Spectral Clustering of Large-scale Data by Directly Solving Normalized Cut

Xiaojun Chen
College of Computer Science and Software
Shenzhen University
Shenzhen 518060, P. R. China
xjchen@szu.edu.cn

Weijun Hong
College of Computer Science and Software
Shenzhen University
Shenzhen 518060, P. R. China
280996118@qq.com

Feiping Nie*
School of Computer Science and OPTIMAL
Northwestern Polytechnical University
Xi'an 710072, Shanxi, P. R. China
feipingnie@gmail.com

Dan He
College of Computer Science and Software
Shenzhen University
Shenzhen 518060, P. R. China
hedanxkhc@hotmail.com

Min Yang
Shenzhen Institutes of Advanced Technology (SIAT)
Chinese Academy of Sciences
Shenzhen 518060, P. R. China
min.yang1129@gmail.com

Joshua Zhexue Huang
College of Computer Science and Software
Shenzhen University
Shenzhen 518060, P. R. China
zx.huang@szu.edu.cn

ABSTRACT

During the past decades, many spectral clustering algorithms have been proposed. However, their high computational complexities hinder their applications on large-scale data. Moreover, most of them use a two-step approach to obtain the optimal solution, which may deviate from the solution by directly solving the original problem. In this paper, we propose a new optimization algorithm, namely Direct Normalized Cut (DNC), to directly optimize the normalized cut model. DNC has a quadratic time complexity, which is a significant reduction comparing with the cubic time complexity of the traditional spectral clustering. To cope with large-scale data, a Fast Normalized Cut (FNC) method with linear time and space complexities is proposed by extending DNC with an anchor-based strategy. In the new method, we first seek a set of anchors and then construct a representative similarity matrix by computing distances between the anchors and the whole data set. To find high quality anchors that best represent the whole data set, we propose a Balanced k-means (BKM) to partition a data set into balanced clusters and use the cluster centers as anchors. Then DNC is used to obtain the final clustering result from the representative similarity matrix. A series of experiments were conducted on both synthetic data and real-world data sets, and the experimental results show the superior performance of BKM, DNC and FNC.

CCS CONCEPTS

- **Information systems → Clustering**;

KEYWORDS

Clustering, normalized cut, large-scale data

ACM Reference Format:
Xiaojun Chen, Weijun Hong, Feiping Nie, Dan He, Min Yang, and Joshua Zhexue Huang. 2018. Spectral Clustering of Large-scale Data by Directly Solving Normalized Cut. In KDD 2018: 24th ACM SIGKDD Int. Conference on Knowledge Discovery & Data Mining, August 19C23, 2018, London, UK. ACM, NY, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3220039

1 INTRODUCTION

During the past decades, researchers have proposed many clustering algorithms for cluster analysis, such as spectral clustering [19], subspace clustering [6, 10] and multi-view clustering [2, 5] etc. Among them, spectral clustering has received a lot of attention because it is easy to implement and often shows good clustering performance. Various spectral clustering algorithms have been proposed, such as ratio cut [8], normalized cut [13], multiclass spectral clustering [21], spectral embedded clustering [17], clustering with adaptive neighbors (CAN)[14], constrained laplacian rank (CLR) [15] and self-balanced min cut [3]. They have been successfully applied to many applications.

Spectral clustering methods usually involve a two-stage process for obtaining the final solution, i.e., performing eigendecomposition of similarity matrix first, and then obtaining the final clustering assignments from eigenvectors by k-means or spectral rotation [21]. Huang et al. have pointed

[*] Corresponding author

KDD '18, August 19–23, 2018, London, United Kingdom
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5552-0/18/08. . . $15.00
https://doi.org/10.1145/3219819.3220039

out that spectral rotation can obtain better clustering result than k-means [9]. Chen et al. improved the original spectral rotation and proposed an improved spectral rotation (ISR) [4]. However, existing spectral rotation involves a two-stage process in which an approximate continuous cluster assignment matrix is first computed, and the final discrete solution is obtained from the approximate continuous cluster assignment matrix. Such two-stage process may result in bad clustering structure which deviates from the clustering structure obtained by directly optimizing the original objective function.

On the other hand, spectral clustering usually has a time complexity of $O(n^3)$ where n is the number of samples, because they often need to perform eigendecomposition first. In recent years, much effort has been devoted for accelerating the spectral clustering. Recently, researchers have proposed two types of methods to address the scalability issue of spectral clustering. One type is to reduce the computational cost of the eigendecomposition step [7, 11], and another type is to sample the original data and perform clustering on the reduced data [18, 20]. However, these methods are based on sampling, and a lot of information will be lost in the sampling step. Recently, Cai et al. proposed a landmark-based spectral clustering (LSC) method [1]. Given a data set with n samples, LSC generates $m \ll n$ representative data points to compute a representative similarity matrix and the eigendecomposition can be performed on the low-size representative matrix. The final discrete clustering result is obtained from eigenvectors by k-means. Chen proposed a Scalable Normalized Cut algorithm, in which an anchor-based method is used to compute a representative similarity matrix and the improved spectral rotation (ISR) is used to obtain the final clustering result from the low-size representative similarity matrix [4]. However, how to effectively construct a representative similarity matrix and how to effectively obtain the final clustering result are still the two problems that need to be solved.

To solve the above problems, we propose two spectral clustering methods. We first propose a new normalized cut optimization algorithm, namely Direct Normalized Cut (DNC), to directly optimize the normalized cut problem. The new method has a lower computational complexity of $O(n^2c)$ than normalized cut with k-means, MSC and normalized cut with ISR. To cope with large-scale data, we further extend DNC to Fast Normalized Cut (FNC) that has linear time and space complexities. In the new method, we first use the anchor-based strategy to construct an representative similarity matrix. To find high quality anchors that best represent the whole data set, we propose a Balanced k-means (BKM) to partition the original data sets into balanced clusters and use the cluster centers as anchors. In BKM, a balance regularizer is incorporated into k-means in order to produce balanced partition. Then DNC is used to obtain the final clustering result from the representative similarity matrix. The convergence of DNC, BKM and FNC are proved. A series of experiments were conducted on both synthetic data and real-world data sets, and the experimental results show the superior performance of DNC, BKM and FNC.

The rest of this paper is organized as follows. Notations are given in Section 2 and related work are summarized in Section 3. We present DNC in Section 4 and FNC along with BKM in Section 5. The experimental results and analysis are presented in Section 6. Conclusions and future work are given in Section 7.

2 NOTATIONS

We first introduce the notations that are used throughout this paper. Matrices are written as boldface uppercase letters. Vectors are written as boldface lowercase letters. For matrix $\mathbf{M} = (m_{ij})$, its i-th row is denoted as $\mathbf{m}^i$, and its j-th column is denoted by $\mathbf{m}_j$. The Frobenius norm of the matrix $\mathbf{M} \in \mathbb{R}^{n\times m}$ is defined as $\|\mathbf{M}\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} m_{ij}^2}$. We call a matrix as cluster indicator matrix, if in which each row consists one and only one element equal to 1 to indicate the cluster membership, while the rest elements are 0. We denote the set of all cluster indicator matrices as $\mathbf{\Psi}$.

3 RELATED WORK

3.1 Spectral Clustering

Given a data set $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, we can construct an affinity matrix $\mathbf{A}$. Let $\mathbf{Y} \in \mathbf{\Psi}^{n\times c}$ be the cluster indicator matrix, in which $y_{il} = 1$ indicates that $\mathbf{x}_i$ is assigned to the l-th cluster. The objective function of classical Ratio Cut clustering can be written as [8]

$$\min_{\mathbf{Y}^T\mathbf{Y}=\mathbf{I}} Tr(\mathbf{Y}^T\mathbf{L}_A\mathbf{Y}) \tag{1}$$

and the objective function of Normalized Cut can be represented by [13]

$$\min_{\mathbf{Y}^T\mathbf{D}_A\mathbf{Y}=\mathbf{I}} Tr(\mathbf{Y}^T\mathbf{L}_A\mathbf{Y}) \tag{2}$$

where $\mathbf{L}_A = \mathbf{D}_A - \mathbf{A}$ is the Laplacian matrix and $\mathbf{D}_A$ is the corresponding degree matrix that is a diagonal matrix with the i-th diagonal element as $d_{ii} = \sum_{j=1}^{n} a_{ij}$. Problems (1) and (2) can be solved by a two-stage process: performing eigendecomposition on $\mathbf{L}_A$ first, and then obtaining the final clustering assignments from eigenvectors by k-means.

In 2014, Nie et al. proposed a clustering method CAN (Clustering with Adaptive Neighbors) [14]. CAN learns a probability matrix $\mathbf{S} \in \mathbb{R}^{n\times n}$, in which s_{ij} is the connected probability between $\mathbf{x}_i$ and $\mathbf{x}_j$. The objective function of CAN is as follows

$$\begin{aligned} &\min_{\mathbf{S}} \sum_{i,j=1}^{n} \left(\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 s_{ij} + \gamma s_{ji}^2\right) \\ &\forall\ i,\ \mathbf{s}^i\mathbf{1} = 1,\ s_{ij} \in [0,1],\ rank(\mathbf{L}_S) = n - c \end{aligned} \tag{3}$$

where a rank constraint $rank(L_S) = n - c$ is imposed to the Laplacian matrix of $\mathbf{S}$ such that the connected components in $\mathbf{S}$ are exactly equal to the number of clusters c and the final clustering result can be obtained from these connected components. The above problem can be solved with an iterative method. In each iteration, we have to perform eigendecomposion on $\mathbf{L}_S$.

In 2016, Nie et al. further improved CAN for a given affinity matrix to propose the Constrained Laplacian Rank (CLR) method [15]. CLR learns $\mathbf{S} \in \mathbb{R}^{n\times n}$ that best approximates the initial affinity matrix $\mathbf{A}$. Two versions of CLR were proposed, one is with the ℓ_2 norm

$$\begin{aligned} &\min_{\mathbf{S}} \|\mathbf{S}-\mathbf{A}\|_2^2 \\ &\forall\ i,\ \mathbf{s}^i\mathbf{1}=1,\ s_{ij}\in[0,1],\ rank(\mathbf{L}_S)=n-c \end{aligned} \tag{4}$$

and the other one is with the ℓ_1 norm

$$\begin{aligned} &\min_{\mathbf{S}} \|\mathbf{S}-\mathbf{A}\|_1 \\ &\forall\ i,\ \mathbf{s}^i\mathbf{1}=1,\ s_{ij}\in[0,1],\ rank(\mathbf{L}_S)=n-c \end{aligned} \tag{5}$$

The above two problems can be solved with iterative methods, in which eigendecomposion is performed on $\mathbf{L}_S$ in each iteration. However, CAN and CLR are time-consuming since they involve multiple eigendecompositions.

In 2017, Chen et al. proposed a self-balanced min-cut algorithm to simultaneously minimize the graph cut and balance the partition across all clusters [3]. In their method, the objective function of their method is as follows

$$\min_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},s} \|\mathbf{A}-s\mathbf{Y}\mathbf{Y}^T\|_F^2 \tag{6}$$

where $s > 0$ is a balance parameter. An iterative method is proposed to solve the above problem.

3.2 Normalized Cut Revisit

Normalized Cut is a very classic spectral clustering method with the objective function in Eq. (2) and much effort has been devoted to improve it. Yu et al. proposed a multiclass spectral clustering (MSC), in which a spectral rotation is proposed to solve the following k-way normalized cut problem. In their method, problem (2) can be written as

$$\max_{\mathbf{Y}^T\mathbf{D}_A\mathbf{Y}=\mathbf{I}} Tr(\mathbf{Y}^T\mathbf{A}\mathbf{Y}) \tag{7}$$

Then they replace $\mathbf{Y}$ with scaled partition matrix $\mathbf{Z}=\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}$ and rewrite the above problem as

$$\max_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},\ \mathbf{Z}=\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}} Tr(\mathbf{Z}^T\mathbf{A}\mathbf{Z}) \tag{8}$$

It is difficult to directly solve problem (8). A well known way is to relax $\mathbf{Z}$ from the discrete values to the continuous ones and form the following new problem

$$\max_{\mathbf{Z}^T\mathbf{D}_A\mathbf{Z}=\mathbf{I}_c,\ \mathbf{Z}=\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}} Tr(\mathbf{Z}^T\mathbf{A}\mathbf{Z}) \tag{9}$$

According to Proposition 1 in [21], the optimal solution of $\mathbf{Z}$ is $\{\mathbf{Z}^*\mathbf{R} : \mathbf{R}^T\mathbf{R}=\mathbf{I}_c\}$ where $\mathbf{Z}^*\in\mathbb{R}^{n\times c}$ consists of the c column vectors of the eigenvectors of $\mathbf{D}_A^{-1}\mathbf{A}$ which correspond to the c biggest eigenvalues.

To obtain the discrete solution $\mathbf{Y}$, they first compute an approximate solution $\mathbf{Y}^* = Diag(\mathbf{Z}^*(\mathbf{Z}^*)^T)^{-\frac{1}{2}}\mathbf{Z}^*$, then suitable $\mathbf{R}$ and $\mathbf{Y}$ can be learned such that $\mathbf{Y}^*\mathbf{R}$ is closest to $\mathbf{Y}$ by solving the following problem

$$\min_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},\ \mathbf{R}\in\mathbb{R}^{c\times c},\ \mathbf{R}^T\mathbf{R}=\mathbf{I}_c} \|\mathbf{Y}-\mathbf{Y}^*\mathbf{R}\|_F^2 \tag{10}$$

However, the final clustering result $\mathbf{Y}$ may deviate from the result by directly optimizing the original objective function since $\mathbf{Y}^*$ is an approximate solution. In 2017, Chen et al. proposed an improved spectral rotation (ISR) to solve problem (8) [4]. They first relax problem (8) to the following continuous problem

$$\max_{\mathbf{F}^T\mathbf{F}=\mathbf{I},\ \mathbf{F}=\mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}} Tr(\mathbf{F}^T\mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}}\mathbf{F}) \tag{11}$$

Then the optimal solution of $\mathbf{F}$ is $\{\mathbf{F}^*\mathbf{R} : \mathbf{R}^T\mathbf{R}=\mathbf{I}_c\}$ where $\mathbf{F}^*\in\mathbb{R}^{n\times c}$ is the c column vectors of the eigenvectors of $\mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}}$ that correspond to the c biggest eigenvalues. With $\mathbf{F}^*$, they proposed to directly obtain the discrete solution $\mathbf{Y}$ such that $\left\|\mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}-\mathbf{F}^*\mathbf{R}\right\|_F^2$ is minimized, i.e., by solving the following problem

$$\max_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},\ \mathbf{R}\in\mathbb{R}^{c\times c},\ \mathbf{R}^T\mathbf{R}=\mathbf{I}_c} Tr((\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}\mathbf{Y}^T\mathbf{D}_A^{\frac{1}{2}}\mathbf{F}^*\mathbf{R}) \tag{12}$$

Then they proposed an alternative method to solve problem (12). To cope with large-scale data, they further incorporated the anchor-based strategy to extend ISR to a Scalable Normalized Cut (SNC) for large-scale data. However, the new method still uses a two-stage approach. In this paper, we propose to directly optimize problem (8).

4 THE DIRECT NORMALIZED CUT ALGORITHM

4.1 The Optimization Algorithm

We first rewrite problem (8) as follows

$$\max_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},\ \mathbf{F}=\mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}} Tr(\mathbf{F}^T\mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}}\mathbf{F}) \tag{13}$$

where $\mathbf{Y}\in\mathbf{\Psi}^{n\times c}$ is the cluster indicator matrix.

It can be verified that

$$\begin{aligned} Tr(\mathbf{F}^T\mathbf{F}) &= Tr(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}\mathbf{Y}^T\mathbf{D}_A\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}) \\ &= Tr(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-1}) = c \end{aligned} \tag{14}$$

Therefore, solving problem (13) is equivalent to solving

$$\max_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},\ \mathbf{F}=\mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}} Tr(\mathbf{F}^T\mathbf{M}\mathbf{F}) \tag{15}$$

where $\mathbf{M}=\mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}}+\lambda\mathbf{I}_n$, in which $\mathbf{I}_n$ is an identity matrix and λ is a sufficiently large constant to make $\mathbf{M}$ positive semi-definite.

In this paper, we propose the following two-step method to solve problem (15):

(1) Update $\mathbf{G}=\mathbf{M}\mathbf{F}$ where $\mathbf{F}=\mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}$.
(2) Solve $\max_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},\ \mathbf{F}=\mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}} Tr(\mathbf{F}^T\mathbf{G})$.

Steps 1 and 2 are iteratively executed until problem (15) converges (problem (13) also converges). In step 2, we need to solve a complex problem . Let $\mathbf{U} = \mathbf{Y}^T\mathbf{D}_A\mathbf{Y}$, we have $u_{lj} = \sum_{i=1}^{n} d_{ii}y_{il}y_{ij}$. Note that $y_{il}y_{ij} = 0$ if $l \neq j$, we know that $u_{lj} = 0$ if $l \neq j$. Therefore, $\mathbf{U}$ should be a $c \times c$ diagonal matrix, and $\mathbf{U}^{-\frac{1}{2}}$ is also a diagonal matrix in which the l-th diagonal element is defined as $\frac{1}{\sqrt{\mathbf{y}_l^T\mathbf{D}_A\mathbf{y}_l}}$. Then the problem in step 2 can be rewritten as

$$\max_{\mathbf{Y}\in\Psi^{n\times c}} \sum_{l=1}^{c} \frac{\mathbf{y}_l^T\sqrt{\mathbf{D}_A}\mathbf{g}_l}{\sqrt{\mathbf{y}_l^T\mathbf{D}_A\mathbf{y}_l}} \tag{16}$$

Since $\sqrt{\mathbf{y}_l^T\mathbf{D}_A\mathbf{y}_l}$ involves all rows of $\mathbf{Y}$, we propose to sequentially solve $\mathbf{Y}$ row by row and fix the other rows of $\mathbf{Y}$ as constants. Suppose we have obtained the optimal solution $\overline{\mathbf{Y}}$. To solve the i-th row $\mathbf{y}^i$, we only need to consider the increment of the objective function value from $y_{il} = 0$ to $y_{il} = 1$. Since $\overline{\mathbf{y}}_l^T\mathbf{D}_A\overline{\mathbf{y}}_l$ and $\overline{\mathbf{y}}_l^T\sqrt{\mathbf{D}_A}\mathbf{g}_l$ can be computed once before we solve $\mathbf{y}^i$, we can compute the increment as

$$s_{il} = \frac{\overline{\mathbf{y}}_l^T\sqrt{\mathbf{D}_A}\mathbf{g}_l + g_{il}\sqrt{d_{ii}}(1-\overline{y}_{il})}{\sqrt{\overline{\mathbf{y}}_l^T\mathbf{D}_A\overline{\mathbf{y}}_l + d_{ii}(1-\overline{y}_{il})}} - \frac{\overline{\mathbf{y}}_l^T\sqrt{\mathbf{D}_A}\mathbf{g}_l - \overline{y}_{il}\sqrt{d_{ii}}g_{il}}{\sqrt{\overline{\mathbf{y}}_l^T\mathbf{D}_A\overline{\mathbf{y}}_l - d_{ii}\overline{y}_{il}}} \tag{17}$$

Then the optimal solution of $\mathbf{y}^i$ can be obtained as

$$y_{il} = < l = \arg\max_{l'\in[1,c]} s_{il'} > \tag{18}$$

where $< . >$ is 1 if the argument is true or 0 otherwise and s_{il} is defined in Eq. (17).

The detailed algorithm to solve problem (12), namely Direct Normalized Cut (DNC), is summarized in Algorithm 1. In the new algorithm, we need $O(nc)$ time to solve $\mathbf{Y}$ according to Eq. (18) because $\overline{\mathbf{y}}_l^T\mathbf{D}_A\overline{\mathbf{y}}_l$ and $\overline{\mathbf{y}}_l^T\sqrt{\mathbf{D}_A}\mathbf{g}_l$ can be computed before solving $\mathbf{Y}$ and updated after solving $\mathbf{y}^i$. Here, the discrete solution $\mathbf{Y}$ converges very fast due to its limited solution space. Although updating $\mathbf{G}$ involves computing $(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}$, it can be finished in $O(c)$ time since $\mathbf{Y}^T\mathbf{D}_A\mathbf{Y}$ is a diagonal matrix. However, computing $\mathbf{G} = \mathbf{MF}$ takes $O(n^2c)$ time since $\mathbf{M}$ is a $n \times n$ matrix. Therefore, DNC has a computational complexity of $O(n^2c)$.

Algorithm 1 Direct Normalized Cut (DNC) to solve problem (15)

```
1: Input: the similarity matrix $\mathbf{A}$.
2: Find a large enough constant $\lambda$ to make $\mathbf{M} = \mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}} + \lambda\mathbf{I}_c$ positive semi-definite.
3: Initialize $\mathbf{Y}$.
4: repeat
5:     $\mathbf{G} = \mathbf{M}\mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}$.
6:     Update $\mathbf{Y}$ according to Eq. (18).
7: until problem (15) converges
8: Output: the clustering result $\mathbf{Y}$.
```

4.2 Convergence Analysis of Algorithm 1

Now we prove the convergence of Algorithm 1 as follows.

THEOREM 1. *Algorithm 1 monotonically increases problem (15) in each iteration until the algorithm converges.*

PROOF. Denote $\mathbf{F} = \mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}$ in the t-th and $t+1$-th iterations as $\mathbf{F}_{t+1}$ and $\mathbf{F}_t$, respectively. According to lines 5 and 6 in Algorithm 1, we have

$$Tr(\mathbf{F}_{t+1}^T\mathbf{M}\mathbf{F}_t) \geq Tr(\mathbf{F}_t^T\mathbf{M}\mathbf{F}_t) \tag{19}$$

Since the matrix $\mathbf{M}$ is positive semi-definite, we can rewrite $\mathbf{M} = \mathbf{Q}_M^T\mathbf{Q}_M$ via Cholesky factorization. Then Eq. (19) can be rewritten as

$$Tr(\mathbf{F}_{t+1}^T\mathbf{Q}_M^T\mathbf{Q}_M\mathbf{F}_t) \geq Tr(\mathbf{F}_t^T\mathbf{Q}_M^T\mathbf{Q}_M\mathbf{F}_t) \tag{20}$$

The inequation $\|\mathbf{Q}_M\mathbf{F}_{t+1} - \mathbf{Q}_M\mathbf{F}_t\|_F^2 \geq 0$ can be rewritten as

$$\begin{aligned} &Tr(\mathbf{F}_{t+1}^T\mathbf{Q}_M^T\mathbf{Q}_M\mathbf{F}_{t+1}) - 2Tr(\mathbf{F}_{t+1}^T\mathbf{Q}_M^T\mathbf{Q}_M\mathbf{F}_t) \\ &+ Tr(\mathbf{F}_t^T\mathbf{Q}_M^T\mathbf{Q}_M\mathbf{F}_t) \geq 0 \end{aligned} \tag{21}$$

Multiplying Eq. (20) by 2 and summing over it and Eq. (21) gives

$$Tr(\mathbf{F}_{t+1}^T\mathbf{Q}_M^T\mathbf{Q}_M\mathbf{F}_{t+1}) \geq Tr(\mathbf{F}_t^T\mathbf{Q}_M^T\mathbf{Q}_M\mathbf{F}_t) \tag{22}$$

which equals to

$$Tr(\mathbf{F}_{t+1}^T\mathbf{M}\mathbf{F}_{t+1}) \geq Tr(\mathbf{F}_t^T\mathbf{M}\mathbf{F}_t) \tag{23}$$

Therefore, Algorithm 1 monotonically increases problem (15) in each iteration until the algorithm converges. □

5 THE FAST NORMALIZED CUT ALGORITHM

Although DNC has lower complexity than normalized cut with k-means, spectral rotation, and improved spectral rotation, its square time complexity hinders its usage in large-scale data. In this section, we extend DNC to a Fast Normalized Cut (FNC) for large-scale data and the new method has linear time and space complexities.

5.1 Anchor Generation with Balanced k-means

In the anchor-based strategy, the key problem is how to effectively obtain a set of anchors which can best represent the whole data set. Note that although a real-world data set usually consists of unbalanced clusters with different different cluster sizes, the number of anchors is usually much larger than the number of clusters so we hope these anchors are evenly distributed in the data. To achieve this goal, we wish to uncover balanced clusters from the data. Although it was shown that k-means can obtain better anchors than random selection [1, 12], the quality of the generated anchors cannot be guaranteed since k-means may produce unbalanced clusters. In this paper, we propose a Balanced k-means to produce balanced partition for anchor generation.

Given a data set $\mathbf{X} \in \mathbb{R}^{d\times n}$, where d is the number of features and n is the number of objects. To uncover c clusters in $\mathbf{X}$, the classical k-means clustering can be reformulated as follows

$$\min_{\mathbf{F}\in\mathbf{\Psi}^{n\times c},\mathbf{H}\in\mathbb{R}^{d\times c}} \left\|\mathbf{X}-\mathbf{H}\mathbf{F}^T\right\|_F^2 \tag{24}$$

where $\mathbf{F}$ is an cluster indicator matrix and $\mathbf{H} \in \mathbb{R}^{d\times c}$ is the c cluster centers.

Given a matrix $\mathbf{F} \in \mathbb{R}^{n\times c}$, Chen et al. have proved the following regularization

$$\|\mathbf{F}\|_e = Tr(\mathbf{F}^T\mathbf{1}_n\mathbf{1}_n^T\mathbf{F}) \tag{25}$$

arrives its minimum when $\sum_{i=1}^n f_{il} = \frac{n}{c}$ if $\frac{n}{c}$ is an integer, or $\sum_{i=1}^n f_{il} = \{\lfloor\frac{n}{c}\rfloor, \lceil\frac{n}{c}\rceil\}$ otherwise ($l \in [1,c]$) [3]. That is, minimizing $\|\mathbf{F}\|_e$ results in the most balanced partition. Therefore, we incorporate the balance regularization into problem (24) to form the objective function of balanced k-means as follows

$$\min_{\mathbf{F}\in\mathbf{\Psi}^{n\times c},\mathbf{H}\in\mathbb{R}^{d\times c}} \left\|\mathbf{X}-\mathbf{H}\mathbf{F}^T\right\|_F^2 + \gamma\|\mathbf{F}\|_e \tag{26}$$

where γ is a regularization parameter.

We present an iterative approach to optimize problem (26), in which $\mathbf{F}$ and $\mathbf{H}$ are alternatively solved. If $\mathbf{F}$ is fixed, by setting the derivative of problem (26) w.r.t $\mathbf{H}$ to zero, we obtain the optimal solution of $\mathbf{H}$ as

$$\mathbf{H} = \mathbf{X}\mathbf{F}(\mathbf{F}^T\mathbf{F})^{-1} \tag{27}$$

Then we fix $\mathbf{H}$ to update $\mathbf{F}$ by solving the following problem

$$\min_{\mathbf{F}\in\mathbf{\Psi}^{n\times c}} \left\|\mathbf{X}-\mathbf{H}\mathbf{F}^T\right\|_F^2 + \gamma\|\mathbf{F}\|_e \tag{28}$$

which can be rewritten as

$$\min_{\mathbf{F}\in\mathbf{\Psi}^{n\times c}} Tr(\mathbf{F}\mathbf{H}^T\mathbf{H}\mathbf{F}^T) + \gamma Tr(\mathbf{F}^T\mathbf{1}_n\mathbf{1}_n^T\mathbf{F}) - 2Tr(\mathbf{X}^T\mathbf{H}\mathbf{F}^T) \tag{29}$$

It can be verified that $Tr(\mathbf{F}^T\mathbf{F}) = n$ and

$$Tr(\mathbf{F}\mathbf{H}^T\mathbf{H}\mathbf{F}^T) = \sum_{i=1}^n\sum_{l=1}^c\sum_{j=1}^d\sum_{t=1}^c f_{il}h_{jl}h_{jt}f_{it} \tag{30}$$

Note that $f_{il}f_{it} = 0$ if $l \neq t$, the above equation equals to

$$\begin{aligned} Tr(\mathbf{F}\mathbf{H}^T\mathbf{H}\mathbf{F}^T) &= \sum_{i=1}^n\sum_{l=1}^c\sum_{j=1}^d f_{il}h_{jl}^2 \\ &= Tr(\mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H})\mathbf{F}^T) \end{aligned} \tag{31}$$

Therefore, problem (28) can be rewritten as

$$\max_{\mathbf{F}\in\mathbf{\Psi}^{n\times c}} Tr(\mathbf{F}^T\mathbf{E}\mathbf{F}) + Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}^T) \tag{32}$$

where $\mathbf{E} = n\gamma\mathbf{I}_n - \gamma\mathbf{1}_n\mathbf{1}_n^T$.

In this paper, we propose the following two-step method to solve the above problem:

(1) Update $\mathbf{G} = \mathbf{E}\mathbf{F}$;

(2) Solve $\max_{\mathbf{F}\in\mathbf{\Psi}^{n\times c}} Tr(\mathbf{F}^T\mathbf{G}) + \frac{1}{2}Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}^T)$.

Note that the problem in step 2 can be rewritten as

$$\max_{\mathbf{F}\in\mathbf{\Psi}^{n\times c}} Tr((\mathbf{G} + \mathbf{X}^T\mathbf{H} - \frac{1}{2}\mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}^T) \tag{33}$$

Then the optimal solution of f_{il} is

$$f_{il} = < t = \arg\max_{l'\in[1,c]} q_{il'} > \tag{34}$$

where $< . >$ is 1 if the argument is true or 0 otherwise, and $\mathbf{Q} = [q_{il}]$ is defined as

$$\mathbf{Q} = \mathbf{G} + \mathbf{X}^T\mathbf{H} - \frac{1}{2}\mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}) \tag{35}$$

The detailed algorithm to solve problem (26), namely the balanced k-means (BKM), is summarized in Algorithm 2. In the new algorithm, the cluster centers $\mathbf{H}$ and cluster indicator matrix $\mathbf{F}$ are iteratively updated until convergence. In this algorithm, since $\mathbf{F}^T\mathbf{F}$ is a diagonal matrix, we only need $O(dnc)$ time to compute $\mathbf{H}$. We need $O(nc)$ time to compute $\mathbf{G} = \mathbf{1}(\mathbf{1}^T\mathbf{F})$. Therefore, the overall computational complexity of BKM is $O(dnc)$ which is the same as the computational complexity of k-means.

Algorithm 2 Balanced k-means Clustering (BKM) to solve problem (26)

```
1:  Input: the data matrix $\mathbf{X}$ and a regularization parameter $\gamma$.
2:  Initialize the cluster indicator matrix $\mathbf{F}$.
3:  repeat
4:      Update $\mathbf{H}$ as $\mathbf{H} = \mathbf{X}\mathbf{F}(\mathbf{F}^T\mathbf{F})^{-1}$.
5:      repeat
6:          Update $\mathbf{G} = (n\gamma\mathbf{I}_n - \gamma\mathbf{1}_n\mathbf{1}_n^T)\mathbf{F}$.
7:          Independently update each row of $\mathbf{F}$ according to Eq. (34) and (35).
8:      until $\mathbf{F}$ does not change
9:  until problem (26) converges
10: Output: the clustering result $\mathbf{F}$.
```

To prove the convergence of Algorithm 2, we need the following lemma.

Lemma 1. *Lines 5 to 8 in Algorithm 2 monotonically increase problem (26) in each iteration.*

Proof. Denote $\mathbf{F}$ in the t-th and $t+1$-th iterations as $\mathbf{F}_{t+1}$ and $\mathbf{F}_t$, respectively. According to lines 6 and 7 in Algorithm 2, we have

$$\begin{aligned} &Tr(\mathbf{F}_{t+1}^T\mathbf{E}\mathbf{F}_t) + \frac{1}{2}Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}_{t+1}^T) \\ &\geq Tr(\mathbf{F}_t^T\mathbf{E}\mathbf{F}_t) + \frac{1}{2}Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}_t^T) \end{aligned} \tag{36}$$

It can be verified that all eigenvalues in $\mathbf{E}$ are non-negative, so $\mathbf{E}$ is positive semi-definite. Then we can rewrite $\mathbf{E} = \mathbf{Q}_E^T\mathbf{Q}_E$ via Cholesky factorization. Then Eq. (36) can be rewritten as

$$\begin{aligned} &Tr(\mathbf{F}_{t+1}^T\mathbf{Q}_E^T\mathbf{Q}_E\mathbf{F}_t) + \frac{1}{2}Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}_{t+1}^T) \\ &\geq Tr(\mathbf{F}_t^T\mathbf{Q}_E^T\mathbf{Q}_E\mathbf{F}_t) + \frac{1}{2}Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}_t^T) \end{aligned} \tag{37}$$

Rewrite the inequation $\|\mathbf{Q}_E\mathbf{F}_{t+1} - \mathbf{Q}_E\mathbf{F}_t\|_F^2 \geq 0$ as

$$\begin{aligned} & Tr(\mathbf{F}_{t+1}^T\mathbf{Q}_E^T\mathbf{Q}_E\mathbf{F}_{t+1}) - 2Tr(\mathbf{F}_{t+1}^T\mathbf{Q}_E^T\mathbf{Q}_E\mathbf{F}_t) \\ & + Tr(\mathbf{F}_t^T\mathbf{Q}_E^T\mathbf{Q}_E\mathbf{F}_t) \geq 0 \end{aligned} \tag{38}$$

Multiplying Eq. (37) by 2 and summing over it with Eq. (38) gives

$$\begin{aligned} & Tr(\mathbf{F}_{t+1}^T\mathbf{Q}_E^T\mathbf{Q}_E\mathbf{F}_{t+1}) + Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}_{t+1}^T) \\ & \geq Tr(\mathbf{F}_t^T\mathbf{Q}_E^T\mathbf{Q}_E\mathbf{F}_t) + Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}_t^T) \end{aligned} \tag{39}$$

which equals to

$$\begin{aligned} & Tr(\mathbf{F}_{t+1}^T\mathbf{E}\mathbf{F}_{t+1}) + Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}_{t+1}^T) \\ & \geq Tr(\mathbf{F}_t^T\mathbf{E}\mathbf{F}_t) + Tr((2\mathbf{X}^T\mathbf{H} - \mathbf{1}_n\mathbf{1}_d^T(\mathbf{H}\circ\mathbf{H}))\mathbf{F}_t^T) \end{aligned} \tag{40}$$

The above inequation can be further rewritten as

$$\begin{aligned} & \left\|\mathbf{X} - (\mathbf{H}\circ\mathbf{H})\mathbf{F}_{t+1}^T\right\|_F^2 + \gamma\left\|\mathbf{F}_{t+1}\right\|_e \\ \leq & \left\|\mathbf{X} - (\mathbf{H}\circ\mathbf{H})\mathbf{F}_t^T\right\|_F^2 + \gamma\left\|\mathbf{F}_t\right\|_e \end{aligned} \tag{41}$$

Therefore, lines 5 to 8 in Algorithm 2 monotonically increase problem (26) in each iteration. □

Then with the above lemma, we can verify the following theorem

Theorem 2. *Algorithm 2 monotonically increases problem (26) in each iteration until the algorithm converges.*

5.2 The Optimization Algorithm

Assume that we have obtained m anchors $\mathbf{W} \in \mathbb{R}^{d\times m}$ with BKM, the next step is to construct a representation matrix $\mathbf{B} \in \mathbb{R}^{n\times m}$. Chen et al. proposed to compute $\mathbf{b}_{ij}$ as follows [4]

$$b_{ij} = \begin{cases} \frac{d_{i,k+1} - \|\mathbf{x}_i - \mathbf{w}_j\|_2^2}{kd_{i,k+1} - \sum_{h=1}^k d_{i,h}} & \mathbf{x}_j \in \mathcal{N}_k(\mathbf{x}_i) \\ 0 & \text{otherwise} \end{cases} \tag{42}$$

where $d_{i,h}$ is the square of Euclidean distance between $\mathbf{x}_i$ and its h-th nearest neighbor, and $\mathcal{N}_k(\mathbf{x}_i)$ contains the k nearest neighbors of $\mathbf{x}_i$.

After obtaining the representative similarity matrix $\mathbf{B}$, we can compute an approximate affinity matrix $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{P}\mathbf{P}^T \tag{43}$$

where $\mathbf{P} \in \mathbb{R}^{n\times m} = \mathbf{B}\Delta^{-\frac{1}{2}}$

$$\mathbf{P} = \mathbf{B}\Delta^{-\frac{1}{2}} \tag{44}$$

where $\Delta \in \mathbb{R}^{m\times m}$ is a diagonal matrix and the j-th entry is defined as $\Delta_{jj} = \sum_{i=1}^n b_{ij}$. Note that in real applications, we need not compute $\mathbf{A}$ but directly perform computation with $\mathbf{B}$.

According to Theorem 1 in [4], the degree matrix of $\mathbf{A}$ computed according to Eq. (43) should be an identity matrix, i.e., $\mathbf{D}_A = \mathbf{I}_n$. Then problem (13) can be rewritten as

$$\max_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},\ \mathbf{F}=\mathbf{Y}(\mathbf{Y}^T\mathbf{Y})^{-\frac{1}{2}}} Tr(\mathbf{F}^T\mathbf{P}\mathbf{P}^T\mathbf{F}) \tag{45}$$

Similar to DNC, we propose the following two-step method to solve the above problem:

(1) Update $\mathbf{G} = \mathbf{P}\mathbf{P}^T\mathbf{F}$ where $\mathbf{F} = \mathbf{Y}(\mathbf{Y}^T\mathbf{Y})^{-\frac{1}{2}}$.
(2) Solve $\max_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c},\mathbf{F}=\mathbf{Y}(\mathbf{Y}^T\mathbf{Y})^{-\frac{1}{2}}} Tr(\mathbf{F}^T\mathbf{G})$.

Steps 1 and 2 are iteratively executed until problem (45) converges. Similar to Eq. (16), the problem in step 2 can be rewritten as

$$\max_{\mathbf{Y}\in\mathbf{\Psi}^{n\times c}} \sum_{l=1}^c \frac{\mathbf{y}_l^T\mathbf{g}_l}{\sqrt{\mathbf{y}_l^T\mathbf{y}_l}} \tag{46}$$

where $\mathbf{G} = \mathbf{P}\mathbf{P}^T\mathbf{Y}(\mathbf{Y}^T\mathbf{Y})^{-\frac{1}{2}}$ is fixed.

It can be verified that the optimal solution of $\mathbf{y}^i$ is

$$y_{il} =< l = \arg\max_{l'\in[1,c]} s_{il'} > \tag{47}$$

where $< . >$ is 1 if the argument is true or 0 otherwise, and s_{il} is defined as

$$s_{il} = \frac{\overline{\mathbf{y}}_l^T\mathbf{g}_l + g_{il}(1-\overline{y}_{il})}{\sqrt{\overline{\mathbf{y}}_l^T\overline{\mathbf{y}}_l + (1-\overline{y}_{il})}} - \frac{\overline{\mathbf{y}}_l^T\mathbf{g}_l - \overline{y}_{il}g_{il}}{\sqrt{\overline{\mathbf{y}}_l^T\overline{\mathbf{y}}_l - \overline{y}_{il}}} \tag{48}$$

$\overline{\mathbf{Y}}$ is the obtained solution in the previous iteration.

The detailed algorithm to solve problem (45), namely Fast Normalized Cut (FNC), is summarized in Algorithm 3. Given a data matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$, we need $O(dnm)$ time to obtain m anchors by BKM and $O(nc)$ time to obtain $\mathbf{Y}$. Solving $\mathbf{Y}$ takes $O(nc)$ time according to Eq. (47) because $\overline{\mathbf{y}}_l^T\overline{\mathbf{y}}_l$ and $\overline{\mathbf{y}}_l^T\mathbf{g}_l$ can be computed before solving $\mathbf{Y}$ and updated after solving $\mathbf{y}^i$. Here, the discrete solution $\mathbf{Y}$ converges very fast due to its limited solution space. We need $O(nmc)$ time to update $\mathbf{G}$. Therefore, the overall computational complexity of FNC is $O(dnm + nmc)$.

Algorithm 3 Fast Normalized Cut (FNC) to solve problem (45)

```
Input: Data matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$, the number of nearest neighbors $k$, the number of anchors $m$ and a regularization parameter $\gamma$.
Run BKM with to obtain $m$ clusters from $\mathbf{X}$, use the resulting $\mathbf{H}$ as $m$ anchors $\mathbf{W}$, construct a sparse representation matrix $\mathbf{B}$ according to Eq. (42) and compute $\mathbf{P}$ according to Eq. (44).
Initialize $\mathbf{Y}$.
repeat
    Update $\mathbf{G} = \mathbf{P}\mathbf{P}^T\mathbf{Y}(\mathbf{Y}^T\mathbf{Y})^{-\frac{1}{2}}$.
    Update $\mathbf{Y}$ according to Eq. (47).
until problem (45) converges
Output: the clustering result $\mathbf{Y}$.
```

Now we prove the convergence of Algorithm 3 as follows

Theorem 3. *Algorithm 3 monotonically increases the objective function value of problem (45) in each iteration until the algorithm converges.*

Proof. Denote $\mathbf{F} = \mathbf{D}_A^{\frac{1}{2}}\mathbf{Y}(\mathbf{Y}^T\mathbf{D}_A\mathbf{Y})^{-\frac{1}{2}}$ in the t-th and $t+1$-th iterations as $\mathbf{F}_{t+1}$ and $\mathbf{F}_t$, respectively. According to lines 3 to 6 in Algorithm 3, we have

$$Tr(\mathbf{F}_{t+1}^T\mathbf{P}\mathbf{P}^T\mathbf{F}) \geq Tr(\mathbf{F}_t^T\mathbf{P}\mathbf{P}^T\mathbf{F}) \tag{49}$$

The inequation $\left\|\mathbf{P}^T\mathbf{F}_{t+1} - \mathbf{P}^T\mathbf{F}_t\right\|_F^2 \geq 0$ can be rewritten as

$$\begin{aligned} &Tr(\mathbf{F}_{t+1}^T\mathbf{P}\mathbf{P}^T\mathbf{F}_{t+1}) - 2Tr(\mathbf{F}_{t+1}^T\mathbf{P}\mathbf{P}^T\mathbf{F}_t) \\ &+ Tr(\mathbf{F}_t^T\mathbf{P}\mathbf{P}^T\mathbf{F}_t) \geq 0 \end{aligned} \tag{50}$$

Multiplying Eq. (49) by 2 and summing over it and Eq. (50) gives

$$Tr(\mathbf{F}_{t+1}^T\mathbf{P}\mathbf{P}^T\mathbf{F}_{t+1}) \geq Tr(\mathbf{F}_t^T\mathbf{P}\mathbf{P}^T\mathbf{F}_t) \tag{51}$$

Therefore, Algorithm 3 monotonically increases problem (45) in each iteration until the algorithm converges. □

Actually, Algorithms 1, 2 and 3 presented in this paper can be considered as applications of the reweighted method for a general optimization framework [16].

6 EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we present the experimental results to demonstrate the efficiency and effectiveness of the proposed three methods.

6.1 Anchor Generation

We first compared BKM with k-means and Bisection k-means for anchor generation. Given a partition $\mathcal{C}$ in which n objects are partitioned into c clusters, we compute the standard deviation of the sizes of clusters in $\mathcal{C}$ as $SD(\mathcal{C})$ to measure the balance degree of $\mathcal{C}$

$$SD(\mathcal{C}) = \sqrt{\frac{1}{c}\sum_{\mathcal{C}_i \in \mathcal{C}}(|\mathcal{C}_i| - \frac{n}{c})^2} \tag{52}$$

where $|\mathcal{C}_i|$ is the number of objects in $\mathcal{C}_i$. The smaller the $SD(\mathcal{C})$ is, the more balanced the partition $\mathcal{C}$ is. We also compute $SSE(\mathcal{C})$ to measure the quality of the clustering result

$$SSE(\mathcal{C}) = \sum_{\mathcal{C}_i \in \mathcal{C}}\sum_{\mathbf{x} \in \mathcal{C}_i}\|\mathbf{c}_i - \mathbf{x}\|^2 \tag{53}$$

where $\mathbf{c}_i$ is the cluster center of $\mathcal{C}_i$. The smaller the $SSE(\mathcal{C})$ is, the better the partition $\mathcal{C}$ is.

To evaluate the performance of BKM, k-means and Bisection k-means for anchor generation, we generated a 5-cluster synthetic data set which consists of 5400 objects and 5 features. In this experiment, the number of anchors was set as 14 numbers from 7 to 20 for three algorithms. The regularizer parameter γ of BKM was set as eight values $\{10^{-1}, 10^{-2}, 6 \times 10^{-3}, 4 \times 10^{-3}, 2 \times 10^{-3}, 10^{-3}, 10^{-4}\}$. The cluster balance results are shown in Figure 1. Figure 1(a) shows that $SD(\mathcal{C})$ decrease with the increase of γ, and the results of BKM with all parameters are better than those of k-means and bisection k-means. From Figure 1(b) and 1(c), we can observe that $SSE(\mathcal{C})$ of bisection k-means and BKM with $\gamma \geq 6 \times 10^{-3}$ are worse than the result of k-means. As γ decrease from 6×10^{-3}, $SSE(\mathcal{C})$ first drops and then rises. The smallest $SSE(\mathcal{C})$ is obtained by BKM with $\gamma = 2 \times 10^{-3}$. In real application, we can select proper γ according to both $SD(\mathcal{C})$ and $SSE(\mathcal{C})$.

Table 1: Characteristics of 7 data sets.

Data sets	Name	#Samples	#Features	#Classes
D_1	nci	61	5244	8
D_2	glass	214	9	6
D_3	USPSdata-20	1854	256	10
D_4	text1	1946	711	2
D_5	uspst	2007	256	10
D_6	segment	2310	19	7
D_7	MnistData-10	6996	784	10

6.2 Clustering Results on Small Data Sets

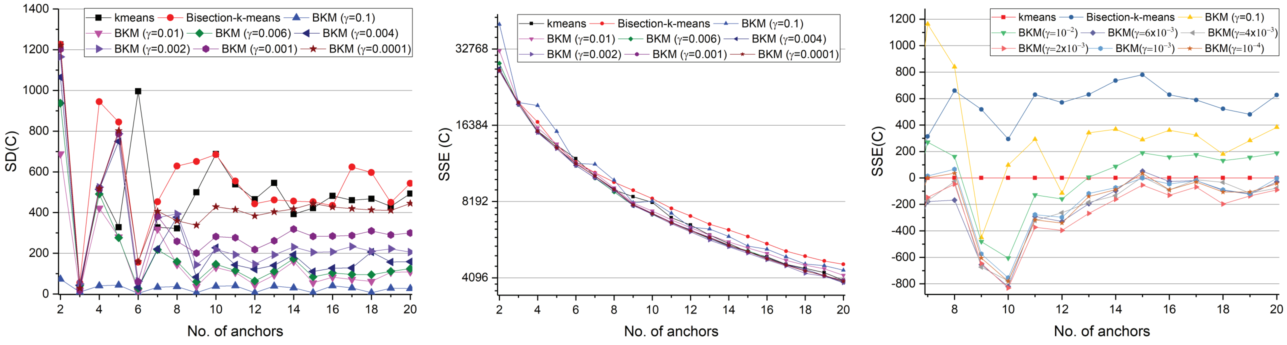
6.2.1 *Experiment results on benchmark data sets.*

In this experiment, 7 benchmark data sets were selected from the UCI Machine Learning Repository and a website [1]. Table 1 summarizes the characteristics of these data sets.

With the seven benchmark data sets, we compared DNC with normalized cut with k-means (NCut+KM), normalized cut with spectral rotation (NCut+SR, actually MSC), normalized cut with improved spectral rotation (NCut+ISR) [4], CAN [14], CLR [15] and SBMC [3]. For each data set, we set the number of nearest neighbors k to $\{10, 20, 30, 40, 50\}$ to construct five affinity matrices with the method in [15], and used these matrices to run all 7 methods. We computed the average accuracy of each algorithm on each data set, and show the results in Table 2. From these results, we can observe that DNC outperformed all the other methods on six data sets. Especially on D_4, DNC has over 29% improvement compared with the second-best method SBMC. On D_2 and D_6, DNC achieves nearly 6% and 14% improvements compared with the second-best methods, respectively. These results indicate the superior performance of DNC.

We selected D_5 to show the convergence curves of the objective function value and the number of iterations for obtaining $\mathbf{Y}$ in each main loop. The results are drawn in Figure 2. Figure 2(a) shows that the objective function value decreases very fast, indicating Algorithm 1 converges rapidly. Figure 2(b) shows that the average number of iterations for obtaining $\mathbf{Y}$ is around 4. Therefore, DNC can quickly obtain the final clustering results.

[1] http://www.escience.cn/people/chenxiaojun/data.html#



(a) $SD(\mathcal{C})$ versus the number of anchors.

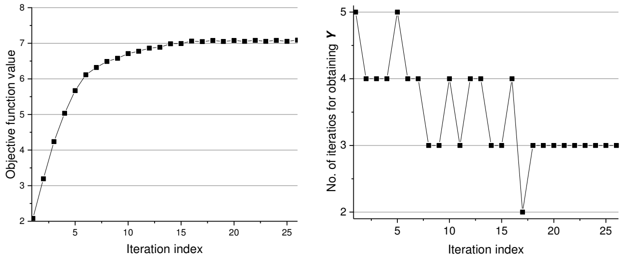
(b) $SSE(\mathcal{C})$ versus the number of anchors.

(c) Difference of $SSE(\mathcal{C})$ versus the number of anchors.

Figure 1: Cluster balance results on a synthetic data set, in terms of $SD(\mathcal{C})$ and $SSE(\mathcal{C})$. In Figure 1(c), the results of k-means are set as 0 and the differences between the results obtained by the other methods and k-means are shown.

Table 2: The average accuracy±standard deviation of four spectral clustering methods (The best result on each data set is highlighted in bold).

Data	NCut+KM	NCut+SR	NCut+ISR	CAN	CLR	SBMC	DNC
D_1	0.718 ± 0.054	0.662 ± 0.067	0.685 ± 0.045	0.675 ± 0.032	0.690 ± 0.000	0.698 ± 0.144	$\mathbf{0.741 \pm 0.058}$
D_2	0.450 ± 0.004	0.445 ± 0.015	0.455 ± 0.006	0.509 ± 0.027	0.487 ± 0.000	0.506 ± 0.019	$\mathbf{0.542 \pm 0.000}$
D_3	0.663 ± 0.006	0.667 ± 0.012	0.666 ± 0.007	0.599 ± 0.230	0.621 ± 0.000	0.641 ± 0.045	$\mathbf{0.670 \pm 0.018}$
D_4	0.502 ± 0.000	0.502 ± 0.000	0.503 ± 0.001	0.503 ± 0.000	0.500 ± 0.000	0.728 ± 0.163	$\mathbf{0.941 \pm 0.008}$
D_5	0.658 ± 0.000	0.652 ± 0.000	0.659 ± 0.001	0.632 ± 0.077	0.607 ± 0.000	0.660 ± 0.137	$\mathbf{0.674 \pm 0.008}$
D_6	0.503 ± 0.030	0.500 ± 0.047	0.430 ± 0.143	0.431 ± 0.037	0.509 ± 0.000	0.529 ± 0.068	$\mathbf{0.603 \pm 0.032}$
D_7	$\mathbf{0.638 \pm 0.021}$	0.611 ± 0.043	0.636 ± 0.019	0.566 ± 0.071	0.591 ± 0.000	0.523 ± 0.108	0.560 ± 0.041



(a) Objective function value of problem (12).

(b) No. of iterations for obtaining $\mathbf{Y}$ in each main loop.

Figure 2: Objective function value of problem (13) and no. of iterations for obtaining Y in each main loop of DNC on D_5.

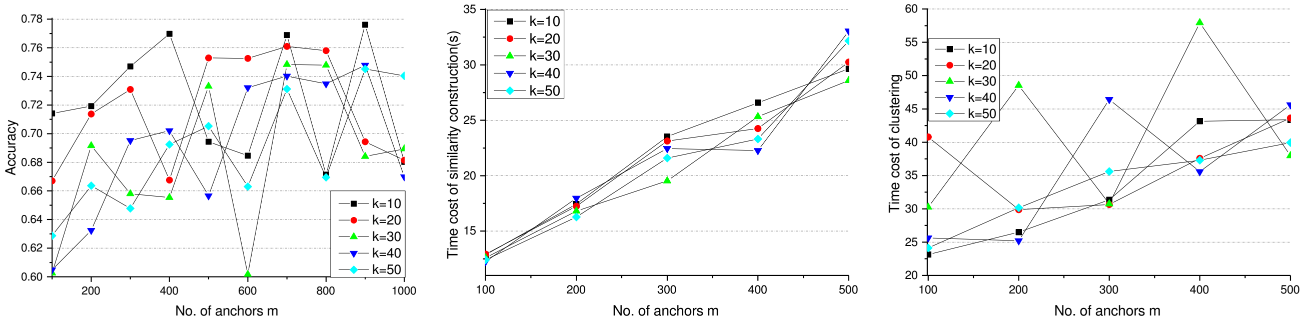
6.3 Clustering Results on Large-scale Data Sets

6.3.1 Benchmark data sets. 5 large-scale benchmark data sets were selected from the UCI Machine Learning Repository and a website [1] for this experiment. Table 3 summarizes the characteristics of these 5 data sets.

Table 3: Characteristics of 5 data sets.

Data sets	Name	#Samples	#Features	#Classes
D_1	uspst	2007	256	10
D_2	segment	2310	19	7
D_3	isolet5	7797	617	26
D_4	caltech101	8641	256	101
D_5	USPSdata	9298	256	10

6.3.2 Results and Analysis. We compared FNC with six spectral clustering methods, including NCut with k-means (NCut+KM) [13], Ratio Cut with k-means (RCut+KM) [8], multiclass spectral clustering (MSC) [21], Scalable Normalized Cut (SNC) [4], LSC [1] and k-means-based approximate spectral clustering (KASP) [20]. The number of nearest neighbors k was set to $\{10, 20, 30, 40, 50\}$ for all data sets, and the number of anchors m was set to $\{100, 200, \cdots, 1000\}$ for LSC, KASP, SNC and FNC. We used the similarity construction method in 5.1 to construct representative similarity matrices for each data set to run NCut+KM, RCut+KM and MSC. For SNC, KASP and LSC, we used k-means to generate anchors and the similarity construction method in 5.1



(a) Accuracy versus m and k.

(b) Time cost of similarity construction versus m and k.

(c) Time cost of clustering versus m and k.

Figure 3: Accuracy and time cost of FNC versus the no. of anchors m and the number of nearest neighbors k on D_5.

Table 4: The average accuracy±standard deviation of seven spectral clustering methods (The best result on each data set is highlighted in bold).

Data	D_1	D_2	D_3	D_4	D_5
Ncut+KM	0.671 ± 0.005	0.457 ± 0.042	0.553 ± 0.008	0.262 ± 0.005	0.666 ± 0.003
Rcut+KM	0.672 ± 0.003	0.457 ± 0.042	0.553 ± 0.005	0.266 ± 0.005	0.666 ± 0.003
MSC	0.674 ± 0.012	0.419 ± 0.101	0.508 ± 0.020	0.092 ± 0.000	0.643 ± 0.045
LSC	0.660 ± 0.037	0.639 ± 0.035	0.480 ± 0.074	0.208 ± 0.016	0.648 ± 0.028
KASP	0.703 ± 0.022	0.526 ± 0.028	0.559 ± 0.043	0.279 ± 0.015	0.703 ± 0.026
SNC	0.591 ± 0.026	0.468 ± 0.015	0.398 ± 0.025	0.252 ± 0.020	0.599 ± 0.027
FNC	$\mathbf{0.707 \pm 0.041}$	$\mathbf{0.644 \pm 0.025}$	$\mathbf{0.580 \pm 0.039}$	$\mathbf{0.286 \pm 0.011}$	$\mathbf{0.706 \pm 0.051}$

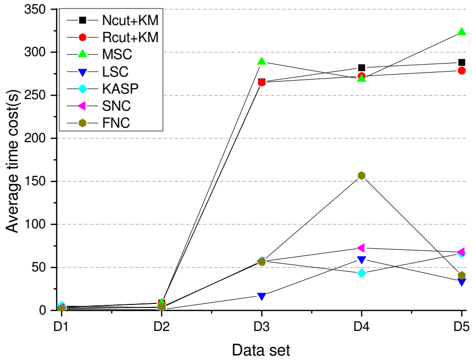


Figure 4: Average time costs of seven spectral clustering methods.

to construct affinity matrices. For each data set with a specific number of anchors m, the regularizer parameter γ in BKM was set to 5 values of $\{10^{-4}, 10^{-3}, \cdots, 1\}$ to produce 5 clustering results and the clustering result with the smallest $SD(\mathcal{C})SSE(\mathcal{C})$ was selected as anchors to run FNC. The average clustering accuracy and standard deviation of seven spectral clustering algorithms on 5 data sets are shown in Table 4 and the time costs of these methods are shown in Figure 4. Table 4 shows that FNC outperformed all the other methods on all data sets. Specifically, FNC achieves a nearly 4% average improvement on D_3 compared with the second-best method NCut+KM. Since D_1 and D_2 in Table 3 are identical as D_5 and D_6 in Table 1, we can directly compare the results of DNC and FNC on the two data sets. We can observe that FNC even produced better results than DNC, which indicates the superiority of FNC. Figure 4 shows that the time costs of FNC are much smaller than NCut+KM, RCut+KM and MSC, and it costs a similar time as SNC, KASP and LSC.

6.3.3 Parameter study. We select D_5 to show the relationship between the clustering accuracy of FNC and the two parameters m and k. The results are drawn in Figure 3. In this figure, we can observe that the accuracy of FNC decreases as k increases. In general, the accuracy of FNC increases as m increases. However, we also observe that the accuracy of FNC is instable as m changes. This indicates that quality of anchors greatly affects the performance of FNC. These results show the requirement of further improvement of FNC. Figure 3(b) shows that the time cost of similarity construction grows linearly as m increases, and do not change substantially as k increases. Figure 3(c) shows that the time cost of clustering is mainly affected by m. In real-world applications, we can carefully set a large m and a small k, or perform hierarchy grid search to choose the proper m and k for better results.

7 CONCLUSIONS

A Direct Normalized Cut (DNC) is proposed to directly optimize the normalized cut model, which has lower computational complexity than the original normalized cut with kmeans and spectral rotation. To cope with large-scale data, a Fast Normalized Cut (FNC) is extend from DNC for large-scale data. In the new method, a balanced k-means (BKM) is proposed to find high quality anchors which can best represent the whole data set, in which a balance regularizer is incorporated into k-means in order to produce balanced partition. BKM is used to seek high quality anchors for constructing of a representative similarity matrix, and then DNC is used to obtain the final clustering assignments from the representative similarity matrix. Experimental results on both synthetic data and benchmark data sets show the superior performance of BKM, DNC and FNC.

In future work, we will extend DNC and FNC to semi-supervised learning task. The investigation of the proposed methods in real applications is also our future work.

ACKNOWLEDGMENT

This research was supported by NSFC under Grant no. 61773268, no. 61772427, no. 61751202, and Tencent Rhinoceros Birds - Scientific Research Foundation for Young Teachers of Shenzhen University.

REFERENCES

[1] Deng Cai and Xinlei Chen. 2015. Large Scale Spectral Clustering Via Landmark-Based Sparse Representation. *IEEE Transactions on Cybernetics* 45, 8 (2015), 1669–1680.

[2] Xiao Cai, Feiping Nie, Heng Huang, and Farhad Kamangar. 2011. Heterogeneous image feature integration via multi-modal spectral clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 1977–1984.

[3] Xiaojun Chen, Joshua Zhexue Huang, Feiping Nie, Renjie Chen, and Qingyao Wu. 2017. A Self-Balanced Min-Cut Algorithm for Image Clustering. In *Proceedings of the International Conference on Computer Vision, ICCV-17*. 2080–2088.

[4] Xiaojun Chen, Feiping Nie, Joshua Zhexue Huang, and Min Yang. 2017. Scalable Normalized Cut with Improved Spectral Rotation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*. 1518–1524.

[5] Xiaojun Chen, Xiaofei Xu, Yunming Ye, and Joshua Zhexue Huang. 2013. TW-k-means: Automated Two-level Variable Weighting Clustering Algorithm for Multi-view Data. *IEEE Transactions on Knowledge and Data Engineering* 25, 4 (2013), 932–944. https://doi.org/10.1109/TKDE.2011.262

[6] Xiaojun Chen, Min Yang, Joshua Zhexue Huang, and Zhong Ming. 2018. TWCC: Automated Two-way Subspace Weighting Partitional Co-Clustering. *Pattern Recognition* 76 (2018), 404–415.

[7] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. 2010. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 26, 2 (2010), 214–225.

[8] Lars Hagen and Andrew B. Kahng. 1992. New spectral methods for ratio cut partitioning and clustering. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 11, 9 (1992), 1074–1085.

[9] Jin Huang, Feiping Nie, and Heng Hu. 2013. Spectral rotation versus K-Means in spectral clustering. In *AAAI Conference on Artificial Intelligence*. 431–437.

[10] Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. 2009. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data* 3, 1 (2009), 1–58. https://doi.org/10.1145/1497577.1497578

[11] Mu Li, James T. Kwok, and Bao Liang Lu. 2010. Making Large-Scale Nyström Approximation Possible. In *International Conference on Machine Learning*. 631–638.

[12] Wei Liu, Junfeng He, and Shih Fu Chang. 2010. Large Graph Construction for Scalable Semi-Supervised Learning. In *International Conference on Machine Learning*. 679–686.

[13] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. 2002. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems* 2 (2002), 849–856.

[14] Feiping Nie, Xiaoqian Wang, and Heng Huang. 2014. Clustering and projected clustering with adaptive neighbors. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 977–986.

[15] Feiping Nie, Xiaoqian Wang, Michael Jordan, and Heng Huang. 2016. The Constrained Laplacian Rank Algorithm for Graph-Based Clustering. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. 1969–1976.

[16] F. Nie, J. Yuan, and H. Huang. 2014. Optimal mean robust principal component analysis. In *International Conference on Machine Learning*. 1062–1070.

[17] Feiping Nie, Zinan Zeng, Ivor W Tsang, Dong Xu, and Changshui Zhang. 2011. Spectral embedded clustering: a framework for in-sample and out-of-sample spectral clustering. *IEEE Transactions on Neural Networks* 22, 11 (2011), 1796–808.

[18] Hiroyuki Shinnou and Minoru Sasaki. 2008. Spectral Clustering for a Large Data Set by Reducing the Similarity Matrix Size. In *International Conference on Language Resources and Evaluation, Lrec 2008, 26 May - 1 June 2008, Marrakech, Morocco*. 201–204.

[19] Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing* 17, 4 (2007), 395–416.

[20] Donghui Yan, Ling Huang, and Michael I Jordan. 2009. Fast approximate spectral clustering. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 907–916.

[21] Stella X. Yu and Jianbo Shi. 2003. Multiclass Spectral Clustering. In *Proceedings of IEEE International Conference on Computer Vision*. 313–319 vol.1.