Pattern Recognition Letters 29 (2008) 2067–2077

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

DIVFRP: An automatic divisive hierarchical clustering method based on the furthest reference points

Caiming Zhong [a,b,*], Duoqian Miao [a,c], Ruizhi Wang [a,c], Xinmin Zhou [a]

[a] School of Electronics and Information Engineering, Tongji University, Shanghai 201804, PR China
[b] College of Science and Technology, Ningbo University, Ningbo 315211, PR China
[c] Tongji Branch, National Engineering and Technology Center of High Performance Computer, Shanghai 201804, PR China

ARTICLE INFO

Article history:
Received 25 September 2007
Received in revised form 21 May 2008
Available online 22 July 2008

Communicated by L. Heutte

Keywords:
Divisive clustering
Automatic clustering
Furthest reference point
Dissimilarity measure
Peak
Spurious cluster

ABSTRACT

Although many clustering methods have been presented in the literature, most of them suffer from some drawbacks such as the requirement of user-specified parameters and being sensitive to outliers. For general divisive hierarchical clustering methods, an obstacle to practical use is the expensive computation. In this paper, we propose an automatic divisive hierarchical clustering method (DIVFRP). Its basic idea is to bipartition clusters repeatedly with a novel dissimilarity measure based on furthest reference points. A sliding average of sum-of-error is employed to estimate the cluster number preliminarily, and the optimum number of clusters is achieved after spurious clusters identified. The method does not require any user-specified parameter, even any cluster validity index. Furthermore it is robust to outliers, and the computational cost of its partition process is lower than that of general divisive clustering methods. Numerical experimental results on both synthetic and real data sets show the performances of DIVFRP.

1. Introduction

Clustering is an unsupervised classification technique in pattern analysis (Jain et al., 1999). It is defined to divide a data set into clusters without any prior knowledge. Objects in a same cluster are more similar to each other than those in different clusters. Many clustering methods have been proposed in the literature (Xu and Wunsch, 2005; Jain et al., 1999). These methods can be roughly classified into following categories: hierarchical, partitional, density-based, grid-based and model-based methods. However, the first two methods are the most significant algorithms in clustering communities. The hierarchical clustering methods can be further classified into agglomerative methods and divisive methods. Agglomerative methods start with each object as a cluster, recursively take two clusters with the most similarity and merge them into one cluster. Divisive methods, proceeding in the opposite way, start with all objects as one cluster, at each step select a cluster with a certain criterion (Savaresi et al., 2002) and bipartition the cluster with a dissimilarity measure.

In general, partitional clustering methods work efficiently, but the clustering qualities are not as good as those of hierarchical methods. The *K*-means (MacQueen, 1967) clustering algorithm is one of well-known partitional approaches. Its time complexity is $O(NKId)$, where N is the number of objects, K is the number of clusters, I is the number of iterations required for convergence, and d is the dimensionality of the input space. In practice, K and d are usually far less than N, it runs in linear time on low-dimensional data. Even though it is computationally efficient and conceptually simple, *K*-means has some drawbacks, such as no guarantee of convergence to the global minimum, the requirement of the number of clusters as an input parameter provided by users, and sensitivity to outliers and noise. To remedy these drawbacks, some variants of *K*-means have been proposed: PAM (Kaufman and Rousseeuw, 1990), CLARA (Kaufman and Rousseeuw, 1990), and CLARANS (Ng and Han, 1994).

To the contrary, hierarchical clustering methods can achieve good clustering results, but only at the cost of intensive computation. Algorithm single-linkage is a classical agglomerative method with time complexity of $O(N^2 \log N)$. Although algorithm CURE (Guha et al., 1998), one improved variant of single-linkage, can produce good clustering quality, the worst-case time complexity of CURE is $O(N^2 \log_2 N)$. Compared to agglomerative methods, divisive methods are more computationally intensive. For bipartitioning a cluster C_i with n_i objects, a divisive method will produce a global optimal result if all possible $2^{n_i-1}-1$ bipartitions are considered. But clearly, the computational cost of the complete enumeration is prohibitive. This is the very reason why divisive methods are seldom applied in practice. Some improved divisive

* Corresponding author. Address: School of Electronics and Information Engineering, Tongji University, Shanghai 201804, PR China. Tel.: +86 21 69589867; fax: +86 21 69589359.
E-mail addresses: zhongcaiming@nbu.edu.cn, charman_zhong@hotmail.com (C. Zhong).

0167-8655/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2008.07.002

methods do not consider unreasonable bipartitions identified by a pre-defined criterion in order to reduce the computational cost (Gowda and Ravi, 1995). Chavent et al. (2007) in a monothetic divisive algorithm use a monothetic approach to reduce the number of admissible bipartitions.

Most traditional clustering methods, such as *K*-means, DBScan (Ester et al., 1996), require some user-specified parameters. Generally, however, the required parameters are unknown to users. Therefore, automatic clustering methods are expected in practical applications. Some clustering methods of this kind have been presented in the literature (Wang et al., 2007; Tseng and Kao, 2005; Garai and Chaudhuri, 2004; Bandyopadhyay and Maulik, 2001; Tseng and Yang, 2001). Roughly these methods can be categorized into two groups: clustering validity index-based methods (Wang et al., 2007; Tseng and Kao, 2005) and genetic scheme-based methods (Garai and Chaudhuri, 2004; Bandyopadhyay and Maulik, 2001; Tseng and Yang, 2001). Wang et al. (2007) iteratively apply the local shrinking-based clustering method with different cluster number *K*s. In the light of CH index and Silhouette index, the qualities of all clustering results are measured. The optimal clustering result with the best cluster quality is selected. Tseng and Kao (2005) use Hubert's Γ index to measure a cluster strength after each adding (or removing) of objects to (or from) the cluster. For genetic scheme-based clustering methods, it is crucial to define a reasonable fitness function. Bandyopadhyay and Maulik (2001) take some validity indices as fitness functions directly. In the methods of Garai and Chaudhuri (2004) and Tseng and Yang (2001), although validity indices are not used directly, the fitness functions are very close to validity indices essentially. So genetic scheme-based methods, in different extents, are dependent on the clustering validity indices. However, clustering validity indices are not a panacea since an index that can deal with different shapes and densities is not available.

Robustness to outliers is an important property for clustering algorithms. Clustering algorithms that are vulnerable to outliers (Patan and Russo, 2002) may use some outlier detection mechanisms (Aggarwal and Yu, 2001; Ramaswamy et al., 2000; Breunig et al., 2000; Knorr and Ng, 1998) to eliminate the outliers in data sets before clustering proceeds. However, since this is an extra task, users prefer to clustering algorithms robust to outliers.

In this paper, we propose an efficient divisive hierarchical clustering algorithm with a novel dissimilarity measure (DIVFRP). Based on the furthest reference points, the dissimilarity measure makes the partition process robust to outliers and reduces the computational cost of partitioning a cluster C_i to $O(n_i \log n_i)$. After a data set being partitioned completely, the algorithm employs a sliding average of differences between neighboring pairs of sum-of-errors to detect potential peaks and determine the candidates of the cluster number. Finally, spurious clusters are removed and the optimal cluster number *K* is achieved. Our experiments demonstrate these performances. The remaining sections are organized as follows: algorithm DIVFRP is presented in Section 2. Section 3 presents experimental results. The performances are studied in Section 4. Section 5 concludes the paper.

2. The clustering algorithm

We begin our discussion of the clustering algorithm DIVFRP by considering the concept of general clustering algorithm.

Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_N\}$ be a data set, where $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ij}, \ldots, x_{id})^T \in \Re^d$ is a feature vector, and x_{ij} is a feature. A general clustering algorithm attempts to partition the data set $\boldsymbol{X}$ into *K* clusters: $C_0, C_1, \ldots, C_{K-1}$ and one outlier C_{outlier} set according to the similarity or dissimilarity measure of objects. Generally, $C_i \neq \emptyset$, $C_i \cap C_j = \emptyset$, $\boldsymbol{X} = C_0 \cup C_1 \cup \ldots \cup C_{K-1} \cup C_{\text{outlier}}$, where $i = 0, 1, \ldots, K-1$, $j = 0, 1, \ldots, K-1$, $i \neq j$.

The algorithm DIVFRP comprises three phases:

1. Partitioning a data set.
2. Detecting the peaks of differences of sum-of-errors.
3. Eliminating spurious clusters.

2.1. Partitioning a data set

2.1.1. The dissimilarity measure based on the furthest reference points

Similarity or dissimilarity measures are essential to a clustering scheme, because the measures determine how to partition a data set. In a divisive clustering method, let C_i be the cluster to be bipartitioned at a step of the partitioning process, $g(C_x, C_y)$ be a dissimilarity function. If the divisive method bipartitions C_i into C_{i1} and C_{i2}, the pair (C_{i1}, C_{i2}) will maximize the dissimilarity function *g* (Theodoridis and Koutroumbas, 2006). According to this definition of dissimilarity, we design our dissimilarity measure as follows.

For a data set consisting of two spherical clusters, our dissimilarity measure is on the basis of the observation: the distances between points in a same cluster and a certain reference point are approximative. We call the distances a representative. For the two clusters, two representatives exist with respect to a same reference point. Assume that there exits a point on the line that passes through the two cluster mean points, and both clusters are on the same side of the point. Taking the point as the reference point, one will get the maximum value of the difference between the two representatives. On the contrary, if the reference point is on the perpendicular bisector of the line segment that ends at the two cluster mean points, one will get the minimum value. However, it is difficult to get the ideal reference point since the cluster structure is unknown. We settle for the furthest point from the centroid of the whole data set instead, because it never lies between the two cluster mean points and two clusters must be on the same side of it. Fig. 1 illustrates the dissimilarity measure based the furthest point and how the cluster being split.

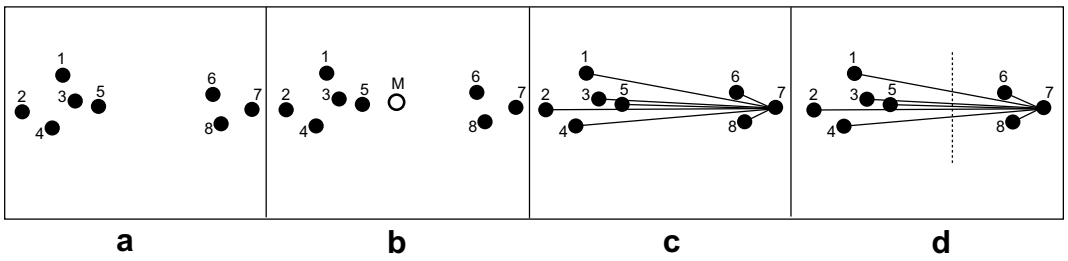

Fig. 1. Illustration of the dissimilarity measure and a split. In (a), a data set with two spherical clusters is shown. In (b), the hollow point *M* is the mean point of the data set; point 7 is the furthest point to the mean and selected as the reference point. In (c), distances from all points including the reference point to the reference are computed. In (d), the neighboring pair $< d_{r6}, d_{r5} >$ with maximum difference between its two elements is selected as the boundary, with which the cluster is split.

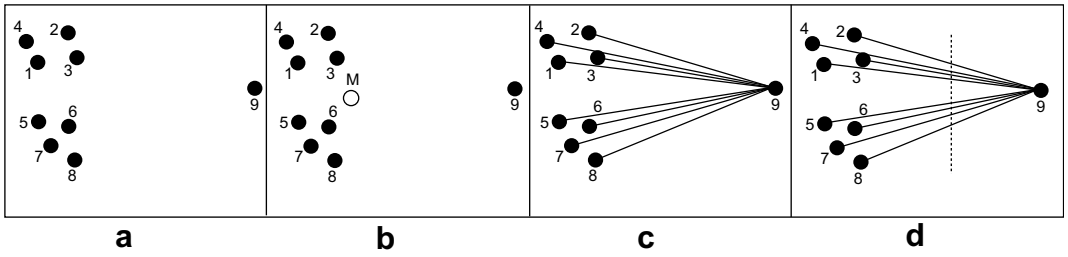

Fig. 2. Illustration of the reference point as an outlier. In (a), two spherical clusters with an outlier is shown. In (b), the hollow point M is the mean point of the data set; point 9 is the furthest point to the mean and selected as the reference point. In (c), distances from all points to the reference are computed. In (d), the neighboring pair $< d_{r9}, d_{r3} >$ with maximum difference between its two elements is selected as the boundary, with which the reference point itself is peeled off.

In Fig. 1b, point 7 is the furthest point to the data set centroid (hollow point M). So it is selected as the reference point r. Let d_{ri} stand for the distance between the reference point r and a point i ($1 \leqslant i \leqslant 8$), the distances are sorted ascendantly as $\langle d_{r7}, d_{r8}, d_{r6}, d_{r5}, d_{r3}, d_{r1}, d_{r4}, d_{r2}\rangle$. Considering all neighboring pairs in the list, we observe that the difference between the elements of the neighboring pair $\langle d_{r6}, d_{r5}\rangle$ is the maximum and select the pair as the boundary. This is the very dissimilarity measure. In accordance with the boundary, the list is then split into two parts: $\langle d_{r7}, d_{r8}, d_{r6}\rangle$ and $\langle d_{r5}, d_{r3}, d_{r1}, d_{r4}, d_{r2}\rangle$. Correspondingly, two clusters are formed: {7,8,6} and {5,3,1,4,2}.

Note that the dissimilarity measure based the furthest reference point does not always discern two clusters well. As aforementioned, when the furthest point is on or close to the perpendicular bisector of the line segment that takes two cluster mean points as its two endpoints, respectively, the dissimilarity measure fails to split the two clusters. Surprisingly, however, the dissimilarity measure acts as an outlier detector in this situation. This property, discussed in Section 4, endows our algorithm with robustness to outliers. In Fig. 2, the sorted list of distances from all of the points to the reference is $\langle d_{r9}, d_{r3}, d_{r6}, d_{r8}, d_{r2}, d_{r7}, d_{r1}, d_{r5}, d_{r4}\rangle$. The neighboring pair $\langle d_{r9}, d_{r3}\rangle$ has the maximum difference and functions as the boundary. So the reference point 9 is peeled off as an outlier.

Then we formally define the dissimilarity function. Suppose C_i is one of the valid clusters, $\boldsymbol{x} \in C_i$, $|C_i| = n_i$.

Definition 1. Let $rp(C_i)$ be the furthest point to the mean of C_i, namely the reference point in C_i, as in

$$rp(C_i) = \arg\sum_{\boldsymbol{x}\in C_i} \|\boldsymbol{x} - \mu(C_i)\| \tag{1}$$

where $\mu(C_i) = \frac{1}{|C_i|}\sum_{\boldsymbol{x}\in C_i}\boldsymbol{x}$

Definition 2. Let $\mathrm{Rank}(C_i)$ be an ordered list as in

$$\mathrm{Rank}(C_i) = \langle \mathrm{near\_ref}(C_i) \circ \mathrm{Rank}(C_i - \{\mathrm{near\_ref}(C_i)\})\rangle \tag{2}$$

where $\circ$ is concatenate operator, $\mathrm{near\_ref}(C_i)$ is the nearest point to the reference point in C_i as in

$$\mathrm{near\_ref}(C_i) = \arg\min_{\boldsymbol{x}\in C_i} \|\boldsymbol{x} - rp(C_i)\| \tag{3}$$

Definition 3. Assume $\mathrm{Rank}(C_i) = \langle \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{n_i}\rangle$, C_i is to be split into C_{i1} and C_{i2}, where $C_{i1} = \langle \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_j\rangle$, $C_{i2} = \langle \boldsymbol{x}_{j+1}, \boldsymbol{x}_{j+2}, \ldots, \boldsymbol{x}_{n_i}\rangle$ and $1 \leqslant j < n$. The dissimilarity function $g(C_{i1}, C_{i2})$ is defined as

$$g(C_{i1}, C_{i2}) = \|\boldsymbol{x}_{j+1} - rp(C_i)\| - \|\boldsymbol{x}_j - rp(C_i)\| \tag{4}$$

This dissimilarity definition can be compared with the distance (dissimilarity) definition of single-linkage algorithm as in

$$\mathrm{dist}(C_i, C_j) = \min_{\boldsymbol{x}_i\in C_i, \boldsymbol{x}_j\in C_j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \tag{5}$$

Because single-linkage is an agglomerative algorithm, the pair of clusters with minimum distance, $\mathrm{dist}(C_i, C_j)$, are merged at each step. Whereas the presented method DIVFRP is a divisive algorithm, the bipartitioned pair (C_{i1}, C_{i2}) of cluster C_i should maximize the dissimilarity function $g(C_{i1}, C_{i2})$.

2.1.2. The partition algorithm

A divisive clustering problem can be divided into two sub-problems (Savaresi et al., 2002):

Problem 1. Selecting which cluster must be split.
Problem 2. How to split the selected cluster.

For Problem 1, generally, the following three criteria can be employed for selecting the cluster to be split at each step (Savaresi et al., 2002):

(1) complete split: every cluster is split;
(2) the cluster with the largest number of objects;
(3) the cluster having maximum variance with respect to its centroid.

Since every cluster is split, criterion (1) is simple, but it does not consider the effect of splitting sequence on the quality of the clusters. Criterion (2) attempts to balance the object numbers of the clusters, but it ignores the "scatter" property. Criterion (3) considers the "scatter" property well, so we use maximum deviation which is similar to criterion (3) in DIVFRPs.

Suppose there are totally M clusters at a certain step, namely $C_0, C_1, \ldots, C_{M-1}$. One of the clusters will be selected for the further bipartition.

Definition 4. Let next_split(CS) be the cluster with the maximum deviation with respect to its centroid, $\mathrm{CS} = \{C_i : 0 \leqslant i \leqslant M-1\}$:

$$\mathrm{next\_split(CS)} = \arg\max_{C_i\in \mathrm{CS}} \|\mathrm{dev}(C_i)\| \tag{6}$$

where $\mathrm{dev}(C_i) = \frac{1}{|C_i|}\sum_{\boldsymbol{x}\in C_i}\|\boldsymbol{x} - \mu(C_i)\|$.

So next_split(CS) is the cluster to be split at next step.

An optimal partition will give the maximum value of the dissimilarity function g (Theodoridis and Koutroumbas, 2006). Bearing this in mind, we can bipartition a cluster as follows.

Definition 5. Let $D(C_i)$ be a set consisting of the differences between elements of all neighboring pairs in $\mathrm{Rank}(C_i)$:

$$D(C_i) = \{d : d = \|\mathrm{next}(\boldsymbol{x}) - rp(C_i)\| - \|\boldsymbol{x} - rp(C_i)\|\} \tag{7}$$

where $\boldsymbol{x}$ is a component in $\mathrm{Rank}(C_i)$, $\mathrm{next}(\boldsymbol{x})$ is the next component to $\boldsymbol{x}$ in $\mathrm{Rank}(C_i)$.

Definition 6. Let $b(C_i)$ be the point in C_i with the maximum difference in $D(C_i)$:

$$b(C_i) = \arg\max_{\boldsymbol{x}\in C_i}(\|\mathrm{next}(\boldsymbol{x}) - rp(C_i)\| - \|\boldsymbol{x} - rp(C_i)\|) \tag{8}$$

The cluster is bipartitioned into C_{i1} and C_{i2} as in:

$$C_{i1} = \{\boldsymbol{x} : \boldsymbol{x} \in C_i \wedge \|\boldsymbol{x} - rp(C_i)\| \leqslant \|b(C_i) - rp(C_i)\|\} \quad (9)$$

$$C_{i2} = C_i - C_{i1} \quad (10)$$

The divisive algorithm (DA) is formally stated as follows.

Divisive algorithm (DA)

Step 1. From Eq. (6), the cluster to be split is determined: $C_i = \text{next\_split}(S)$. The first cluster to be split is the initial data set which includes whole points.
Step 2. Partition C_i into two clusters with Eqs. (8)–(10), record the partition process.
Step 3. If each cluster has only one object, stop; otherwise go to step 1.
Note that recoding the partition process is for the later analysis of the optimal K.

2.2. Detecting the peaks of sum-of-error differences

After partitioning the whole data set, we will figure out the proper clusters. We classify splits into two categories: essential splits and inessential splits. The split in Fig. 1 is essential, because two expected clusters are achieved after the split. Correspondingly, the split in Fig. 2 is inessential, because only an outlier is detected with the split. Intuitively, the number of the proper essential splits is equal to the optimal number of clusters minus 1. Accordingly, the task of finding the optimal number of clusters is equivalent to that of determining the number of essential splits. By inspecting the split process, we find that the difference of sum-of-errors between an essential split and the prior inessential split has a local maximum. We call the local maximum a peak. Then the problem of determining the essential splits can be transformed to that of detecting the peaks.

As what we observed, peaks generally become lower with the split process going on. Under this circumstance, the sliding averages are employed to detect the peaks.

Definition 7. Let $J_e(i)$ be the sum-of-error after ith bipartition:

$$J_e(i) = \sum_{j=0}^{i} J_{ce}(C_j) \quad (11)$$

where $0 \leqslant i \leqslant N-1$, $J_{ce}(C_i)$ is an effective error of each cluster defined as

$$J_{ce}(C_i) = \sum_{\boldsymbol{x} \in C_i} \|x - \mu(C_i)\| \quad (12)$$

Definition 8. Let Diff be a list that consists of the differences between neighboring sum-of-errors:

$$\text{Diff} = \langle d_1, d_2, \ldots, d_i \rangle \quad (13)$$

where $1 \leqslant i \leqslant N-1$, $d_i = J_e(i-1) - J_e(i)$.

Fig. 3a illustrates the bipartitions (only first two) of the data set in Fig. 2. Fig. 3b shows $J_e(i)$ of each bipartitions. It seems difficult to perceive some information to detect cluster number K only from Fig. 3b. But in Fig. 3c two points A, B, which correspond to $J_e(0) - J_e(1)$ and $J_e(1) - J_e(2)$, respectively, are very different from the remaining points, because the first two bipartitions lead to large decreases of J_e.

Definition 9. Let $P = \langle p_0, p_1, \ldots, p_j, \ldots, p_{t-1} \rangle$ be a peak list, $p_j \in \{d_i : d_i$ is an element of Diff$\}$.

Note that if p_{j-1} and p_j in P correspond to d_u and d_v in Diff, respectively, then $v > u$ holds. Obviously, the following fact exists:

Fact 1: If the peak list P has t elements, the optimal cluster number K should be $t+1$.

The task of this phase is to construct the peak list P.

Suppose that an element of Diff, say d_j, is selected as an element of P, say p_m, if the following holds:

$$d_j \geqslant \lambda \,\text{avg}\, d(m) \quad (14)$$

where λ is a parameter; avg$d(m)$ is the average of the elements between d_e and d_j in Diff, d_e corresponds to the prior peak in P, namely, p_{m-1}. Two exceptions exist. When d_j is next to d_e in Diff, i.e. $j = e+1$, no elements exist between d_e and d_j; when p_m is first element in P, i.e. $m = 0$, d_e does not exist. As remedies, the previous average avg$d(m-1)$ is used instead of avg$d(m)$ in Eq. (14) for the former exception and the global average for the latter. Consequently, the sliding average is defined as

$$\text{avg}d(m) = \begin{cases} \frac{1}{j-e-1} \sum_{f=e+1}^{j-1} d_f & : \text{if } m \neq 0 \text{ and } j > e+1 \\ \text{avg}d(m-1) & : \text{if } m \neq 0 \text{ and } j = e+1 \\ \frac{1}{N-1} \sum_{f=1}^{N-1} d_f & : \text{if } m = 0 \end{cases} \quad (15)$$

Clearly, the window width of the sliding average is not fixed. In Fig. 3c, when we consider point A and determine whether it is a peak, we will compare its value with global average since currently the peak list P is empty and no sliding average is available.

The parameter λ in Eq. (14) is a crucial factor for detecting the peaks. Different values of λ lead to different peak lists. However, it is difficult to select a proper value for the parameter λ, because it would be different for data sets with different density distributions. Since for a given λ it is computational simple (the computational cost is linear to N) to construct a peak list, we can construct the peak lists greedily with all the possible values of λ. Intuitively, the parameter must be great than 1, and less than the value which results in the whole data set taken as one cluster. Being a real number, the λ can take a small increment, say σ, when we construct the peak lists iteratively.

Consider a function $f : S \to T$ with the Fact 1 in mind. T is a set of possible values of K, while S is a set of possible values of λ. Then $S = \{\lambda : \lambda = \omega + \sigma \times \eta, \lambda < f^{-1}(1), \omega \geqslant 1, \eta \in \mathbb{N}\}$, $T = \{k : 1 \leqslant k \leqslant N, k \in \mathbb{N}\}$, ω is the initial value of λ and σ is the increment in the construction of the peak lists. We will discuss the values of the parameters σ and ω in Section 3.1. Fig. 3d illustrates the graph of function f on the same data set.

In general, the function f is monotonically decreasing. When λ is small, the element number of the corresponding peak list is large. When the element number is greater than the optimal K, some of discovered clusters are spurious.

Definition 10. Let LBD(S) be a list of binary relations of $(\lambda, f(\lambda))$ defined as

$$\text{LBD}(S) = \langle (\min_{\lambda \in S}(\lambda), f(\min_{\lambda \in S}(\lambda))) \circ \text{LBD}(S - \{\min_{\lambda \in S}(\lambda)\}) \rangle \quad (16)$$

The list LBD(S) collects all binary relations of $(\lambda, f(\lambda))$ in an ascending order with respect to λ. In next subsection, we will consider how to eliminate the spurious clusters and consequently discover optimal cluster number K from the list LBD(S).

2.3. Eliminating the spurious clusters

By inspecting the list LBD(S), we find that some different λs produce the same number of clusters K. We call this local stability. Suppose λ_i and λ_j are the first and the last value, respectively, that lead to $f(\lambda_i) = f(\lambda_j) = K'$ in LBD(S). The local stability can be mea-

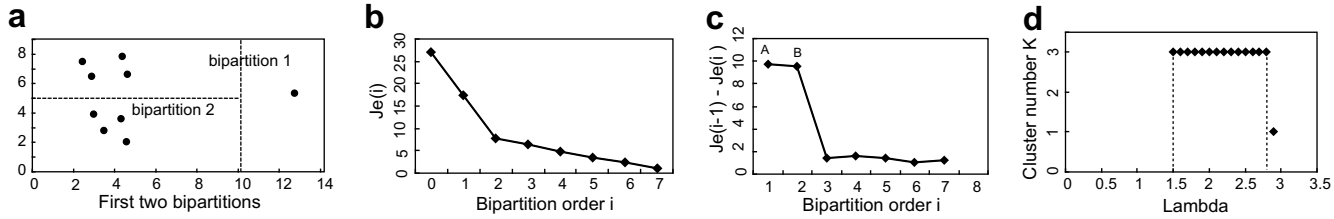

Fig. 3. Illustration of the clustering process. (a) The bipartition process of data set in Fig. 2; (b) the eight J_es for total eight bipartitions; (c) the differences of neighboring J_es, the points A and B are two potential peaks; and (d) the graph of function f.

sured by $\Delta\lambda = \lambda_j - \lambda_i$. In Fig. 3d, $f(1.5) = f(2.8) = 3$, and $\Delta\lambda = 1.3$. If the $\Delta\lambda$ is great than a threshold, say γ, the corresponding K' is regarded as a candidate of K. However, some spurious clusters may exist for a candidate of K. There are two kinds of spurious clusters. The one are the clusters consisting of outliers, the other one are the clusters partitioned from the normal clusters when K' is great than the real number K. Compared to a normal cluster, in general, a spurious cluster consists of a small number of objects and has a small effective error J_{ce} no matter how dense or sparse it is. Under this observation, we define a criterion to identify the spurious clusters.

Note that after the divisive algorithm (DA) is applied, every object is a cluster. Suppose K' is a candidate of K, the $(K'-1)$th peak in P corresponds to d_j in Diff. When we detect spurious clusters from K' clusters, the last $N-j-1$ partitioning operations are not needed. Otherwise, every cluster contains one object, it is meaningless to detect spurious clusters under this situation. For instance, considering $K' = 3$ in Fig. 3d. The 2th peak in Fig. 3c corresponds to d_2 (this is a special case, here $j = K' - 1 = 2$, generally $j \geqslant K' - 1$). When we determine that if there exist spurious clusters in the $K' = 3$ clusters which are produced by the first two bipartition (Fig. 3a), we need the object number and J_{ce} of the three clusters, but these information is not available after complete partitioning. There are two solutions for this problem. One is that the last $N-j-1 = 6$ partitions are rolled back, the other one is to record all needed information in the process of bipartitioning. For the sake of decreasing space complexity, we employ the first one, that is, the last $N-j-1$ partitioning operations are undone.

Definition 11. Let SC be a set that consists of the K' clusters, $Q(\mathrm{SC}) = \langle q_0, q_1, \ldots, q_i, \ldots, q_{K'-1}\rangle$ be an ordered list, where q_i $(0 \leqslant i \leqslant K'-1)$ is the product of the cluster number and the sum-of-error with respect to a cluster in SC, $Q(\mathrm{SC})$ is defined to be

$$Q(\mathrm{SC}) = \langle \min_{C_i \in \mathrm{SC}}(|C_i| \times J_{ce}(C_i)) \circ Q(\mathrm{SC} - \{\arg\min_{C_i \in \mathrm{SC}}(|C_i| \times J_{ce}(C_i))\})\rangle \tag{17}$$

The criterion of identifying spurious cluster is given as follows. Suppose m spurious clusters exist in SC. Because a spurious cluster comprises a small number of objects and has a small effective error, the m spurious clusters are corresponding to the first m elements in $Q(\mathrm{SC})$, namely, $q_0, q_1, \ldots, q_{m-1}$. The element q_{m-1} must satisfy:

1. $q_m \geqslant \alpha q_{m-1}$,
2. If the cluster C_{m-1} in SC is corresponding to q_{m-1} in $Q(\mathrm{SC})$, then $\max_{C_i \in \mathrm{SC}}|C_i| > \beta|C_{m-1}|$,

where α and β are two real number parameters.The criterion includes the above two conditions, which are similar to the definition of large or small clusters (He et al., 2003). The first condition indicates a relative change ratio of the normal cluster and the spurious cluster near the boundary. The second one is an absolute constraint on spurious clusters.When we apply the criterion to SC, the spurious clusters are detected, but this is not final result since there may exist a number of candidates of K. As we know, a candidate of K and the corresponding set SC is decided by λ. We repeatedly increase λ by the step of σ and apply the criterion until it converges. For a candidate of K, K'_i, suppose that s_i spurious clusters are detected from K'_i clusters in terms of the criterion, the next candidate of K is K'_{i+1}, then the convergence is defined as:

1. according to the criterion of identifying spurious cluster, no spurious cluster in SC is detected, i.e. $s_i = 0$, or
2. after the spurious clusters being removed from SC, the number of normal clusters is equal to or greater than the next candidate of K, i.e. $K'_i - s_i \geqslant K'_{i+1}$, or
3. the candidate K'_{i+1} does not exist, i.e. K'_i is the last candidate of K.

For the first situation, all clusters in SC have relatively consistent object number and J_{ce}. For the second situation, if $K'_i - s_i < K'_{i+1}$, some of spurious clusters still exist in the K'_{i+1} clusters, and we must continue to consider the candidate K'_{i+1}; otherwise, all of spurious clusters are excluded from K'_{i+1} clusters, and it is meaningless to consider K'_{i+1} for removing spurious clusters. The last situation is obvious. Based on the definition of convergence, the spurious clusters detection mechanism (SCD) is formally presented as follows. The parameters α, β, γ will be discussed in Section 3.1.

Spurious clusters detection algorithm (SCD)

Step 1. Scan the list LBD(S) from the left to the right. Find out the pair of λ, λ_i and λ_j, which satisfy:
  (1) $f(\lambda_i) = f(\lambda_j)$, subject to $i < j$, $f(\lambda_i) > f(\lambda_{i-1})$ and $f(\lambda_j) < f(\lambda_{j+1})$;
  (2) $\Delta\lambda > \gamma$, where $\Delta\lambda = \lambda_j - \lambda_i$;
Step 2. Construct the set SC which consists of K' clusters, and $Q(\mathrm{SC})$, where $K' = f(\lambda_i) = f(\lambda_j)$;
Step 3. Determine the spurious clusters according to the spurious cluster criterion;
Step 4. If the convergence is achieved, stop; otherwise go to step 1.

3. Numerical experiments

3.1. Setting parameters

From Eq. (14), the parameter λ directly determines the class number K. Since discovering the number K automatically is one of our objectives, the proper selected λ is vital for us. Greedily, we inspect all possible values of λ to find optimal K. The parameter ω decides the start point of λ. It is meaningless to assign λ a value less than or equal to 1, because each one object will be a cluster in this situation. For covering more λs, we can easily set a value to ω, usually a value between 1.2 and 2.0 is selected. In LBD(S), the parameter σ adjusts the interval between the previous λ and the next one, and consequently determines different change rate of the number K. Generally, if the interval is small, the number K will change slowly. However, a too small interval will be not helpful to reflect the change of K but result in time-consuming. The parameter

σ is set to 0.1 in all experiments. Our experiments indicate that if it is set to a smaller value, for instance 0.05, the clustering result does not change at all. The parameter γ is always 0.5. Empirically, the two parameters α and β are set to 2 and 3, respectively. In (He et al., 2003), large clusters are defined as those containing 90% of objects, or the minimum ratio of the object number of a cluster in the large group to that of a cluster in small group is 5. In our divisive method, however, the objects in the outer shell of clusters are peeled off gradually in the process of partition. Only the kernels remain. In general, the difference of the object numbers of two kernels is not as large as that of the object numbers of the two corresponding clusters. So the two parameters are relatively small. All the parameters remain unchanged in the following four experiments.

3.2. Comparing DIVFRP with other methods

In this section, two performances of DIVFRP will be compared to other algorithms. The one is the difference between the discovered cluster number K and the real K. The other is how the partitions produced by the algorithms are consistent with the real partitions. To measure the consistency, three external validity criteria, namely Fowlkes and Mallows (FM) index, Jaccard coefficient and Rand statistic, are employed. However, the three external criteria do not consider the outliers. In order to use the criteria, we redistribute each object partitioned as an outlier in DIVFRP into the corresponding cluster of which the mean is closest to the object. The two well-known clustering algorithms DBScan and K-means are selected for the comparison. To obtain the cluster number K with the two algorithms, we combine DBScan and K-means with two relative validity criteria: Calinski–Harabasz (CH) index and Silhouette index. For a same clustering method with the different input parameters, the best clustering scheme can be determined by the relative criteria (Halkidi et al., 2002), therefore the optimal cluster number K is discovered. We can change the input parameters Eps and MinPts for DBScan and the cluster number K for K-means to produce the best clustering scheme. Note that DBScan can produce outliers too. Similarly, the outliers are redistributed into the nearest clusters for the purpose of comparison.

CH Index is defined as follows

$$\mathrm{CH} = \frac{Tr(B)/(K-1)}{Tr(W)/(N-K)} \tag{18}$$

where K is the cluster number and N is the object number, and

$$Tr(B) = \sum_{i=1}^{K} |C_i| \times \|\mu(C_i) - \mu(C)\|^2 \tag{19}$$

where $\mu(C) = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i$;

$$Tr(W) = \sum_{k=1}^{K} \sum_{i=1}^{|C_i|} \|\boldsymbol{x}_i - \mu(C_i)\|^2 \tag{20}$$

Silhouette index is defined as follows

$$S = \frac{1}{N}\sum_{i=1}^{N} \frac{b_i - a_i}{\max(a_i, b_i)} \tag{21}$$

where K is the cluster number and N is the object number and

$$a_i = \frac{1}{|C_j| - 1} \sum_{\boldsymbol{y}\in C_j,\ \boldsymbol{y}\neq \boldsymbol{x}_i} \|\boldsymbol{y} - \boldsymbol{x}_i\| \tag{22}$$

$$b_i = \min_{l\in H,\ l\neq j} \frac{1}{|C_l|} \sum_{\boldsymbol{y}\in C_l} \|\boldsymbol{y} - \boldsymbol{x}_i\| \tag{23}$$

where $\boldsymbol{x}_i \in C_j,\ H = \{h : 1 \leqslant h \leqslant K,\ h \in \mathbb{N}\}$.

First we demonstrate the performances of DIVFRP with two synthetic 2D data sets R15 and D31 (Veenman et al., 2002) of which the coordinates are transformed linearly and slightly. The data set R15 which is shown in Fig. 4a has 15 Gaussian clusters which are arranged in rings, and each cluster contains 40 objects. Fig. 4b shows the clustering result of DIVFRP with the outliers redistributed into the nearest clusters. Fig. 5 shows the curve of the cluster number K and the λ based on LBD(S). Clearly, only horizontal line segments in the curve may denote the candidates of K, because a horizontal line segment denotes $f(\lambda_i) = f(\lambda_j) = K'$, where (λ_i, K') and (λ_j, K') are the two endpoints of the line segment. In general, the first candidate of K lies on the knee of the curve. In Fig. 5, the first horizontal line segment $\overline{AB}$ is selected to produce the initial candidate of K, as the length of horizontal line segment (namely $\Delta\lambda$) is 12.7 and much greater than γ. The K' determined by

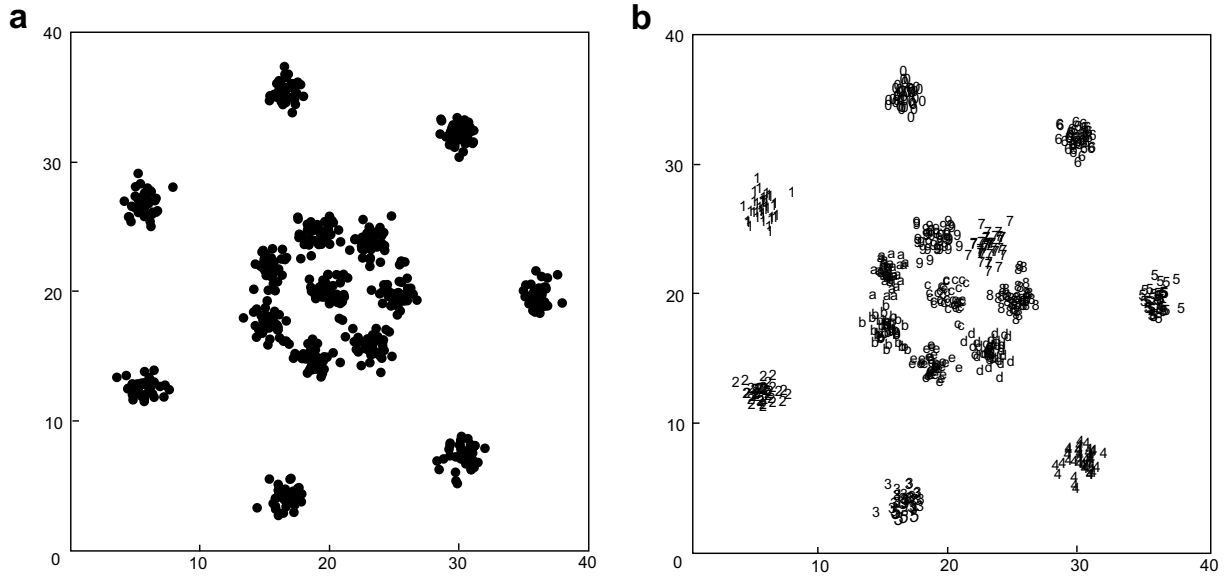

Fig. 4. (a) The original data set of R15 and (b) the clustering result of DIVFRP for R15 with the outliers redistributed into the closest clusters.

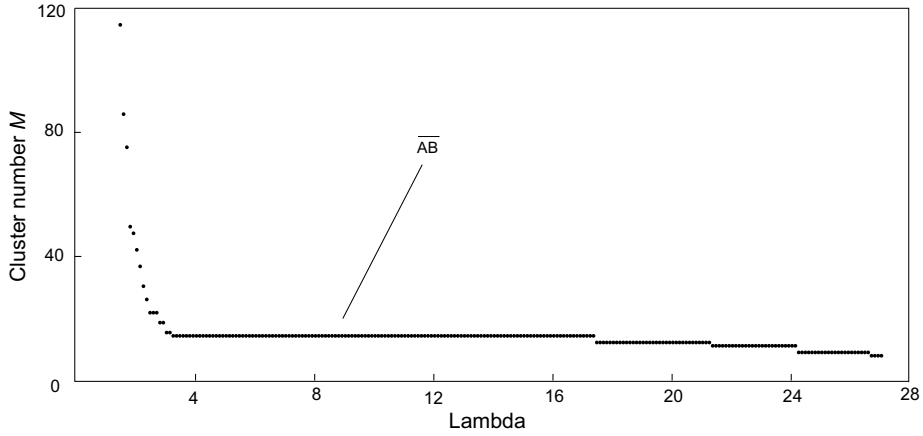

Fig. 5. The curve of λ and the cluster number K for the data set R15. The horizontal line segment $\overline{AB}$ is qualified to produce a candidate of K, and the algorithm SCD converges at this line segment.

$\overline{AB}$ is 15. According to the criterion of spurious cluster, no spurious cluster exists in the corresponding $Q(SC)$. Consequently, the spurious clusters detection algorithm (SCD) converges after first iteration, and the final cluster number K is 15. Then we consider DBScan algorithm. To achieve optimal clustering result, we run DBScan algorithm repeatedly with the change of the parameter Eps from 0.1 to 10 and the parameter MinPts from 5 to 50. Two increments for the changes are 0.1 and 1, respectively. For K-means algorithm, we change the parameter K from 2 to $N-1$ with increment of 1 in all experiments. Even for a same parameter K, the clustering results of K-means probably are quit different because of the initial cluster centers selected randomly. Accordingly, we consider the average performance of 10 runs of K-means in all the comparisons. CH index and Silhouette index are computed in each run of both DBScan and K-means algorithms. The optimal results have maximum CH index or Silhouette index values. Table 1 provides the comparison of DIVFRP, DBScan and K-means. The comparison focuses on two performances mentioned above, namely the cluster number K and partition consistency. Clearly, DIVFRP, CH index-based DBScan and Silhouette index-based DBScan achieve same perfect results. They discover the cluster number K correctly, and three external indices are close to the expected value 1. However, because K-means algorithm may converge at local minimum, both CH index-based K-means and Silhouette index-based K-means have unsatisfied results.

Table 1
Performances of applying the methods to R15 data set

Method	Estimated K	Rand	Jaccard	FM
DIVFRP	15	0.9991	0.9866	0.9932
DBScan–CH	15	0.9991	0.9866	0.9932
DBScan–Silhouette	15	0.9991	0.9866	0.9932
K-means–CH	18.7	0.9917	0.8728	0.9333
K-means–Silhouette	18.1	0.9908	0.8688	0.9302

The second synthetic data set D31 comprises 31 Gaussian clusters, and each cluster has 100 objects. Fig. 6a shows the original data set and clustering result of DIVFRP is shown in Fig. 6b. In Fig. 7, the curve of the cluster number and λ is shown. $\overline{AB}$ is the first horizontal line segment with the length greater than γ. As a result, the K' determined by $\overline{AB}$ is the initial candidate of K. The succeeding horizontal line segments $\overline{CD}$ and $\overline{EF}$ are qualified to produce the

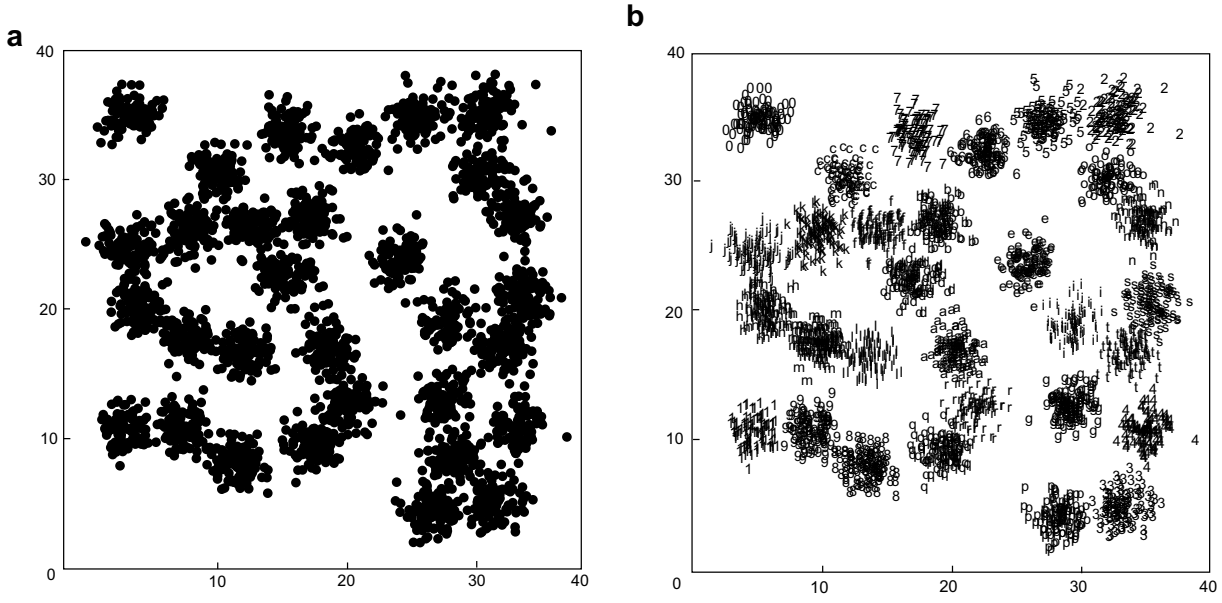

Fig. 6. (a) The original data set of D31 and (b) the clustering result of DIVFRP for D31 with the outliers redistributed into the closest clusters.

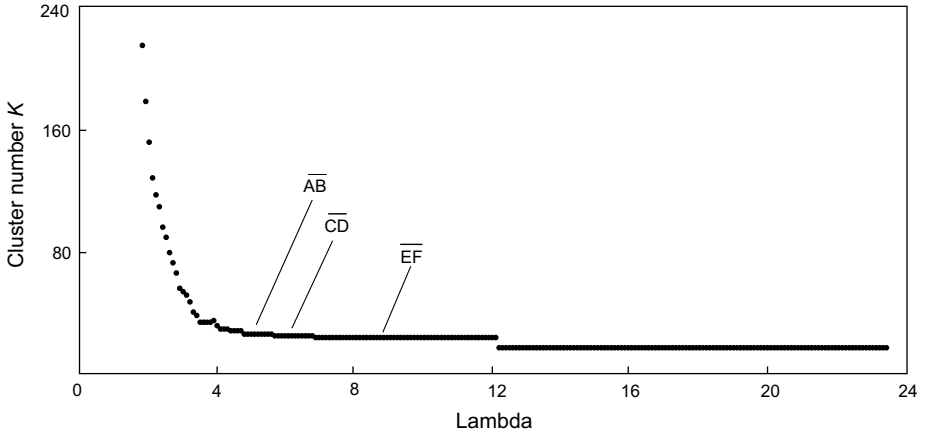

Fig. 7. The curve of λ and the cluster number K for the data set D31. The horizontal line segment $\overline{AB}$, $\overline{CD}$ and $\overline{EF}$ are qualified to produce the candidates of K. The algorithm SCD converges at line segment $\overline{EF}$.

candidates of K, since the lengths of the two line segments are 1.2 and 54, respectively. We apply SCD to the candidates of K. For the first $K' = 33$ denoted by $\overline{AB}$, there are two spurious clusters detected. We move to the line segment $\overline{CD}$, the corresponding K' is 32. Similarly, one spurious cluster is discerned by SCD. Then we consider the line segment $\overline{EF}$ with $K' = 31$. Since this iteration of SCD gives no spurious cluster, SCD converges at this line segment, and accordingly the final number of clusters is 31. As for DBScan, we set the parameter Eps on D31 from 0.1 to 5.0 with increment 0.1, MinPts from 10 to 40 with increment 1, and run it repeatedly. From the Table 2, we observe that both DIVFRP and DBScan determine the cluster number K precisely, but K-means algorithms do not. Moreover, the three external indices demonstrate that the partition qualities of the two DBScan algorithms are slimly better than that of DIVFRP, and at the same time the Silhouette index-based DBScan outperforms CH index-based DBScan slightly.

Next, we experiment on two real data sets IRIS and WINE. The well-known data set IRIS has three clusters of 50 objects each, and each object has four attributes. The curve about K and λ, which is produced by applying DIVFRP to IRIS data set, is displayed in Fig. 8a. Both horizontal line segment $\overline{AB}$ and $\overline{CD}$ in the curve have the length of 1.3 which is greater than γ. Therefore, the two K's denoted by $\overline{AB}$ and $\overline{CD}$, respectively, are the two candidates of K. As SCD identifies one spurious cluster from K' ($K' = 4$) clusters corresponding to $\overline{AB}$ and no spurious cluster from K' ($K' = 3$) clusters corresponding to $\overline{CD}$, the final cluster number is determined to 3.

Table 2
Performances of applying the methods to D31 data set

Method	Estimated K	Rand	Jaccard	FM
DIVFRP	31	0.9969	0.9083	0.9520
DBScan–CH	31	0.9970	0.9116	0.9537
DBScan–Silhouette	31	0.9971	0.9127	0.9543
K-means–CH	38.9	0.9909	0.7442	0.8541
K-means–Silhouette	28.7	0.9803	0.5854	0.7469

Table 3
Performances of applying the methods to IRIS data set

Method	Estimated K	Rand	Jaccard	FM
DIVFRP	3	0.8859	0.7071	0.8284
DBScan–CH	3	0.8797	0.6958	0.8208
DBScan–Silhouette	2	0.7762	0.5951	0.7714
K-means–CH	2.9	0.7142	0.4768	0.6570
K-means–Silhouette	2	0.7636	0.5723	0.7504

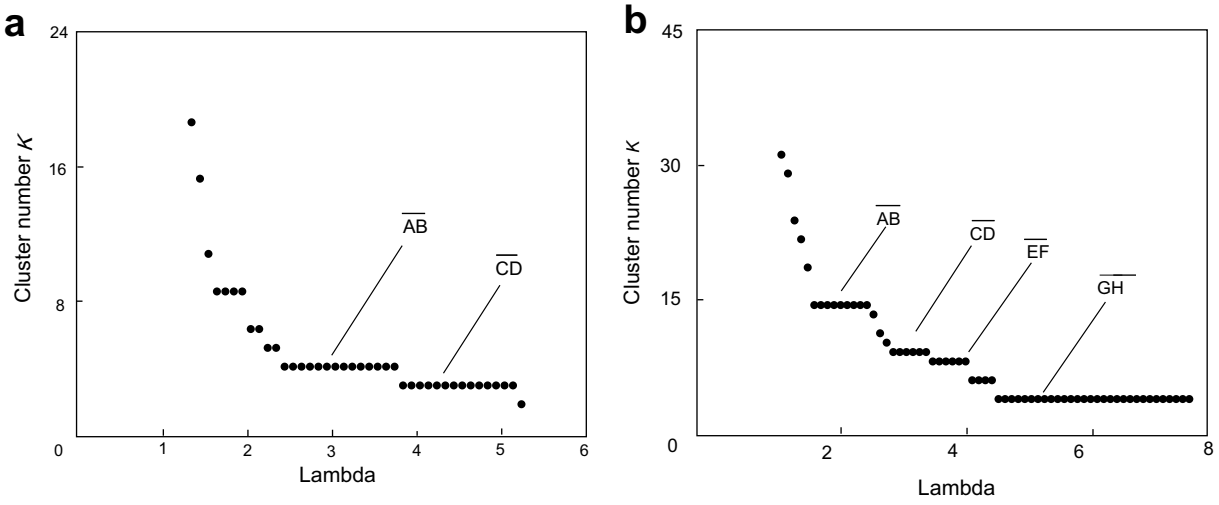

Fig. 8. (a) The curve of λ and the cluster number K for the data set IRIS. The algorithm SCD converges at line segment $\overline{CD}$. (b) The curve of λ and the cluster number K for the data set WINE. The algorithm SCD converges at line segment $\overline{EF}$.

Table 4
Performances of applying the methods to WINE data set

Method	Estimated K	Rand	Jaccard	FM
DIVFRP	3	0.7225	0.4167	0.5883
DBScan–CH	6	0.7078	0.2656	0.4467
DBScan–Silhouette	2	0.6836	0.4623	0.6474
K-means–CH	40.6	0.6660	0.0453	0.1639
K-means–Silhouette	2	0.6688	0.4638	0.6549

Table 3 depicts the comparison of the performances of the algorithms. The parameter ranges and increments of DBScan for IRIS data set are the same as for D31 data set. DIVFRP and CH index base DBScan achieve similar results and outperform the three remaining methods.

The data set WINE contains the chemical analyses of 178 kinds of wines from Italy on 13 aspects. The 178 entries are categorized into three clusters with 48, 59, and 71 entries for each cluster. SCD

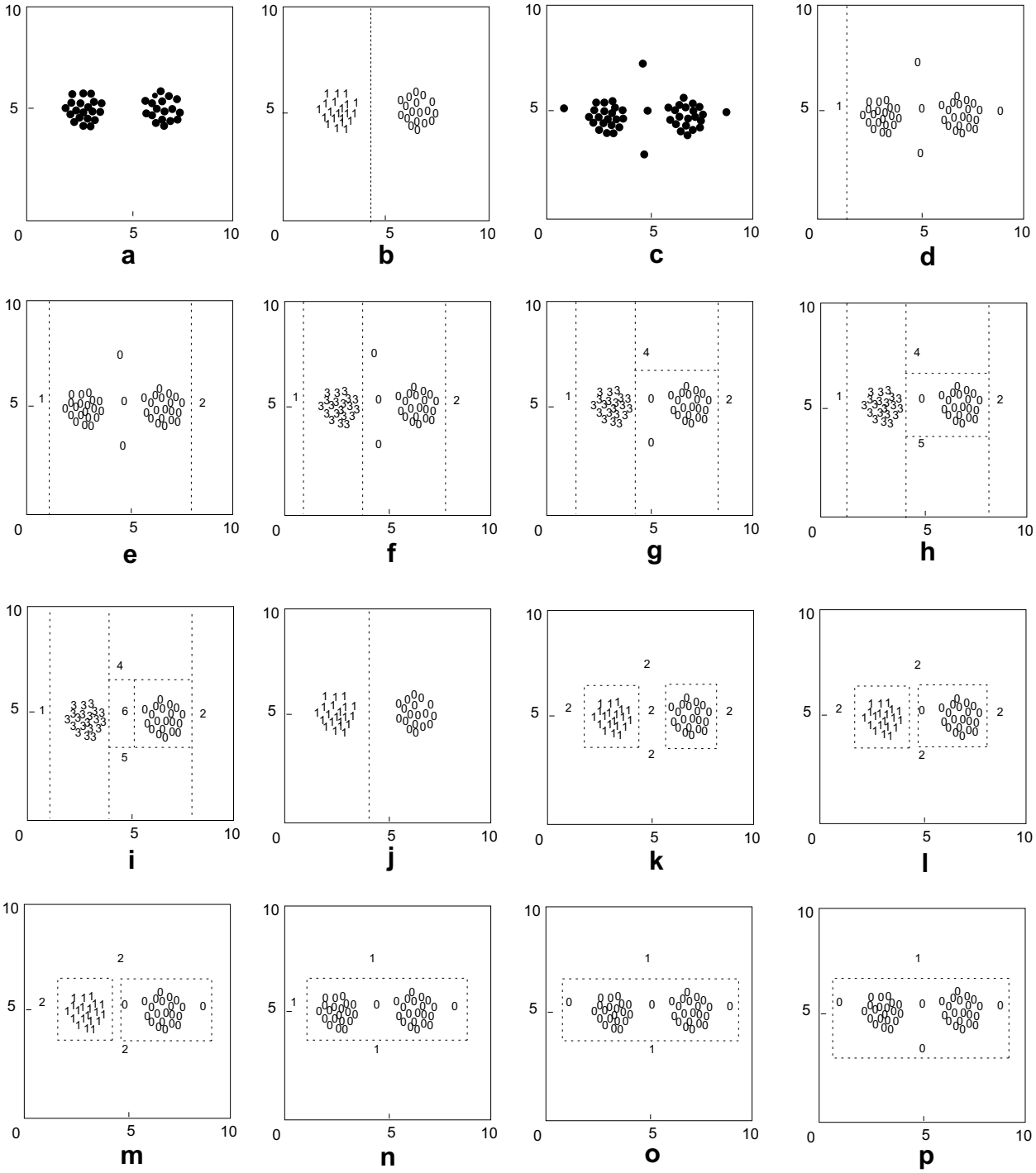

Fig. 9. The figures illustrate the effect of outliers on DIVFRP and DBScan. In (a), a data set contains two clusters. In (b), the first partition of DIVFRP identifies the two clusters. In (c), a data set contains five outliers and two clusters. In (d, e) and (g–i), each partition peels off one outlier, while an expected cluster is partitioned in Fig. 9f. In (j), DBScan detects the two clusters with MinPts = 5 and $0.60 \leqslant \text{Eps} \leqslant 2.10$. In (k), the outliers are identified by DBScan with MinPts = 5 and $0.60 \leqslant \text{Eps} \leqslant 1.01$. In (l), DBScan with MinPts = 5 and $1.02 \leqslant \text{Eps} \leqslant 1.20$. In (m), DBScan with MinPts = 5 and $1.21 \leqslant \text{Eps} \leqslant 1.24$. In (n), DBScan with MinPts = 5 and $1.25 \leqslant \text{Eps} \leqslant 1.29$. In (o), DBScan with MinPts = 5 and $1.30 \leqslant \text{Eps} \leqslant 1.72$. In (p), DBScan with MinPts = 5 and $1.73 \leqslant \text{Eps} \leqslant 2.10$.

converges at the horizontal line segment $\overline{GH}$ to which the corresponding K' is 4, see Fig. 8b. Although SCD identifies a spurious cluster, $K' = 4$ is the last candidate of K in this data set. In terms of item three of convergence criterion, SCD converges and the final estimated number of clusters is $K = K' - 1 = 3$. In Table 4, only DIVFRP estimates the cluster number correctly. Silhouette index-based DBScan and K-means achieve better results than CH index-based DBScan and K-means.

4. Discussion

From the four experiments, we observe that DIVFRP figures out the optimum cluster number and achieves good results compared to validity indices-based DBScan and K-means. In addition, DIVFRP is robust to outliers.

Outliers in a data set are those random objects that are very different from others and do not belong to any clusters (Lin and Chen, 2005). In general, the presence of outliers may deteriorate the result of a clustering method. For this kind of clustering methods, some outlier detection mechanisms can be employed to remove the outliers before the partition process. However, a clustering method robust to outliers is more expected. DIVFRP is equipped with this property. Fig. 9 shows the partition processes of DIVFRP on two similar data sets. The one data set contains two clusters only and is shown in Fig. 9a. The other one combines the two clusters with another five outliers and is shown in Fig. 9c. For the data set in Fig. 9a, one split is enough for partitioning the two clusters with DIVFRP, see Fig. 9b. When there exist extra five outliers in the data set, six splits are needed, which are depicted from Fig. 9d to Fig. 9i. Actually, the effect of five splits of the six is to peel off the five outliers. In other words, the existence of outliers does not change final clustering result but only leads to more partitions, and none parameters are involved in the whole partitioning process. Note that some adjacent outliers peeled off at one partition may be regarded as a cluster in the peak detection process. However, it can be removed as a spurious cluster in SCD. Fig. 9j–p illustrates the clustering of DBScan on the same data sets. When no outliers exist, in Fig. 9j, DBScan can discover the two clusters with MinPts $= 5$ and $0.60 \leqslant \text{Eps} \leqslant 2.10$. For the data set with five outliers, although DBScan can still discover the two clusters with MinPts $= 5$ in Fig. 9k, the range of possible values of Eps is reduced to $[0.60, 1.01]$. Fig. 9l–p shows that when MinPts keeps unchanged, the remaining values of Eps, $[1.02, 2.10]$, result in poor clusters. The experiments on DBScan indicate that the parameters of DBScan are sensitive to outliers. Therefore, compared with DBScan, we can say that DIVFRP is robust to outliers.

Generally, the computational cost of a divisive clustering method is very expensive, because there are $2^{n_i-1} - 1$ possibilities to bipartition a cluster C_i with n_i objects in a divisive algorithm. However, DIVFRP employs furthest reference points to bipartition a cluster optimally, and does not need to consider the complete enumeration of all possible bipartitions. With this property, the computational cost of the partition scheme of DIVFRP is lower than that of general divisive clustering methods, even some agglomerative methods. For a data set with N objects, for example, the computational cost of DIVFRP is lower than that of single-linkage ($O(N^2 \log N)$). Because in DIVFRP, totally there are $N - 1$ bipartitions after every object becomes a cluster, and the computational cost of each bipartition is $O(n_i \log n_i)$, where n_i is the object number of a cluster to be bipartitioned and it is less than N except the first bipartition in which n_i is N.

However, DIVFRP has some drawbacks. It can only find out clusters in spherical shape. This drawback results from that DIVFRP uses Euclidean distance and takes the furthest points as reference points. With different reference point, clusters in different shapes may be detected. For example, if we take the mean points as reference points instead of the furthest points, the clusters in ring shape with close centers can be identified. Another drawback is that when some valid clusters are quit dense, they may be mis-detected as spurious clusters. If lots of objects locate at an almost identical position, their $|C_i| \times J_{ce}(C_i)$ would be very small, which is employed to construct $Q(\text{SC})$. In the future work, we will explore and improve the criterion of detecting spurious cluster to overcome this drawback. In addition, similar to classical hierarchical clustering methods, DIVFRP is also incapable of dealing with large-scale data sets because of its quadratic computational cost. For high-dimensional data sets, distance-based clustering algorithms are ineffective (Xu and Wunsch, 2005). DIVFRP is very a distance-based algorithm, but we can employ dimensionality reduction methods to preprocess high-dimensional data, and then apply DIVFRP to the dimensionality reduced data sets.

5. Conclusion

In this paper, we present an automatic divisive hierarchical algorithm based on furthest reference points (DIVFRP). It contains three phases: partitioning data set, detecting the peaks of sum-of-error differences and eliminating spurious clusters. In partitioning data set phase, other than general divisive hierarchical clustering algorithms, DIVFRP employs a novel dissimilarity function which takes the furthest points as reference points. With the dissimilarity function, the computational cost of partitioning data set is less than $O(N^2 \log N)$. Sliding average is used to detect the peaks in second phase. In the third phase, the spurious clusters are removed and the optimal cluster number K is determined. The experiments on both artificial and real data sets show that DIVFRP can automatically and precisely detect the number of clusters and achieve a good clustering quality. In addition, the presence of outliers does not degrade the quality of clustering results, since DIVFRP can peel off the outliers bit by bit.

Acknowledgements

The paper is supported by the National Natural Science Foundation of China, Grant No. 60775036, and Ningbo University, Grant No. 200460. The authors are grateful to Jian Yu, Cor J. Veenman and Yuntao Qian for their valuable comments and suggestions.

References

Aggarwal, C., Yu, P., 2001. Outlier detection for high dimensional data. In: Proc. SIGMOD'01, Santa Barbara, CA, USA, pp. 37–46.

Bandyopadhyay, S., Maulik, U., 2001. Nonparametric genetic clustering: Comparison of validity indices. IEEE Trans. Systems Man Cybernet. – Part C: Appl. Rev. 31 (1).

Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J., 2000. LOF: Identifying density-based local outliers. In: Proc. SIGMOD'00, Dallas, Texas, pp. 427–438.

Chavent, M., Lechevallier, Y., Briant, O., 2007. DIVCLUS-T: A monothetic divisive hierarchical clustering method. Comput Statist. Data Anal. 52 (2), 687–701.

Ester, M., Kriegel, H.P., Sander, J., Xu, X.,1996. A density based algorithm for discovering clusters in large spatial databases. In: Proc. KDD'96, Portland OR, USA, pp. 226–231.

Garai, G., Chaudhuri, B.B., 2004. A novel genetic algorithm for automatic clustering. Pattern Recognition Lett. 25, 173–187.

Gowda, K.C., Ravi, T.V., 1995. Divisive clustering of symbolic objects using the concept of both similarity and dissimilarity. Pattern Recognition 28 (8), 1277–1282.

Guha, S., Rastogi, R., Shim, K., 1998. CURE: An efficient clustering algorithm for large databases. In: Proc. ACM SIGMOD Internat. Conf. Management Data. ACM Press, New York, pp. 73–84.

Halkidi, M., Batistakis, Y., Vazirgiannis, M., 2002. Clustering validity checking methods: Part II. ACM SIDMOD Record 31 (3), 19–27.

He, Z., Xu, X., Deng, S., 2003. Discovering cluster-based local outliers. Pattern Recognition Lett. 24, 641–1650.

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

Kaufman, L., Rousseeuw, P.J., 1990. Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, New York.

Knorr, E.M., Ng, R.T., 1998. Algorithms for mining distance based outliers in large datasets. In: Proc. VLDB'98, New York, USA, pp. 392–403.

Lin, C.R., Chen, M.S., 2005. Combining partitional and hierarchical algorithms for robust and efficient data clustering with cohesion self-merging. IEEE Trans. Knowledge Data Eng. 17 (2), 145–159.

MacQueen, J.B., 1967. Some methods for classification and analysis of multivariate observations. In: Proc. 5th Berkeley Symposium Mathematical Statistical Probability, vol. 1, pp. 281–297.

Ng, R.T., Han, J., 1994. Efficient and effective clustering methods for spatial data mining. In: Bocca, J.B., Jarke, M., Zaniolo, C. (Eds.), Proc. 20th Internat. Conf. on Very Large Data Bases (VLDB'94). Morgan Kaufmann, Santiago, pp. 144–155.

Patan, G., Russo, M., 2002. Fully automatic clustering system. IEEE Trans. Neural Networks 13 (6).

Ramaswamy, S., Rastogi, R., Kyuseok, S., 2000. Efficient algorithms for mining outliers from large data sets. In: Proc. SIGMOD'00, Dallas, Texas, pp. 93–104.

Savaresi, S.M., Boley, D.L., Bittanti, S., Gazzaniga, G., 2002. Cluster selection in divisive clustering algorithms. In: Proc. 2nd SIAM ICDM, Arlington, VA, pp. 299–314.

Theodoridis, S., Koutroumbas, K., 2006. Pattern Recognition, third ed. Academic Press Inc., Orlando, FL.

Tseng, V.S., Kao, C.P., 2005. Efficiently mining gene expression data via a novel parameterless clustering method. IEEE/ACM Trans. Comput. Bioinformatics 2 (4) (October–December).

Tseng, L.Y., Yang, S.B., 2001. A genetic approach to the automatic clustering problem. Pattern Recognition 34, 415–424.

Veenman, C.J., Marcel, J.T., Reinders, M.J.T., Backer, E., 2002. A maximum cluster algorithm. IEEE Trans. Pattern Anal. Machine Intell. 24 (9), 1273–1280.

Wang, X., Qiu, W., Zamar, R.H., 2007. CLUES: A non-parametric clustering method based on local shrinking. Comput. Statist. Data Anal. 52 (1), 286–298.

Xu, R., Wunsch II, D., 2005. Survey of clustering algorithms. IEEE Trans. Neural Networks 16 (3), 645–678.