International Conference on Machine Learning, volume 1, 2000, pp. 727–734.
[32] X. Zhu, C.C. Loy, S. Gong, Constructing robust affinity graphs for spectral clustering, in: IEEE International Conference on Computer Vision and Pattern Recognition, 2014, pp. 1450–1457.
[33] P. Fränti, S. Sieranoja, How much can k-means be improved by using better initialization and repeats, Pattern Recognit. 93 (2019) 95–112.
[34] E. Schubert, J. Sander, M. Ester, H.P. Kriegel, X. Xu, Dbscan revisited, revisited: why and how you should (still) use dbscan, ACM Trans. Database Syst. 42 (3) (2017) 1–21.
[35] J. Sander, M. Ester, H.-P. Kriegel, X. Xu, Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications, Data Min. Knowl. Discov. 2 (2) (1998) 169–194.

Jian Hou is a professor with the School of Computer Science and Technology, Dongguan University of Technology, Dongguan, China. His research interests include pattern recognition, machine learning, computer vision and image processing.

Huaqiang Yuan is a professor with the School of Computer Science and Technology, Dongguan University of Technology, Dongguan, China. His research interests include intelligent computation and virtual reality.

Marcello Pelillo is a Professor of Computer Science at the University of Venice, Italy, where he directs the European Centre for Living Technology and leads the Computer Vision and Pattern Recognition group, which he founded in 1995. He held visiting research positions at Yale University (USA), McGill University (Canada), the University of Vienna (Austria), York University (UK), the University College London (UK), and the National ICT Australia (NICTA) (Australia). He serves (or has served) on the editorial boards of IEEE Transactions on Pattern Analysis and Machine Intelligence, IET Computer Vision, Pattern Recognition, Brain Informatics, and is on the advisory board of the International Journal of Machine Learning and Cybernetics. He has initiated several conferences series as Program Chair (EMMCVPR, IWCV, SIMBAD) and will serve as a General Chair for ICCV 2017. He is (or has been) scientific coordinator of several research projects, including SIMBAD, a highly successful EU-FP7 project devoted to similarity-based pattern analysis and recognition. He is a Fellow of the IEEE and a Fellow of the IAPR, and has been appointed IEEE Distinguished Lecturer (2016-2017 term).