Int. J. Mach. Learn. & Cyber. (2017) 8:1537–1554
DOI 10.1007/s13042-016-0526-y

ORIGINAL ARTICLE

k-Proximal plane clustering

Li-Ming Liu[1] · Yan-Ru Guo[2] · Zhen Wang[3] · Zhi-Min Yang[2] · Yuan-Hai Shao[2]

Received: 7 March 2015 / Accepted: 22 March 2016 / Published online: 13 April 2016

Abstract Instead of clustering data points to cluster center points in k-means, k-plane clustering (kPC) clusters data points to the center planes. However, kPC only concerns on within-cluster data points. In this paper, we propose a novel plane-based clustering, called k-proximal plane clustering (kPPC). In kPPC, each center plane is not only close to the objective data points but also far away from the others by solving several eigenvalue problems. The objective function of our kPPC comprises the information from between- and within-clusters data points. In addition, our kPPC is extended to nonlinear case by kernel trick. A determinative strategy using a Laplace graph to initialize data points is established in our kPPC. The experiments conducted on several artificial and benchmark datasets show that the performance of our kPPC is much better than both kPC and k-means.

✉ Yuan-Hai Shao
shaoyuanhai21@163.com

Li-Ming Liu
llm5609@163.com

Yan-Ru Guo
Guoyanru211@163.com

Zhen Wang
wangzhen@imu.edu.cn

Zhi-Min Yang
yzm9966@126.com

[1] Capital University of Economics and Business, Beijing 100083, People's Republic of China

[2] Zhijiang College, Zhejiang University of Technology, Hangzhou 310024, People's Republic of China

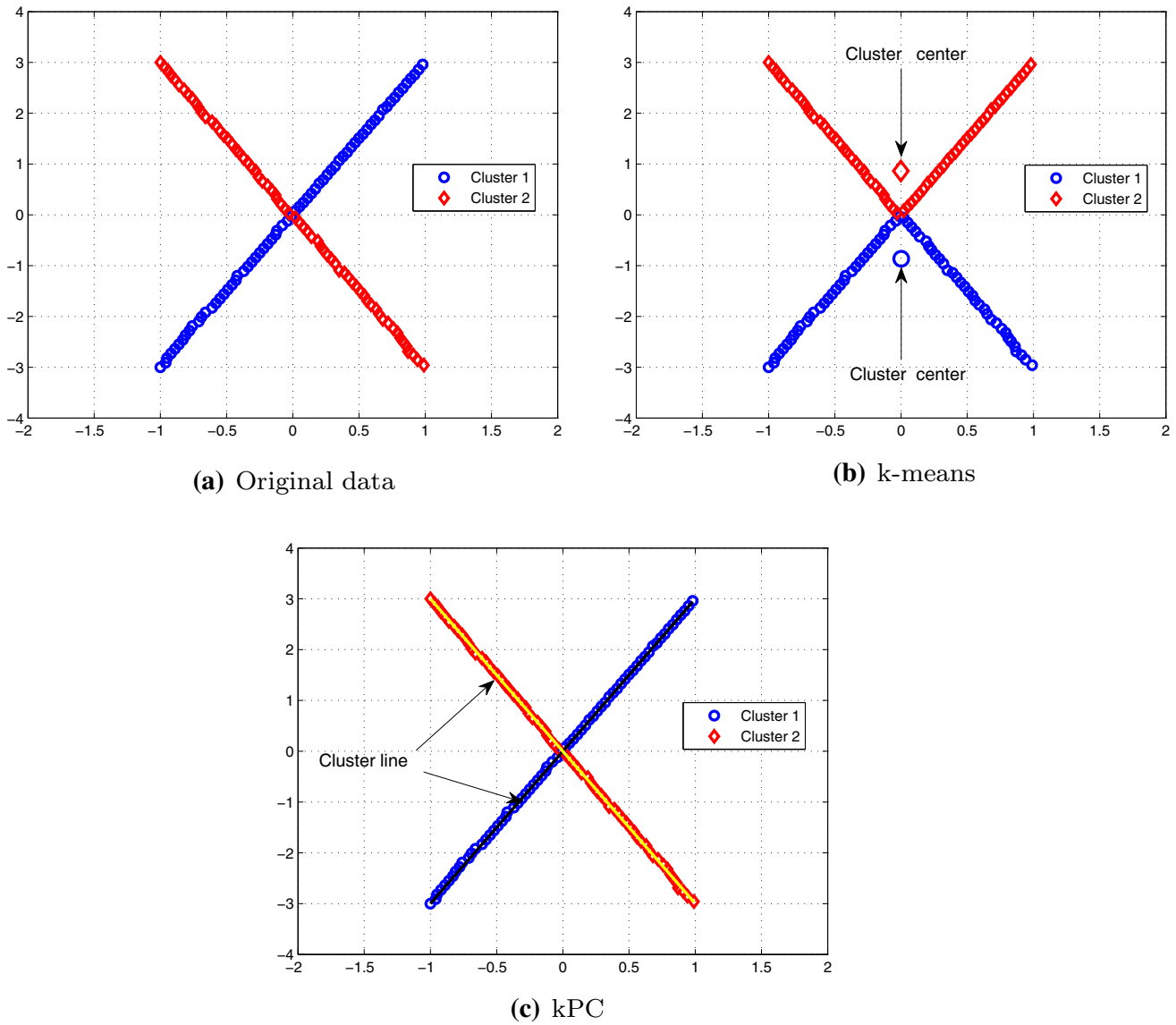
[3] School of Mathematical Sciences, Inner Mongolia University, Hohhot 010021, People's Republic of China

Keywords Clustering · k-means · k-Plane clustering · eigenvalue problem · Laplace graph

1 Introduction

Clustering [1] is the process of grouping data points into clusters, so that data points within a cluster have high similarity and data points are very dissimilar in the different clusters. Clustering has been applied in many areas, including data mining, statistics, biology, and information retrieval [2–4]. There are many methods for clustering, such as partitioning methods [5–7], hierarchical methods [8, 9], and density-based methods [10, 11]. Many clustering methods [5, 6, 10] aim at clustering data points to several clusters by center points, such as k-means [3, 12] and fuzzy C-means [13], which may perform better when data points distribute as spherical shapes. However, data points may not be clustered around a data point, e.g., a toy example shown in Fig. 1a [14], where each cluster's data points distribute along a straight line.

k-plane clustering (kPC) [15, 16] is proposed to handle the case that data points fall into clusters around planes. To obtain the final center planes, kPC alternates between center plane updates and cluster assignments. Therefore, in Figure 1, kPC performs better than k-means. However, kPC only consider the within-cluster compactness may leads difficulties. A toy example is given by Fig. 2, in which kPC constructs two overlapped lines that misclassify the data points of cluster 2 into the cluster 1. Hence, kPC fails to find reasonable clusters and ends up estimating the number of clusters to 2. In addition, selecting the initial data points randomly in both k-means and kPC will affect the cluster results in a great extent [17–19]. A set of poor quality initial seeds may lead to a poor quality clustering result.



(a) Original data

(b) k-means

(c) kPC

Fig. 1 Different performance of k-means and kPC on the artificial data points, which distribute along two cross straight lines. The data points with the same color and shape belong to the same cluster

Recent studies [20–22] show that the between-cluster data points could be used to improve the cluster performance. Therefore, we introduce the between-cluster data points in kPC and propose a novel plane-based clustering, called k-proximal plane clustering (kPPC). Similar to kPC, kPPC also updates the center plane and cluster assignments alternately. The difference is, in the updating process, each center plane is not only close to its data points but also far away from the others, leading to solve several eigenvalue problems. In addition, we extend the above procedure to nonlinear case by kernel trick [14, 23–27]. Compared with kPC, the main advantages of our kPPC are: (1) the center plane in kPPC not only close to its data points but also far away from the others; (2) instead of the random initialization in kPC, kPPC constructs a Laplace graph to give the efficient initial clusters; (3) a nonlinear model of kPPC is also proposed by the kernel trick;

The remainder of this paper is divided into the following parts. Section 2 gives a brief review of the related work. Section 3 presents the details of our kPPC. In Sect. 4, we report the experimental results on both artificial and benchmark datasets. Finally, Sect. 5 concludes the paper.

2 Related works

In this paper, we consider a set of m unlabeled data points in the n-dimensional real space represented by the matrix A in $R^{m\times n}$. For the clustering problem, the purpose is to

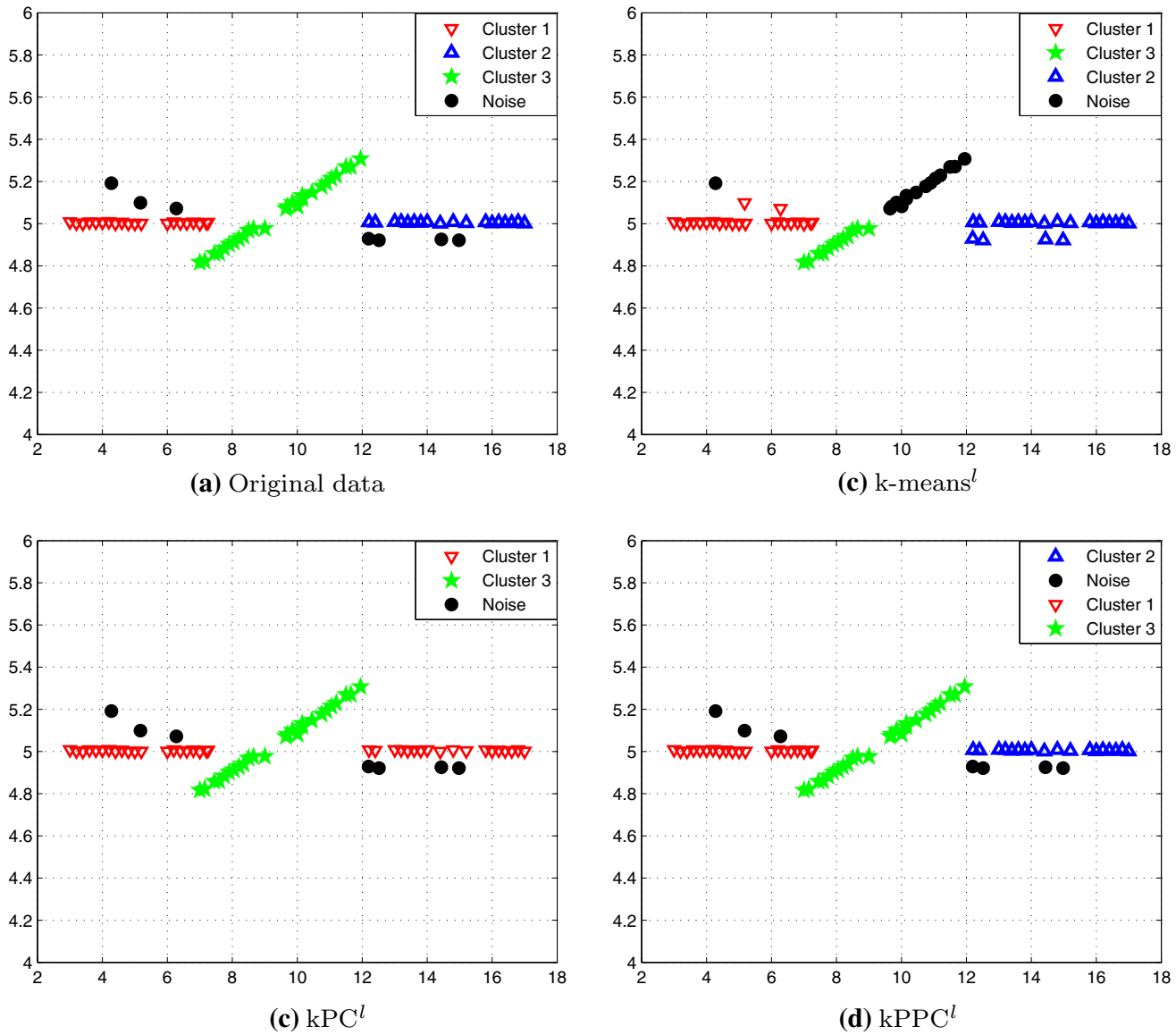


Fig. 2 Different methods on artificial dataset, who's cluster 1 and cluster 2 lie approximately on the same line

cluster A into k clusters with the labels $y_i \in N, i = 1, 2, \ldots, m$ represented by the vector $\mathbf{y} \in N^m$. Next, we briefly review k-means and kPC.

2.1 k-means

k-means [28, 29] wishes to cluster m data points into k clusters so that data points of each cluster lie near the cluster center data point.

k-means keeps each data point from the same cluster being the closest to its cluster center data point, which uses the sum of the squared error to represent the dissimilarity measure by solving the following problem

$$\min_{\mathbf{A}_i} \sum_{i=1}^{k} \sum_{\mathbf{x} \in \mathbf{A}_i} ||\mathbf{x} - \mathbf{A}_i||_2^2, \tag{1}$$

where $\mathbf{x}$ is the data point in space representing a given data point, $\mathbf{A}_i$ is the cluster center points of the *ith* cluster, $\mathbf{A}_i$ contains the rows of $\mathbf{A}$ that belong to the *ith* cluster, and $||\cdot||_2$ means the L_2 norm.

By using the Lagrangian multiplier method [30, 31], cluster centers points can be updated according to

$$\mathbf{A}_i = \frac{1}{m_i} \sum_{\mathbf{x} \in \mathbf{A}_i} \mathbf{x}, i = 1, \ldots, k. \tag{2}$$

where m_i is the number of data points corresponding to $\mathbf{A}_i$.

After getting the cluster center points, a data point $\mathbf{x}$ is labeled by its nearest center as follow

$$y_i = \arg\min_i ||\mathbf{x} - \mathbf{A}_i||, i = 1, \ldots, k. \tag{3}$$

Given a set of m data points, k-means randomly choose k data points as the initial cluster centers. In turn, every data point is assigned to its cluster center by (3). Then, k cluster centers points are updated by (2). The final k clusters are obtained if no redistribution of data points or cluster's centroid in any cluster.

2.2 kPC

The kPC [15] wishes to cluster $\mathbf{A}$ into k clusters such that the data points of each cluster lie close to a cluster center plane defined as follows

$$f_i(\mathbf{x}) = \mathbf{x}\mathbf{w}_i + b_i = 0, \quad i = 1, 2, \ldots, k, \tag{4}$$

where $\mathbf{w}_i \in R^n$ is the weighted vector and $b_i \in R$ is the bias.

kPC keeps each data point from the same cluster being the closest to the same cluster center plane, which leads to solve the following programming problems with $i = 1, 2, ..., k$.

$$\begin{aligned} \min_{\mathbf{w}_i, b_i} \quad & ||\mathbf{A}_i\mathbf{w}_i + b_i\mathbf{e}_i||_2^2 \\ s.t. \quad & ||\mathbf{w}_i||_2^2 = 1, \end{aligned} \tag{5}$$

where $\mathbf{A}_i$ contains the rows of $\mathbf{A}$ that belong to the *ith* cluster, $\mathbf{e}_i$ is a vector of ones of appropriate dimension. The geometric interpretation is clear, i.e., given the ith cluster, the objective of (5) minimizes the sum of the distances between the data points of the ith cluster and the ith center plane, and the constraint normalizes the normal vector of the center plane. The solution of the problem (5) can be obtained by solving k eigenvalue problems [15].

Once we obtained the k cluster center planes, a data point $\mathbf{x}$ is assigned to the *ith* cluster by

$$y = \arg\min_i ||\mathbf{x}\mathbf{w}_i + b_i||, \quad i = 1, \ldots, k. \tag{6}$$

Given the original data points $\mathbf{A}$, kPC starts from randomly initializing $(\mathbf{w}_i, b_i)$, where $||\mathbf{w}_i||_2^2 = 1$. In turn, every data point is assigned a label by (6). Then, k center planes are updated by solving (5). The final k cluster center planes are obtained if a repeated overall assignment of data points to cluster center plane or a non-decrease in the overall objective function occur.

3 kPPC formulation

3.1 Linear kPPC

For each cluster, kPPC not only requests the objective data points close to the corresponding center plane but also requests the other data points far away from this cluster center plane, which leads to solve the following programming problems with $i = 1, 2, ..., k$

$$\begin{aligned} \min_{(\mathbf{w}_i, b_i)} \quad & ||\mathbf{A}_i\mathbf{w}_i + b_i\mathbf{e}_i||_2^2 - c||\mathbf{B}_i\mathbf{w}_i + b_i\bar{\mathbf{e}}_i||_2^2 \\ s.t. \quad & ||\mathbf{w}_i||_2^2 = 1, \end{aligned} \tag{7}$$

where $\mathbf{A}_i$ contains the rows of $\mathbf{A}$ that belong to the *ith* cluster and $\mathbf{B}_i$ is the data points not belonging to the *ith* cluster; $c > 0$ is a parameter to weight the proximal term and the aloof term; $\mathbf{e}_i$ and $\bar{\mathbf{e}}_i$ are vectors of ones with an appropriate dimensions. Once we obtained the k cluster center planes, the data points can be assigned the same as kPC by (6).

Now, we give the details to solve (7). For $i = 1, 2, \ldots, k$, the problem (7) can be solved by the Lagrangian multiplier method [23, 30, 31] below. The Lagrangian function of (7) is

$$L = ||\mathbf{A}_i\mathbf{w}_i + b_i\mathbf{e}_i||_2^2 - c||\mathbf{B}_i\mathbf{w}_i + b_i\bar{\mathbf{e}}_i||_2^2 - \lambda(||\mathbf{w}_i||_2^2 - 1), \tag{8}$$

where λ is the Lagrangian multiplier. Set the partial derivatives of L with respect to $(\mathbf{w}_i, b_i)$ to zero, we get the following two equalities:

$$\frac{1}{2}\frac{\partial L}{\partial \mathbf{w}_i} = (\mathbf{A}_i^\top\mathbf{A}_i - c\mathbf{B}_i^\top\mathbf{B}_i)\mathbf{w}_i + (\mathbf{A}_i^\top\mathbf{e}_i - c\mathbf{B}_i^\top\bar{\mathbf{e}}_i)b_i - \lambda\mathbf{w}_i = 0, \tag{9}$$

$$\frac{1}{2}\frac{\partial L}{\partial b_i} = (\mathbf{e}_i^\top\mathbf{A}_i - c\bar{\mathbf{e}}_i^\top\mathbf{B}_i)\mathbf{w}_i + (\mathbf{e}_i^\top\mathbf{e}_i - c\bar{\mathbf{e}}_i^\top\bar{\mathbf{e}}_i)b_i = 0. \tag{10}$$

When $\mathbf{e}_i^\top\mathbf{e}_i - c\bar{\mathbf{e}}_i^\top\bar{\mathbf{e}}_i = 0$, b_i will disappear, and thus there are two cases of the solution of (7):

Case (1). $\mathbf{e}_i^\top\mathbf{e}_i - c\bar{\mathbf{e}}_i^\top\bar{\mathbf{e}}_i \neq 0$:

$$\mathbf{Q}w_i = \lambda w_i, \tag{11}$$

$$b_i = \frac{(\mathbf{e}_i^\top\mathbf{A}_i - c\bar{\mathbf{e}}_i^\top\mathbf{B}_i)\mathbf{w}_i}{-\mathbf{e}_i^\top\mathbf{e}_i + c\bar{\mathbf{e}}_i^\top\bar{\mathbf{e}}_i}, \tag{12}$$

where $\mathbf{Q} = \mathbf{A}_i^\top\mathbf{A}_i - c\mathbf{B}_i^\top\mathbf{B}_i + \frac{(\mathbf{e}_i^\top\mathbf{A}_i - c\bar{\mathbf{e}}_i^\top\mathbf{B}_i)^\top(\mathbf{e}_i^\top\mathbf{A}_i - c\bar{\mathbf{e}}_i^\top\mathbf{B}_i)}{-\mathbf{e}_i^\top\mathbf{e}_i + c\bar{\mathbf{e}}_i^\top\bar{\mathbf{e}}_i}$.

Case (2). $\mathbf{e}_i^\top\mathbf{e}_i - c\bar{\mathbf{e}}_i^\top\bar{\mathbf{e}}_i = 0$:

$$(\mathbf{A}_i^\top\mathbf{A}_i - c\mathbf{B}_i^\top\mathbf{B}_i)\mathbf{w}_i = \lambda\mathbf{w}_i, \tag{13}$$

$$b_i = \frac{1}{m_i}\sum_{\mathbf{x}\in\mathbf{A}_i}\mathbf{x}\mathbf{w}_i, \tag{14}$$

where m_i is the number of data points corresponding to $\mathbf{A}_i$. It is easy to conclude that $\mathbf{w}_i$ is the eigenvector of the smallest eigenvalue of the matrix $\mathbf{Q}$ in (Eqs. 11, 12) (or $\mathbf{A}_i^\top\mathbf{A}_i - c\mathbf{B}_i^\top\mathbf{B}_i$ of (Eqs. 13, 14, 15].

Now, we turn to the initialization of the proposed kPPC. As we know, the initial centers selection is very

important in both k-means and kPC [18, 32–35]. Therefore, we propose an effective way to give the initial centers. Specifically, we introduce a similarity graph to initialize the centers. The following steps are adopted:

1. Calculate the P nearest neighbors undirected graph G [36] for $\mathbf{A}$ by

$$G_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_i \in N(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in N(\mathbf{x}_i) \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

   where $N(\mathbf{x}_i) = \{\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, ...\mathbf{x}_i^{(P)}\}$ is the set of its P nearest neighbors. The data points are labeled into $\bar{c}$ connected components by tagging them the same label if they are connected in G, and different labels if they are not connected.
2. Suppose k is the number of clusters one wanted. If $\bar{c} < k$, repeatedly find two neighbors with the largest distance and break their neighborhood to divide a connected component into two, until the number of the current connected components is k; if $\bar{c} > k$, find two connected components of data points with the closest distance and merge them into one repeatedly, until the number of the current connected components is k.
3. Export the current cluster labels as the initial cluster.

According the above procedure, we obtain the following algorithm 1.

Algorithm 1 Linear kPPC.

Input: $\mathbf{A}$: Dataset; P: number of nearest neighbors;
k: number of clusters; c: appropriate parameter.
Output: $\mathbf{w}_i$, b_i, $i = 1, \ldots, k$; $\mathbf{y}$: the label of $\mathbf{A}$.
Process:
1. initialize $\mathbf{w}_i^0$, b_i^0 by using the above Laplace graph procedure.
2. for j = 1,2, . . ., update $\mathbf{w}_i^j$, b_i^j by solving (7), where the ith cluster is represented by $\mathbf{A}_i^j$ and the other clusters is represented by $\mathbf{B}_i^j = \mathbf{A} \backslash \mathbf{A}_i^j$.
3. obtain the label of $\mathbf{A}_i$ by using $\mathbf{w}_i^j$, b_i^j from (6).
4. if the algorithm converges, then terminate the iteration. Otherwise, go to Step2.

We consider the algorithm converges if a repeated overall assignment of data points to cluster center plane or a non-decrease in the overall objective function occur.

3.2 Nonlinear kPPC

When the data points distribution in the nonlinear manifold (such as Figs. 5 and 6), the linear methods may not doing well. In this section, we extend our linear kPPC to manifold clustering by kernel trick. The thought of kernel trick [31, 37, 38] is to map the input into a higher-dimensional feature space with some non-linear transformation. The trick is that this mapping is never given explicitly, but implicitly applied in this newly-mapped data space. The nonlinear kPPC formulation also follows this process.

Instead of the aforementioned linear plane in input space, we consider the following kernel-generated surface [39–41]

$$K(\mathbf{x}^\top, \mathbf{A}^\top)\mathbf{w}_i + b_i = 0, i = 1, 2, \ldots, k, \tag{16}$$

where K is an appropriately chosen kernel.

Nonlinear kPPC is to minimize the following programming problems with $i = 1, 2, ..., k$.

$$\min_{(\mathbf{w}_i, b_i)} ||K(\mathbf{A}_i, \mathbf{A}^\top)\mathbf{w}_i + \mathbf{e}_i b_i||_2^2 - c||K(\mathbf{B}_i, \mathbf{A}^\top)\mathbf{w}_i + \bar{\mathbf{e}}_i b_i||_2^2 \tag{17}$$

$$s.t. \quad ||\mathbf{w}_i||_2^2 = 1, \tag{18}$$

where $\mathbf{A}_i$ is the data points of the ith cluster and $\mathbf{B}_i$ is the data points not in the ith cluster, $c > 0$ is a parameter to weight the proximal term and the aloof term, $\mathbf{e}_i$ and $\bar{\mathbf{e}}_i$ are vectors of ones of appropriate dimensions. Once we obtained k clustering surfaces, a data point $\mathbf{x}$ is assigned to the ith cluster by

$$y = \arg\min_i ||K(\mathbf{x}^\top, \mathbf{A}^\top)\mathbf{w}_i + b_i||, \quad i = 1, \ldots, k. \tag{19}$$

The above problems (17, 18) also can be solved by the Lagrangian multiplier method below

$$L = ||K(\mathbf{A}_i, \mathbf{A}^\top)\mathbf{w}_i + \mathbf{e}_i b_i||_2^2 - c||K(\mathbf{B}_i, \mathbf{A}^\top)\mathbf{w}_i + \bar{\mathbf{e}}_i b_i||_2^2 - \lambda(||\mathbf{w}_i||_2^2 - 1), \tag{20}$$

where λ is the Lagrangian multiplier. Set the partial derivatives of L with respect to $(\mathbf{w}_i, b_i)$ Ù:

Case (i). $\mathbf{e}_i^\top \mathbf{e}_i - c\bar{\mathbf{e}}_i^\top \bar{\mathbf{e}}_i \neq 0$:

$$\mathbf{Q}w_i = \lambda w_i \tag{21}$$

$$b_i = \frac{(\mathbf{e}_i^\top K(\mathbf{A}_i, \mathbf{A}^\top) - c\bar{\mathbf{e}}_i^\top K(\mathbf{B}_i, \mathbf{A}^\top))\mathbf{w}_i}{-\mathbf{e}_i^\top \mathbf{e}_i + c\bar{\mathbf{e}}_i^\top \bar{\mathbf{e}}_i}, \tag{22}$$

where $\mathbf{Q} = (K(\mathbf{A}_i, \mathbf{A}^\top)^\top K(\mathbf{A}_i, \mathbf{A}^\top) - cK(\mathbf{B}_i, \mathbf{A}^\top)^\top K(\mathbf{B}_i, \mathbf{A}^\top) + \frac{(\mathbf{e}_i^\top K(\mathbf{A}_i, \mathbf{A}^\top) - c\bar{\mathbf{e}}_i^\top K(\mathbf{B}_i, \mathbf{A}^\top))^\top (\mathbf{e}_i^\top K(\mathbf{A}_i, \mathbf{A}^\top) - c\bar{\mathbf{e}}_i^\top K(\mathbf{B}_i, \mathbf{A}^\top))}{-e_i^\top e_i + c\bar{e}_i^\top \bar{e}_i})$.

Case (ii). $\mathbf{e}_i^\top \mathbf{e}_i - c\bar{\mathbf{e}}_i^\top \bar{\mathbf{e}}_i = 0$:

$$(K(\mathbf{A}_i, \mathbf{A}^\top)^\top K(\mathbf{A}_i, \mathbf{A}^\top) - cK(\mathbf{B}_i, \mathbf{A}^\top)^\top K(\mathbf{B}_i, \mathbf{A}^\top))\mathbf{w}_i = \lambda \mathbf{w}_i \tag{23}$$

$$b_i = \frac{1}{m_i} \sum_{\mathbf{x} \in K(\mathbf{A}_i, \mathbf{A}^\top)} \mathbf{x}\mathbf{w}_i, \tag{24}$$

where m_i is the number of data points corresponding to $\mathbf{A}_i$.

We also use a similarity graph to construct the initial cluster center planes instead of the randomly initialization for nonlinear kPPC. The process contains the following steps: (1) we choose an appropriately kernel function

$K(\cdot,\cdot)$ to map the data points into a higher-dimensional feature space; (2) the same strategy as linear kPPC is used to construct the initial center planes. Then, we obtain the algorithm 2.

From algorithm 1 and 2, we can see that the main computational cost in our kPPC is to compute one P nearest neighbors undirected graph G and some eigenvalue decomposition problems. The computational complexity of each is at most $O(n^3)$. Thus, the complexity of our kPPC is $O(t * n^3)$, where t is the number of iterations.

Algorithm 2 Nonlinear kPPC.

Input: $\mathbf{A}$: Dataset; P: number of nearest neighbors; k: number of clusters;
$K(\cdot,\cdot)$: kernel function; c: appropriate parameter.
Output: $\mathbf{w}_i$, b_i, $i = 1,\ldots,k$; $\mathbf{y}$: the label of $\mathbf{A}$.
Process:
1. map $\mathbf{A}$ into a higher-dimensional feature space using $K(\cdot,\cdot)$.
2. initialize $\mathbf{w}_i^0$, b_i^0 by using the above Laplace graph procedure.
3. for $j = 1,2,\ldots$, update $\mathbf{w}_i^j$, b_i^j by solving (17), where the *ith* cluster is represented by $\mathbf{A}_i^j$ and the other clusters is represented by $\mathbf{B}_i^j = \mathbf{A}\backslash\mathbf{A}_i^j$.
4. obtain the label of the $\mathbf{A}_i$ by using $\mathbf{w}_i^j$, b_i^j from (19).
5. if the algorithm converges, then terminate the iteration. Otherwise, go to Step3.

The convergence criterion of algorithm 2 is the same as algorithm 1.

4 Experimental results

In this section, to validate our kPPC, experiments are carried out on several artificial datasets and benchmark datasets [42]. We compare our kPPC with other clustering methods, including kmeans [28, 29], kPC [15], and updated versions of DBSCAN [43, 44]. All these methods are implemented by Matlab (2008a) [45] on a PC with an Intel P4 processor (2.5 GHz) with 2 GB RAM. Clustering accuracy and F-measure [46] are used as the performance evaluation criterions, which are defined as follows. Consider $\widetilde{C} = (\widetilde{c_1},...,\widetilde{c_k})$ a prediction partition of a clustering method and $\widetilde{H} = (\widetilde{h_1},...,\widetilde{h_k})$ is a artificial defined partition of $\mathbf{A}$. We refer to a pair of data points $(\mathbf{x}_i, \mathbf{x}_j)$ from $\mathbf{A}$ using the following terms:

SS: if both data points belong to the same cluster of the clustering structure $\widetilde{C}$ and to the same group of $\widetilde{H}$.

SD: if data points belong to the same cluster of $\widetilde{C}$ and to different group of $\widetilde{H}$.

DS: if data points belong to different cluster of $\widetilde{C}$ and to the same group of $\widetilde{H}$.

DD: if data points belong to different cluster of $\widetilde{C}$ and to different group of $\widetilde{H}$.

Assuming now that f_{11}, f_{10}, f_{01} and f_{00} are the number of SS, SD, DS and DD pairs respectively. Now we can define the following indices to measure the degree of similarity between $\widetilde{C}$ and $\widetilde{H}$:

$$\text{R and Statistic} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}. \tag{25}$$

In this paper, the clustering accuracy is computed by the Rand statistic [46] as above.

F-measure [46] is computed as follow:

$$F - measure = \frac{2 * \widetilde{P} * \widetilde{R}}{\widetilde{P} + \widetilde{R}}. \tag{26}$$

where $\widetilde{P}$ is the precision, $\widetilde{R}$ is the recall.

$$\widetilde{P} = \frac{f_{11}}{f_{11} + f_{10}}. \tag{27}$$

$$\widetilde{R} = \frac{f_{11}}{f_{11} + f_{01}}. \tag{28}$$

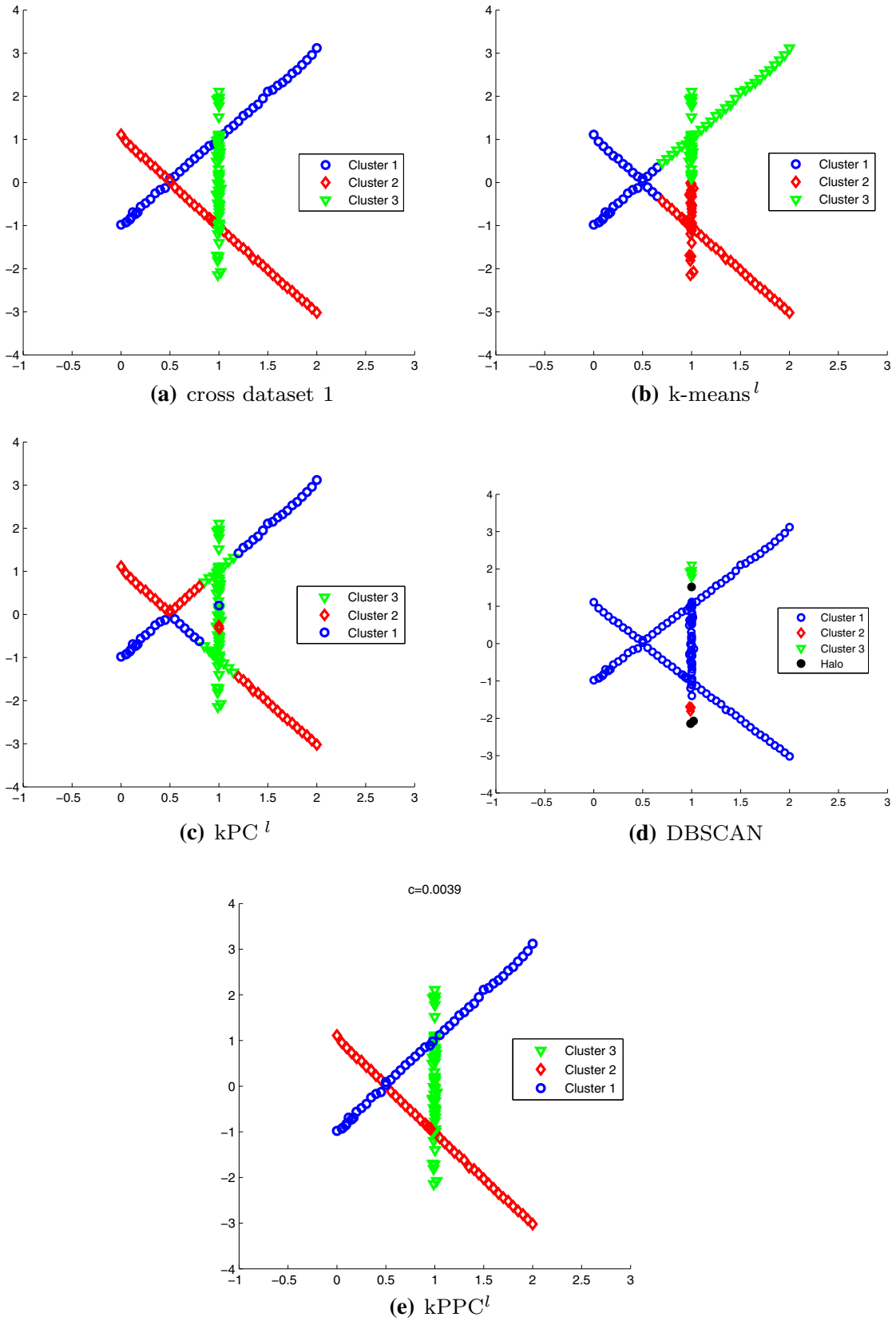
The parameter c in both linear and nonlinear kPPC is selected from $\{2^i | i = -8,-7,\ldots,8\}$. For the nonlinear cluster methods, we choose the RBF kernel and select the kernel parameter p in $\{2^i | i = -8,-7,\ldots,8\}$. The parameter P nearest neighbors undirected graph based initialization is selected from $\{P = 1,3,5,7,9\}$.

4.1 Artificial datasets

In this subsection, four artificial datasets are generated. The first two datasets (see Figs. 3a and 4a are composed of three clusters and rest two datasets (see Figs. 5a and 6a are composed of two clusters. The number of data points are 160, 160, 126, and 150, respectively. Figs. 3, 4, 5 and 6b–e show the results of the different cluster methods performed on these datasets. Tables 1 and 2 show the clustering accuracies and the F-measures of these methods, respectively.

The first two datasets are cross-plane type datasets, and we use the linear cluster methods to cluster these datasets. From Figs. 3 and 4 , we observe that both kPC and our linear kPPC are better than k-means and DBSCAN on these two datasets, and our kPPC get the best performance, while kPC performs not as good as our kPPC. This shows that our linear kPPC is more reasonable than the others when the dataset has the linear structure. Table 1 shows that our linear kPPC gives the best performance and both the clustering accuracies and F-measure are highest.

The rest two datasets have nonlinear structure, and we use the nonlinear cluster methods to cluster these datasets. From Figs. 5 and 6, we observe that DBSCAN get the best performance, and our nonlinear kPPC clusters two clusters almost the same as the original datasets, while the other two methods do not perform well in the nonlinear case. This shows DBSCAN and our nonlinear kPPC can catch

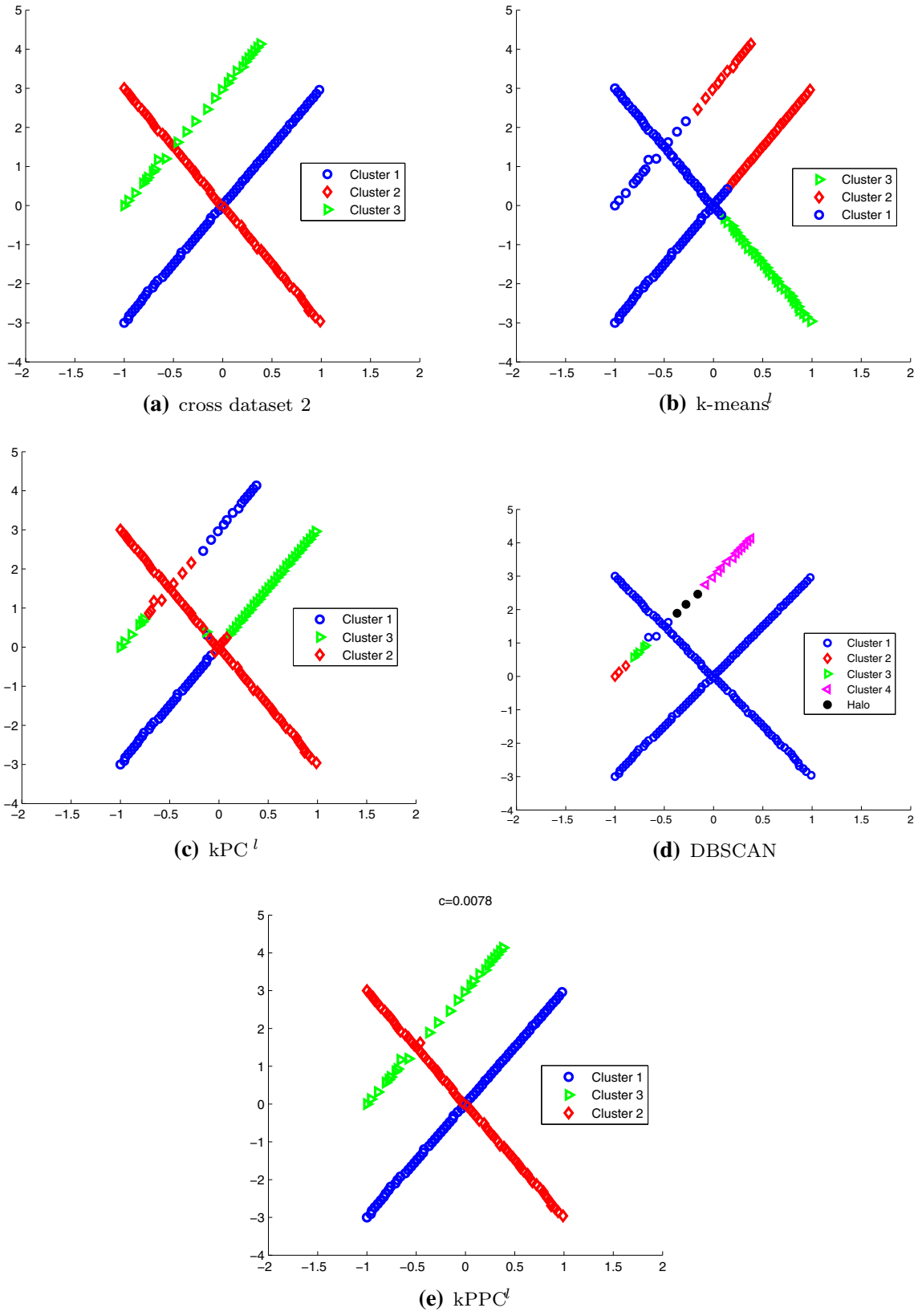


(a) cross dataset 1 **(b)** k-means^{l}

(c) kPC^{l} **(d)** DBSCAN

(e) kPPC^{l}

Fig. 3 Different methods on cross dataset 1. $(\cdot)^{l}$ denotes the linear cases. In **e**, the value of parameter c is corresponding to linear kPPC's clustering accuracy

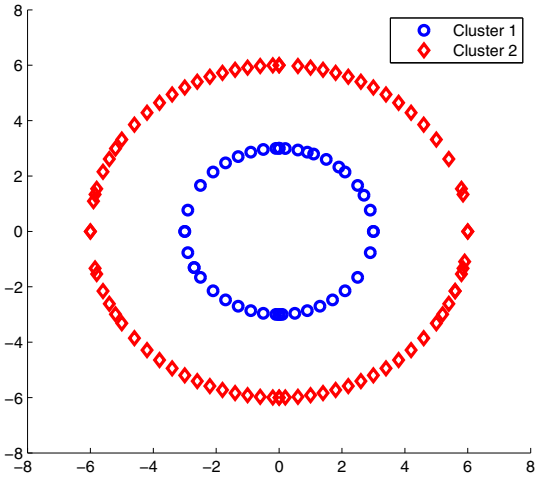


(a) cross dataset 2 **(b)** k-means^l

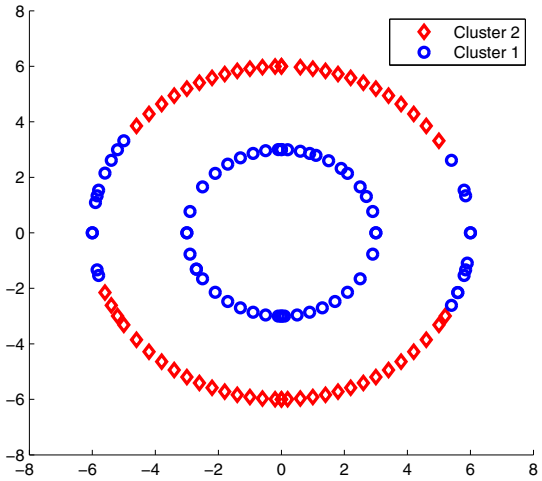
(c) kPC^l **(d)** DBSCAN

(e) kPPC^l

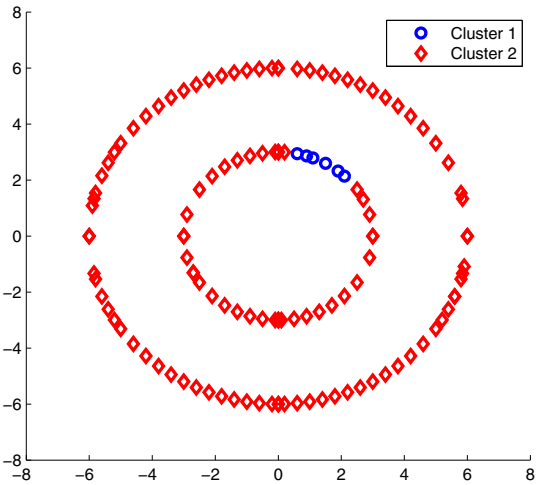
Fig. 4 Different methods on cross dataset 2. $(\cdot)^l$ denotes the linear cases. In **e**, the value of parameter c is corresponding to linear kPPC's clustering accuracy



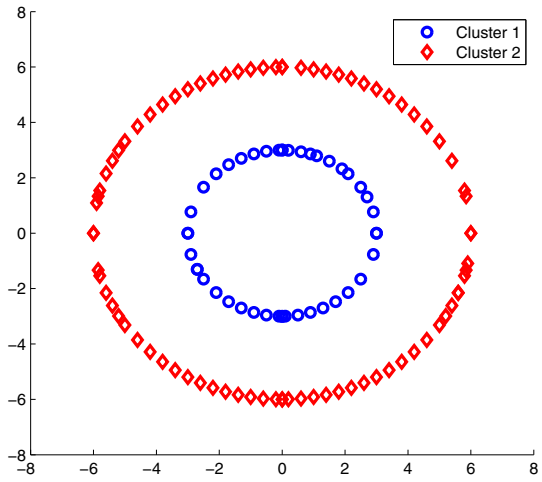
(a) two-circles



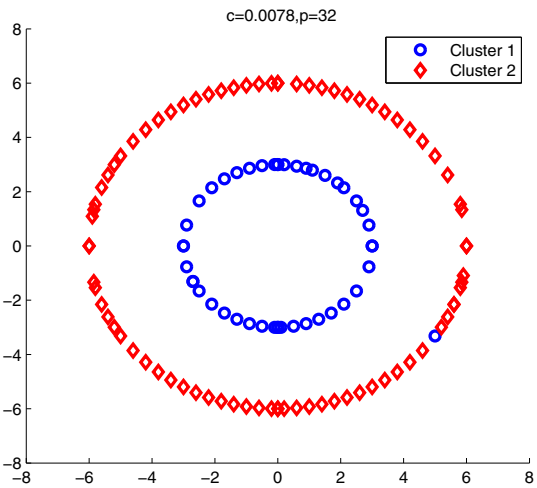
(b) k-means^n



(c) kPC^n

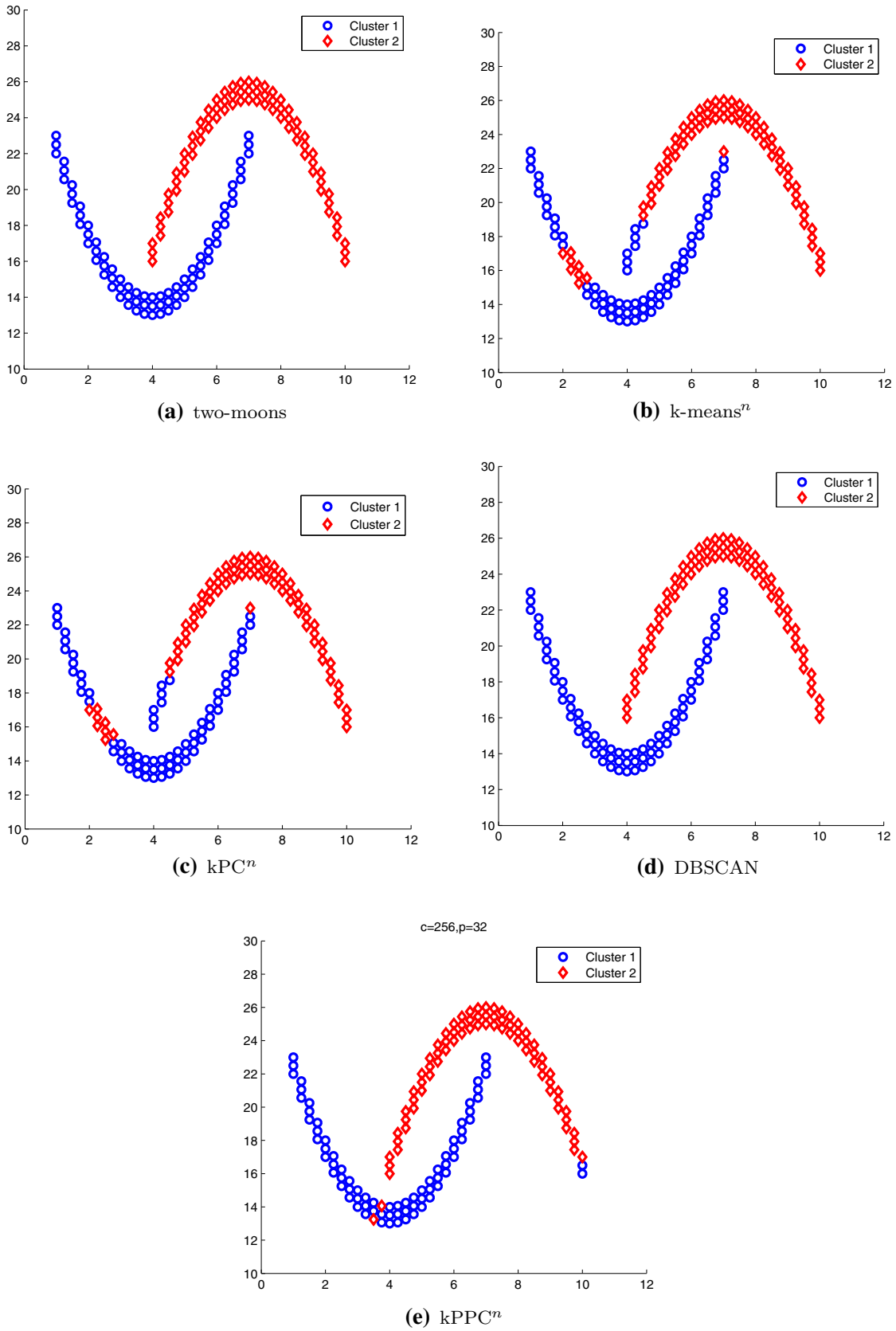


(d) DBSCAN



(e) kPPC^n

Fig. 5 Different methods on two-circles dataset. $(\cdot)^n$ denotes the nonlinear cases. In **e**, the value of parameters c and p are corresponding to nonlinear kPPC's clustering accuracy



(a) two-moons **(b)** k-means^n **(c)** kPC^n **(d)** DBSCAN **(e)** kPPC^n

Fig. 6 Different methods on two-moons dataset. $(\cdot)^n$ denotes the nonlinear cases. In **e**, the value of parameters c and p are corresponding to nonlinear kPPC's clustering accuracy

Table 1 Experiments on linear artificial datasets, where $(\cdot)^l$ denotes linear case

Dataset	k-means^l Acc (%) F-measure (%)	kPC^l Acc (%) F-measure (%)	DBSCAN Acc (%) F-measure (%)	kPPC^l Acc (%) F-measure (%)
Cross dataset1	65.42	76.42	36.43	**96.64**
	53.10	78.04	49.76	**97.50**
Cross dataset2	59.39	75.60	58.66	**99.28**
	53.92	67.83	66.11	**99.37**

Highest value in each row is in bold

Table 2 Experiments on nonlinear artificial datasets, where $(\cdot)^n$ denotes nonlinear case

Dataset	k-means^n Acc (%) F-measure (%)	kPC^n Acc (%) F-measure (%)	DBSCAN Acc (%) F-measure (%)	kPPC^n Acc (%) F-measure (%)
Two-circles	70.95	56.91	**100.00**	98.41
	62.51	53.29	**100.00**	99.21
Two-moons	80.81	67.79	**100.00**	94.77
	81.20	70.32	**100.00**	97.33

Highest value in each row is in bold

the distribution of the manifold datasets more precisely. However, DBSCAN achieves better clustering performance than the kPPC in Table 2. As we know, DBSCAN is a density-based spatial clustering technique and can merge the overlapping datasets into one cluster. Therefore, we observe that DBSCAN get the worse performance in Table 1. While we have two non-overlapping datasets and the spatial clustering density of the datasets is uniform in Table 2. Therefore, the clustering effect of DBSCAN is a little better.

4.2 Benchmark datasets

In this subsection, we compare our kPPC with the others in their ability to recover cluster labels [15] on several benchmark datasets listed in Table 3. We use accuracy and F-measure as cluster evaluation criteria in the following tables. Both the random initialization and graph-based initialization k-means, kPC, and kPPC are experimented. For random initialization, all these methods are run ten times, and we record the average accuracy, the standard deviation and the one-run CPU time in Tables 4 and 9. The better results are marked by bold. The p value was calculated by performing a paired t test [47–49] that comparing our kPPC to the other methods under the assumption of the null hypothesis that there is no difference between the dataset accuracy distributions.

From Table 4, it is seen that our kPPC has better ability to recover cluster labels than the other methods, especially the linear kPPC gets the six highest accuracies on these eight datasets, and the accuracy of our kPPC is almost higher than kPC on both random and graph-based initialization. This indicates that our kPPC is better than kPC, which means that the introduction of the between-cluster information can improve the clustering ability of kPC. Furthermore, graph-based clusters are more stable because their standard deviation are always zero, while the bias of random clusters are ranging from 0.02 to 2.17%. Therefore, our graph-based initialized kPPC are more reasonable and stable than randomly initialized kPPC. From Table 5, it is seen that kmeans is more stable than other methods with random initialization, which implies the F-measures of these plane-based methods, includes kPC and kPPC, strongly depend on the initialization. our linear kPPC also gets the five highest F-measures on these eight datasets. From Tables 4 and 5, we can seen that our kPPC owns the highest cluster results compared with the others. One

Table 3 Details of the benchmark datasets

Dataset	Points	Features	Clusters
(a) Ionosphere	351	33	2
(b) Seeds	210	7	3
(c) Glass	214	9	6
(d) Spiral	312	2	3
(e) Jain	373	2	2
(f) Flame	240	2	2
(g) Ecoli	336	7	8
(h) Haberman	306	3	2

Table 4 The clustering accuracy of the linear clustering methods on the benchmark datasets

Data	k-means^{r} Acc (%) p value	k-means^{l} Acc (%) p value	kPC^{r} Acc (%) p value	kPC^{l} Acc (%) p value	kPPC^{r} Acc (%) p value	kPPC^{l} Acc (%) p value
(a)	58.41 ± 0.00	58.65	50.12 ± 0.29	57.95	56.64 ± 0.42	**59.89**
	0.0000	0.0000	0.0000	0.0000	0.0270	–
(b)	**89.97** ± 0.00	**89.97**	57.24 ± 1.14	53.78	83.34 ± 0.65	82.37
	0.0000	0.0000	0.0000	0.0000	0.0074	–
(c)	65.88 ± 1.36	67.08	65.78 ± 1.48	66.25	65.73 ± 2.16	**69.29**
	0.0001	0.0000	0.0000	0.0000	0.0018	–
(d)	55.38 ± 0.02	55.46	55.32 ± 0.29	55.50	55.87 ± 0.25	**56.27**
	0.0000	0.0000	0.0000	0.0000	0.6550	–
(e)	65.91 ± 0.00	66.21	61.41 ± 0.00	61.41	68.35 ± 0.03	**69.43**
	0.0000	0.0000	0.0000	0.0000	0.8137	–
(f)	71.55 ± 0.00	73.24	55.07 ± 1.83	70.46	73.98 ± 2.17	**74.39**
	0.0000	0.0000	0.0000	0.0000	0.0226	–
(g)	79.79 ± 1.65	83.06	27.60 ± 0.95	27.60	70.65 ± 0.00	72.38
	0.0000	0.0000	0.0000	0.0000	0.0000	–
(h)	49.89 ± 0.04	63.21	52.67 ± 0.73	65.59	62.30 ± 0.93	**63.55**
	0.0000	0.0000	0.0000	0.0000	0.0021	–

$(\cdot)^{r}$ denotes random initialization and $(\cdot)^{l}$ denotes the Laplace graph based initialization

Highest value in each row are in bold

Table 5 The F-measure of the linear clustering methods on the benchmark datasets

Data	k-means^{r} F-measure (%) p value	k-means^{l} F-measure (%) p value	kPC^{r} F-measure (%) p value	kPC^{l} F-measure (%) p value	kPPC^{r} F-measure (%) p value	kPPC^{l} F-measure (%) p value
(a)	71.19 ± 0.00	71.48	57.14 ± 3.85	64.23	72.21± 5.13	**78.13**
	0.0000	0.0000	0.0000	0.0000	0.0053	–
(b)	89.09 ± 0.24	**89.54**	42.88 ± 2.14	45.43	81.79 ± 0.58	83.51
	0.0000	0.0000	0.0000	0.0000	0.0000	–
(c)	47.95 ± 1.82	**48.64**	29.32 ± 1.80	40.50	44.56 ± 3.43	44.38
	0.0000	0.0000	0.0000	0.0000	0.8698	–
(d)	34.98 ± 0.37	35.26	35.24 ± 0.88	36.22	43.94 ± 4.78	**48.44**
	0.0000	0.0000	0.0000	0.0000	0.0156	–
(e)	79.61 ± 0.00	79.86	85.05 ± 0.15	85.05	82.44 ± 0.24	**94.47**
	0.0000	0.0000	0.0000	0.0000	0.0000	–
(f)	83.24 ± 0.00	82.83	67.48 ± 0.20	82.43	76.31 ± 11.94	**85.29**
	0.0000	0.0000	0.0000	0.0000	0.0414	–
(g)	**66.67** ± 4.05	64.88	41.96 ± 0.74	42.46	49.63 ± 2.31	60.26
	0.0000	0.0000	0.0000	0.0000	0.0000	–
(h)	59.55 ± 6.45	71.15	51.63 ± 2.21	57.55	68.30 ± 1.80	**72.67**
	0.0000	0.0000	0.0000	0.0000	0.0000	–

$(\cdot)^{r}$ denotes random initialization and $(\cdot)^{l}$ denotes the Laplace graph based initialization

Highest value in each row is in bold

shortcoming of kPPC is that the time consuming is larger than other methods from Table 6.

For the nonlinear case, Tables 7 and 9 have the similar results with the above linear case. Therefore confirm our conclusions further.

4.3 Parameter influence

In this subsection, we show the influence of parameters c and p in our kPPC of the above benchmark datasets. Figs. 7 and 8 show the relations between the parameters

Table 6 The time of the linear clustering methods on the benchmark datasets

Data	k-means^{r}, Time (s)	k-means^{l}, Time (s)	kPC^{r}, Time (s)	kPC^{l}, Time (s)	kPPC^{r}, Time (s)	kPPC^{l}, Time (s)
(a)	**0.03**	0.09	0.05	0.19	0.05	0.22
(b)	**0.05**	0.18	0.07	0.62	0.58	0.74
(c)	**0.06**	0.21	0.58	1.58	1.47	1.67
(d)	0.04	0.28	**0.01**	0.18	0.04	0.58
(e)	0.06	0.68	**0.02**	0.74	0.04	0.28
(f)	0.02	0.06	**0.01**	0.42	0.28	0.56
(g)	**0.01**	0.02	**0.01**	0.03	0.58	0.66
(h)	0.02	0.03	**0.01**	0.12	0.28	0.36

$(\cdot)^{r}$ denotes random initialization and $(\cdot)^{l}$ denotes the Laplace graph based initialization

Fastest one in each row are in bold

Table 7 The clustering accuracy of the manifold clustering methods on the benchmark datasets

Data	k-means^{r} Acc (%) p value	k-means^{l} Acc (%) p value	kPC^{r} Acc (%) p value	kPC^{l} Acc (%) p value	DBSCAN Acc (%) p value	kPPC^{r} Acc (%) p value	kPPC^{l} Acc (%) p value
(a)	60.67 ± 0.00	60.67	54.18 ± 0.00	54.34	**68.67**	59.64 ± 0.01	60.67
	NAN	NAN	0.0000	0.0000	0.0000	0.0000	–
(b)	90.96 ± 0.61	**91.65**	56.28 ± 0.31	86.66	33.01	80.17 ± 3.85	80.31
	0.0000	0.0000	0.0000	0.0000	0.0000	0.1440	–
(c)	**69.86** ± 0.56	69.56	57.43 ± 7.02	62.58	27.17	62.75 ± 0.02	62.77
	0.0000	0.0000	0.5988	0.0000	0.0000	0.1316	–
(d)	56.17 ± 0.94	57.74	56.04 ± 0.06	59.62	**100.00**	65.12 ± 2.27	**100.00**
	0.0000	0.0000	0.0000	0.0000	NAN	0.0000	–
(e)	82.46 ± 2.31	86.53	79.59 ± 11.83	88.06	90.00	79.14 ± 3.45	**90.79**
	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	–
(f)	91.01 ± 0.01	**95.10**	79.79 ± 3.84	88.97	64.81	91.38 ± 1.25	**95.10**
	0.3453	NAN	0.0000	0.0000	0.0000	0.0027	–
(g)	81.03 ± 0.36	**82.39**	70.25 ± 2.54	72.11	31.95	73.27 ± 0.23	75.35
	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	–
(h)	60.74 ± 0.54	61.26	61.20 ± 0.41	61.57	60.08	58.33 ± 4.56	**62.21**
	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	–

$(\cdot)^{r}$ denotes random initialization and $(\cdot)^{l}$ denotes the Laplace graph based initialization

Highest values in each row are in bold

and the accuracies of our kPPC on these datasets, where Fig. 7 corresponds to the linear kPPC and Fig. 8 corresponds to the nonlinear kPPC, respectively. In Fig. 8, we conduct experiments on four selected benchmark datasets listed in Table 3.

From Fig. 7, we obtain three interesting observations: (1) we know that kPPC with commonly setting parameter $P = 1$ performs better than other values of P, and kPPC often works not well if one increase P from 5, 7, and 9. Therefore, we suggest one to set P smaller to get a better performance. (2) the parameter c has a great influence for the clustering accuracy, and with the increase of c, the clustering accuracy tends to be stable. (3) kPPC performs well when c is relatively small but larger than zero and it can be seen that when $c = 0$, the clustering accuracies are mostly lower. The reason lies in that when $c = 0$, our kPPC degrades to kPC. Therefore, we conclude that the parameter c can be set more flexible, such as from 0 to 64.

There are three parameters c, p, and P in the nonlinear kPPC. From Fig. 8, we can see that (1) the parameter $P = 3$ can makes nonlinear kPPC perform better than other values of P. (2) both c and p have great influence for the clustering accuracy. Moreover, $c \in [1, 64]$ may be a good option to obtain a well performance. (3) the best accuracy of the different datasets is obtained with different p, and $p \in [0.02, 32]$ often makes it perform well.

Table 8 The F-measure of the manifold clustering methods on the benchmark datasets

Data	k-means^{r} F-measure (%) p value	k-means^{l} F-measure (%) p value	kPC^{r} F-measure (%) p value	kPC^{l} F-measure (%) p value	DBSCAN F-measure (%) p value	kPPC^{r} F-measure (%) p value	kPPC^{l} F-measure (%) p value
(a)	73.66 ± 0.00	73.66	67.90 ± 1.78	77.56	**88.24**	71.71 ± 8.41	78.13
	0.0000	0.0000	0.0000	0.0000	0.0000	0.0390	–
(b)	89.15 ± 0.25	**90.45**	83.98 ± 8.50	86.78	50.00	83.15 ± 0.73	83.38
	0.0000	0.0000	0.8289	0.0000	0.0000	0.3434	–
(c)	46.89 ± 2.91	52.20	50.22 ± 5.39	**52.41**	52.30	51.5 ± 0.59	**52.41**
	0.0000	0.0000	0.2300	NAN	0.0000	0.0155	–
(d)	43.40 ± 11.30	71.27	51.01 ± 3.97	**100.00**	**100.00**	44.38 ± 4.02	**100.00**
	0.0000	0.0000	0.0000	NAN	NAN	0.0000	–
(e)	77.45 ± 1.29	73.75	77.75 ± 4.35	92.01	85.05	85.05 ± 0.00	**90.00**
	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	–
(f)	94.96 ± 5.33	97.47	90.12 ± 8.57	97.47	78.26	92.70 ± 1.57	**97.48**
	0.1687	0.0000	0.0237	0.0000	0.0000	0.0000	–
(g)	64.63 ± 2.38	**89.16**	53.50 ± 7.90	60.22	60.98	58.83 ± 3.70	61.51
	0.0025	0.0000	0.0107	0.0000	0.0000	0.0477	–
(h)	72.66 ± 0.00	72.54	76.86 ± 6.79	73.05	85.22	70.66 ± 4.65	**89.16**
	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	–

$(\cdot)^{r}$ denotes random initialization and $(\cdot)^{l}$ denotes the Laplace graph based initialization

Highest values in each row are in bold

Table 9 The time of the manifold clustering methods on the benchmark datasets

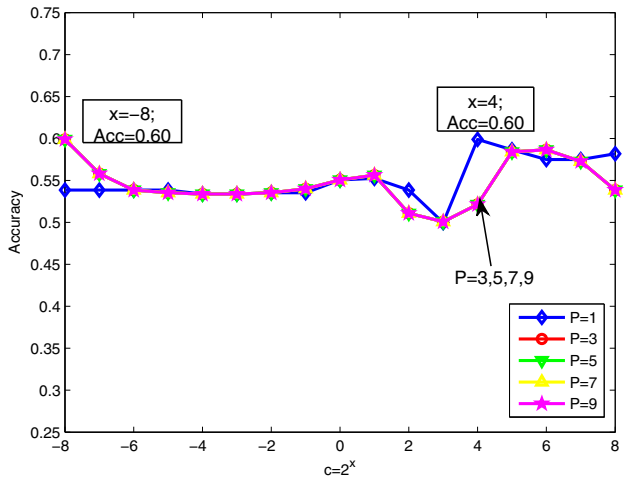
Data	k-means^{r}, Time (s)	k-means^{l}, Time (s)	kPC^{r}, Time (s)	kPC^{l}, Time (s)	DBSCAN,Time (s)	kPPC^{r}, Time (s)	kPPC^{l}, Time (s)
(a)	**0.06**	3.44	1.24	6.67	0.14	1.35	1.60
(b)	**0.04**	0.76	0.33	2.72	0.99	7.52	16.18
(c)	**0.03**	0.77	3.96	4.00	0.92	10.89	104.95
(d)	**0.05**	2.51	0.52	9.59	0.75	12.25	17.21
(e)	**0.08**	3.00	1.17	8.57	0.76	4.01	3.17
(f)	**0.06**	1.05	1.93	2.60	0.81	0.64	0.68
(g)	0.02	0.06	**0.01**	0.42	0.04	0.28	0.56
(h)	0.02	0.06	**0.01**	0.42	0.02	0.28	0.56

$(\cdot)^{r}$ denotes random initialization and $(\cdot)^{l}$ denotes the Laplace graph based initialization

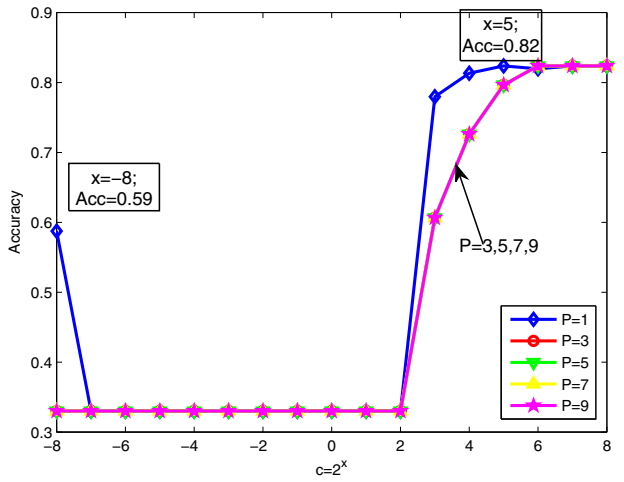
Fastest one in each row is in bold

5 Conclusions

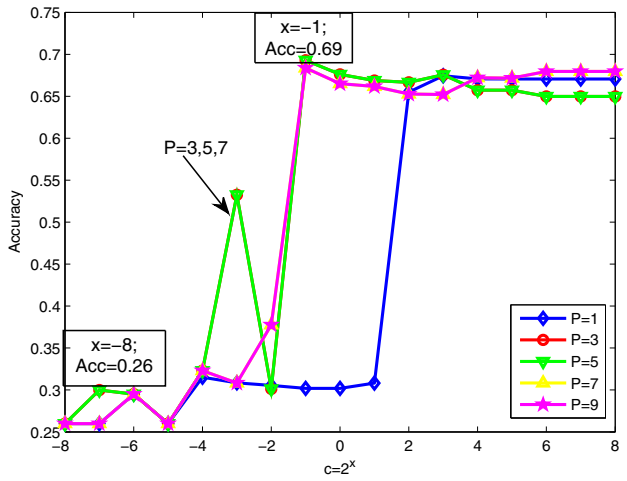
In this paper, we have proposed an improved version of kPC, called kPPC. In kPPC, each cluster center plane is not only close to the objective data points but also far away from the other data points. Our kPPC has been extended to nonlinear clustering. In addition, the initial labels of the data points in kPPC are computed efficiently by a Laplacian graph. Preliminary experiments confirm the merits of our kPPC. Our kPPC Matlab codes can be downloaded from: http://www.optimal-group.org/Resource/kPPC.html. For the future work, it seems worth to find the way to explore the localized representations of the plane such as [50, 51], and use other methods to construct the graph such



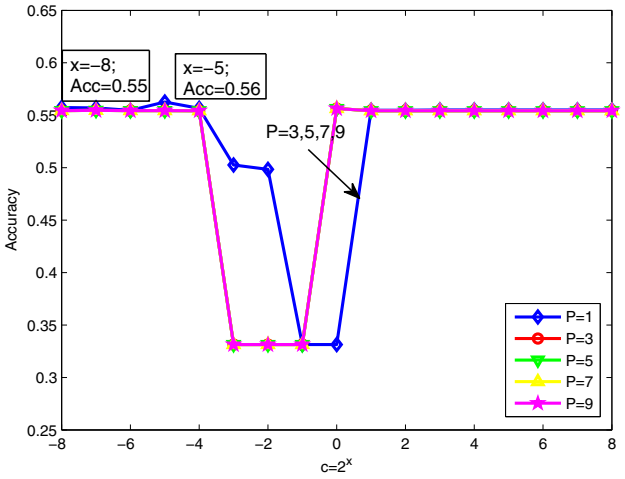
(a) Ionosphere



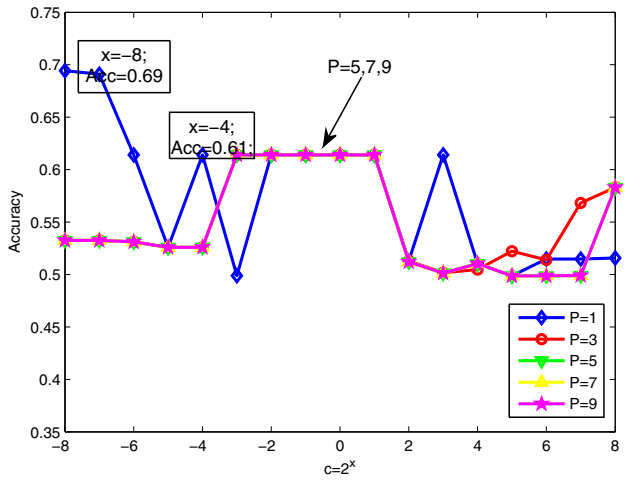
(b) Seeds



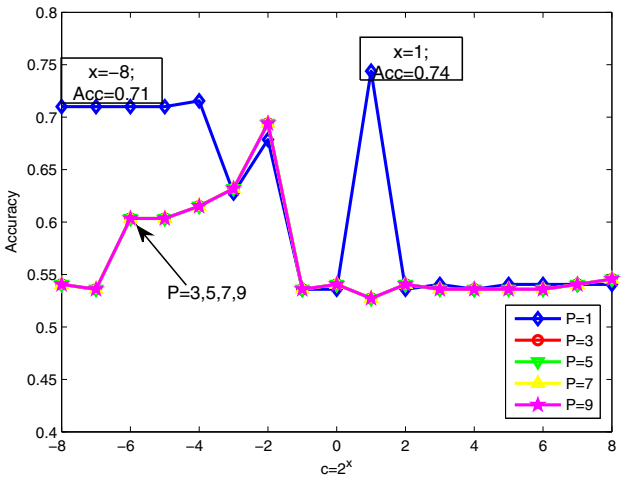
(c) Glass



(d) Spiral

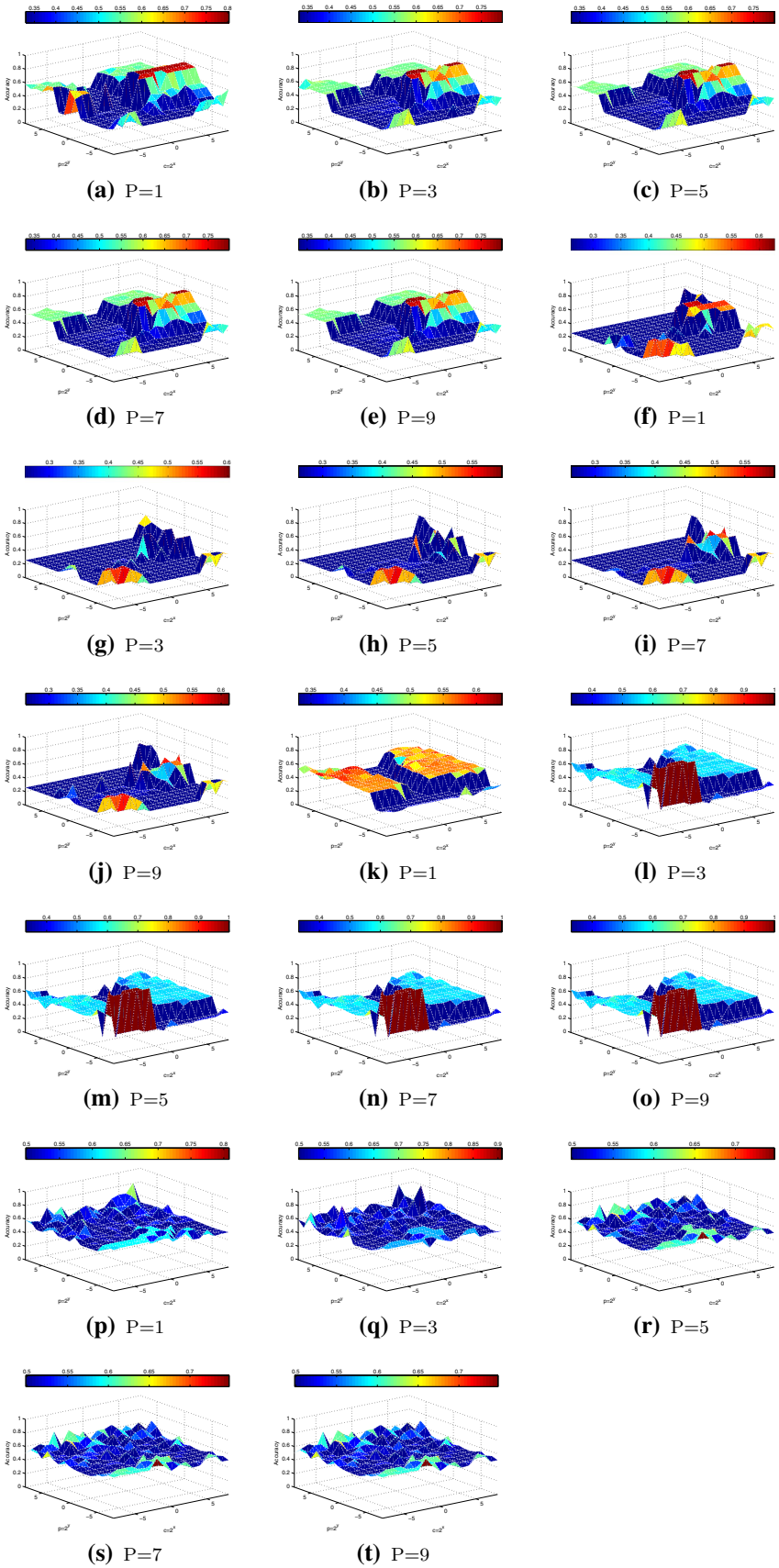


(e) Jain



(f) Flame

Fig. 7 Relationship of parameters c, P, and clustering accuracy for linear cases



(a) P=1 **(b)** P=3 **(c)** P=5

(d) P=7 **(e)** P=9 **(f)** P=1

(g) P=3 **(h)** P=5 **(i)** P=7

(j) P=9 **(k)** P=1 **(l)** P=3

(m) P=5 **(n)** P=7 **(o)** P=9

(p) P=1 **(q)** P=3 **(r)** P=5

(s) P=7 **(t)** P=9

Fig. 8 Relationship of parameters c, p, P, and clustering accuracy for nonlinear cases on Seeds:(a)-(e), Glass:(f)-(j), Spiral:(k)-(o), Jain:(p)-(t)

as [52]. Also, extending kPPC by using other plane-based classifiers such as [53, 54] and by considering fuzzy clustering algorithm such as fuzzy linear c-means [55], fuzzy c-lines [56], and fuzzy c-varieties [57] are also interesting.

Acknowledgments This work is supported by the National Natural Science Foundation of China (Nos. 11201426, 11371365, and 11501310), the Zhejiang Provincial Natural Science Foundation of China (Nos. LY15F030013, LQ14G010004, and LY16A010020), the National Statistical Science Research Project of China (No. 2013LZ13), and the Natural Science Foundation of Inner Mongolia Autonomous Region of China (No. 2015BS0606).

References

1. Han J, Kamber M (2006) Data mining concepts and techniques. Morgan Kaufmann, San Francisco
2. Wang Z, Shao Y, Bai L et al (2015) Twin support vector machine for clustering. IEEE Trans Neural Netw Learn Sys 26(10):2583–2588
3. Anderberg M (1973) Cluster analysis for applications. Academic Press, New York
4. Aldenderfer M, Blashfield R (1985) Cluster analysis. Sage Publications, Los Angeles
5. Andrews H (1972) Introduction to mathematical techniques in pattern recognition. Wiley, New York
6. Celeux G, Govaert G (1995) Gaussian parsimonious clustering models. Pattern Recognit 28(5):781–793
7. Jain A, Dubes R (1988) Algorithms for clustering data. Englewood Cliffs, NJ
8. Fisher D (1987) Knowledge acquisition via incremental conceptual clustering. Mach Learn 2(2):139–172
9. Hassoun M (1995) Fundamentals of artificial neural networks. MIT, Cambridge
10. Bradley P, Mangasarian O, Street W (1997) Clustering via concave minimization. Adv Neural Inf Process Syst 9:368–374
11. Rao M (1987) Cluster analysis and mathematical programming. Am Stat Assoc 66(335):622–626
12. Selim S, Ismail M (1984) K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. IEEE Trans Pattern Anal Mach Intell PAMI 6(1):81–87
13. Bezdek JC, Hathaway RJ, Sabin MJ, Tucker WT (1987) Convergence theory for fuzzy c-means: counterexamples and repairs. Syst Man Cybern IEEE Trans 17(5):873–877
14. Mangasarian O, Wild E (2006) Multisurface proximal support vector classification via generalize eigenvalues. IEEE Trans Pattern Anal Mach Intell 28(1):69–74
15. Bradley P, Mangasarian O (2000) k-Plane clustering. J Glob Optim 16(1):23–32
16. Tseng P (2000) Nearest q-flat to m points. J Optim Theory Appl 105(1):249–252
17. Amaldi E, Coniglio S (2013) A distance-based point-reassignment heuristic for the k-hyperplane clustering problem. Eur J Oper Res 227(1):22–29
18. Rahman MA, Islam MZ, Bossomaier T (2014) Denclust: a density based seed selection approach for k-means. Artif Intell Soft Comput 8468:784–795
19. Bezdek JC, Pal NR (1998) Some new indexes of cluster validity. Syst Man Cybernet Part B Cybern IEEE Trans 28(3):301–315
20. Li C, Kuo B, Chin T (2011) Lda-based clustering algorithm and its application to an unsupervised feature extraction. Fuzzy Syst IEEE Trans 19(1):152–163
21. Pang Y, Wang S, Yuan Y (2014) Learning regularized lda by clustering. Neural Netw Learn Syst IEEE Trans 25(12):2191–2201
22. Yang ZM, Guo YR, Li CN, Shao YH (2015) Local k-proximal plane clustering. Neural Comput Appl 26(1):199–211
23. Shao Y, Deng N, Chen W, Wang Z (2013) Improved generalized eigenvalue proximal support vector machine. IEEE Signal Process Lett 20(3):213–216
24. Shao Y, Zhang C, Wang X, Deng N (2011) Improvements on twin support vector machines. IEEE Trans Neural Netw 22(6):962–968
25. Shao Y, Chen W, Deng N (2014) Nonparallel hyperplane support vector machine for binary classification problems. Inf Sci 263:22–35
26. Qi Z, Tian Y, Shi Y (2012) Twin support vector machine with universum data. Neural Netw 36:112–119
27. Qi Z, Tian Y, Shi Y (2012) Robust twin support vector machine for pattern classification. Pattern Recognit 46(1):305–316
28. Han J, Kamber M (2006) Data mining: concepts and techniques, 2nd edn. China Machine Press, Beijing
29. Ding C, He X (2004) K-means clustering via principal component analysis, in: Proceedings of the twenty-first international conference on machine learning, ACM, p 29
30. Scarborough J (1958) Numerical mathematical analysis, 4th edn. Johns Hopkins Press, New York
31. Deng N, Tian Y, Zhang C (2013) Support vector machines: optimization based theory, algorithms, and extensions. CRC Press, Boca Raton
32. Naldi M, Campello R (2014) Evolutionary k-means for distributed datasets. Neurocomputing 127(3):30–42
33. Bradley P, Fayyad U (1998) Refining initial points for k-means clustering, in: Proceedings of the 15th international conference on machine learning (ICML98), pp. 91–99
34. Fayyad U, Reina C, Bradley B (1998) Initialization of iterative refinement clustering algorithms In: Proc 14th Intl Conf on machine learning (ICML), pp. 194–198
35. Shao Y-H, Bai L, Wang Z, Hua X-Y, Deng N-Y (2013) Proximal plane clustering via eigenvalues. Proc Comput Sci 17:41–47
36. Luxburg U (2007) A tutorial on spectral clustering. Stat Comput 17(4):395–416
37. Hathaway RJ, Bezdek JC, Huband JM (2005) Kernelized non-euclidean relational c-means algorithms. Neural Parallel Sci Comput 13(3):305–326
38. Scholköpf B, Smola A (2002) Learning with kernels. MIT Press, Cambridge, MA
39. Shao Y, Deng N (2012) A coordinate descent margin based-twin support vector machine for classification. Neural Netw 25:114–121
40. Qi Z, Tian Y, Shi Y (2012) Laplacian twin support vector machine for semi-supervised classification. Neural Netw 35:46–53
41. Shao Y-H, Wang Z, Chen W-J, Deng N-Y (2013) A regularization for the projection twin support vector machine. Knowl Based Syst 37:203–210
42. Blake CL, Merz CJ (199 8) UCI repository for machine learning databases. http://www.ics.uci.edu/mlearn/MLRepository.html. Accessed Jan 2015
43. Derya B, Alp K (2007) ST-DBSCAN: an algorithm for clustering spatial–temporal data. Data Knowl Eng 60(1):208–221
44. Zhou A, Zhou S, Cao J, Fan Y, Hu Y (2000) Approaches for scaling DBSCAN algorithm to large spatial databases. J Comput Sci Technol 15(6):509–526
45. The MathWorks Inc (1994–2001) Matlab, User's guide. http://www.mathworks.com
46. Halkidi M, Batistakis Y, Vazirgiannis M (2001) On clustering validation techniques. Intell Inf Syst J 17:107–145

47. Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mac Learn Res 7(1):1–30
48. Hodges J Jr, Lehmann EL (1956) The efficiency of some nonparametric competitors of the t-test. Ann Math Stat 27(2):324–335
49. Hollander M, Wolfe D, Chicken E (1973) Nonparametric statistical methods, 2nd edn. Wiley, New York
50. Wang Y, Jiang Y, Wu Y, Zhou Z (2011) Localized k-flats. In: Proceedings of the twenty-fifth AAAI conference on artificial intelligence, pp. 525–530
51. Huang P, Zhang D (2010) Locality sensitive c-means clustering algorithms. Neurocomputing 73:2935–2943
52. Yang B, Chen S (2010) Sample-dependent graph construction with application to dimensionality reduction. Neurocomputing 74:301–314
53. Tian Y, Shi Y, Liu X (2012) Recent advances on support vector machines research. Technol Econ Dev Econ 18(1):5–33
54. Shao Y-H, Deng N-Y, Chen W-J (2013) A proximal classifier with consistency. Knowl Based Syst 49:171–178
55. Bezdek JC, Gunderson R, Ehrlich R, Meloy T (1978) On the extension of fuzzy k-means algorithms for detection of linear clusters. Decision and control including the 17th symposium on adaptive processes 17(1):1438–1443
56. Bezdek JC, Coray C, Gunderson R, Watson J (1981) Detection and characterization of cluster substructure i. linear structure: fuzzy c-lines. SIAM J Appl Math 40(2):339–357
57. Bezdek JC, Coray C, Gunderson R, Watson J (1981) Detection and characterization of cluster substructure ii. fuzzy c-varieties and complex combinations thereof. SIAM J Appl Math 40(2):358–372