

# An anchor-based spectral clustering method\*

Qin ZHANG<sup>1,2</sup>, Guo-qiang ZHONG<sup>†‡1</sup>, Jun-yu DONG<sup>1</sup>

<sup>1</sup>Department of Computer Science and Technology, Ocean University of China, Qingdao 266100, China

<sup>2</sup>Science and Information College, Qingdao Agricultural University, Qingdao 266109, China

<sup>†</sup>E-mail: gqzhong@ouc.edu.cn

Received Apr. 17, 2017; Revision accepted Aug. 15, 2017; Crosschecked Nov. 27, 2018

**Abstract:** Spectral clustering is one of the most popular and important clustering methods in pattern recognition, machine learning, and data mining. However, its high computational complexity limits it in applications involving truly large-scale datasets. For a clustering problem with  $n$  samples, it needs to compute the eigenvectors of the graph Laplacian with  $O(n^3)$  time complexity. To address this problem, we propose a novel method called anchor-based spectral clustering (ASC) by employing anchor points of data. Specifically,  $m$  ( $m \ll n$ ) anchor points are selected from the dataset, which can basically maintain the intrinsic (manifold) structure of the original data. Then a mapping matrix between the original data and the anchors is constructed. More importantly, it is proved that this data-anchor mapping matrix essentially preserves the clustering structure of the data. Based on this mapping matrix, it is easy to approximate the spectral embedding of the original data. The proposed method scales linearly relative to the size of the data but with low degradation of the clustering performance. The proposed method, ASC, is compared to the classical spectral clustering and two state-of-the-art accelerating methods, i.e., power iteration clustering and landmark-based spectral clustering, on 10 real-world applications under three evaluation metrics. Experimental results show that ASC is consistently faster than the classical spectral clustering with comparable clustering performance, and at least comparable with or better than the state-of-the-art methods on both effectiveness and efficiency.

**Key words:** Clustering; Spectral clustering; Graph Laplacian; Anchors

<https://doi.org/10.1631/FITEE.1700262>

**CLC number:** TP311

## 1 Introduction

Spectral clustering is one of the most popular clustering algorithms in many related areas due to its simplicity and high accuracy, such as machine learning, image processing, and data mining, and has solid theoretical support known as spectral theory.

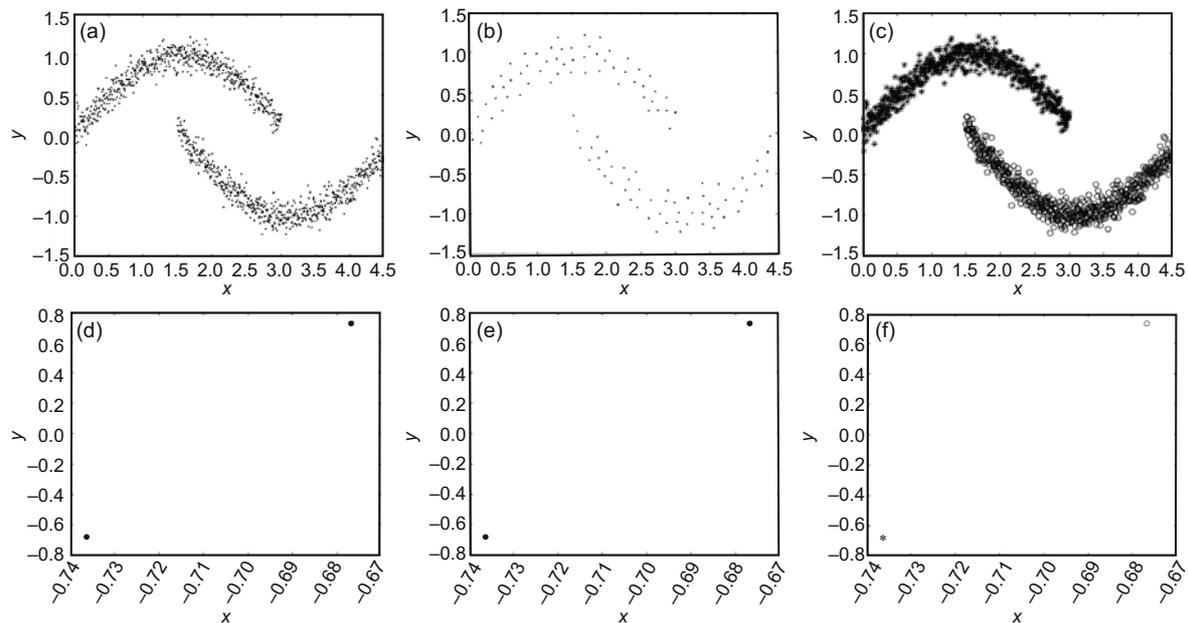
<sup>‡</sup> Corresponding author

\* Project supported by the National Natural Science Foundation of China (Nos. 61271405, 61403353, 41576011, and U1706218), the PhD Program Foundation of the Ministry of Education of China (No. 20120132110018), the International Science & Technology Cooperation Program of China (No. 2014DFA10410), the Science and Technology Program of Qingdao, China (No. 17-3-3-20-nsh), and the Fundamental Research Funds for the Central Universities, China

 ORCID: Guo-qiang ZHONG, <http://orcid.org/0000-0002-2952-6642>

© Zhejiang University and Springer-Verlag GmbH Germany, part of Springer Nature 2018

The main idea of spectral clustering is to model the data points by a graph, whose vertices and edges represent the data points and the similarity between pairwise data points, respectively. The new representation of data can be obtained using the eigenvectors of the graph Laplacian matrix. This new, often low-dimensional and more separable, representation is known as spectral embedding of data. Running  $K$ -means on the low-dimensional representation usually leads to clustering results better than that in the original space. Details about spectral clustering can be found in several papers and tutorials (Shi and Malik, 2000; Ng et al., 2002; von Luxburg, 2007; Jia et al., 2014). Fig. 1 shows the classical two-moon example and the two-dimensional (2D) spectral embedding of the original data points. It can be seen that



**Fig. 1** Two-moon dataset (a) and its 2D spectral embedding  $U$  (eigenvectors of  $W$ ) (d); anchor set (b) and its 2D spectral embedding  $U_a$  (eigenvectors of  $W_a$ ) (e); recovered spectral clustering of the whole dataset (c) and clustering result of  $PU_a$  (f). Markers of different shapes stand for different clusters

spectral embedding preserves the intrinsic structure of the data and causes data belonging to the same cluster to be as close as possible.

The spectral clustering method is easy to implement and outperforms traditional clustering methods in many applications. Hence, it has been widely studied for many problems. For example, Xiao et al. (2016) used the spectral clustering method to partition a large network into several small software-defined networking (SDN) domains to solve the SDN deployment problem in wide area networks or large-scale networks. Li et al. (2016) proposed a spectral clustering method to address the high dimensionality and sparsity of the annotating data, and solved the tag clustering problems in social tagging systems.

The following describes some recent research in spectral clustering. Yang et al. (2011) used a nonnegative constraint to relax the elements of the cluster indicator matrix for spectral clustering. Liu et al. (2013) proposed an efficient clustering algorithm for large-scale graph data, whose key idea was to compress the original graph into a sparse bipartite graph by repeatedly generating a small number of super nodes connected to the regular nodes. Spectral clustering was then performed on the bipartite graph instead. Xia et al. (2014) proposed a

multi-view spectral clustering method based on low-rank and sparse decomposition. Tian et al. (2014) employed deep learning in graph clustering, which learned a non-linear embedding of the original graph by a stacked autoencoder, and then ran the  $K$ -means algorithm on the embedding to obtain clustering results. Chang et al. (2015) proposed a novel convex formulation of spectral shrunk clustering, which contributed to more precise structural information for clustering based on the low-dimensional space.

Many other studies tried to learn the graph Laplacian matrix instead of the predefined graph Laplacian matrix using the Gaussian function. Yang et al. (2010) proposed to learn a new Laplacian matrix by exploiting both manifold structure and local discriminant information. Yang et al. (2012) proposed to learn a robust Laplacian matrix for data ranking.

However, the main problem in spectral clustering is the computational complexity. It needs to compute the smallest  $k$  eigenvectors of the graph Laplacian, which has  $O(n^3)$  time complexity. This limits the spectral clustering method in applications involving truly large-scale datasets. Many techniques have been proposed to deal with this bottleneck, which can be divided into three basic categories.

The first category is based on the approximation to the eigenvectors. Fowlkes et al. (2004) adopted the classical Nyström method to efficiently compute the numerical solution of the eigenfunction problem, whereas Boutsidis et al. (2015) used an iterative algorithm called the power method to approximate the eigenvectors.

The second category is based on sub-sampling a small subset of the original dataset, often called anchors or landmarks. The idea is to reduce the size of the problem based on sampling techniques. The main problem in this category is how to select the subset. Yan et al. (2009) provided a general framework for fast approximate spectral clustering. They developed two concrete instances based on local  $K$ -means clustering and random projection trees, respectively. Wang et al. (2009) examined and proposed three schemes for approximate spectral clustering, and made an empirical comparison of those schemes in combination with four sampling strategies. Chen and Cai (2011) proposed a novel approach for large-scale spectral clustering based on landmarks. A novel algorithm called FURS was proposed in Mall et al. (2013a), which greedily selects nodes with high-degree centrality from a given graph. Mall et al. (2013b) selected a smaller subgraph that could preserve the overall graph structure to construct the large kernel matrix, and used the kernel spectral clustering method to detect the community in big data networks. Li et al. (2015) used the landmarks to develop a constrained spectral clustering algorithm, which is scalable to handle moderate and large datasets. Zhang et al. (2016) proposed an incremental sampling method to select landmarks one by one.

The third category is related to parallelization. Song et al. (2008) and Chen et al. (2011) parallelized both memory and computation on distributed computers. The size of data in their experiments reached hundreds of thousands or even millions. Mall et al. (2014) proposed a distributed environment to convert big data into a nearest-neighbor graph.

In this study, we propose a novel spectral clustering algorithm called anchor-based spectral clustering (ASC), which is motivated by anchor-based machine learning methods (Liu et al., 2010). It focuses on how to approximate the clustering result of the original data based on the eigenvectors of the similarity matrix constructed with the anchor points.

Specifically,  $m$  ( $m \ll n$ ) anchor points are selected from the dataset, which can approximately retain the intrinsic (manifold) structure of the original data. Then a mapping matrix from data to the anchor points is constructed. Using this mapping matrix, it is easy and efficient to approximate the spectral embedding of the original data. The proposed method scales linearly with the size of data with low performance degradation. Experimental results show that the proposed method performs significantly better than classical spectral clustering in terms of runtime, and achieves better or at least comparable performance compared to the state-of-the-art methods.

Note that our proposed method belongs to the second category mentioned above, but it is different from the previous methods in the second category. Fig. 2 illustrates the flow diagram of the proposed ASC algorithm. It can be seen from Fig. 2 that the proposed method does not focus on how to form a similarity matrix from anchors whose eigenvectors can be easily computed. In contrast, it recovers the clustering structure of the original data using the data-anchor mapping matrix. Because its computational complexity is linear with respect to the amount of data,  $n$ , the ASC algorithm is much more efficient than the original spectral clustering algorithm (Ng et al., 2002). The contributions of this paper are summarized as follows:

1. We propose an anchor-based spectral clustering algorithm, which is much more efficient than the original spectral clustering algorithm in Ng et al. (2002).
2. We build a linear relationship between the spectral embedding of the original data and that of the anchors.
3. We prove that the data-anchor mapping matrix preserves the clustering structure.

## 2 Background

In this section, we briefly review the spectral clustering algorithm, the methods to deal with the time complexity bottleneck of spectral clustering, and the anchor graph.

### 2.1 Spectral clustering

Given a dataset  $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]^T \in \mathbb{R}^{n \times d}$  grouped into  $K$  clusters, Ng et al. (2002) proposed the following algorithm named spectral clustering:

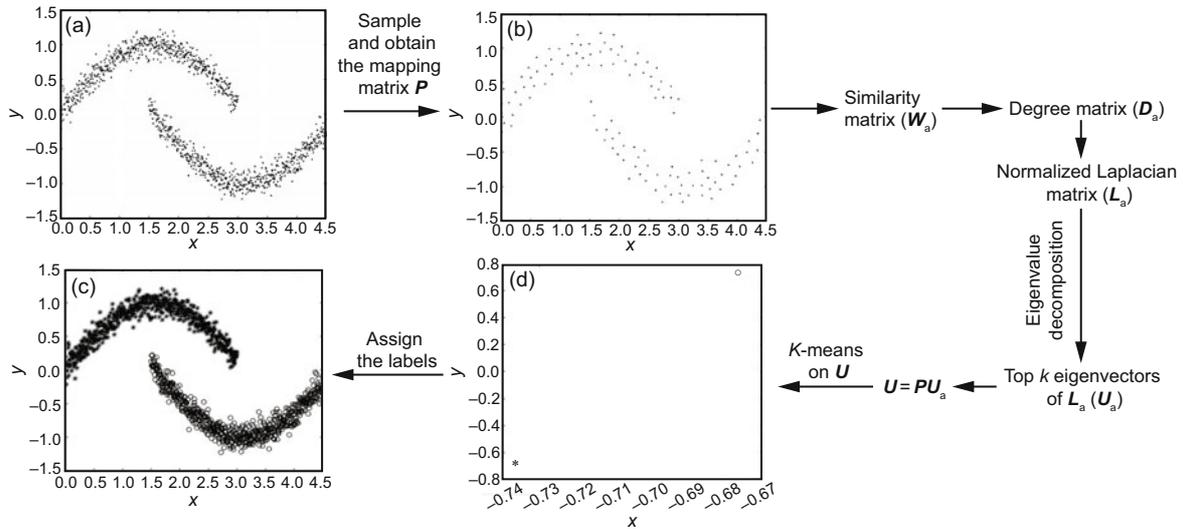


Fig. 2 Framework of the proposed anchor-based spectral clustering algorithm: (a) two-moon dataset; (b) anchor set; (c) recovered spectral clustering of the whole dataset; (d)  $k$  clusters of  $U$

Step 1: Construct the similarity matrix  $W \in \mathbb{R}^{n \times n}$  defined by a Gaussian kernel function as  $W_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2)$  ( $i \neq j$ ),  $W_{ii} = 0$ , and  $\sigma$  is given by the user.

Step 2: Construct the diagonal degree matrix  $D \in \mathbb{R}^{n \times n}$  as  $D_{ii} = \sum_i W_{ij}$ , and then normalize  $W$  as  $\tilde{W} = D^{-1/2}WD^{-1/2}$ . Here,  $L = D - W$  is the graph Laplacian matrix and  $\tilde{L} = I - \tilde{W}$  is the normalized graph Laplacian matrix.

Step 3: Find eigenvectors corresponding to the largest  $k$  eigenvalues of  $\tilde{W}$  (eigenvectors corresponding to the largest  $k$  eigenvalues of  $\tilde{W}$  are the same as that corresponding to the smallest  $k$  eigenvalues of  $\tilde{L}$ ) and assign them as columns to form the matrix  $Y$ , and then normalize each of  $Y$ 's rows to have unit length  $\tilde{Y}_{ij} = Y_{ij}/(\sum_j Y_{ij}^2)^{1/2}$ .

Step 4: Apply  $K$ -means on the rows of  $\tilde{Y}$  into  $K$  clusters and use this clustering result to cluster the original data accordingly.

Following the work of Ng et al. (2002), we use the same value for the number of selected eigenvectors of  $\tilde{W}$  and that of the clusters to be grouped in this study, i.e.,  $K = k$ . Furthermore, we adopt Ng, Jordan, and Weiss's spectral clustering algorithm as our baseline and name it SC\_NJW.

### 2.2 Acceleration for spectral clustering

In this subsection, we introduce two representative methods, power iteration clustering (PIC) and landmark-based spectral clustering (LSC), from two

different categories, which are used for comparison in the experiments.

Based on the power method, Lin and Cohen (2010) presented an algorithm to accelerate the eigenvalue decomposition and named the new algorithm power iteration clustering (PIC). They used the power method to iteratively obtain the approximate eigenvectors as the left singular vectors of the matrix  $B' = (\tilde{W}\tilde{W}^T)^p\tilde{W}S = \tilde{W}^{(2p+1)}S$ , where  $S \in \mathbb{R}^{n \times k}$  is a matrix with independent and identically distributed random Gaussian initialization and  $p$  is a positive integer. Boutsidis et al. (2015) proved that solving the  $K$ -means clustering problem on the approximate eigenvectors obtained via the power method gave an additive-error approximation to solving the  $K$ -means problem on the true eigenvectors.

Chen and Cai (2011) used a different way to reduce the size of the problem and proposed a method called landmark-based spectral clustering (LSC). They selected  $m$  ( $m \ll n$ ) representative data points as landmarks and represented the original data points as the linear sparse combinations of these landmarks, of which the weights can be denoted as  $Z \in \mathbb{R}^{m \times n}$ . Then the normalization of  $Z$  was computed, i.e.,  $\hat{Z} = D^{-1/2}Z$ , where  $D$  is the degree matrix of  $Z$ . Finally, singular value decomposition was applied on  $\hat{Z}$  as  $\hat{Z} = V_l \Sigma V_r^T$ , where  $\Sigma$  is a diagonal matrix with the singular values on the diagonal and  $V_l$  and  $V_r$  are the left and right

singular vector matrices, respectively. It can be proved that  $\mathbf{V}_r$  is the eigenvector matrix of the similarity matrix  $\mathbf{W} = \hat{\mathbf{Z}}^T \hat{\mathbf{Z}}$ . Hence, spectral clustering on  $\mathbf{V}_r$  is equivalent to that on the original data.

In our experiments, we use the classical spectral clustering proposed in Ng et al. (2002) as the baseline, and compare our algorithm with the two methods mentioned above, i.e., PIC and LSC. PIC belongs to the first category, which tries to approximate the eigenvectors rapidly. Our algorithm, ASC, and the compared algorithm, LSC (belonging to the second category), use a small subset of the data. However, their ideas are quite different. LSC constructs a similarity matrix based on anchors, whose eigenvectors can be easily computed. However, our algorithm obtains the clustering result of the original data from anchors using a data-anchor mapping matrix, which preserves the clustering structure.

### 2.3 Anchor graph

The idea of anchor graph is from Delalleau et al. (2005) and Zhu and Lafferty (2005). They worked with large-scale data and made the label prediction function be a weighted average of the labels on a subset of anchor (landmark) samples. As such, the label prediction function  $f$  can be represented by a subset  $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_m]^T$ , in which each  $\mathbf{a}_j \in \mathbb{R}^d$  is an anchor point:

$$f(\mathbf{x}_i) = \sum_{j=1}^m Z_{ij} f(\mathbf{a}_j), \quad (1)$$

where  $Z_{ij}$  is the data-adaptive weight. If we define two vectors  $\mathbf{f} = [f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_n)]^T$  and  $\mathbf{f}_a = [f(\mathbf{a}_1), f(\mathbf{a}_2), \dots, f(\mathbf{a}_m)]^T$ , then Eq. (1) can be rewritten as

$$\mathbf{f} = \mathbf{Z} \mathbf{f}_a, \quad \mathbf{Z} \in \mathbb{R}^{n \times m}, \quad m \ll n. \quad (2)$$

This formula reduces the solution space of unknown labels from larger  $\mathbf{f}$  to smaller  $\mathbf{f}_a$ . The problem here is how to choose the anchor points. Liu et al. (2010) suggested using  $K$ -means clustering centers as anchors instead of randomly sampled points because  $K$ -means clustering centers have a stronger representation power to adequately cover the full dataset.

Another problem here is how to design the matrix  $\mathbf{Z}$ . Liu et al. (2010) proposed a method called local anchor embedding (LAE) to reconstruct any data point  $\mathbf{x}_i$  as a convex combination of its closest

anchors, while Chen and Cai (2011) used a predefined method to compute  $\mathbf{Z}$ .

Using matrix  $\mathbf{Z}$ , the adjacency matrix can be designed as  $\mathbf{W} = \mathbf{Z} \mathbf{A}^{-1} \mathbf{Z}^T$ , in which the diagonal matrix  $\mathbf{A} \in \mathbb{R}^{m \times m}$  is defined as  $A_{kk} = \sum_{i=1}^n Z_{ik}$  (Liu et al., 2010). This is where the name anchor graph comes from because a graph can be fully represented by its adjacency matrix.

## 3 Anchor-based spectral clustering

In this section, we first introduce the notations used in the remainder of this paper. Then we describe the key steps of the proposed ASC algorithm in detail, including selecting anchors, local anchor embedding, and approximate clustering. Finally, we theoretically analyze the ASC algorithm and prove its correctness.

### 3.1 Notations

Given a set of data points denoted as  $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]^T \in \mathbb{R}^{n \times d}$ , the similarity matrix  $\mathbf{W}$  can be computed by the Gaussian kernel as  $W_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2)$  ( $i \neq j$ ) and  $W_{ii} = 0$ , where  $\sigma$  is a parameter to be tuned.  $\tilde{\mathbf{W}}$  denotes the normalization of  $\mathbf{W}$  as  $\tilde{\mathbf{W}} = \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$ , where  $\mathbf{D}$  is the diagonal degree matrix, i.e.,  $D_{ii} = \sum_j W_{ij}$ .  $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_m]^T \in \mathbb{R}^{m \times d}$  stands for the anchor set and  $\mathbf{W}_a$  is the corresponding similarity matrix computed by the Gaussian kernel on the anchors.  $\tilde{\mathbf{W}}_a$  denotes the normalization of  $\mathbf{W}_a$ . We use  $\mathbf{X}$  and  $\mathbf{A}$  to represent the data matrix and anchor matrix with one sample in a row, which are easy to distinguish from the context.

The eigenvalue decompositions of  $\tilde{\mathbf{W}}$  and  $\tilde{\mathbf{W}}_a$  are  $\tilde{\mathbf{W}} = \mathbf{U} \mathbf{\Sigma} \mathbf{U}^T$  and  $\tilde{\mathbf{W}}_a = \mathbf{U}_a \mathbf{\Sigma}_a \mathbf{U}_a^T$ , respectively. Our goal is to obtain a similar clustering result from eigenvectors  $\mathbf{U}_a$  of  $\tilde{\mathbf{W}}_a$  without computing  $\tilde{\mathbf{W}}$  or its eigenvalue decomposition.

Table 1 summarizes these notations.

Table 1 Notations

Symbol	Description
$\mathbf{X}$	Data matrix with a data point in each row
$\mathbf{W}$	Similarity matrix of $\mathbf{X}$
$\mathbf{U}$	Eigenvector matrix of $\mathbf{W}$
$\mathbf{A}$	Anchor matrix with an anchor point in each row
$\mathbf{W}_a$	Similarity matrix of $\mathbf{A}$
$\mathbf{U}_a$	Eigenvector matrix of $\mathbf{W}_a$

### 3.2 Selecting anchors

How to effectively select the anchors is critical in representing the intrinsic structure of the whole dataset. Random sampling is a simple and common method (Fowlkes et al., 2004), but the representational ability of the randomly selected anchors is not satisfactory with respect to the clustering tasks. Yan et al. (2009) and Chen and Cai (2011) used  $K$ -means centers with a larger  $K$ . In this study, we use a probabilistic sampling method, which was employed in the initialization step of  $K$ -means++ (Arthur and Vassilvitskii, 2007). Let  $\text{dist}(\mathbf{a}, \mathbf{X})$  denote the shortest distance from anchor point  $\mathbf{a}$  to dataset  $\mathbf{X}$ . The anchors can be selected as follows:

Step 1: Choose initial data point  $\mathbf{a}$  uniformly at random from  $\mathbf{X}$ , and set  $\mathbf{A} = \{\mathbf{a}\}$ .

Step 2: Choose the next data point  $\mathbf{a} \in \mathbf{X}$  with the probability  $p(\mathbf{a}) = \frac{[\text{dist}(\mathbf{a}, \mathbf{A})]^2}{\sum_{\mathbf{a}' \in \mathbf{X}} [\text{dist}(\mathbf{a}', \mathbf{A})]^2}$  and add it to  $\mathbf{A}$ .

Step 3: Repeat the above two steps until a total of  $m$  anchor points have been chosen.

Here,  $\mathbf{A}$  stands for the anchor set. We can set every anchor as a row to form an anchor matrix, and denote it as  $\mathbf{A}$ . Thus, we use  $\mathbf{A}$  to denote the anchor set and the anchor matrix in the rest of this paper.

### 3.3 Local anchor embedding

The data-anchor mapping problem can be formulated as follows:

$$\begin{aligned} \min_{\mathbf{P} \in \mathbb{R}^{n \times m}} \quad & J(\mathbf{P}) = \frac{1}{2} \|\mathbf{X} - \mathbf{P}\mathbf{A}\|^2 \\ \text{s.t.} \quad & P_{ij} \geq 0, \mathbf{P}\mathbf{1} = \mathbf{1}, \forall i, \forall j, \end{aligned} \quad (3)$$

where  $\mathbf{X} \in \mathbb{R}^{n \times d}$  stands for the data matrix in which every row is a data sample,  $\mathbf{A} \in \mathbb{R}^{m \times d}$  is the anchor matrix in which every row is an anchor, and  $\mathbf{P} \in \mathbb{R}^{n \times m}$  is the data-anchor mapping matrix to be learned. Any standard quadratic programming solver can be used to solve Eq. (3), but to achieve faster optimization, we adopt the local anchor embedding method described in Liu et al. (2010). The concrete algorithm is briefly described as follows:

For every data point  $\mathbf{x}_i$ :

Step 1: Find  $s$  nearest neighbors of  $\mathbf{x}_i$  in the anchor set and save the index set as  $\langle i \rangle$ .

Step 2: Define the objective function  $g(\mathbf{p}) = \|\mathbf{x}_i - \mathbf{A}_{i, \langle i \rangle} \mathbf{p}\|^2 / 2$  and its gradient  $\nabla g(\mathbf{p}) = \mathbf{A}_{i, \langle i \rangle}^T \mathbf{A}_{i, \langle i \rangle} \mathbf{p} - \mathbf{A}_{i, \langle i \rangle}^T \mathbf{x}_i$ .

Step 3: Repeat until convergence; i.e., update  $\mathbf{p}_i^{(t+1)} = \Pi_S(\mathbf{p}_i^{(t)} - \eta_t \nabla g(\mathbf{p}_i^{(t)}))$ , where the symbol  $\Pi_S$  denotes the simplex projection operator which is formulated as  $\Pi_S(\mathbf{p}) = \arg \min_{\mathbf{p}' \in S} \|\mathbf{p}' - \mathbf{p}\|$ .

Step 4: Set  $\mathbf{P}_{i, \langle i \rangle} = \mathbf{p}_i^T$  and  $\mathbf{P}_{i, \langle \bar{i} \rangle} = \mathbf{0}$  for the remaining entries of  $\mathbf{P}$ .

We notice that matrix  $\mathbf{P}$  captures the data-anchor relationship, and use this matrix  $\mathbf{P}$  to recover the clustering structure from anchors.

### 3.4 Approximate clustering

So far we have obtained the anchor matrix  $\mathbf{A}$  and data-anchor mapping matrix  $\mathbf{P}$ . Applying spectral clustering on  $\mathbf{A}$ , we can obtain the clustering result denoted by label vector  $\mathbf{f}_a \in \mathbb{R}^{m \times 1}$ . Then using Eq. (1), we can obtain the approximate spectral clustering result for the original data as  $\mathbf{f} = \mathbf{P}\mathbf{f}_a$ .  $\mathbf{f} \in \mathbb{R}^{n \times 1}$  is a label vector that contains float labels, and we need to round them to the nearest integers.

Note that as the low-dimensional embedding of  $\mathbf{X}$ , the eigenvectors  $\mathbf{U}_{(k)} \in \mathbb{R}^{n \times k}$  corresponding to the  $k$  largest eigenvalues of  $\tilde{\mathbf{W}}$  completely retain the clustering structure of the original data  $\mathbf{X}$ . Intuitively, the clustering structure can be recovered by  $\mathbf{P}\mathbf{U}_{a \langle k \rangle}$  where the eigenvectors  $\mathbf{U}_{a \langle k \rangle}$  are corresponding to the  $k$  largest eigenvalues of  $\tilde{\mathbf{W}}_a$  as the low-dimensional embedding of  $\mathbf{A}$ . We show a two-moon example to demonstrate our findings in Fig. 1. Figs. 1a and 1d are the two-moon dataset and its spectral embedding respectively, Figs. 1b and 1e are the anchor set and its spectral embedding respectively, and Figs. 1c and 1f are the approximate clustering recovered from  $\mathbf{P}\mathbf{U}_a$  and the clustering result of  $\mathbf{P}\mathbf{U}_a$  respectively. Hereinafter, we use  $\mathbf{U}$  and  $\mathbf{U}_a$  instead of  $\mathbf{U}_{(k)}$  and  $\mathbf{U}_{a \langle k \rangle}$  for short.

We can see that  $\mathbf{U}_a$  completely preserves the clustering structure as  $\mathbf{U}$  except for a few magnitude differences, and  $\mathbf{P}\mathbf{U}_a$  preserves the same clustering structure as  $\mathbf{U}$ . Inspired by these observations, we use  $\mathbf{P}\mathbf{U}_a$  to obtain the approximate clustering of the original dataset. We formulate this as

$$K\text{means}(\mathbf{U}) \simeq K\text{means}(\mathbf{P}\mathbf{U}_a), \quad (4)$$

where  $\mathbf{U} \in \mathbb{R}^{n \times k}$  is the eigenvector matrix corresponding to the  $k$  largest eigenvalues of  $\tilde{\mathbf{W}}$  and  $\mathbf{U}_a \in \mathbb{R}^{m \times k}$  is the eigenvector matrix corresponding to the  $k$  largest eigenvalues of  $\tilde{\mathbf{W}}_a$ .  $K\text{means}(\cdot)$  stands for the clustering result obtained by  $K$ -means.

The symbol “ $\simeq$ ” denotes the relationship of approximate equivalence. Eq. (4) is the main finding of this study, which shows that  $K$ -means clustering on  $\mathbf{PU}_a$  is equivalent to  $K$ -means clustering on  $\mathbf{U}$ .

### 3.5 Computational complexity analysis

Supposing that we have  $n$  data points with dimensionality  $d$  and use  $m$  anchors, we need  $O(mnd)$  to select anchors,  $O(m^3)$  to do eigenvalue decomposition on  $\mathbf{W}_a$ ,  $O(smn + s^2Tn)$  to compute the mapping matrix  $\mathbf{P}$ , and  $O(tmnd)$  to do  $K$ -means. Algorithm 1 summarizes our method and Table 2 shows the computational complexity. Note that  $s$  is the number of nearest anchors in LAE,  $T$  is the number of iterations in LAE, and  $t$  is the number of iterations in  $K$ -means ( $n \gg m \gg s$ ).

---

#### Algorithm 1 Anchor-based spectral clustering

---

**Input:** data points  $\{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^d$ , clustering number  $k$ .

**Output:**  $k$  clusters.

- 1: Select  $m$  anchors to form anchor matrix  $\mathbf{A}$
  - 2: Compute the similarity matrix  $\mathbf{W}_a$ , and its eigenvalue decomposition  $\mathbf{U}_a \mathbf{\Sigma}_a \mathbf{U}_a^T$
  - 3: Solve problem (3) by LAE to obtain matrix  $\mathbf{P}$
  - 4: Run  $K$ -means on  $\mathbf{PU}_a$  to obtain  $k$  clusters
- 

**Table 2 Time complexity analysis of the anchor-based spectral clustering method**

Selecting anchors	Eigenvalue decomposition	Computing matrix $\mathbf{P}$	$K$ -means
$O(mnd)$	$O(m^3)$	$O(smn + s^2Tn)$	$O(tmnd)$

### 3.6 Algorithm analysis

Here, we give an analysis of the correctness of the proposed method, which uses data-anchor mapping matrix  $\mathbf{P}$  to recover the spectral clustering of the original dataset from anchor set  $\mathbf{A}$ .

The eigen-decomposition of the normalized similarity matrix  $\tilde{\mathbf{W}}_a$  of anchor set  $\mathbf{A}$  is  $\tilde{\mathbf{W}}_a = \mathbf{U}_a \mathbf{\Sigma}_a \mathbf{U}_a^T$ . We find that  $\mathbf{PU}_a$  can preserve the same clustering structure as  $\mathbf{U}$ , where  $\mathbf{U}_a$  is the eigenvector matrix corresponding to the  $k$  largest eigenvalues of  $\tilde{\mathbf{W}}_a$ ;  $\mathbf{U}$  is the eigenvector matrix corresponding to the  $k$  largest eigenvalues of  $\tilde{\mathbf{W}}$ , which is the normalized similarity matrix of the dataset  $\mathbf{X}$ .

The data matrix  $\mathbf{X}$  can be represented by the anchor matrix  $\mathbf{A}$  and data-anchor mapping matrix

$\mathbf{P}$ , and it can be formulated as

$$\mathbf{X} \simeq \mathbf{PA}, \quad (5)$$

where  $\mathbf{X} \in \mathbb{R}^{n \times d}$ ,  $\mathbf{A} \in \mathbb{R}^{m \times d}$ , and  $\mathbf{P} \in \mathbb{R}^{n \times m}$ . Based on spectral clustering, we obtain the following relationships:

$$\text{cluster}(\mathbf{X}) \simeq \text{cluster}(\mathbf{U}) \simeq K\text{means}(\mathbf{U}), \quad (6)$$

$$\text{cluster}(\mathbf{A}) \simeq \text{cluster}(\mathbf{U}_a) \simeq K\text{means}(\mathbf{U}_a), \quad (7)$$

where  $\text{cluster}(\cdot)$  is a mapping function to represent the true clustering structure.

From Eq. (2), we assume that the clustering function can be seen as a weighed average of the labels on a subset of anchors. It can be formulated as

$$\text{cluster}(\mathbf{X}) \simeq \mathbf{P} \text{cluster}(\mathbf{A}). \quad (8)$$

From Eqs. (5) and (8), we can obtain

$$\text{cluster}(\mathbf{X}) \simeq \text{cluster}(\mathbf{PA}) \simeq \mathbf{P} \text{cluster}(\mathbf{A}). \quad (9)$$

Then we obtain the important property of matrix  $\mathbf{P}$  from Eq. (9) that  $\mathbf{P}$  can preserve the clustering structure. This means that

$$\text{cluster}(\mathbf{PA}) \simeq \mathbf{P} \text{cluster}(\mathbf{A}), \quad (10)$$

where  $\mathbf{A}$  is a sub-matrix of  $\mathbf{X}$ , and  $\mathbf{A}$  corresponds to an anchor set that can retain the intrinsic (manifold) structure of the original dataset.

Then from Eqs. (6), (7), (9), and (10), we can obtain

$$\begin{aligned} \text{cluster}(\mathbf{X}) &\simeq \text{cluster}(\mathbf{U}) \simeq \mathbf{P} \text{cluster}(\mathbf{U}_a) \\ &\simeq \text{cluster}(\mathbf{PU}_a). \end{aligned} \quad (11)$$

As a result, we can use the clustering result of  $\mathbf{PU}_a$  to obtain the clustering of the original dataset.

## 4 Experiments

In this section, we report the experimental settings and results.

### 4.1 Datasets

We use 10 real-world datasets from the UCI Machine Learning Repository (<http://archive.ics.uci.edu/ml/>) to evaluate the proposed method. These datasets are from different application domains, including medicine, biology, meteorology, physics, image processing, and handwritten character recognition. Table 3 summarizes the properties of these 10 datasets.

**Table 3 Ten datasets used in the experiments**

Dataset	Number of data points	Dimensionality	Number of classes
Protein	116	20	6
Thyroid	215	5	3
Ionosphere	351	34	2
Dermatology	366	34	6
Balance	625	4	3
Yeast	1489	8	10
Segmentation	2310	19	7
Waveform21	5000	21	3
Satimage	6435	36	6
Letter	20 000	16	26

## 4.2 Quality metrics of clustering

There are two categories of quality metrics in clustering. One category is external quality metrics, which need to know the true labels of the dataset. However, clustering is an unsupervised learning task, and in many real-world applications the true labels are unknown and difficult to obtain. Therefore, the other category of quality metrics called internal quality metrics is more suitable.

We evaluate the clustering results using one kind of external quality metric, i.e., purity, and one kind of internal quality metric, i.e., the Davis-Bouldin index (Davies and Bouldin, 1979). We list these quality metrics below. Details about these metrics can be found in the above-mentioned reference.

Purity is the simplest and most commonly used metric to measure clustering performance:

$$\text{Purity}(\Omega, C) = \frac{1}{n} \sum_k \max_j |\omega_i \cap c_j|, \quad (12)$$

where  $\Omega$  is the obtained cluster set,  $C$  is the true cluster set, and  $n$  is the number of data points. A bad clustering has a purity value close to 0, and a good clustering has a purity value close to 1.

The Davis-Bouldin index is based on a ratio of within- and between-cluster distances, defined as

$$\text{DB}(C) = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \{D_{i,j}\}, \quad (13)$$

where  $D_{i,j}$  is the within-to-between cluster distance ratio for the  $i^{\text{th}}$  and  $j^{\text{th}}$  clusters:

$$D_{i,j} = \frac{\bar{d}_i + \bar{d}_j}{d_{i,j}}, \quad (14)$$

where  $\bar{d}_i$  is the average distance between each point in the  $i^{\text{th}}$  cluster and the center of the  $i^{\text{th}}$  cluster,  $\bar{d}_j$

is the average distance between each point in the  $j^{\text{th}}$  cluster and the center of the  $j^{\text{th}}$  cluster, and  $d_{i,j}$  is the Euclidean distance between the centers of the  $i^{\text{th}}$  and  $j^{\text{th}}$  clusters. The optimal clustering solution has the smallest Davies-Bouldin index value.

## 4.3 Experimental settings

We run the experiments under the same environment: Intel® Xeon® CPU E5506 @ 2.13 GHz, 72 GB memory, Windows Server 2012 64-bit operating system, and Matlab version 7.12.0.

We compare our ASC algorithm with two state-of-the-art algorithms called PIC and LSC from two different categories, and use the classical spectral clustering algorithm (Ng et al., 2002) as a baseline, SC\_NJW for short. PIC (Lin and Cohen, 2010) uses the power method to approximate the eigenvectors. LSC (Chen and Cai, 2011) uses a sampling method to do that. The two algorithms are representative of the two categories (introduced in Section 2.2). The codes of PIC (<http://www.cs.cmu.edu/~frank/>) and LSC (<http://www.cad.zju.edu.cn/home/dengcai/Data/Clustering.html>) can be downloaded from the authors' web pages. We use the same number of anchors in LSC and ASC for fair comparison. The original PIC method uses cosine similarity to avoid parameter selection. In our experiments, we use the Gaussian kernel function to compute the similarity in both PIC and ASC, whose parameter  $\sigma$  is chosen by 10-fold cross validation on the validation set. All three algorithms involve  $K$ -means, which depends on initialization. So, for every dataset we run each method 20 times and report the average results.

For further analysis, we use the Friedman test and Wilcoxon signed-rank test to verify if the differences between the four methods are significant (Demšar, 2006). The Wilcoxon signed-rank test is used for comparison of two methods over multiple datasets, and the Friedman test is used for comparison of more methods over multiple datasets. Detailed information about these significant difference tests can be found in Demšar (2006).

## 4.4 Experimental results

Tables 4–6 show the experimental results. Bold numbers in these tables represent the best results in LSC, PIC, and ASC, excluding the baseline.

Table 4 shows the average values of purity (larger values are better). Note that the following statistical test results do not include the dataset “letter”. We could not obtain the result on that dataset by SC\_NJW because the runtime was too long and there was no sufficient memory. The last row of Table 4 is the mean rank (smaller values are better). Based on the Friedman test, the performance differences of the four methods are significant at the 5% level. Furthermore, the performance difference between ASC and PIC is significant, and ASC is comparable to the baseline and LSC. Then we perform pairwise Wilcoxon signed-rank tests between pairs of the four methods. Based on the Wilcoxon test, ASC is comparable to SC\_NJW and LSC, and performs significantly better than PIC at the 5% level.

**Table 4 Average purity of 10 datasets**

Dataset	Average purity			
	SC_NJW (baseline)	LSC	PIC	ASC
Protein	0.6466	0.4582	0.4418	<b>0.5095</b>
Thyroid	0.9721	0.8047	0.8672	<b>0.9419</b>
Ionosphere	0.7236	0.7014	0.6410	<b>0.7236</b>
Dermatology	0.8661	<b>0.8699</b>	0.7258	0.8552
Balance	0.7376	0.6545	0.5894	<b>0.6796</b>
Yeast	0.5480	<b>0.5343</b>	0.4041	0.4686
Segmentation	0.7407	<b>0.7016</b>	0.5497	0.6855
Waveform21	0.5184	0.5826	0.5620	<b>0.5928</b>
Satimage	0.7243	0.7338	0.6781	<b>0.7352</b>
Letter	–	0.3570	0.1230	<b>0.4126</b>
Mean rank	1.6667	2.4444	3.7778	2.1111

The best results are highlighted in boldface

Table 5 shows the average values of the Davis-Bouldin index (smaller values are better). The last row is the mean rank (larger values are better). Based on the Friedman test, the proposed ASC method is comparable to the other three methods.

Table 6 shows the average runtime values (smaller values are better). The last row is the mean rank (larger values are better). Based on the Friedman test, the performance differences of the four methods are significant at the 5% level. Furthermore, the performance difference between SC\_NJW and PIC is significant, and the performance difference between SC\_NJW and ASC is significant. Note that a larger mean rank here means a shorter runtime. From the mean rank, we find that the runtime of SC\_NJW is the largest. Then we perform

**Table 5 Average Davis-Bouldin index of 10 datasets**

Dataset	Average Davis-Bouldin index			
	SC_NJW (baseline)	LSC	PIC	ASC
Protein	2.4215	2.9778	3.0313	<b>2.9632</b>
Thyroid	0.9900	2.3980	<b>0.8295</b>	1.0715
Ionosphere	2.3494	1.5495	<b>1.3282</b>	2.9879
Dermatology	1.5283	1.6269	2.4315	<b>1.5755</b>
Balance	1.7529	<b>1.7431</b>	2.4860	1.8484
Yeast	1.5257	1.6693	3.4283	<b>1.6022</b>
Segmentation	1.3125	1.3534	1.8304	<b>1.1296</b>
Waveform21	1.4905	<b>1.4816</b>	2.1689	1.9026
Satimage	1.3534	1.0545	1.6821	<b>0.9803</b>
Letter	–	2.0089	7.1854	<b>1.8434</b>
Mean rank	3.1111	2.5556	1.6667	2.6667

The best results are highlighted in boldface

**Table 6 Average runtime of 10 datasets**

Dataset	Average runtime (s)			
	SC_NJW (baseline)	LSC	PIC	ASC
Protein	0.0772	<b>0.0218</b>	0.0402	0.0291
Thyroid	0.2530	0.0365	0.0526	<b>0.0284</b>
Ionosphere	0.4657	0.0900	<b>0.0195</b>	0.0379
Dermatology	0.5783	0.0951	0.1950	<b>0.0580</b>
Balance	2.2365	0.1750	0.1364	<b>0.0742</b>
Yeast	21.7880	1.2232	0.5232	<b>0.3882</b>
Segmentation	67.4973	0.9031	<b>0.5142</b>	0.6364
Waveform21	546.9019	2.6208	<b>0.2073</b>	2.2664
Satimage	1181.9291	3.9559	5.3411	<b>3.4922</b>
Letter	–	32.4293	<b>9.8302</b>	51.3016
Mean rank	1.0000	2.5556	2.8889	3.5556

The best results are highlighted in boldface

Friedman tests on the other three methods and find that ASC is comparable to PIC and LSC, and performs significantly better than SC\_NJW.

Note that LSC uses an accelerated version of  $K$ -means, which was written by its authors; for ASC and PIC, we use the  $K$ -means function of Matlab to implement them.

#### 4.5 Parameter sensitivity

There are several parameters in our algorithm. The first is the length scale parameter  $\sigma$  used in the Gaussian kernel function to compute the similarity matrix. The simplest method to choose this parameter, suggested by Ng et al. (2002), is repeatedly running the algorithm several times for a number of values of  $\sigma$  and choosing one with the best performance. In our experiments, we choose the values of  $\sigma$  from set  $\{0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 10\}$ .

The second parameter is the number of anchor points,  $m$ . We try to find the relationship between the number of anchor points and the performance on four datasets, including balance, ionosphere, dermatology, and yeast. For every dataset, we run our algorithm 10 times with the number of anchor points from 10% to 100% of the original data. Fig. 3 shows the average accuracy. We can see that our algorithm is not sensitive to the number of anchor points. Hence, we fix  $m$  at 10% of the original data.

The third parameter is the number of the nearest neighbors  $s$  used to optimize the mapping matrix  $P$ . Similarly, we try to find the relationship between the parameter  $s$  and the performance on the same four datasets as above. For every dataset, we run our algorithm 10 times with  $s$  from 1 to 10. Fig. 4 shows the average accuracy. We can see that our algorithm is not sensitive to the number of nearest neighbors. Therefore, we fix this number at 3 in our experiments.

#### 4.6 Predefined versus learned graph Laplacian matrix

The proposed method adopts the traditional way to use a predefined graph Laplacian matrix by the Gaussian function. However, this leads to a sensitive model with respect to the parameter settings. Some studies tried to learn an adaptive graph Laplacian from data. For example, the LDMGI method (Yang et al., 2010) learns a new graph Laplacian

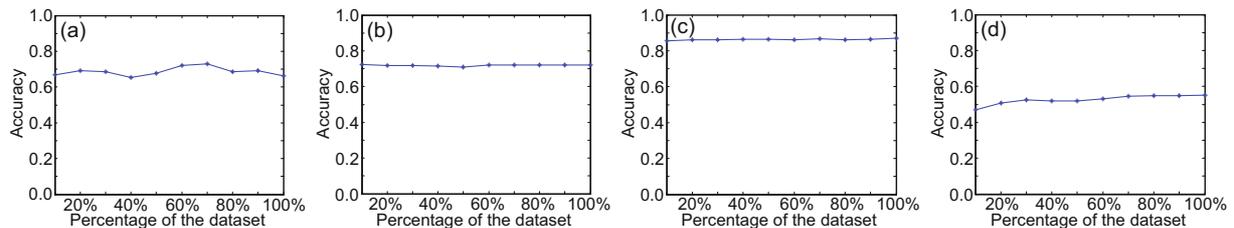
matrix by exploiting both manifold structure and local discriminant information. The main difference between the two methods is how to obtain the graph Laplacian matrix. Table 7 shows the comparison between the proposed method and the LDMGI method. According to Yang et al. (2010), the regularization parameter of LDMGI is chosen from the set  $\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}, 10^0, 10^2, 10^4, 10^6, 10^8\}$ . The  $\sigma$  of ASC is chosen from the same set mentioned above. We do a paired  $t$ -test to check whether the difference between the two sets of results is significant at the 5% significance level. The returned value of the paired  $t$ -test is 0, indicating that the paired  $t$ -test does not reject the null hypothesis at the 5% significance level. That is, the proposed method achieves comparable results to the LDMGI method.

**Table 7 Purity comparison between the LDMGI and ASC methods**

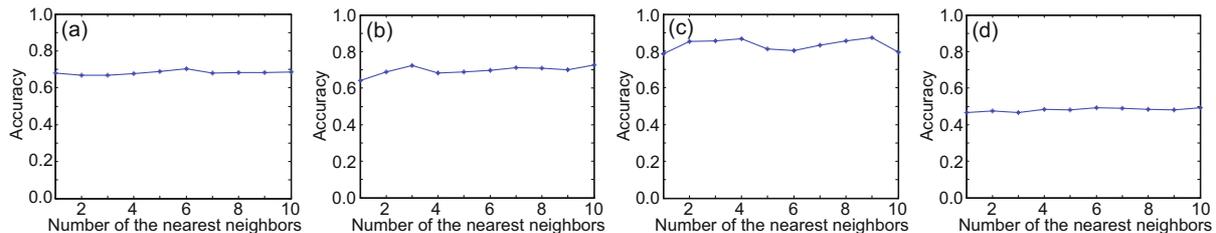
Dataset	Purity	
	LDMGI	ASC
Protein	0.5560	0.5095
Ionosphere	0.7467	0.7236
Waveform21	0.5016	0.5928
Satimage	0.7315	0.7352

## 5 Conclusions and future work

We have focused on the problem of time complexity bottleneck of the spectral clustering method and have proposed an anchor-based spectral



**Fig. 3 Relationship between the number of anchors and the performance of ASC evaluated on four datasets: (a) balance; (b) ionosphere; (c) dermatology; (d) yeast**



**Fig. 4 Relationship between the number of nearest neighbors and the performance of ASC evaluated on four datasets: (a) balance; (b) ionosphere; (c) dermatology; (d) yeast**

clustering method to obtain the approximate clustering. We first selected a small subset of the original dataset as anchors; the rest of the data points can be represented by the linear combination of these anchor points. The data-anchor mapping matrix (i.e., the linear combination coefficient matrix)  $\mathbf{P}$  not only captures the relationship between the data and anchors, but also projects the clustering of anchor points to the original data points. We compared the proposed method, ASC, to two state-of-the-art methods, PIC and LSC, on 10 real-world applications using three evaluation metrics. Experimental results showed that the proposed method performed significantly better than classical spectral clustering in terms of runtime, and achieved better or at least comparable performance (i.e., purity and runtime), compared to the state-of-the-art methods.

However, there is still some work to be done in the future. Why matrix  $\mathbf{P}$  can project the clustering from anchors to data still needs more theoretical analysis and exploration, and the most important problem in our algorithm is how to find the anchors efficiently and rapidly. We will explore more accurate and efficient ways to find the anchors.

Another problem to be explored in the future is how to select informative eigenvectors as the low-dimensional embedding of data. Analysis in Xiang and Gong (2008) showed that not every eigenvector of the similarity matrix was informative and relevant to clustering, and eigenvector selection was critical because using uninformative eigenvectors could lead to poor clustering results. In our problem settings, selecting informative eigenvectors of the anchors' similarity matrix can help improve the clustering accuracy.

Furthermore, we used the predefined weight matrix by the Gaussian function in the proposed method, and the width parameter of the Gaussian function was carefully chosen by cross validation, which needed extra computational time and made the model more sensitive to the parameter settings. Luo et al. (2017) characterized the intrinsic geometry structure of each neighborhood through a reconstruction nonnegative weight graph. If the local structure information can be embedded into the proposed method, that is, learning the adaptive weight matrix via data points, the advantages are two-fold: one is to avoid Gaussian function parameter tuning and the other is to obtain a better performance.

## References

- Arthur D, Vassilvitskii S, 2007. *K*-means++: the advantages of careful seeding. 18<sup>th</sup> Annual ACM-SIAM Symp on Discrete Algorithms, p.1027-1035. <https://doi.org/10.1145/1283383.1283494>
- Boutsidis C, Gittens A, Kambadur P, 2015. Spectral clustering via the power method—provably. 32<sup>nd</sup> Int Conf on Machine Learning, p.40-48.
- Chang XJ, Nie FP, Ma ZG, et al., 2015. A convex formulation for spectral shrunk clustering. 29<sup>th</sup> AAAI Conf on Artificial Intelligence, p.2532-2538.
- Chen WY, Song YQ, Bai HJ, et al., 2011. Parallel spectral clustering in distributed systems. *IEEE Trans Patt Anal Mach Intell*, 33(3):568-586. <https://doi.org/10.1109/TPAMI.2010.88>
- Chen XL, Cai D, 2011. Large scale spectral clustering with landmark-based representation. 25<sup>th</sup> AAAI Conf on Artificial Intelligence, p.313-318.
- Davies DL, Bouldin DW, 1979. A cluster separation measure. *IEEE Trans Patt Anal Mach Intell*, 1(2):224-227. <https://doi.org/10.1109/TPAMI.1979.4766909>
- Delalleau O, Bengio Y, Le Roux N, 2005. Efficient non-parametric function induction in semi-supervised learning. 10<sup>th</sup> Int Workshop on Artificial Intelligence and Statistics, p.96-103.
- Demšar J, 2006. Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res*, 7:1-30.
- Fowlkes C, Belongie S, Chung F, et al., 2004. Spectral grouping using the Nystrom method. *IEEE Trans Patt Anal Mach Intell*, 26(2):214-225. <https://doi.org/10.1109/TPAMI.2004.1262185>
- Jia HJ, Ding SF, Xu XZ, et al., 2014. The latest research progress on spectral clustering. *Neur Comput Appl*, 24(7-8):1477-1486. <https://doi.org/10.1007/s00521-013-1439-2>
- Li HZ, Hu XG, Lin YJ, et al., 2016. A social tag clustering method based on common co-occurrence group similarity. *Front Inform Technol Electron Eng*, 17(2):122-134. <https://doi.org/10.1631/FITEE.1500187>
- Li JY, Xia YJ, Shan ZY, et al., 2015. Scalable constrained spectral clustering. *IEEE Trans Knowl Data Eng*, 27(2):589-593. <https://doi.org/10.1109/TKDE.2014.2356471>
- Lin F, Cohen WW, 2010. Power iteration clustering. 27<sup>th</sup> Int Conf on Machine Learning, p.655-662.
- Liu JL, Wang C, Danilevsky M, et al., 2013. Large-scale spectral clustering on graphs. 23<sup>rd</sup> Int Joint Conf on Artificial Intelligence, p.1486-1492.
- Liu W, He JF, Chang SF, 2010. Large graph construction for scalable semi-supervised learning. 27<sup>th</sup> Int Conf on Machine Learning, p.679-686.
- Luo MN, Nie FP, Chang XJ, et al., 2017. Adaptive unsupervised feature selection with structure regularization. *IEEE Trans Neur Netw Learn Syst*, 29(4):944-956. <https://doi.org/10.1109/TNNLS.2017.2650978>
- Mall R, Langone R, Suykens JAK, 2013a. FURS: fast and unique representative subset selection retaining large-scale community structure. *Soc Netw Anal Min*, 3(4):1075-1095. <https://doi.org/10.1007/s13278-013-0144-6>
- Mall R, Langone R, Suykens JAK, 2013b. Kernel spectral clustering for big data networks. *Entropy*, 15(5):1567-1586. <https://doi.org/10.3390/e15051567>

- Mall R, Jumutc V, Langone R, et al., 2014. Representative subsets for big data learning using  $K$ -NN graphs. *IEEE Int Conf on Big Data*, p.37-42.  
<https://doi.org/10.1109/BigData.2014.7004210>
- Ng AY, Jordan MI, Weiss Y, et al., 2002. On spectral clustering: analysis and an algorithm. *Advances in Neural Information Processing Systems*, p.849-856.
- Shi JB, Malik J, 2000. Normalized cuts and image segmentation. *IEEE Trans Patt Anal Mach Intell*, 22(8):888-905.  
<https://doi.org/10.1109/34.868688>
- Song YQ, Chen WY, Bai HJ, et al., 2008. Parallel spectral clustering. In: Daelemans W, Goethals B, Morik K (Eds.), *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, p.374-389.  
[https://doi.org/10.1007/978-3-540-87481-2\\_25](https://doi.org/10.1007/978-3-540-87481-2_25)
- Tian F, Gao B, Cui Q, et al., 2014. Learning deep representations for graph clustering. 28<sup>th</sup> AAAI Conf on Artificial Intelligence, p.1293-1299.
- von Luxburg U, 2007. A tutorial on spectral clustering. *Stat Comput*, 17(4):395-416.  
<https://doi.org/10.1007/s11222-007-9033-z>
- Wang L, Leckie C, Ramamohanarao K, et al., 2009. Approximate spectral clustering. In: Theeramunkong T, Kijssirikul B, Cercone N, et al. (Eds.), *Advances in Knowledge Discovery and Data Mining*. Springer Berlin Heidelberg, p.134-146.  
[https://doi.org/10.1007/978-3-642-01307-2\\_15](https://doi.org/10.1007/978-3-642-01307-2_15)
- Xia RK, Pan Y, Du L, et al., 2014. Robust multi-view spectral clustering via low-rank and sparse decomposition. 28<sup>th</sup> AAAI Conf on Artificial Intelligence, p.2149-2155.
- Xiang T, Gong SG, 2008. Spectral clustering with eigenvector selection. *Patt Recog*, 41(3):1012-1029.  
<https://doi.org/10.1016/j.patcog.2007.07.023>
- Xiao P, Li ZY, Guo S, et al., 2016. A  $K$  self-adaptive SDN controller placement for wide area networks. *Front Inform Technol Electron Eng*, 17(7):620-633.  
<https://doi.org/10.1631/FITEE.1500350>
- Yan DH, Huang L, Jordan MI, 2009. Fast approximate spectral clustering. 15<sup>th</sup> Int Conf on Knowledge Discovery and Data Mining, p.907-916.  
<https://doi.org/10.1145/1557019.1557118>
- Yang Y, Xu D, Nie FP, et al., 2010. Image clustering using local discriminant models and global integration. *IEEE Trans Image Process*, 19(10):2761-2773.  
<https://doi.org/10.1109/TIP.2010.2049235>
- Yang Y, Shen HT, Nie FP, et al., 2011. Nonnegative spectral clustering with discriminative regularization. 25<sup>th</sup> AAAI Conf on Artificial Intelligence, p.555-560.
- Yang Y, Nie F, Xu D, et al., 2012. A multimedia retrieval framework based on semi-supervised ranking and relevance feedback. *IEEE Trans Patt Anal Mach Intell*, 34(4):723-742.  
<https://doi.org/10.1109/TPAMI.2011.170>
- Zhang XC, Zong LL, You QZ, et al., 2016. Sampling for Nystrom extension-based spectral clustering: incremental perspective and novel analysis. *ACM Trans Knowl Discov Data*, 11(1):1-25.  
<https://doi.org/10.1145/2934693>
- Zhu XJ, Lafferty J, 2005. Harmonic mixtures: combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. 22<sup>nd</sup> Int Conf on Machine Learning, p.1052-1059.  
<https://doi.org/10.1145/1102351.1102484>