Pattern Recognition 98 (2020) 107063

Contents lists available at ScienceDirect

Pattern Recognition

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

Clustering by connection center evolution

Xiurui Geng[a,b,c,*], Hairong Tang[a,b,c]

[a] Institute of Electronics, Chinese Academy of Sciences, Beijing 100191, China
[b] Key Laboratory of Technology in Geo-spatial Information Processing and Application System, Chinese Academy of Sciences, Beijing 100191, China
[c] University of Chinese Academy of Sciences, Beijing 101407, China

ARTICLE INFO

Article history:
Received 14 February 2019
Revised 5 September 2019
Accepted 22 September 2019
Available online 24 September 2019

Index Terms:
Clustering center
Clustering
Connected graph
Connectivity

ABSTRACT

The determination of clustering centers generally depends on the observation scale that we use to analyze the data to be clustered. An inappropriate scale usually leads to unreasonable clustering centers and thus unreasonable results. In this study, we first consider the similarity of elements in the data as the connectivity of vertices in an undirected graph, then present the concept of connection center and regard it as the clustering center of the data. Based on this definition, the determination of clustering centers and the assignment of class become very simple, natural and effective. One more crucial finding is that the clustering centers of different scales can be obtained easily by different powers of a similarity matrix, and the change of power from small to large leads to the dynamic evolution of clustering centers from local (microscopic) to global (macroscopic). Further, in this process of evolution, the number of clusters changes discontinuously, which means that the presented method can automatically skip the unreasonable number of clusters, suggest appropriate observation scales and provide corresponding clustering results.

1. Introduction

Clustering analysis has been widely used in pattern recognition, feature extraction, data compression and image segmentation. The determination of both the clustering centers and the number of clusters are two key problems in clustering analysis.

There are a large number of clustering approaches which have been extensively explored. The popular k-means method [1–7] iteratively calculates clustering centers by minimizing the within-cluster sum of squares. However, the number of clusters must be specified in advance. Moreover, it could be trapped in poor local optima. The distribution-based Gaussian mixture model (GMM) [8–10] finds clustering centers through maximum likelihood estimation, but a preset of the number of clusters is also required. There are many density-based clustering methods [11–13], among which Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [11] is a typical method. DBSCAN searches the density reachable areas in feature space and clusters data points by density reachability. However, the number of clusters is sensitive to the size of a neighborhood and the threshold of the density. Mean-shift [14–17] is a mode-seeking and also density-based method. It gradually shifts each data point to the average of the data points in its neighborhood. Different sizes of neighborhood will result in different clustering centers and numbers of clusters. Fast Search and Find of Density Peaks (FSFDP) [18] is a new clustering approach in which clustering centers are characterized by a higher density than their neighbors and by a relatively large distance from those points with higher densities. It suffers from the same problem as the above density-based methods in the determination of the number of clusters. In order to avoid manual intervention, the latest version of FSFDP, which automatically determines thresholds, has recently been published in [19]. Graph-based methods [20–24] consider each point to be clustered as the vertex of a graph. Among this family, the Minimum Spanning Tree (MST) method [20] is popular. MST determines the minimum spanning tree of a graph and then removes the edges that are extraordinarily large compared with neighboring edges. Therefore, its number of clusters is easily influenced by the threshold used for removing the inconsistent edges. In addition, MST is extremely sensitive to outliers as well as the non-uniformity of density. The spectral clustering (SC) approaches [25–31], a recently popular clustering family, study the properties of a graph via the eigenvalues and eigenvectors of the associated graph matrices such as the adjacency matrix and the Laplacian matrix. The most widely used approaches in this family include the work in [26,27]. Spectral clustering algorithms can also be interpreted from the angle of random walks in [31] where the similarity matrix is interpreted as the Markov walk's transition matrix, and the properties of its steady-state can be studied by the

* Corresponding author at: Institute of Electronics, Chinese Academy of Sciences, Beijing 100191, China.
E-mail address: gengxr@sina.com.cn (X. Geng).

https://doi.org/10.1016/j.patcog.2019.107063
0031-3203/

eigenvalues and eigenvectors of the transition matrix. Spectral clustering algorithms have achieved great success in a number of applications and have been significantly developed [32–34], but the number of clusters still needs to be specified in advance. Path-based clustering algorithms [35,36], which have recently attracted some attention, use a path-based criterion to define the (dis)similarity between points to form a pairwise (dis)similarity matrix, and then use a pairwise clustering method to group the data. The number of clusters is also dependent on the selection of the (dis)similarity threshold and of the corresponding clustering method. Hierarchical clustering algorithms can produce a hierarchy of nested clusters either in agglomerative mode or in divisive mode, and the clusters of the lower level are nested in the clusters of the upper level [37]. However, if one clustering result at the earlier level is poor, this will heavily influence all the clustering results at the later levels.

Most of the clustering methods mentioned above try to understand the data points to be clustered from a local and static angle. Once a group of parameters is given, a single and fixed clustering result will be obtained. However, it is difficult or even impossible for us to automatically set the proper parameters for these clustering algorithms in many cases. In addition, for some practical datasets more than one reasonable clustering result may exist. In this case, if only one group of parameters is provided, most of the existing algorithms will fail to obtain all the desired results.

Here we present a new and original viewpoint of clustering, named Connection Center Evolution (CCE). We consider that the clustering centers are evolving and dynamic rather than fixed and static, and whether a point is a clustering center depends on the scale we use to observe the data sets. By mapping the data points to be clustered into the vertices of an undirected graph, CCE extends the concept of the number of walks in graph theory, and intelligently determines the clustering centers and class assignments through a simple comparison between entries of the similarity matrix. Furthermore, reasonable results at different scales can be easily obtained simply by using the different powers of the similarity matrix.

The remainder of this paper is organized as follows. In Section 2, several basic concepts used in CCE are first presented and then the clustering process of CCE and its computational complexity are given. The experimental results with some synthetic and real data are demonstrated in Section 3. In Section 4, the impact of unbalanced sample distribution in different categories on CCE is discussed. The conclusion is drawn in Section 5.

2. Clustering by connection center evolution (CCE)

2.1. Motivation and definition

The CCE algorithm is informed by the concept of the number of walks in graph theory [38–40]. The only difference is that, in CCE, each vertex in an undirected graph is self-connected. Let G be an undirected graph with a self-connected adjacency matrix $\mathbf{A}$, then the entry $a_{ij}^{(k)}$ of the kth power of $\mathbf{A}$ (denoted $\mathbf{A}^k$) is the number of walks of length k from the ith vertex ($\mathbf{v}_i$) to the jth vertex ($\mathbf{v}_j$). A simple example is shown in Fig. 1. For $\mathbf{A}^3$, $a_{22}^{(3)} = 10$ implies that the number of walks of length 3 from $\mathbf{v}_2$ to $\mathbf{v}_2$ is 10, and all the walks of length 3 from $\mathbf{v}_2$ to itself are listed in Table 1.

For real data to be clustered, we can only construct the pairwise similarity matrix of data rather than its adjacency matrix in general. Therefore, it motivates us to extend the above conclusion for the adjacency matrix to that for the similarity matrix. The concept of connectivity presented below is simply the extension of the concept of the number of walks. The difference is that the number of walks is calculated by the adjacency matrix of an undirected graph,

Table 1
All the walks of length 3 from $\mathbf{v}_2$ to $\mathbf{v}_2$.

	All the walks of length 3 from $\mathbf{v}_2$ to $\mathbf{v}_2$
1	$\mathbf{v}_2 \to \mathbf{v}_2 \to \mathbf{v}_2 \to \mathbf{v}_2$
2	$\mathbf{v}_2 \to \mathbf{v}_1 \to \mathbf{v}_2 \to \mathbf{v}_2$
3	$\mathbf{v}_2 \to \mathbf{v}_3 \to \mathbf{v}_2 \to \mathbf{v}_2$
4	$\mathbf{v}_2 \to \mathbf{v}_4 \to \mathbf{v}_2 \to \mathbf{v}_2$
5	$\mathbf{v}_2 \to \mathbf{v}_1 \to \mathbf{v}_1 \to \mathbf{v}_2$
6	$\mathbf{v}_2 \to \mathbf{v}_3 \to \mathbf{v}_3 \to \mathbf{v}_2$
7	$\mathbf{v}_2 \to \mathbf{v}_4 \to \mathbf{v}_4 \to \mathbf{v}_2$
8	$\mathbf{v}_2 \to \mathbf{v}_2 \to \mathbf{v}_1 \to \mathbf{v}_2$
9	$\mathbf{v}_2 \to \mathbf{v}_2 \to \mathbf{v}_3 \to \mathbf{v}_2$
10	$\mathbf{v}_2 \to \mathbf{v}_2 \to \mathbf{v}_4 \to \mathbf{v}_2$

whereas the connectivity here is defined by the similarity matrix constructed from a graph. Therefore, the original number of walks between two vertices in an undirected graph must be an integer, and the connectivity of them here, however, can be a real number.

Given a set of points $G = \{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$ in R^L we use the element s_{ij} of the pairwise similarity matrix $\mathbf{S}$ ($\mathbf{S} \in \mathrm{R}^{n \times n}$) to represent the *connectivity* between $\mathbf{v}_i$ and $\mathbf{v}_j$. Further, we also introduce the concept of *k-order connectivity* based on the kth power of the similarity matrix ($\mathbf{S}^k$). The strict definition is given by:

Definition 1 (k-order connectivity)**.** Given a pairwise similarity matrix $\mathbf{S}$. The entry $s_{ij}^{(k)}$ of the kth power ($\mathbf{S}^k$) of the similarity matrix is defined as the *k-order connectivity* between $\mathbf{v}_i$ and $\mathbf{v}_j$ (denoted $con^{(k)}(\mathbf{v}_i, \mathbf{v}_j)$). In particular, the diagonal entry $s_{ii}^{(k)}$ is defined as *k-order connectivity* of vertex $\mathbf{v}_i$ (denoted $con^{(k)}(\mathbf{v}_i, \mathbf{v}_i)$).

Obviously, $con^{(k)}(\mathbf{v}_i, \mathbf{v}_j)$ can be approximately regarded as the number of walks with length k from $\mathbf{v}_i$ to $\mathbf{v}_j$. It should be noted that the value of $con^{(k)}(\mathbf{v}_i, \mathbf{v}_j)$ could be a real number.

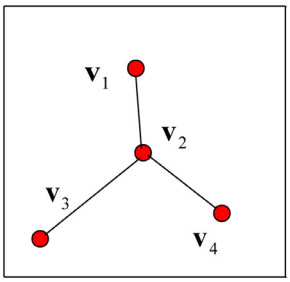
If each vertex in a graph can be considered as a station in a transportation network, it is taken for granted that we use the number of walks or connectivity of a vertex to represent how busy the station is. Obviously, the greater the number of walks or the connectivity, the more likely the station is to be a traffic center or a pivot. It is natural to take the vertice with the greatest number of walks as the center of the graph. As shown in Fig. 1, $a_{22}^{(k)}$ is the only *diagonally maximal element* (it will be defined in the following) in $\mathbf{A}^k$ when $k \geq 2$, and it can therefore be considered as the center of the graph. By inspection, it can be seen that it is indeed the pivot of the graph. The above fact motivates us to define the clustering center of data through the number of walks or connectivity of a vertex in the graph. We consider any vertex $\mathbf{v}_i$ to be a connection center of the graph, and thus a clustering center of the data, if the number of walks with length k from $\mathbf{v}_i$ to $\mathbf{v}_i$ is greater than that from $\mathbf{v}_i$ to $\mathbf{v}_j (j = 1, \ldots, n, \ j \neq i)$. Specifically, the definition of a clustering center can be given by

Definition 2 (clustering center)**.** If one vertex $\mathbf{v}_i$ satisfies the following inequality, it will be a connection center of the graph and is defined as a *k-order clustering center* of the data.

$$con^{(k)}(\mathbf{v}_i, \ \mathbf{v}_i) > con^{(k)}(\mathbf{v}_i, \ \mathbf{v}_j), j = 1, \cdots, n (j \neq i). \tag{1}$$

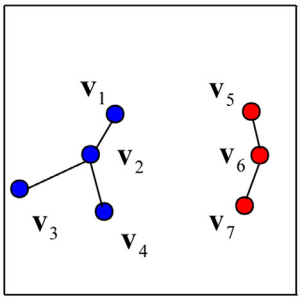
For the kth power ($\mathbf{S}^k$) of a similarity matrix $\mathbf{S}$ formed from a collection of vertices $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$, if a vertex satisfies inequality (1), we refer to $s_{ii}^{(k)}$ as a *diagonally maximal element* of $\mathbf{S}^k$. If all the vertices satisfy inequality (1), $\mathbf{S}^k$ is named the *diagonally maximal matrix*.

In many cases, there may exist more than one clustering center in a graph. For example, Fig. 2 demonstrates a simple two-centers situation. The graph consists of two connected subgraphs, but there is no connection between them and thus the adjacency matrix $\mathbf{A}$ of the graph is a two-block-diagonal matrix. After a



$$\mathbf{A}=\begin{bmatrix}1&1&0&0\\1&1&1&1\\0&1&1&0\\0&1&0&1\end{bmatrix}\quad \mathbf{A}^2=\begin{bmatrix}2&2&1&1\\2&4&2&2\\1&2&2&1\\1&2&1&2\end{bmatrix}\quad \mathbf{A}^3=\begin{bmatrix}4&6&3&3\\6&\boxed{10}&6&6\\3&6&4&3\\3&6&3&4\end{bmatrix}$$

Fig. 1. A simple example illustrating the number of walks of a self-connected undirected graph. The entry $a_{ij}^{(k)}$ of $\mathbf{A}^k$ represents the number of walks of length k from $\mathbf{v}_i$ to $\mathbf{v}_j$. The diagonal entries of the adjacency matrix ($\mathbf{A}$) are all 1 since each vertex is self-connected here.



$$\mathbf{A}=\begin{bmatrix}1&1&0&0&0&0&0\\1&1&1&1&0&0&0\\0&1&1&0&0&0&0\\0&1&0&1&0&0&0\\0&0&0&0&1&1&0\\0&0&0&0&1&1&1\\0&0&0&0&0&1&1\end{bmatrix}\quad \mathbf{A}^3=\begin{bmatrix}4&6&3&3&0&0&0\\6&\boxed{10}&6&6&0&0&0\\3&6&4&3&0&0&0\\3&6&3&4&0&0&0\\0&0&0&0&4&5&3\\0&0&0&0&5&\boxed{7}&5\\0&0&0&0&3&5&4\end{bmatrix}$$

Fig. 2. An example with two connected subgraphs to illustrate multi-center situation. $\mathbf{v}_2$ and $\mathbf{v}_6$ are naturally taken as the pivot of each subgraph since $a_{22}^{(3)}$ and $a_{66}^{(3)}$ are the diagonally maximal elements of $\mathbf{A}^3$.

simple calculation, $\mathbf{v}_2$ and $\mathbf{v}_6$ are found to be the centers of the two subgraphs from the third power matrix $\mathbf{A}^3$.

After acquiring all the clustering centers, we next classify the remaining points to the corresponding clustering centers by the connectivity of vertices. Intuitively, a vertex should be assigned to the clustering center with which it has the greatest connectivity. As we can see from Fig. 2, when k is greater than 2, $\mathbf{v}_2$ and $\mathbf{v}_6$ are always two stable centers. For the remaining vertices, such as $\mathbf{v}_1$, since $\mathbf{v}_1$ and $\mathbf{v}_6$ are not connected, that is, their connectivity is 0, the *k-order connectivity* of $\mathbf{v}_1$ and $\mathbf{v}_2$ is always greater than that of $\mathbf{v}_1$ and $\mathbf{v}_6$, and therefore $\mathbf{v}_1$ can be naturally assigned to $\mathbf{v}_2$. In the same manner, $\mathbf{v}_3$ and $\mathbf{v}_4$ are also assigned to $\mathbf{v}_2$, while $\mathbf{v}_5$ and $\mathbf{v}_7$ can be assigned to $\mathbf{v}_6$. In order to eliminate the influence of the connectivity of the centers themselves, we next introduce the concept of relative connectivity and use it as the basis of class assignment. It is assumed that we have m clustering centers $\mathbf{v}_{c_i}$, where $c_i \in \{1, 2, \ldots, n\}$ and $i = 1, 2, \ldots, m$. For any non-cluster-center vertex $\mathbf{v}_j$, in order to determine its assignment, we present the concept of the *k-order* relative connectivity between $\mathbf{v}_i$ and $\mathbf{v}_j$, defined as follows:

Definition 3 (relative connectivity)**.** For any vertices $\mathbf{v}_i$ and $\mathbf{v}_j$($i, j = 1, 2, \ldots, n$) in the dataset, the *k-order relative connectivity* of $\mathbf{v}_j$ relative to $\mathbf{v}_i$ is defined by

$$rcon^{(k)}(\mathbf{v}_i, \mathbf{v}_j) = con^{(k)}(\mathbf{v}_i, \mathbf{v}_j)/con^{(k)}(\mathbf{v}_i, \mathbf{v}_i).$$

Based on the concept of *relative connectivity*, the presented classification rule is as follows:

Classification rule: If we have m clustering centers $\mathbf{v}_{c_i}(c_i \in \{1, 2, \ldots, n\}$ and $i = 1, 2, \ldots, m)$, for any vertex $\mathbf{v}_j$, it will be assigned to $\mathbf{v}^*$, where $\mathbf{v}^*$ satisfies

$$\mathbf{v}^* = \underset{\mathbf{v}_{c_i}}{\operatorname{argmax}}(rcon^{(k)}(\mathbf{v}_{c_i}, \mathbf{v}_j)). \tag{2}$$

This classification rule indicates that each vertex ($\mathbf{v}_j$) is assigned to the center ($\mathbf{v}^*$) with respect to which it has the largest relative connectivity. This assignment is natural and reasonable, and it makes the classification process simple and elegant. It shoud be noted that expression (2) may produce more than one maximum in theory. In this case, the vertex can be assigned to any of the centers. In our code, we classify it to the first center with maximum. In fact, we have never encountered such a situation in practice.

2.2. Clustering process of CCE

The implementation of CCE can be summarized as follows: For each k, clustering centers can be found by (1), and then each of the remaining vertices is assigned to one clustering center according to (2). When $k = 1$, all vertices are clustering centers because the similarity matrix is generally a diagonal maximal matrix. Therefore, the initial number of categories is equivalent to the number of vertices, which is the extreme situation for a clustering problem. As k increases, the acquired clustering centers and the corresponding clustering result gradually reflect the connectivity among the vertices from micro to macro. When k approaches infinity, for a connected graph, the number of ultimate clustering centers will be generally reduced to one, which indicates that all the vertices belong to the same cluster. Therefore, k can be regarded as a scale-regulating factor which determines how we observe the data to be clustered, and the changing process of k from small to large corresponds to the evolving process of connection center of the graph from local to global. This is why the algorithm is named connection center evolution. In short, the evolution of the clustering center follows the rule below: in the early stages, all vertices are mainly affected by their neighboring vertices, which leads to a larger number of local clustering centers. As the observation scale (k) becomes larger, the connectivity between vertices is propagated more widely, so each vertex will be affected by a wider range of vertices and, in general, the number of clustering centers will gradually decrease. When k is large enough, the number of clustering centers is further reduced and eventually reduced to one.

The above datasets in Figs. 1 and 2 are used to illustrate the classification process of CCE. For the 4-vertices dataset (Fig. 3), when $k \geq 2$, $\mathbf{v}_2$ is found to be the center of the cluster, and $\mathbf{v}_1$, $\mathbf{v}_3$,$\mathbf{v}_4$ are all assigned to $\mathbf{v}_2$ according to the presented

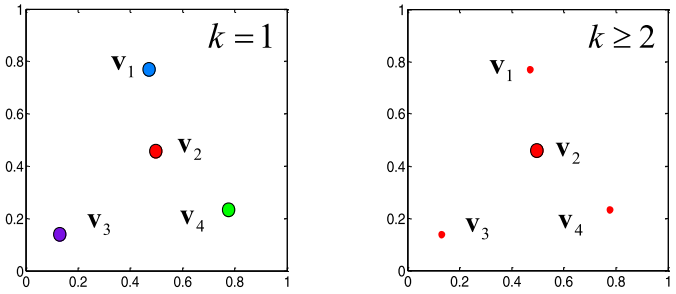


Fig. 3. The classification process of CCE for the data in Fig. 1. Here, a Gaussian kernel ($H_{Gauss}: s_{ij} = \exp(-\|\mathbf{v}_i - \mathbf{v}_j\|^2/\sigma^2), \sigma = 0.8$) is used. Connection centers are marked with larger dots and the remaining vertices are marked with smaller ones and colored according to the clustering centers to which they are assigned. When $k \geq 2$, the unique global clustering center $\mathbf{v}_2$ is found because $s_{22}^{(k)}$ is the only diagonally maximal element of $\mathbf{S}^k$. The corresponding number of classes is 1. All the other vertices are assigned to $\mathbf{v}_2$.

classification rule. For the 7- vertices dataset (Fig. 4), when $k = 2, 3, 4$, $\mathbf{v}_2$ and $\mathbf{v}_6$ are the centers. $\mathbf{v}_1$, $\mathbf{v}_3$, $\mathbf{v}_4$ are assigned to $\mathbf{v}_2$, and $\mathbf{v}_5$, $\mathbf{v}_7$ are assigned to $\mathbf{v}_6$. When $k \geq 5$,$\mathbf{v}_2$ is found to be the global clustering center.

As shown above, the number of categories for a given dataset to be clustered is often determined by the scale of observation. When the observation scale is large enough, in general all the vertices are eventually classified as a single class and they will be assigned to a global clustering center. However, compared with the global clustering center, we usually tend to pay more attention to the local clustering center of data. In fact, in the process of k changing from small to large, some vertices will meet the definition of clustering center (Definition 2), which we call *local* or *metastable clustering centers*.

After obtaining the local clustering centers, we can next give the corresponding clustering results of CCE according to (2). Usually, the clustering results that remain unchanged in a certain scale range are what we want. However, for some datasets, we find that although the different scales correspond to the same number of categories, the corresponding clustering centers and the clustering results may be slightly different. In this case, we introduce the normalized cut (*Ncut*) [28] as an evaluation index to further determine which result is more reasonable. Assume that the vertices $G = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ can be divided into m categories according to (2), which are denoted $V_1, V_2, \ldots, V_m$ ($G = \bigcup_{l=1}^{m} V_l$). The *Ncut* index is defined as

$$Ncut(V_1, V_2, \ldots, V_m) = \sum_{l=1}^{m} \sum_{\mathbf{v}_i \in V_l, \mathbf{v}_j \in \bar{V}_l} s_{ij}^{(k)} / Vol(V_l), \tag{3}$$

where $\bar{V}_l$ represents the complement of V_l in G and $Vol(V_l) = \sum_{\mathbf{v}_i \in V_l, \mathbf{v}_j \in G} s_{ij}^{(k)}$ is the sum of k-order connectivity between all vertices in V_l and all vertices in G. It is apparent that, under the condition that the number of categories is fixed, the smaller the *Ncut* index, the more reasonable the corresponding clustering result. Therefore, for CCE, when multiple scales correspond to the same number of categories, we choose the result with the smallest *Ncut* value as the output of CCE for that category number.

It should be noted that in many cases, the number of samples of different categories is usually different in the dataset to be clustered. When this difference is very large, it is easy to have a 'large cluster swallowing small cluster' phenomenon, that is, the clusters with a few samples are more likely to be quickly merged with clusters having a large number of samples. In this case, we can first normalize the similarity matrix as follows

$$\tilde{\mathbf{S}} = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}, \tag{4}$$

where $\mathbf{D} = diag(d_1, d_2, \ldots, d_n)$ is the degree matrix of $\mathbf{S}$ and $d_i = \sum_{j=1}^{n} s_{ij}$ is the degree of the ith vertex ($\mathbf{v}_i$).

2.3. Computational complexity

After constructing the similarity matrix of vertices, the execution of CCE is mainly composed of three parts: the kth power of the similarity matrix, the determination of the clustering center, and the assignment of categories. Compared to the first item, the computational complexity of the latter two items can be negligible. The computational complexity of the kth power of an $n \times n$ matrix is $O(n^3)$. If the total number of iterations is T, the overall computational complexity of CCE is $O(Tn^3)$. As is generally known, the computational complexity of spectral clustering and of mean-shift is $O(n^3)$ and $O(Tn^2)$, respectively. It can be seen that CCE needs a higher computational complexity than spectral clustering and mean-shift to achieve the final clustering results. However, spectral clustering and mean-shift can only achieve a static clustering result. In contrast, CCE can obtain a group of dynamic clustering results, and is able to suggest a reasonable number of categories. In addition, CCE has a simple and elegant clustering mechanism, so in many cases, CCE will be the better choice.

3. Experiments

In this section, some synthetic and real data are used to evaluate the performance of CCE. Five popular clustering algorithms (k-means [1], mean-shift [15], the spectral cluster method [26], the spectral clustering method based on random walks [31], and FSFDP [18]) are selected here for comparison. For convenience, the spectral clustering method [26] is named SC1 and the spectral clustering method based on random walks is named SC2. For CCE, SC1 and SC2, a Gaussian kernel function is used to measure the similarity between vertices. For mean-shift, we use the Gaussian kernel function to calculate the new clustering center. And for FSFDP, we also use the Gaussian kernel function to obtain the density of vertices. In addition, for k-means, due to its local extremum problem, we select the result with the least cost function from its 5000 runs as its final output. Since SC1 and SC2 both involve the use of k-means, we also execute the k-means algorithm 5000 times for each and take the optimal solution as their final results.

3.1. Synthetic data

3.1.1. The Galaxy data

From the analysis mentioned above, given a collection of data points, CCE may provide various clustering results for different k. However, which one is reasonable? In order to address this issue, we perform the CCE algorithm on the data points designed by us, named the Galaxy data, illustrated in Fig. 5 and the corresponding number of clusters is recorded in Fig. 5A. When $2 \leq k \leq 8$, the Galaxy data are grouped as 31 clusters (Fig. 5B). In the next iteration interval ($15 \leq k \leq 23$), the data points are grouped as 6 clusters (Fig. 5C). Finally, all the points are merged into 1 cluster. Apparently, the results of 31 clusters and of 6 clusters are both correct. In fact, the reasonable number of clusters can be suggested by the platforms of the curves as shown in Fig. 5A. Again, CCE jumps many states just as it does in the previous example. In particular, the number of clusters is reduced to 31 only after two iterations. That is to say, CCE is able to intelligently knock out the most unreasonable classification results.

The five algorithms are also applied to this dataset for comparison. In order to ensure the fairness of comparison, the Gaussian function used in SC1, SC2, mean-shift and FSFDP is parameterized using the same variance ($\sigma = 2.5$) as that in CCE. From Fig. 6, it can be seen that k-means, SC1 and SC2 can obtain reasonable

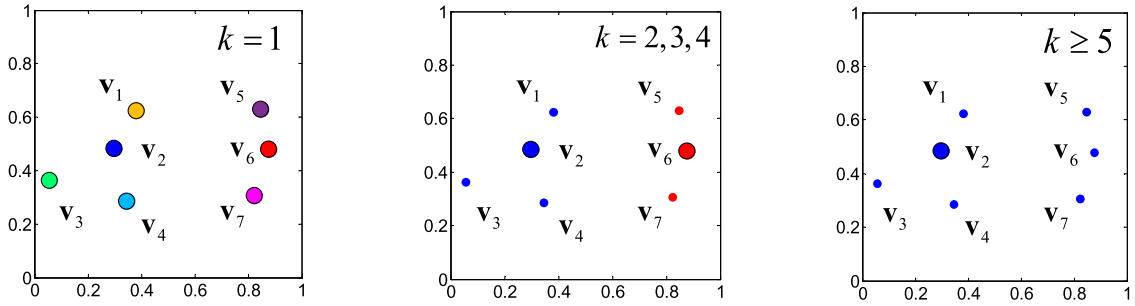


Fig. 4. The classification process of CCE for the data in Fig. 2. Here, a Gaussian kernel ($H_{Gauss}: s_{ij} = \exp(-\|\mathbf{v}_i - \mathbf{v}_j\|^2/\sigma^2), \sigma = 0.5$) is used. Clustering centers are marked with larger dots while the remaining vertices are marked with smaller ones. And the vertices are colored according to the clustering centers to which they are assigned. When $k = 2, 3, 4$, $\mathbf{v}_2$ and $\mathbf{v}_6$ are the centers. $\mathbf{v}_1$, $\mathbf{v}_3$, $\mathbf{v}_4$ are assigned to $\mathbf{v}_2$, and $\mathbf{v}_5$, $\mathbf{v}_7$ are assigned to $\mathbf{v}_6$. When $k \geq 5$,$\mathbf{v}_2$ is found to be the unique global clustering center, and all the other vertices are assigned to $\mathbf{v}_2$.

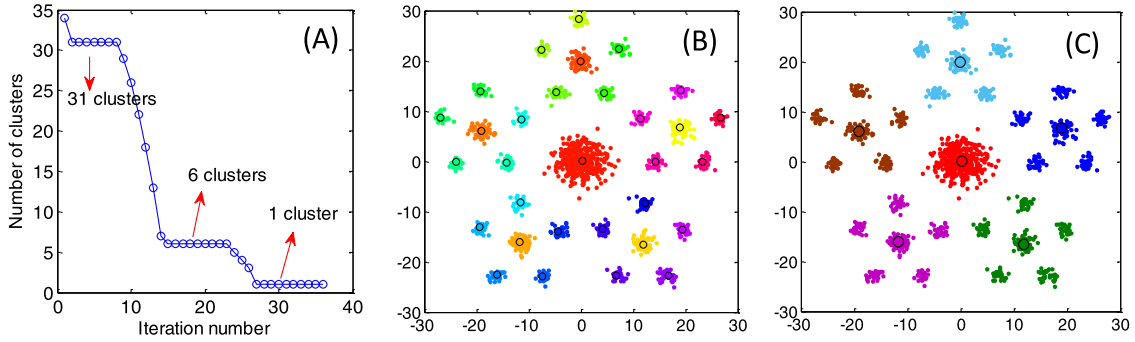


Fig. 5. The results of CCE for the Galaxy data using a Gaussian kernel with a fixed parameter, $\sigma = 2.5$. When k increases, the points are grouped as different clusters. Connection centers are marked with black circles. (A) The number of clusters versus iteration number. (B) Iteration #2∼8, 31 clusters. (C) Iteration #15∼23, 6 clusters.

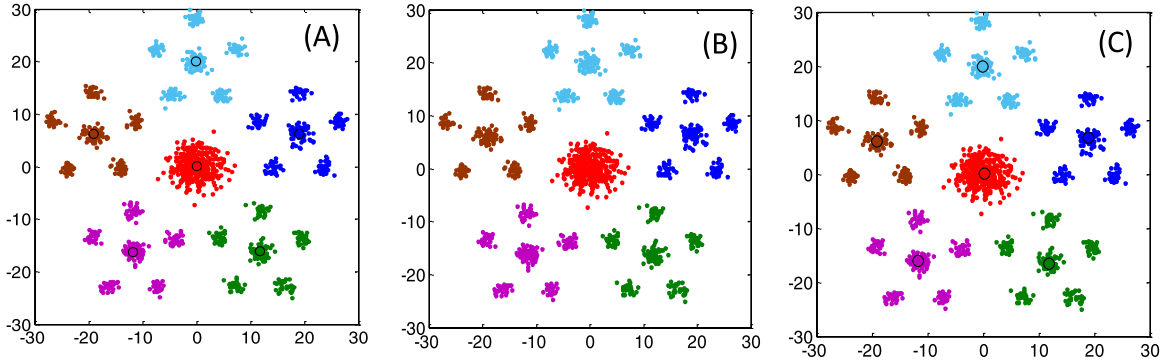


Fig. 6. The clustering results (6 clusters) of the compared algorithms for the Galaxy data. (A) k-means. (B) SC1 and SC2, $\sigma = 2.5$. (C) FSFDP $\sigma = 2.5$.

clustering results under the condition that the number of categories is specified as 6. Through manual selection, FSFDP can also correctly classify these points into 6 clusters. For a given variance, mean-shift will arrive at a unique corresponding clustering result. When the variance is fixed to $\sigma = 2.5$, it happens that mean-shift groups these vertices into 31 categories as shown in Fig. 7D. After careful manual selection, FSFDP again obtains the ideal clustering result for the situation of 31 clusters. However, even if the correct number of categories is given (31 clusters), it is still difficult for k-means, SC1 and SC2 to obtain satisfactory clustering results.

In the following the Adjusted Rand (AR) index [41] is used to further quantitatively compare the performance of different algorithms on a range of variances. The AR index is a scalar and its value is between minus one and one. A large AR value corresponds to a better clustering effect.

Figs. 8 and 9 illustrate the AR curves of the algorithms as a function of variance in the cases of 6 clusters and 31 clusters respectively, and the variance ranges from 1 to 4, with an interval of 0.25. It can be seen that the number of categories in mean-shift is determined only by the given variance used in the Gaussian kernel function. For the given range of variance, mean-shift cannot classify the data into 6 clusters, so its corresponding AR index is much lower than that of other methods. And mean-shift can only obtain reasonable results of 31 clusters when $1.5 \leq \sigma \leq 3.75$. A too-large variance ($\sigma$=4) or too-small variance (σ=1) will lead to an apparent reduction in its AR value. These phenomena imply that the results of mean-shift heavily depend on the choice of variance. For FSFDP, it can achieve a perfect result in the case of 6 clusters for all the given variances and can obtain reasonable results in the case of the 31 clusters when $1 \leq \sigma \leq 3.5$. However, when $\sigma \geq 3.75$, even after careful manual selection in the density-distance scatter plot, its AR value still decreases sharply as shown in Fig. 9. Taking together Figs. 8 and 9, compared with mean-shift and FSFDP, the performance of the two spectral clustering methods (SC1 and SC2)

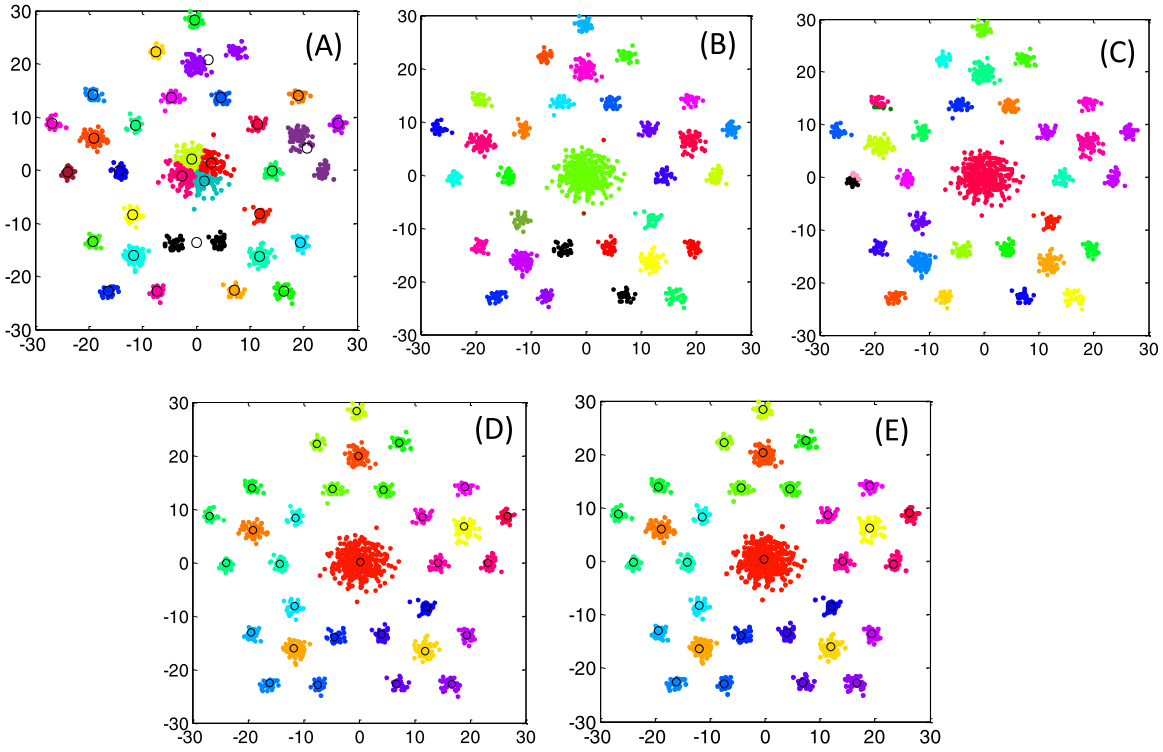


Fig. 7. The clustering results (31 clusters) of the compared algorithms for the Galaxy data. (A). k-means. (B) SC1,$\sigma = 2.5$. (C) SC2,$\sigma = 2.5$. (D) mean-shift,$\sigma = 2.5$ (E) FSFDP, $\sigma = 2.5$.

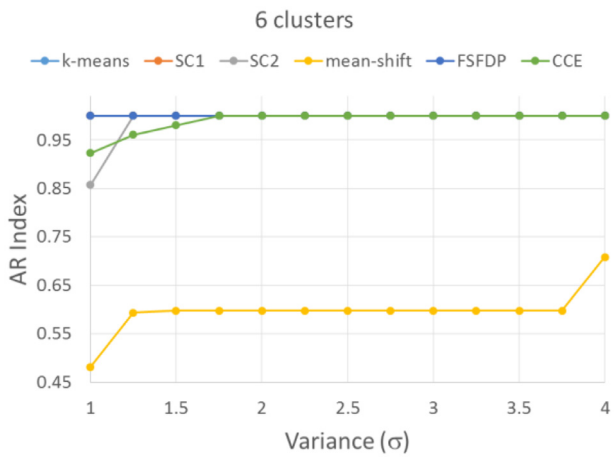


Fig. 8. AR index as a function of variance for the different algorithms when the number of categories is 6.

Fig. 9. AR index as a function of variance for the different algorithms when the number of categories is 31.

are more stable. Nevertheless, in the case of the 31 clusters, even if the correct number of categories is given, it is difficult for SC1 or SC2 to obtain ideal clustering results for all the given variances. For CCE, except that its AR values are slightly lower than 1 at a few parameters ($\sigma = 1, 1.25, 1.5$) in the case of the 6 clusters, it shows a perfect and stable clustering performance on the whole. In addition, a reasonable number of categories can automatically be found from the CCE's curve similar to Fig. 5A. As for k-means, its output heavily depends on the distribution of the data to be clustered. For these data points, its clustering result is perfect in the case of 6 clusters. However, when the number of categories is set to 31, no matter how many times we implement it, k-means can never obtain satisfactory clustering results.

Table 2 gives the statistics of AR index for several comparative methods in the case of $\sigma = 2.5$, from which we can see that the clustering results of mean-shift, of FSFDP and of CCE are deterministic when the parameter (σ) is given. As for k-means, SC1 and SC2, their AR indices vary over a certain range. It is worth noting that in the situation of the 6 clusters, the worst result of SC1 or SC2 is even worse than that of the k-means algorithm. This means that when the distribution of data meets the applicable condition for k-means, it is not necessary to first use spectral clustering to project the data into the space spanned by the first few eigenvectors of the pairwise similarity matrix and then cluster the projected data.

3.1.2. The public data

In this part, we use three commonly used data sets, namely Flame, Dual-ring and Iris, to further evaluate the performance of CCE. For each data set, we specify a fixed variance for the Gaussian function shared by all the methods except k-means. And

Table 2
The statistics of AR Index for several compared methods ($\sigma = 2.5$).

		k-means	SC1	SC2	mean-shift	FSFDP	CCE
Galaxy (6 clusters)	Min	0.6950	0.4641	0.0929	0.5972	1	1
	Max	1	1	1	0.5972	1	1
	Mean	0.9404	0.9101	0.6973	0.5972	1	1
	STD	0.1080	0.1036	0.1931	0	0	0
Galaxy (31 clusters)	Min	0.3065	0.7055	0.5370	1	1	1
	Max	0.7291	0.9870	0.9934	1	1	1
	Mean	0.4344	0.9286	0.8959	1	1	1
	STD	0.0548	0.0299	0.0526	0	0	0

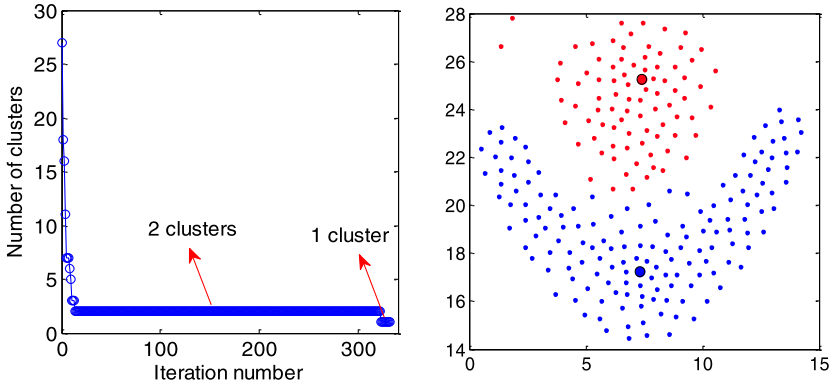


Fig. 10. The results of CCE for the Flame data using a Gaussian kernel with a fixed variance, $\sigma = 1.2$.

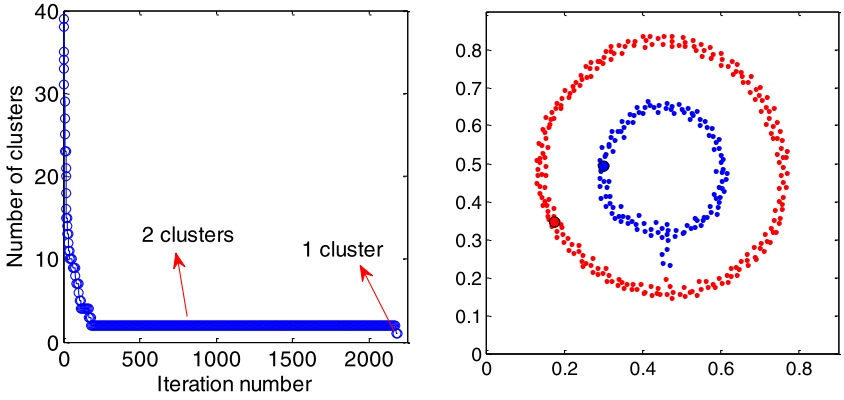


Fig. 11. The results of CCE for the Dual-ring data using a Gaussian kernel with a fixed variance, $\sigma = 0.02$.

the variance in Flame, Dual-ring and Iris is set to 1.2, 0.02 and 1, respectively. As can be seen from Figs. 10 and 11, the curve platforms of CCE clearly give a reasonable number of categories for the Flame and Dual-ring data. In addition, the AR indices of CCE and of the two spectral clustering methods are very close, both showing obvious advantages over mean-shift and FSFDP as shown in Fig. 13A and 13B. This is probably because both mean-shift and FSFDP are dedicated to discovering the density center of data, but the distribution of scatter points for the Flame and Dual-ring data is fairly uniform, so it is difficult for them to obtain satisfactory clustering results for these two pieces of data. As for the Iris data, it seems that CCE encounters some problems. It is known that the Iris data set contains three types, namely Iris Setosa, Iris Versicolour and Iris Virginica. However, as can be seen from Fig. 12A, CCE strongly recommends that this dataset should be divided into two categories, while the platform corresponding to the three types only lasts four times ($k = 3, 4, 5, 6$). After exploring the distribution of data in the space spanned by its first two principal components (Fig. 12B), we find that the suggestion of CCE is reasonable since the points of Iris Versicolour and Iris Virginica are very close, even mixed together, and they are apparently separated from that of Iris Setosa. Even so, when this dataset is divided into three categories, the AR index of CCE is still better than those of other comparable methods, as shown in Fig. 13C.

As can be seen from Table 3, when the parameter is fixed, the results of mean-shift, FSFDP and CCE are still deterministic for all data. For the datasets of Flame and Dual-ring, the AR indices of the two spectral clustering methods remain unchanged in all 5000 runs and their clustering accuracy is far superior to that of k-means. This shows that for some irregular data, it is necessary to first use spectral clustering to project the data into the space spanned by the first several eigenvectors of the pairwise similarity matrix, and then cluster the projected data using k-means. In this way, the clustering effect will be significantly improved. For the Iris data, the AR indices of k-means, SC1 and SC2 change and the performance of spectral clustering is slightly better than

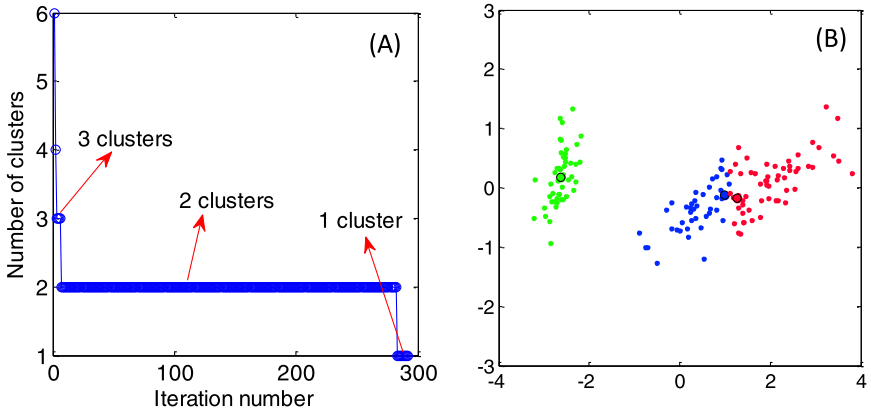


Fig. 12. The results of CCE for the Iris data using a Gaussian kernel with a fixed variance, $\sigma = 1$.

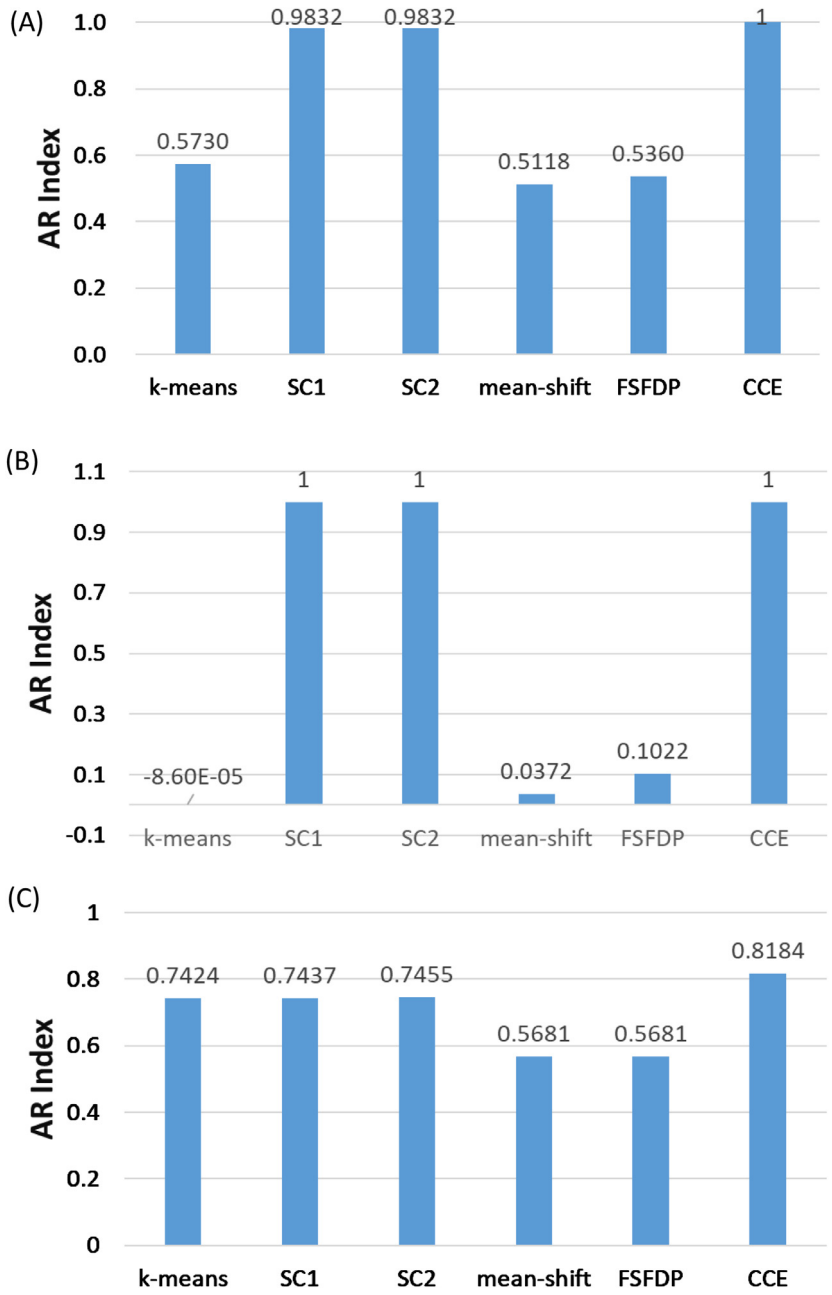


Fig. 13. AR index of the algorithms for the different datasets. (A) The Flame data, $\sigma = 1.2$. (B) The Dual-ring data, $\sigma = 0.02$. (C) The Iris data, $\sigma = 1$.

Table 3
The statistics of AR Index for several compared methods.

		k-means	SC1	SC2	mean-shift	FSFDP	CCE
Flame	Min	0.0185	0.9832	0.9832	0.5118	0.5360	1
	Max	0.5730	0.9832	0.9832	0.5118	0.5360	1
	Mean	0.4694	0.9832	0.9832	0.5118	0.5360	1
	STD	0.0427	0	0	0	0	0
Dual-ring	Min	−0.0028	1	1	0.0372	0.1022	1
	Max	−8.60E-05	1	1	0.0372	0.1022	1
	Mean	−0.0019	1	1	0.0372	0.1022	1
	STD	0.0007	0	0	0	0	0
Iris	Min	0.4205	0.4471	0.4499	0.5681	0.5681	0.8184
	Max	0.7424	0.7437	0.7455	0.5681	0.5681	0.8184
	Mean	0.6703	0.7249	0.7189	0.5681	0.5681	0.8184
	STD	0.1214	0.0710	0.0829	0	0	0

that of k-means from all the three statistics (min, max and mean).

3.2. Real data

The real data used here is for the high-speed rail transport network in China. We hope to rank the importance of cities, or whether they are transport hubs by analyzing the connectivity among all the stations. The data was downloaded from the website [42] in May, 2016. Due to geological and geographical factors, high-speed railway stations are primarily distributed in the east of China. Here, the similarity s_{ij} is set to the number of direct routes between station i and j without any intermediate station. s_{ii} is weighted according to the following rules: If station i is the beginning or terminal station of one route, s_{ii} is added by the number of stations on the route in order to reflect its importance; if station i is an intermediate station of one route, s_{ii} is added by 2. Fig. 14A shows the number of connection centers versus the iteration number. The curve descends sharply at the beginning, and then it begins to decrease slowly as the number of iterations increases. When $k = 8$, eighteen cities are chosen as connection centers (Fig. 14B), all of which are important cities, including provincial capitals, busy seaports and area economy centers. After some iterations, a short-lasting platform ($13 \le k \le 15$) emerges and the number of connection centers is reduced to 9 (Fig. 14C). When $21 \le k \le 24$, a new locally stable result with four connection centers (Shanghai, Chengdu, Guangzhou and Beijing) is presented (Fig. 14D). Interestingly, they correspond to the four centers in the eastern, western, southern and northern part of China. In the following iterations, when $25 \le k \le 42$, a steady platform with a larger interval appears in the curve. In this interval, the number of connection centers falls to 2, and the selected cities are, unsurprisingly, Beijing and Shanghai (Fig. 14E), which are the biggest and most important cities in China. The capital, Beijing is connected by widely distributed cities and its sphere of influence appears greater.

By contrast, the number of stations assigned to Shanghai is significantly fewer than those assigned to Beijing. However, Shanghai finally defeats Beijing and is chosen as the ultimate connection center of all the stations. This can be attributed to the two following facts. First, Shanghai itself has a population of about 25 million, and it is the most economically developed city in China. Second, Shanghai is the center of the Yangtze River Delta which is the most prosperous and advanced region in China. The metropolitan area of the Yangtze River Delta contains some of the most economically dynamic cities such as Nanjing, Hangzhou, Suzhou, Wuxi, Ningbo, Wenzhou and Hefei, which makes the distribution of high-speed railway routes in this region far denser than those in other regions. It should be noted that one city may be assigned to a more remote center because the assignment of class in CCE is based on the actual high-speed railway routes rather than the geographical adjacency of the cities. For instance, one city (Dandong) marked in blue is assigned to Beijing though it is geographically nearer to Dalian (Fig. 14B), because the routes between Beijing and Dandong are twice as many as those between Dalian and Dandong.

4. Discussion

In Section 2.2, we mentioned that when the number of points between categories is seriously unbalanced, the similarity matrix used in CCE is recommended to be normalized first. In this section, we will discuss the impact of normalization on CCE. For this purpose, we generated three sets of data, each containing 200 points and visually consisting of two clusters. The first set of data consists of two clusters with the same number of points, that is, the number of samples in both classes is 100 (Fig. 15). The ratio of the number of points of the two clusters in the second dataset is 7:3 (Fig. 16) and the ratio in the third group is 9:1 (Fig. 17). The three sets of data share the same Gaussian kernel similarity matrix, and the variance is fixed to $\sigma = 0.2$. In the following discussion, we refer to CCE using the normalized similarity matrix as the normalized CCE. For each set of data, two versions of CCE (original version, and normalized version) are compared. From Figs. 15, 16 and 17, it can be seen that both versions of CCE can successfully divide each group of data into two clusters, and the corresponding clustering results are also completely consistent. However, the duration of the curve's platform in Figs. 15–17 reveals the difference between these two versions. The original CCE converges faster and can enter a stable platform only after two or three iterations, while the normalized CCE converges slowly and needs more iterations to get the desired results. In addition, for the original CCE, the length of the platform of the clustering curve depends on whether all the samples are roughly evenly distributed in each category. When the sizes of the categories are comparable as the dataset in Fig. 15, the curve will have a longer stable platform. However, for the normalized CCE, no matter how unbalanced the distribution of samples, its platform length is always long enough.

It should be noted that we adopted the original version of CCE in all the experiments of Sections 3.1 and 3.2. In practical applications, when we know nothing about the data to be clustered,

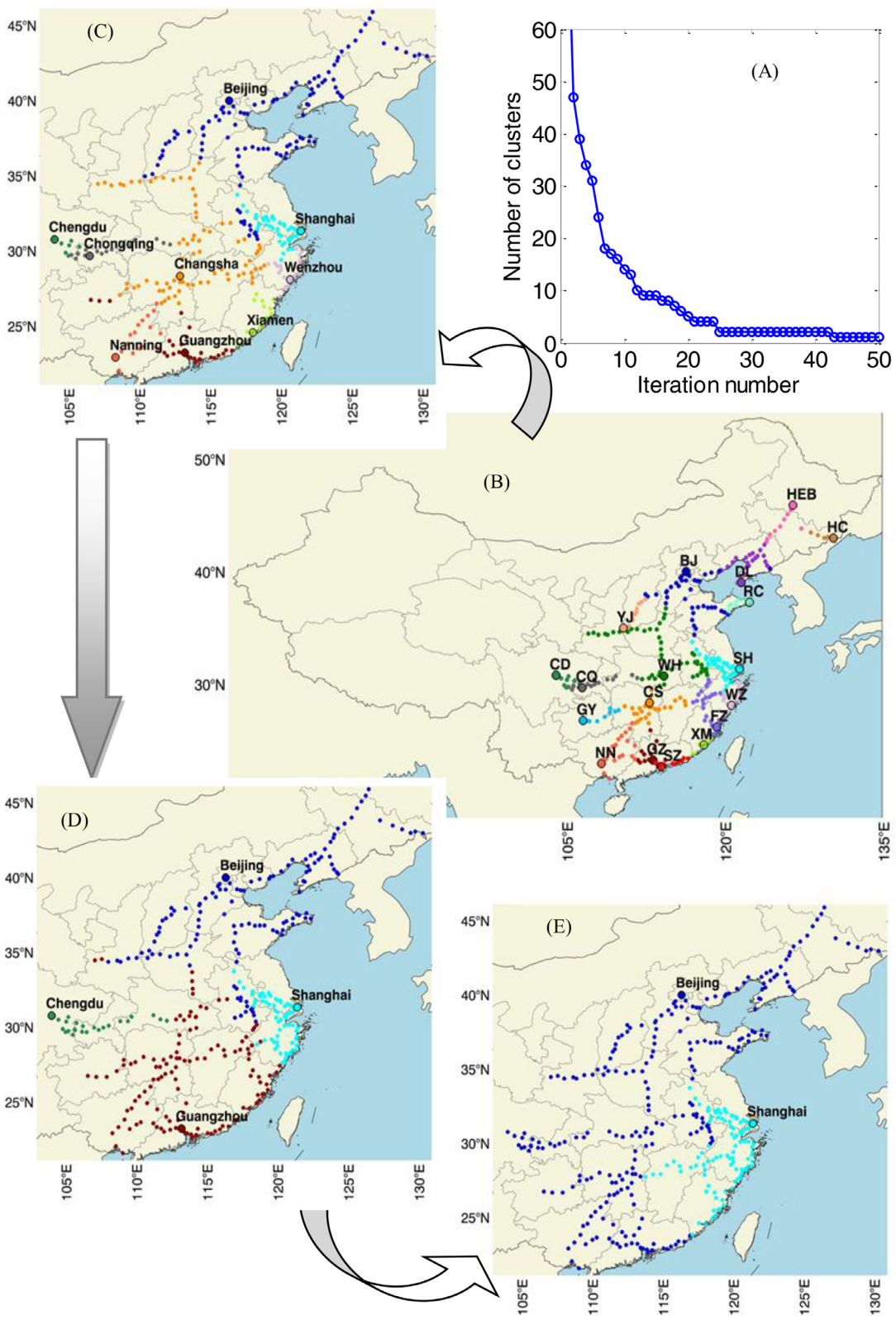


Fig. 14. The high-speed railway network in China. CCE is used to explore the connection centers of all the stations. (A) The number of clusters versus iteration number (k). (B) Eighteen connection centers and the corresponding clustering result when $k = 8$. (C) Nine connection centers and the corresponding clustering result when $13 \leq k \leq 15$. (D) Four connection centers (Shanghai, Chengdu, Guangzhou and Beijing) and the corresponding clustering result when $21 \leq k \leq 24$. (E) Two connection centers (Shanghai and Beijing) and the corresponding clustering result when $25 \leq k \leq 42$.

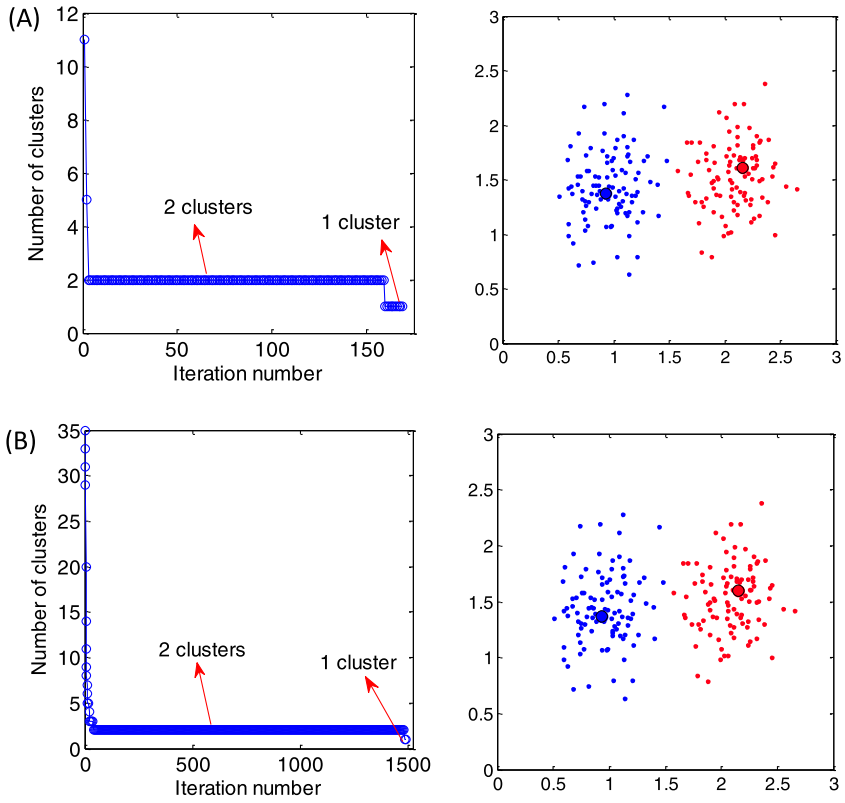


Fig. 15. Impact of normalization on CCE in the case that the ratio of the number of points of the two clusters is 5:5. (A) The clustering result for the original data. (B) The clustering result for the normalization data.

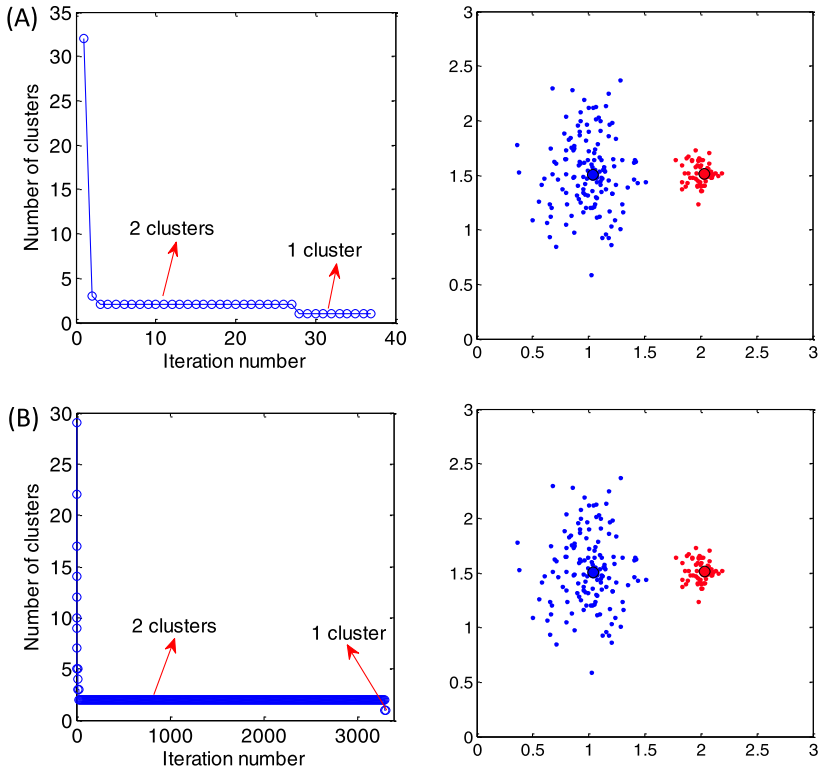


Fig. 16. Impact of normalization on CCE in the case that the ratio of the number of points of the two clusters is 7:3. (A) The clustering result for the original data. (B) The clustering result for the normalization data.

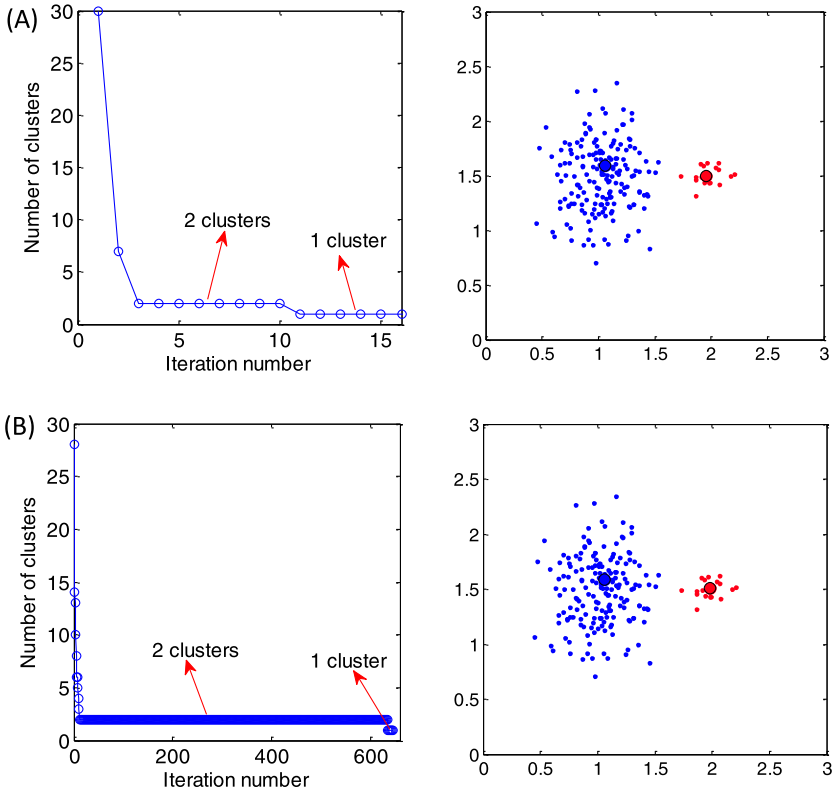


Fig. 17. Impact of normalization on CCE in the case that the ratio of the number of points of the two clusters is 9:1. (A) The clustering result for the original data. (B) The clustering result for the normalization data.

it is difficult for us to recommend which version of CCE is more appropriate. Fortunately, for many data, these two versions of CCE usually come to the same or similar clustering results.

5. Conclusion

For many clustering problems, the true number of clusters is actually ambiguous due to the difference in the cluster scale. In general, the larger the scale, the smaller the number of categories, and vice versa. Most clustering algorithms work only in those cases where the scale is specified in advance and end with an unreasonable clustering result if the scale selection is not appropriate. CCE presents a natural strategy for clustering problems and it can intelligently provide all reasonable results from local to global scale. Additionally, when the pairwise similarity matrix of the data is given, the implementation of CCE involves only the power of the matrix and does not require any manual interference, and thus is very simple and efficient.

It should be pointed out that, given the similarity matrix, CCE can automatically solve all the remaining issues and the final result is deterministic. However, how to construct a similarity matrix or how to determine the appropriate variance of the Gaussian function is also a very important issue in the field of clustering, which will be the focus of our next research.

Acknowledgment

The author (Xiurui Geng) expresses special thanks to his dearest daughter Yihan Geng, whose birth has brought him endless happiness and energy.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2019.107063.

References

[1] J.A. Hartigan, M.A. Wong, Algorithm as 136: a k-means clustering algorithm, Appl. Stat. (1979) 100–108.
[2] S.Z. Selim, M.A. Ismail, k-means type algorithms: a generalized convergence theorem and characterization of local optimality, IEEE Trans. Pattern Anal. Mach. Intell. 6 (1984) 81–87.
[3] A.K. Jain, Data clustering: 50 years beyond K-means, Pattern Recognit. Lett. 31 (2010) 651–666.
[4] J. Newling, F. Fleuret, Nested mini-batch k-means, Adv. Neural Inf. Process. Syst. (2016).
[5] C. Boutsidis, A. Zouzias, M.W. Mahoney, P. Drineas, Randomized dimensionality reduction for k-Means clustering, IEEE Trans. Inf. Theory 61 (2015) 1045–1062.
[6] F.P. Malinen, M, K-means: clustering by gradual data transformation, Pattern Recognit. 47 (2014) 3376–3386.
[7] P. Fränti, S. Sieranoja, How much can k-means be improved by using better initialization and repeats? Pattern Recognit. 93 (2019) 95–112.
[8] G. McLachlan, D. Peel, Finite Mixture Models, JohnWiley & Sons, New York, 2000.
[9] X. Zhuang, Y. Huang, K.Z.Y. Palaniappan, Gaussian mixture density modelling, decomposition and applications, IEEE Trans. Image Process. 5 (1996) 1293–1302.
[10] Rosa Altilio, P. Lorenzo, Massimo Panella, Distributed data clustering over networks, Pattern Recognit. 93 (2019) 603–620.
[11] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR, 1996, pp. 226–231.
[12] A.K.D.A. Hinneburg, An efficient approach to clustering large multimedia databased with noise, in: Proceedings of the 4th ACM SIGKDD, New York, NY, 1998, pp. 58–65.
[13] A.C. Bryant, K.J. Cios, RNN-DBSCAN: a density-based clustering algorithm using reverse nearest neighbor density estimates, IEEE Trans. Knowl. Data Eng. 30 (2017) 1109–1121.
[14] K. Fukunaga, L.D. Hostetler, The estimation of the gradient of a density function, with applications in pattern recognition, IEEE Trans. Inf. Theory 21 (1) (1975) 32–40.
[15] Y. Cheng, Mean shift, mode seeking, and clustering, IEEE Trans. Pattern Anal. Mach. Intell. 17 (1995) 790–799.
[16] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 603–619.
[17] S. Anand, S. Mittal, O. Tuzel, P. Meer, Semi-Supervised kernel mean shift clustering, IEEE Trans. Pattern Anal. Mach. Intell. 36 (6) (2013) 1201–1215.

[18] Alex Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (2014) 1492–1496.
[19] M. D'Errico, E. Facco, A. Laio, A. Rodriguez, Automatic topography of high-dimensional data sets by non-parametric density peak clustering, arXiv:1802.10549, (2018).
[20] C.T. Zahn, Graph-theroretical methods for detecting and describing gestalt clusters, IEEE Trans. Comput. C-20 (1971) 68–86.
[21] G.T. T., The relative neighborhood graph of a finite planar set, Pattern Recognit. 12 (1980) 261–268.
[22] S. Tamura, Clustering based on multiple paths, Pattern Recognit. 15 (1982) 477–483.
[23] S. Van Dongen, Graph Clustering by Flow Simulation, University of Utrecht, 2000.
[24] A. Aksaca, T. Özyerb, R. Alhajj, CutESC: cutting edge spatial clustering technique based on proximity graphs, Pattern Recognit. (2019) 96.
[25] Y. Weiss, Segmentation using eigenvectors: a unifying view, in: Proceedings of the Seventh IEEE International Conference on Computer Vision, IEEE, 1999, pp. 975–982.
[26] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, Adv. Neural Inf. Process. Syst. (2002) 849–856.
[27] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–895.
[28] U. Von Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (2007) 395–416.
[29] R. Kannan, S. Vempala, A. Vetta, On clusterings: good, bad and spectral, J. ACM (JACM) 51 (2004) 497–515.
[30] P. Perona, W. Freeman, A factorization approach to grouping, in: Proceedings of the European Conference on Computer Vision, 1998, pp. 655–670.
[31] M. Meila, J. Shi, A random walks view of spectral segmentation, in: Proceedings of the 8th International Workshop on Artificial Intelligence and Statistics (AISTATS), 2001.
[32] J. Ponttuset, P. Arbelaez, J. Barron, F. Marques, J. Malik, Multiscale combinatorial grouping for image segmentation and object proposal generation, IEEE Trans. Pattern Anal. Mach. Intell. 39 (2017) 128–140.
[33] A.G. Christos Boutsidis, Prabhanjan Kambadur, Spectral clustering via the power method - Provably, in: Proceedings of the 32nd International Conference on Machine Learning, JMLR: W&CP, Lille, France, 2015.
[34] D. Hofmeyr, Clustering by minimum cut hyperplanes, IEEE Trans. Pattern Anal. Mach. Intell. 39 (2016) 1547–1560.
[35] B. Fischer, J.M. Buhmann, Path-Based clustering for grouping of smooth curves and texture segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 25 (2003) 513–518.
[36] H. Chang, D.-.Y. Yeung, Robust path-based spectral clustering, Pattern Recognit. 41 (2008) 191–203.
[37] Sergios Theodoridis, K. kourtroumbas, Pattern Recognition, 4th ed., Elsevier Inc., Singapore, 2009.
[38] C. Godsil, G. Royle, Algebraic Graph Theory, Cambridge University Press, 1974.
[39] F.R.K. Chung, Spectral Graph Theory, American Mathematical Society, Providence, Rhode Island, 1994.
[40] A.E. Brouwer, W.H. Haemers, Spectra of Graphs, Springer, New York, 2012.
[41] L. Hubert, P. Arabie, Comparing partitions, J. Classif. 2 (1985) 193–218.
[42] The high-speed rail transport network data in China are from the following website: http://checi.tieyou.com/g_index.html.

Xiurui Geng received his B.S. and M.S. degrees in applied mathematics from Beihang University, in 1999 and 2002 respectively, and received the Ph.D. degree in hyperspectral remote sensing from the Institute of Remote Sensing Applications, Chinese Academy of Sciences, Beijing, China, in 2005. Currently, he is a Professor in the Key Laboratory of Technology in Geo-spatial Information Process and Application Systems, Institute of Electronics, Chinese Academy of Sciences. His-research interests are data understanding and algorithm development in feature extraction, dimension reduction, image matching, target detection and classification.

Hairong Tang received the Ph.D. degree in signal processing from the Institute of Electronics, Chinese Academic of Sciences, Beijing, China, in 2003. She is currently an Associate Professor with the Key Laboratory of Technology in Geo-Spatial Information Process and Application Systems, Institute of Electronics, Chinese Academy of Sciences. Her research interests include hyperspectral data processing and optimization, cloud detection, feature extraction, and clustering analysis.