Front. Comput. Sci. China 2011, 5(3): 268–278
DOI 10.1007/s11704-011-0023-0

RESEARCH ARTICLE

An improved spectral clustering algorithm based on random walk

Xianchao ZHANG (✉), Quanzeng YOU

School of Software, Dalian University of Technology, Dalian 116623, China

Abstract The construction process for a similarity matrix has an important impact on the performance of spectral clustering algorithms. In this paper, we propose a random walk based approach to process the Gaussian kernel similarity matrix. In this method, the pair-wise similarity between two data points is not only related to the two points, but also related to their neighbors. As a result, the new similarity matrix is closer to the ideal matrix which can provide the best clustering result. We give a theoretical analysis of the similarity matrix and apply this similarity matrix to spectral clustering. We also propose a method to handle noisy items which may cause deterioration of clustering performance. Experimental results on real-world data sets show that the proposed spectral clustering algorithm significantly outperforms existing algorithms.

Keywords spectral clustering, random walk, probability transition matrix, matrix perturbation

1 Introduction

Spectral clustering has recently become one of the most popular clustering algorithms. Compared with traditional clustering techniques, spectral clustering exhibits many advantages and is applicable to different types of data set. Though spectral clustering algorithms are simple and efficient, their performance is highly dependent on the construction of a similarity matrix.

In an ideal similarity matrix, the entries between intra-cluster data points are assigned 1, while the entries between inter-cluster data points are assigned 0, we call this the *ideal matrix*. The use of such an ideal matrix will enable the spectral clustering algorithm to find the exact *real* clusters. According to Ref. [1], we can obtain better clustering performance if the similarity matrix is closer to the ideal matrix. Therefore, the generation of a similarity matrix that is closer to the ideal matrix is critical to the success of spectral clustering.

There have been numerous methods proposed to improve the similarity matrix [2–5]. In general, these methods can be categorized into two classes: unsupervised and semi-supervised. In unsupervised methods, no labeled or constraint information is known, whereas semi-supervised methods try to work with labeled or constraint information for the construction of similarity matrix. In this paper, we propose an approach which belongs to the unsupervised class.

Though many studies [6,7] have mentioned the relationship between spectral clustering and random walk, they don't consider the problem of how to obtain a good similarity matrix. In Ref. [6], they reveal how random walk can be related to spectral clustering and also give the condition under which MNCut [6] can be optimized ($\mathrm{MNCut} = \sum_{i=1,..,k} cut(A_i,\overline{A}_i)/vol(A_i)$, we will discuss this further in Section 2). In Ref. [7], they give a more concrete analysis, and provide a method to automatically determine the number of clusters as well as the parameter σ of Gaussian kernel function.

In this paper, we apply random walk theory to process the Gaussian kernel similarity matrix. We view the

Received March 23, 2010; accepted October 11, 2010

E-mail: xczhang@dlut.edu.cn

normalized similarity matrix as a stochastic matrix, and then begin a random walk process based on the matrix. This has the effect of emphasizing intra-cluster similarity and lightening inter-cluster similarity, thus the processed matrix is closer to the ideal matrix. To minimize MNCut [6–8], we make further efforts to process the probability transition matrix. As a result, the performance of the spectral clustering algorithm is improved. We give a theoretical analysis of the similarity matrix and apply this similarity matrix to spectral clustering. Experimental results on real-world data sets show that the proposed spectral clustering algorithm can achieve much better clustering performance than existing spectral clustering methods.

We give a brief review of spectral clustering in Section 2. In Section 3, we give some theoretical analysis of the preprocess procedure. Finally we provide experimental results in Section 4 and conclude this paper in Section 5.

2 Overview of spectral clustering

In this section we give a brief review of spectral clustering algorithms. We focus on graph cut theory and its relationship to spectral clustering.

The objective of clustering is to divide data points into different clusters, where data points in the same cluster are similar to each other. We can construct a graph from the similarity matrix, where the vertexes represent the data points, and the edge weights represent similarities between data points. With this representation, the clustering problem is equivalent to the corresponding graph's cut problem. Given a weighted graph G, we want to find a cut of the graph, such that the cut will be minimized. In this paper we denote $\Delta = \{A_1,A_2,...,A_K\}$ as a clustering result, where A_i includes all the data points that belong to cluster i.

According to spectral graph theory, there can be many different objective functions for cluster analysis, such as MNCut [9], RatioCut [10], NCut [9], and MinCut [11]. In this paper, we focus on MNCut, where the objective is to achieve a rational minimum cut. Given a graph with similarity matrix W, where w_{ij} denotes the i,j-th entry of W. Then the problem of minimizing MNCut is defined as [8]

$$\mathrm{MNCut}(\Delta) = \sum_{i=1}^{k} \frac{cut\left(A_i,\overline{A}_i\right)}{vol(A_i)}, \tag{1}$$

where $\overline{A}_i$ refers to the complement of A, $cut\left(A_i,\overline{A}_i\right) = \frac{1}{2}\sum_{j\in A_i,k\in\overline{A}_i} w_{jk}$, and $vol(A_i)$ represents the total weights of A_i.

It was shown in Ref. [9] that the minimization of Eq. (1) is NP-hard. According to Rayleigh-Ritz theory, it is possible to find an approximate solution of Eq. (1). In solving MNCut, we need to define Laplacian matrix $L = I - D^{-1}W$, where I denotes the entity matrix and D is a diagonal matrix with $D_{ii} = \sum_j w_{ij}$. Then, the approximate solution can be derived from the leading eigenvectors of L. The use of a Laplacian matrix eigenvector for approximating the graph minimum cut is called spectral clustering, as described in Algorithm 1 [12].

Algorithm 1 Spectral clustering

```
Input A set of points, $S = \{s_1,s_2,...,s_n\}$, cluster number $K$.
Output A partition, $\Delta = \{A_1,A_2,...,A_K\}$.
1. Compute similarity matrix $W$.
2. Compute Laplacian matrix $L = D - W$.
3. Compute Normalized Laplacian matrix $L = I - D^{-1}W$
4. Compute the first $K$ eigenvectors of $L$, denote as $U$.
5. Consider the rows of $U$ as data points, and use $k$-means to cluster them into $K$ clusters.
6. Assign $s_i$ to cluster $A_j$ if and only if row $i$ of the matrix $U$ was assigned to cluster $A_j$.
```

In spectral clustering, the pair-wise similarities are first computed through a similarity function. The Gaussian kernel function is one of the most popular used similarity functions. Denote w_{ij} as the similarity between two points s_i and s_j, then

$$w_{ij} = \exp\left(\frac{-d^2(s_i,s_j)}{\sigma^2}\right), \tag{2}$$

where $d(s_i,s_j)$ denotes the Euclidean distance between two points, s_i and s_j, the parameter σ controls the width of the neighborhood [8]. The method for choosing an appropriate σ is also critical in spectral clustering. In general, one can choose σ in the order of the mean distance of a point to its k-th nearest neighbor (k~log(n) + 1) [8]. Zelnik proposed an alternate method for selecting σ in [13], where σ is not a fixed value

$$\sigma_i = d(s_i,s_K), \tag{3}$$

where s_K is the K-th nearest neighbor of s_i, and σ_i is said to be the local scale of data point s_i. How to choose an optimal parameter σ is beyond the scope of this paper. We adopt the method in (3) to choose parameter σ. On the one hand, such a local scale method can improve the applicability of spectral clustering. On the other hand, it may encounter problems on some data sets. For instance, it generates incorrect clustering results on the spiral data set in Fig. 1. Our proposed algorithm in the next section is capable of handing such cases.

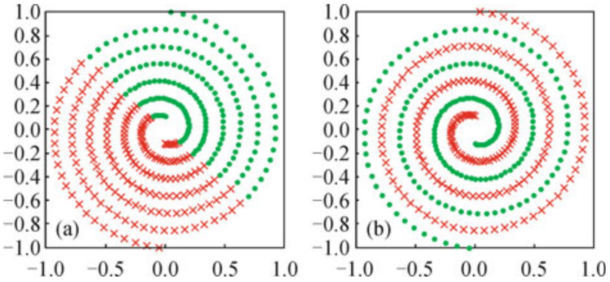


Fig. 1 Clustering results on spiral set. (a) Self-tuning spectral clustering result; (b) traditional spectral clustering result

3 Improved spectral clustering algorithm

Before the discussion of the proposed algorithm, we first give a brief review of random walk theory. Given a graph $G=(V,E)$, a random walk on G can be defined as: randomly choose a vertex s_0, with some probability p_{0i} a random walker will jump to one of its neighbors s_i, and at vertex s_i, the random walker will jump to one of its neighbor s_j with some probability p_{ij}. After time t, we get a chain $(s_0,s_i,\ldots,s_m)$, and the chain is a Markov chain.

Definition 1 (Stochastic Matrix [14]) A row stochastic matrix is a square matrix whose rows consists of nonnegative real numbers, with each row summing to 1, while a column stochastic matrix is a square matrix whose columns consists of nonnegative real numbers, with each column summing to 1.

According to the definition, $P=D^{-1}W$ is a row stochastic matrix. P can also be considered a transition probability matrix, where the entry p_{ij} gives the probability that the random walker jumps from vertex s_i to vertex s_j. If we define vector $\mathbf{s}(t)$ as the probability distribution of the random walk at time t, where $s_i(t)$ represents the random walk to point i at time t. Then we get [15]:

$$s_i(t+1)=\sum\nolimits_j s_j(t)p_{ji}. \tag{4}$$

Eq. (4) is equivalent to

$$\mathbf{s}(t+1)=(P^{\mathrm{T}})\mathbf{s}(t). \tag{5}$$

This implies that the probability $s_i(t)$ is related to all of its neighbors. And P^M will be the transition probability matrix of a random walker after M steps.

According to the definition above, the largest eigenvalue of a stochastic matrix is 1. This indicates the spectral radius of the matrix is 1, namely $\rho(P)=1$. The Perron-Frobenius theorem [16] ensures that the absolute value of any other eigenvalue is strictly smaller than $\rho(P)$. Let λ_i,v_i be the i-th eigenvalue and eigenvector, and then it's known that λ_i^M,v_i will be the i-th eigenvalue and eigenvector of matrix P^M [8]. If M is big enough, then λ_i^M will be close to zero (if $|\lambda_i|$ is strictly less than 1). With this observation [7], provided an analysis of the properties of matrix P^M. Their main results are stated below. Fig. 2 gives an intuitive illustration of the matrix P^M. We can see that the clustering structure is clearly revealed by the transition probability matrix after random walk.

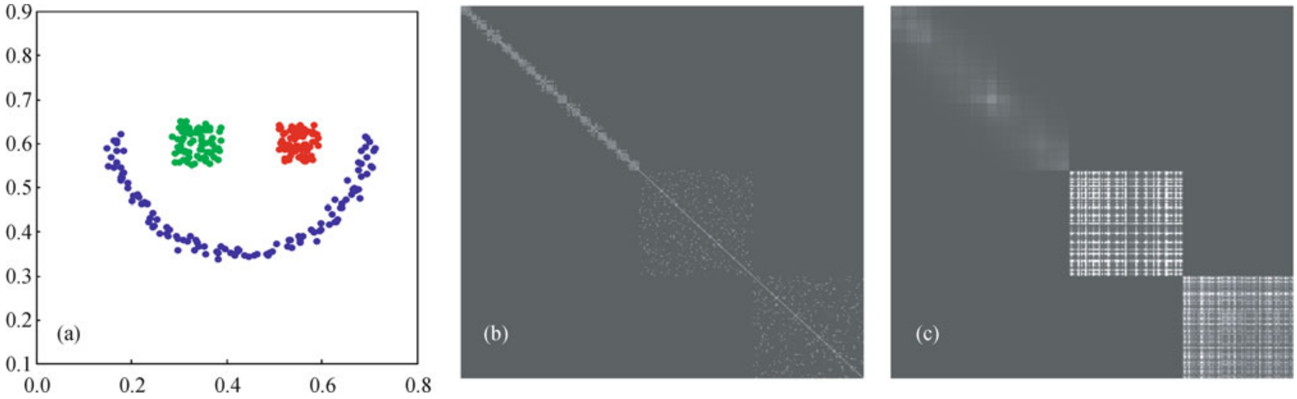


Fig. 2 Example of similarity matrix (white point means bigger similarity). (a) Data points with different colors represent different clusters; (b) the original similarity matrix; (c) the similarity matrix after random walk

3.1 When does spectral clustering perform well?

The similarity matrix plays a key role in the performance of the spectral clustering algorithm. In Refs. [1,17], the authors use matrix perturbation theory to analyze spectral clustering. They show that if the affinity matrix is close to an ideal matrix then spectral clustering will perform well [7]. In Refs.[6,8], the random walk is to exploited to obtain the following results.

Lemma 1 (Multicut lemma [8]) *Let W be an $n \times n$ symmetric matrix with nonnegative elements, and let P be the stochastic matrix, with $P = D^{-1}W$. Assume that P has K piecewise constant eigenvectors with regard to a partition $\Delta = (A_1,A_2,...,A_K)$. Assume also that P has n distinct eigenvalues and that the K largest eigenvalues of P are all nonzero. Then the minimum of MNCut for W is given by the partition Δ.*

Definition 2 (Piecewise constant eigenvectors (PCE) [6–8]) Let $\boldsymbol{v}$ be an eigenvector of P and $\Delta = (A_1,A_2,...,A_K)$ be a partition of a data set S. Then $\boldsymbol{v}$ is said to be a piecewise constant eigenvectors of P with regard to Δ, if $v(i) = v(j) \quad \forall i,j \in A_k$ and $k \in 1,2,...,K$.

According to Lemma 1, spectral clustering will be successful, even the similarity matrix is far from ideal. As long as the eigenvectors of P are PCE, spectral clustering is guaranteed to give a partiton which will minimize MNCut.

3.2 Using random walk to reduce MNCut

According to Lemma 1 if the K leading eigenvectors are PCE, then the spectral clustering is guaranteed to perform well. However, eigenvectors are usually not PCE. Thus we need to generate eigenvectors that are at least quasi-PCE.

It was shown in Ref. [8] that the objective function of MNCut can be reformulated as

$$\text{MNCut}(\Delta) = K - \sum_{k=1}^{K} \Pr[A_k \rightarrow A_k | A_k], \tag{6}$$

where $\Delta = (A_1,A_2,...,A_K)$ is an arbitrary partition and Pr denotes the conditional probabilities. In other words, the MNCut can be equally interpreted as to minimize the probability that a random walk jumps to another cluster. Ideally, the conditional probability is equal to 1, thus MNCut is minimized.

Using the above insights, the minimization of MNCut is equivalent to maximizing the conditional probability. $\Pr[A_k \rightarrow A_k | A_k]$ is related to the items belonging to cluster A_k. Thus, if we can increase the conditional probability for each item in A_k, the MNCut can be reduced. We express the statement in the following form:

$$\begin{aligned} PR(\Delta) &= \sum_{k=1}^{K} \Pr[A_k \rightarrow A_k | A_k] \\ &= \sum_{k=1}^{K} \Pr[r_1 \in A_k | r_0 \in A_k] \\ &= \sum_{k=1}^{K} \sum_{s_i \in A_k} \Pr[r_1 \in A_k | r_0 = s_i] \\ &= \sum_{k=1}^{K} \sum_{s_i \in A_k} \sum_{s_j \in A_k} \Pr[r_1 = s_j | r_0 = s_i], \end{aligned} \tag{7}$$

where r_i gives the position of a random walk at time i.

Definition 3 (Principal matrix component [7]) We refer to the matrix $T_n = \dfrac{v_n v_n^T D}{v_n^T D v}$ as the n-th principal matrix component of P^M, and λ_n^M as its weight. $\{\lambda_n, v_n\}$ is the eigensystem of P.

Then P^M can be written as

$$P^M = \sum_{n=1}^{N} \lambda_n^M T_n. \tag{8}$$

Notice that $|\lambda_n| \leqslant 1$, with a large enough M, only if λ_n is close to 1, then T_n survives after the random walk. So PMC reveals structure in multiscales, with λ_n^M as an indicator of component stability [7]. This explains why we choose the K leading eigenvectors for clustering in the spectral clustering algorithm.

As mentioned above, $P^M(i,j)$ is the probability that a random walker reaches j after M steps when it starts from item i. Similarly, we have

$$PR^M(\Delta) = \sum_{k=1}^{K} \sum_{s_i \in A_k} \sum_{s_j \in A_k} \Pr\left[r_M = s_j | r_0 = s_i\right].$$

From Fig. 2, we know that a random walk of M steps leads to a similarity matrix that is close to a block diagonal matrix. This indicates that the transition probability between two items will increase after a random walk if these two items belong to the same cluster , i.e., there is a good path that connects them [18]. Here a good path means the weight of each edge in the path should not be too small. We also notice that a bad path will have the opposite effect. The transition probability that an item jumps into another cluster is supposed to be a very small value. If we ignore these small values, then PR^M is supposed to be increased. Thus we can improve the performance of spectral clustering. We formulate the idea here in the following theorem.

Theorem 1 *Let P^M denote the transition matrix after M steps random walk, and if we ignore the small values of P^M(set $P^M(i,j) = 0$), then* MNCut *will be reduced.*

Proof Let $PR^M(\Delta)'$ denote the transition probability if we ignore the small values of $PR^M(\Delta)$. Then we have

$$\begin{aligned}\mathrm{MNCut}(\Delta) &= K - PR^M \\ &= \sum_{k=1}^{K}\left(1 - \sum_{s_i\in A_k}\sum_{s_j\in A_k}\Pr[r_M = s_j|r_0 = s_i]\right) \\ &= \sum_{k=1}^{K}\sum_{s_i\in A_k}\sum_{s_j\notin A_k}\Pr[r_M = s_j|r_0 = s_i]. \qquad (9)\end{aligned}$$

Note, after we ignore the small values, we will normalize the transition matrix. Let $\mathrm{MNCut}(\Delta)'$ denote the new multiway normalized cut and $\Omega = [\omega_1,\omega_2,...,\omega_N]^{\mathrm{T}}$ denote the sum of small values of each row we ignored.

As we need to normalize the new transition matrix P^M, we let $D' = [d_1,d_2,...,d_N]^{\mathrm{T}}$, with $d_i = \sum_{j=1}^{N} P^{M\prime}(i,j)$. Since we discard some small values, so $d_i \leqslant 1$. Then,

$$\begin{aligned}\mathrm{MNCut}(\Delta)' &= K - PR^M(\Delta)' \\ &= \sum_{k=1}^{K}\left(1 - \left(\sum_{s_i\in A_k}\sum_{s_j\in A_k}\Pr[r_M = s_j|r_0 = s_i]\right)/d_i\right) \\ &\leqslant \sum_{k=1}^{K}\left(1 - \sum_{s_i\in A_k}\sum_{s_j\in A_k}\Pr[r_M = s_j|r_0 = s_i]\right) \\ &= \mathrm{MNCut}(\Delta). \qquad (10)\end{aligned}$$

As we will at least ignore the smallest values of $PR^M(\Delta)$, so the equation will not hold. Thus we have MNCut $(\Delta)' <$ MNCut(Δ).

Here we provide an easy method to determine the threshold ξ for discarding small values. The threshold is used to reduce the effect of outliers and inter-cluster transmission. One can choose ξ as the smallest gap of all the biggest gaps within each row

$$\xi = \min_{i=1\cdots n}\left(\max_{w_{ij},j=1\cdots(n-1)}(w_{ij} - w_{i(j+1)})\right). \qquad (11)$$

3.3 Robustness to noise

In this section we propose a simple method to detect noisy data points from the data set. Each column of a transition matrix P represents the probability of a random walk jumping to the corresponding item. It is evident that there will be little chance for other items to jump to a noisy item. In other words, there is no good path for connecting items to a noisy item.

Based on above observations, we can check whether an item is noisy using the sum of transition probabilities from other items. Let $PN_i = \sum_{j=1}^{N} P_{ji}^{M}$ denote the probability. It is straightforward to consider an item s_i as an outlier if PN_i is less than a threshold Θ. To find a proper threshold, here we adopt a similar strategy to that used in the determination of ξ.

If an item s_i is a noisy item, then PN_i will have a big gap to other item transition probabilities. Thus we can find such gap to determine noisy items by sorting the transition probability. Formally, we have

$$\begin{aligned}&k = \arg\ \max_i(PN_{i+1} - PN_i), \\ &\Theta = PN_k.\end{aligned} \qquad (12)$$

To reduce the impact of noisy items on other items, we will temporarily ignore those items and cluster the remaining items. After we get the partition $\Delta = \{A_1,A_2,...,A_K\}$ of non-noisy items, we use Eq. (13) to determine the cluster number of a given outlier v_i.

$$k = \arg\ \max_k\left(\sum_{s_j\in A_k} P_{ji}^{M}\right). \qquad (13)$$

Then we set $s_i \in A_k$. From Eq. (6), we know this will minimize MNCut.

We summarize our results in Algorithm 2.

Algorithm 2 Spectral clustering based on random walk

```
Input Data $S = \{s_1,s_2,\cdots,s_N\}$, number of clusters $K$, number of steps for random walk $M$
Output A partition with $\Delta = \{A_1,A_2,\cdots,A_K\}$.
1. Compute local scale $\sigma_i$ of each $s_i \in S$ with (3).
2. Compute similarity matrix $P$ according to (2) using local scale.
3. Compute threshold $\xi$ according to (11).
4. Set $P_{ij}^M = 0$ where $P_{ij}^M < \xi$, then after the normalization we get $P^{M\prime}$.
5. Compute threshold $\Theta$ according to (12), and ignore the corresponding rows and columns of the noisy items. We get $P^{M\prime\prime}$.
6. Call steps 2–5 of algorithm 1 to cluster the non-noisy items. We get partition $\Delta''$
7. Use (13) to determine the correct cluster for each noisy item. Then we get final cluster results $\Delta$.
```

The complexity of Algorithm 2 is comparable to spectral clustering. The increased computational complexity is mainly in step 4. In fact, it's not necessary to calculate P^M directly. Note that if $Px = \lambda x$, then we have

$P^M x = \lambda^M x$. Thus P^M can be calculated by $P^M = \sum_{i=1,\cdots,n} x^{\mathrm{T}} x$. This will greatly reduce the complexity.

In the proposed algorithm, the number of clusters is given. In Ref. [7], they proposed a method which would search the maximal eigengap of matrix P^M to determine the desired cluster number, i.e., $K(M) = \arg\max_k(\lambda_k^M - \lambda_{k+1}^M)$. This can also be used in algorithm 2 to determine the cluster number automatically. However, the method has to search a huge space, which will reduce the speed of spectral clustering. Therefore, the number of clusters is given by the user in Algorithm 2. In the next section, we will compare our algorithm with other algorithms over a wide range of data sets.

4 Experiments

We implement the proposed spectral clustering algorithm with random walk (SCRW) and compare with the traditional spectral clustering (SC) [1], local scale spectral clustering (LSSC) [13] and *k*-means clustering. We test the performance of these algorithms on some benchmark data sets.

We use both error rate and the normalized mutual information (*NMI*) [19] as our evaluation metric. The normalized mutual information is defined as follows. Given the true label of a data set $\{C_1,C_2,...,C_c\}$ and $|C_i| = n_i$, suppose the clustering result is $\{S_1,S_2,...,S_k\}$ and $|S_i| = n_i'$, then *NMI* can be written as Eq. (14) where $n_{ij} = |C_i \cap S_j'|$:

$$NMI = \frac{\sum_{i=1}^{c}\sum_{j=1}^{k} n_{ij}\log\frac{nn_{ij}}{n_i n_j'}}{\sqrt{\left(\sum_{i=1}^{c} n_i \log\frac{n_i}{n}\right)\left(\sum_{j=1}^{k} n_j' \log\frac{n_j'}{n}\right)}}. \quad (14)$$

According to the definition, we get $0 \leqslant NMI \leqslant 1$. And a larger *NMI* implies a better clustering result. If the clustering result is exactly the same as the true label, then *NMI* equals 1.

4.1 Results on synthetic data sets

We use three widely used synthetic data sets (FourLine, ThreeCircle, and CirclePoints) to evaluate the proposed algorithm. The steps of the random walk is set to 50. The clustering results are shown in Fig. 3. Different colors refer to different clusters. The leftmost column shows the correct cluster results. The right four columns are the results of different algorithms (SCRW, LSSC, SC, *k*-means). Since the data sets are not convex, the results of *k*-means clustering are poor. In the SC algorithm, the delta of the Gaussian kernel function is chosen from a wide range of intervals. The results of SC algorithm in Fig. 3 show the best performance. But SC algorithm still fails to cluster the CirclePoints data set correctly.

As the proposed algorithm adopts the same strategyas LSSC, using Eq. (3) to choose the Gaussian kernel parameter σ, for the construction of similarity matrix, the proposed algorithm, SCRW, and LSSC correctly cluster all of the three data sets. But as noted in Section 3.3, the random walk will enlarge the eigengap which provides a method to determine the cluster number automatically.

Figure 4 shows the top 20 smallest eigenvalues of the Laplacian matrix in both algorithms. From Fig. 4 we can see the eigengap is almost the biggest, between the *K*-th and the $(K+1)$-th eigenvalues of SCRW, where *K* is the desired cluster number: *K* = 4, 3, 3 respectively. Thus, in SCRW the correct cluster number can be evaluated. We can calculate the eigengaps and try the peaks of the eigengaps which the correct cluster number can be.

4.2 Results on real data sets

Here we give some description of the data sets used in our experiment.

● **Iris** Iris data set contains 150 instances from three classes. Each class has 50 instances, and each class refers to a type of iris plant. Each instance has 4 attributes.

● **Wine** The wine data set comes from a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The data set has three types of wines. Compared with Iris, the Wine data set has more attributes. Thus the Wine data set is more challenging.

● **USPS** USPS data set contains 7291 training instances and 2007 test instances. Each instance contains 256 attributes of the handwritten information. We choose digits{1,7}, {8, 9}, {0, 8} and {1, 2, 3, 4} as subsets for experiment separately.

● **Ionosphere** Ionosphere data set contains 351 instances. Each instance contains 34 attributes of some radar data.

● **Glass** Glass data set contains 214 instances. Each instance contains 10 attributes.

In spectral clustering we choose σ from 1 to 200, and then use the best one as final result. In LSSC, we choose the distance to its 7-th nearest neighbor, as the local scale. While in SCRW, we choose the random walk steps *M* to be 101.

The clustering results of SCRW, SC, LSSC and *k*-means

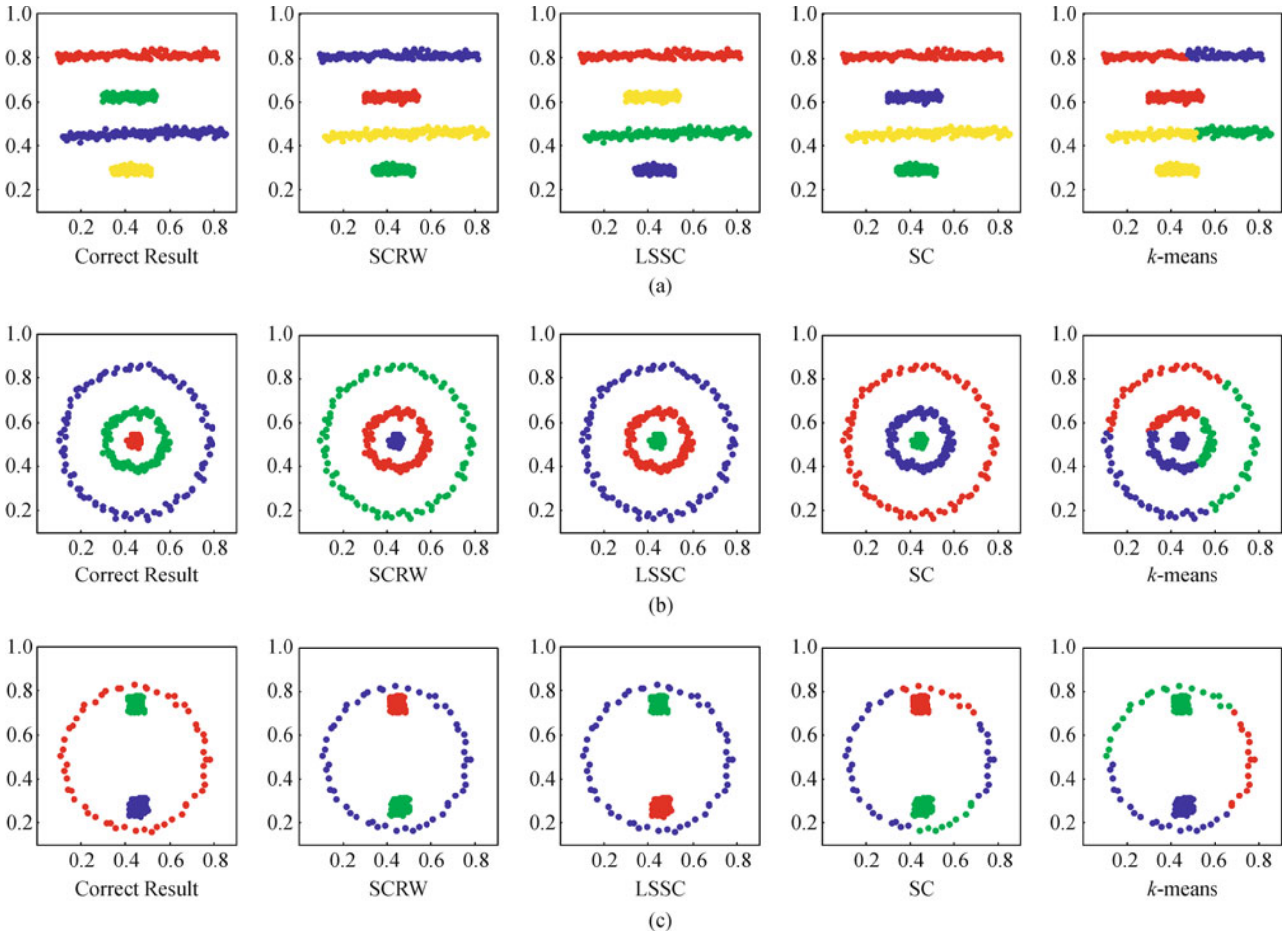


Fig. 3 Results on three synthetic data sets. (a) FourLine; (b) ThreeCircle; (c) CirclePoints

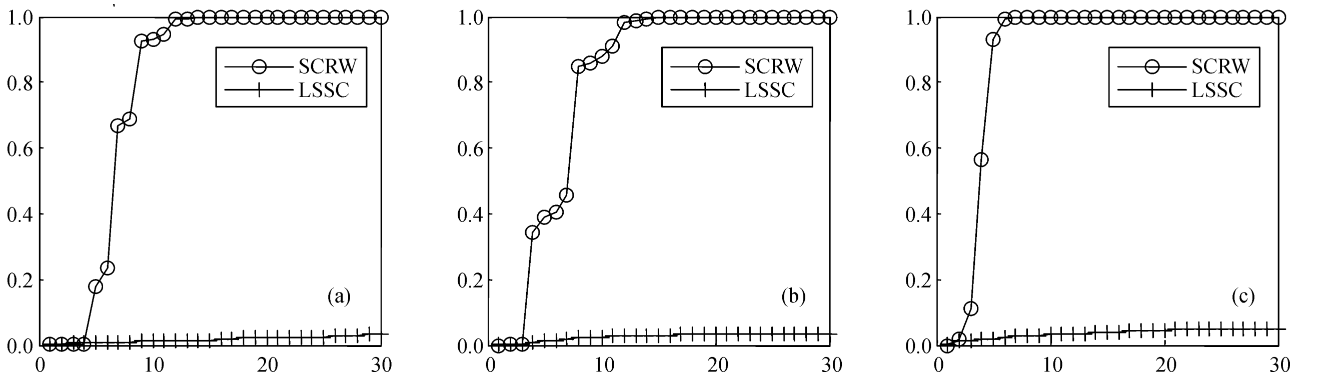


Fig. 4 Top-20 eigenvalues of Laplacian matrix in both SCRW and LSSC algorithms. (a) FourLine; (b) ThreeCircle; (c) CirclePoints

are summarized in Table 1 and Fig. 5. In Table 1, error rate is used to evaluate the performance of different clustering methods. Compared to other algorithms, the proposed algorithm has a lower error rate and is more robust for clustering.

The results in Fig. 5 show the clustering results when NMI is used as the evaluation measure. We see that SCRW provides better clustering results. Especially on two challenging USPS subsets {0,8} and {8,9}, SCRW substantially outperforms other algorithms. Based on the above observation, we conclude that the SCRW is robust and stable.

Table 1 Error rate of each clustering algorithm

Data	SCRW	LSSC	SC	*k*-means
Iris	0.093330	0.093330	0.32000	0.11333
Wine	0.275280	0.452510	0.38547	0.46980
{1, 7}	0.026760	0.038930	0.01945	0.00973
{0, 8}	0.030470	0.190460	0.18667	0.01714
{8, 9}	0.023320	0.160350	0.17492	0.07871
{1, 2, 3, 4}	0.038640	0.655790	0.09299	0.10024
Glass	0.457944	0.500000	0.50000	0.514019
Ionosphere	0.210826	0.264957	0.54700	0.293447

Spectral clustering algorithms can also be applied to image segmentation [5,20,21]. It is noted that applying literal spectral clustering to image segmentation is generally infeasible [20]. In Ref. [5], they actually adopt a technique to reduce the image to a small size (in their implementation, the images are resized to 160 × 160), whose results are not quite satisfactory. In this paper, we adopt a sample based method, *NSC*, proposed in Ref. [21], for image segmentation. We simultaneously start a random walk on the similarity matrix of the sampled points, and the similarity matrix between the sampled and the unsampled points.

The similarities between image pixels are computed using the χ^2 -distance proposed in Ref. [22]. Before the construction of the similarity matrix, a color quantization operation should be applied to the images. In the experiments, we apply the technique proposed in Ref.

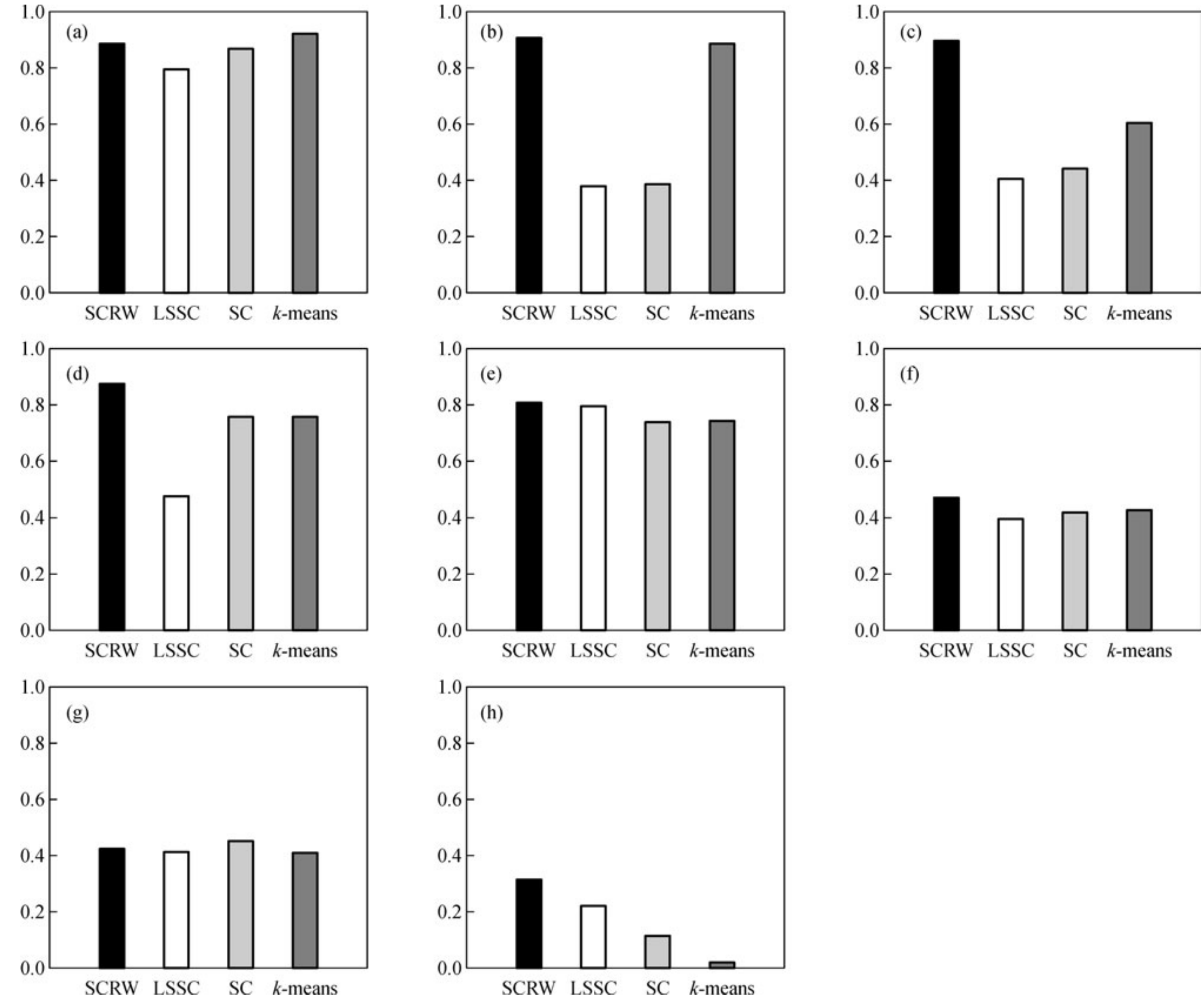
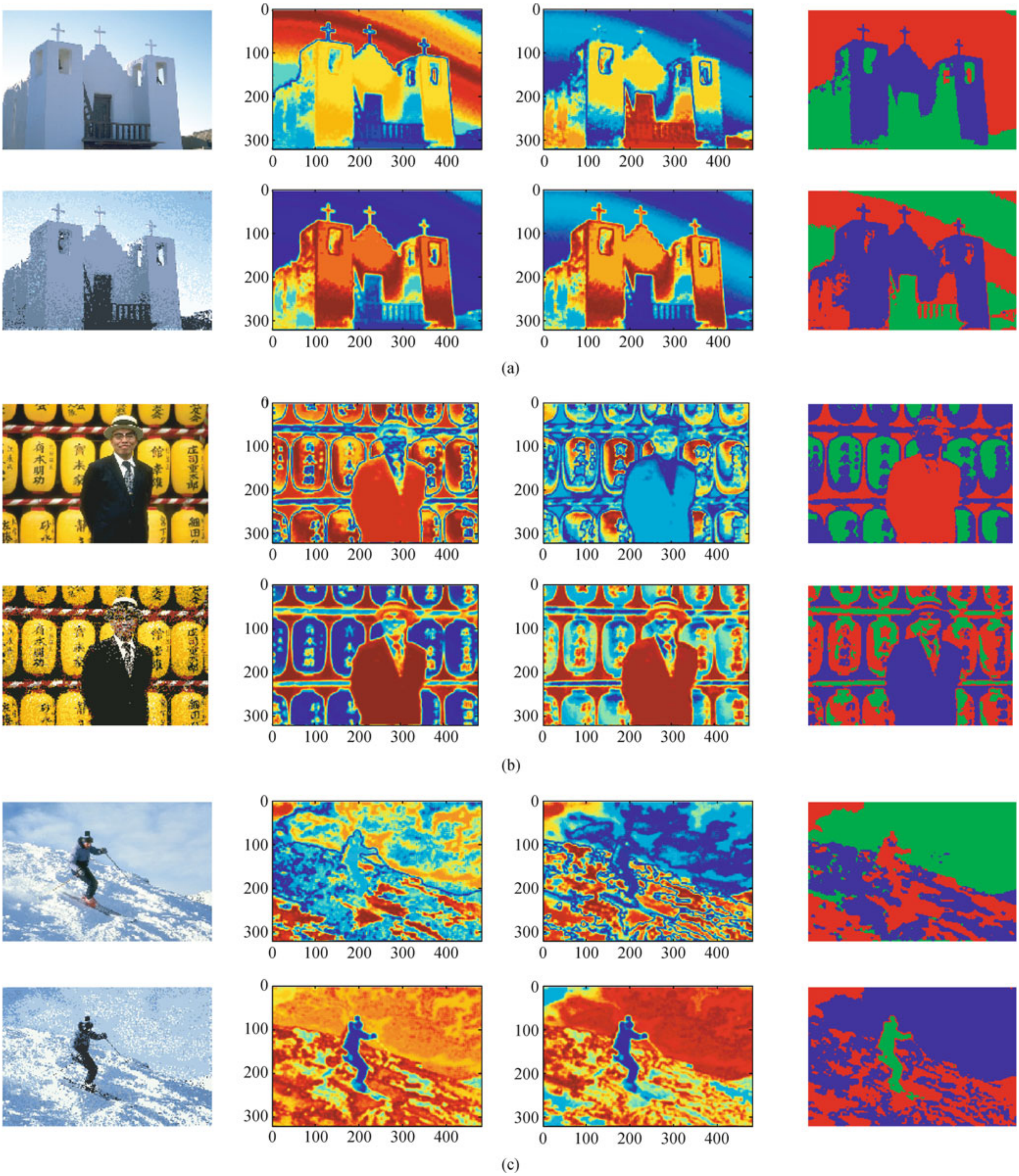


Fig. 5 Clustering results for different clustering methods and using NMI as a metric to evaluate the performance. (a) Iris. (b) Wine; (c) subset {1, 7} of USPS; (d) subset {0, 8} of USPS; (e) subset {8, 9} of USPS; (f) subsets {1, 2, 3, 4} of USPS; (g) Glass; (h) Ionosphere



(a)

(b)

(c)

Fig. 6 Image segmentation results. (a) Railings; (b) tie; (c) man

[23], spatial color quantization[1], for decreasing the color depth of the images. The χ^2-distance considers both the locally-windowed color and texture histogram which has been shown to be a very robust measure [21]. The χ^2-

[1] http://www.cs.berkeley.edu/~dcoetzee/downloads/scolorq/

distance between pixels i and j is given by

$$\chi_{ij}^2 = \frac{1}{2}\sum_{s=1}^{S}\frac{\left(h_i(s)-h_j(s)\right)^2}{h_i(s)+h_j(s)}, \tag{15}$$

where $h_i(s)$ and $h_j(s)$ is the normalized histograms of pixels i and j individually, and S controls the number of colors considered in the quantization operation.

The similarity between pixels i and j is defined as $W_{ij} = e^{-\chi_{ij}^2}$, and it is proven that this kernel is positive and definite [21]. We compare the algorithms on three 481 × 321 images[1)], and the results are shown in Fig. 6. For each of the subfigures, the leftmost column is the original color image (top) and the corresponding image after quantization (bottom) and the right three columns are respectively the second and third eigenvectors, and segmentation results where the top row represents without and bottom row with random walk. In the experiments, the cluster number is set to 3 and the number of sampled pixels is set to 100. The clustering error cannot be evaluated in this case [20]. Thus, the clustering results are given using different colors (the right most column of Fig. 6). Different colors represent different meaningful components.

We also give the eigenvectors corresponding to the second smallest and the third smallest eigenvalues of the Laplacian matrix. It is interesting to notice that the proposed algorithm will give more detailed segmentation compared with the methods in Ref. [21] (spectral clustering with Nyström extension). For example, the proposed algorithm successfully finds the railings in Fig. 6 (a), the man's tie in Fig. 6(b) and the man in Fig. 6(c), whereas the NSC fails to find these segments. This seems to imply again that the proposed algorithm could be superior for image segmentation.

5 Conclusion

In this paper, we have proposed a novel approach to improving spectral clustering by applying random walk theory to processing the Gaussian kernel similarity matrix. Experimental result shows the improved spectral clustering algorithm outperforms traditional and other improved spectral clustering algorithms. The method used here can also be applied to other clustering techniques based on similarity matrices.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant No. 60873180).

References

1. Ng A Y, Jordan M I, Weiss Y. On spectral clustering: analysis and an algorithm. In: Proceedings of Advances in Neural Information Pressing Systems 14. 2001, 849–856
2. Wang F, Zhang C S, Shen H C, Wang J D. Semi-supervised classification using linear neighborhood propagation. In: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2006, 160–167
3. Wang F, Zhang C S. Robust self-tuning semi-supervised learning. Neurocomputing, 2006, 70(16–18): 2931–2939
4. Kamvar S D, Klein D, Manning C D. Spectral learning. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence. 2003, 561–566
5. Lu Z D, Carreira-Perpiňán M A. Constrained spectral clustering through affinity propagation. In: Proceedings of 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2008, 1–8
6. Meila M, Shi J. A random walks view of spectral segmentation. In: Proceedings of 8th International Workshop on Artificial Intelligence and Statistics. 2001
7. Azran A, Ghahramani Z. Spectral methods for automatic multiscale data clustering. In: Proceedings of 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2006, 190–197
8. Meila M. The multicut lemma. UW Statistics Technical Report 417, 2001
9. Shi J, Malik J. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22(8): 888–905
10. Hagen L, Kahng A B. New spectral methods for ratio cut partitioning and clustering. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1992, 11(9): 1074–1085
11. Ding C H Q, He X F, Zha H Y, Gu M, Simon H D. A min-max cut algorithm for graph partitioning and data clustering. In: Proceedings of 1st IEEE International Conference on Data Mining. 2001, 107–114
12. von Luxburg U. A tutorial on spectral clustering. Statistics and Computing, 2007, 17(4): 395–416
13. Zelnik-Manor L, Perona P. Self-tuning spectral clustering. In. Proceedings of Advances in Neural Information Processing Systems 17. 2004, 1601–1608

[1)] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/

14. Huang T, Yang C. Matrix Analysis with Applications. Beijing: Scientific Publishing House, 2007 (in Chinese)

15. Lovász L, Lov L, Erdos O. Random walks on graphs: a survey. Combinatorics, 1993, 2: 353–398

16. Gong C H. Matrix Theory and Applications. Beijing: Scientific Publishing House, 2007 (in Chinese)

17. Tian Z, Li X B, Ju Y W. Spectral clustering based on matrix perturbation theory. Science in China Series F: Information Sciences, 2007, 50(1): 63–81

18. Fouss F, Pirotte A, Renders J, Saerens M. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Transactions on Knowledge and Data Engineering, 2007, 19(3): 355–369

19. Banerjee A, Dhillon I, Ghosh J, Sra S. Generative model-based clustering of directional data. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2003, 19–28

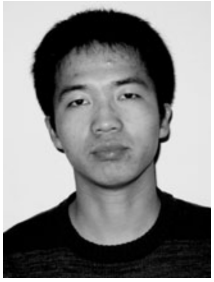
20. Wang L, Leckie C, Ramamohanarao K, Bezdek J C. Approximate spectral clustering. In: Proceedings of 13th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining. 2009, 134–146

21. Fowlkes C, Belongie S, Chung F, Malik J. Spectral grouping using the Nyström method. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26(2): 214–225

22. Puzicha J, Belongie S. Model-based halftoning for color image segmentation. In: Proceedings of 15th International Conference on Pattern Recognition. 2000, 629–632

23. Puzicha J, Held M, Ketterer J, Buhmann J M, Fellner D W. On spatial quantization of color images. IEEE Transactions on Image Processing, 2000, 9(4): 666–682

Xianchao Zhang is a full professor at Dalian University of Technology, China. He received his B.S degree in Applied Mathematics and M.S. degree in Computational Mathematics from National University of Defense Technology in 1994 and 1998, respectively. He received his Ph.D. in Computer Theory and Software from University of Science and Technology of China in 2000. He joined Dalian University of Technology in 2003 after 2 years of industrial working experience at international companies. He worked as Visiting Scholar at The Australian National University and The City University of Hong Kong in 2005 and 2009, respectively. His research interests include algorithms, machine learning, data mining and information retrieval.



Quanzeng You received his B.S. degree from School of Software, Dalian University of Technology, China in 2009. He is Current a master candidate at Dalian University of Technology, China. He joined the Lab of Intelligent Information Processing at DUT in 2009, under the supervision of Prof. Xianchao Zhang. His research interests include spectral clustering, clustering, semi-supervised learning and other data mining techniques. He is especially interested in spectral methods. Currently, his research mainly focuses on the improvement of spectral clustering and how to apply spectral clustering to large scale problems.