SIAM J. IMAGING SCIENCES
Vol. 13, No. 3, pp. 1157–1178

Hypergraph Clustering Using a New Laplacian Tensor with Applications in Image Processing*

Jingya Chang†, Yannan Chen‡, Liqun Qi§, and Hong Yan¶

Abstract. In this paper, we consider the multiclass clustering problem involving a hypergraph model. Fundamentally, we study a new normalized Laplacian tensor of an even-uniform weighted hypergraph. The hypergraph's connectivity is related with the second smallest Z-eigenvalue of the proposed Laplacian tensor. Particularly, an analogue of fractional Cheeger inequality holds. Next, we generalize the Laplacian tensor based approach from biclustering to multiclass clustering. A tensor optimization model with an orthogonal constraint is established and analyzed. Finally, we apply our hypergraph clustering approach to image segmentation and motion segmentation problems. Experimental results demonstrate that our method is effective.

Key words. hypergraph clustering, Laplacian tensor, image processing, hypergraph partitioning, optimization, Stiefel manifold, tensor eigenvalue

AMS subject classifications. 05C50, 05C65, 65F15, 65K05, 90C35

DOI. 10.1137/19M1291601

1. Introduction.

Given a set of objects, partitioning is to sort the items into several groups for certain purposes. It is a useful tool for analyzing or solving problems in science, technology, humanity, medicine, and engineering. Partitioning the unstructured data by machine is an effective way to reduce noise, extract meaningful information, and select the key component. If some of the labels of the set of objects are known in advance, the task is to construct a classifier based on the given information and predict classification labels of other objects. This kind of partitioning is termed supervised classification [23]. The unsupervised classification

*Received by the editors October 7, 2019; accepted for publication (in revised form) April 27, 2020; published electronically July 13, 2020.

https://doi.org/10.1137/19M1291601

Funding: The work of the first author was partially supported by the National Natural Science Foundation of China grant 11901118 and Guangdong Basic and Applied Basic Research Foundation 2020B1515310001. The work of the second author was partially supported by the National Natural Science Foundation of China grant 11771405 and Guangdong Basic and Applied Basic Research Foundation 2020A1515010489. The work of the third author was partially supported by the Hong Kong Research Grants Council grants PolyU 15302114, 15300715, 15301716. The work of the third and fourth authors was partially supported by the Hong Kong Research Grants Council grant C1007-15G. The work of the fourth author was partially supported by the City University of Hong Kong project 9610308.

†School of Applied Mathematics, Guangdong University of Technology, Guangzhou, China (jychang@gdut.edu.cn).

‡Corresponding author. School of Mathematical Sciences, South China Normal University, Guangzhou, China (ynchen@scnu.edu.cn).

§Department of Mathematics, School of Science, Hangzhou Dianzi University, Hangzhou, 310018, China; Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong (maqilq@polyu.edu.hk).

¶Department of Electrical Engineering, City University of Hong Kong, Kowloon, Hong Kong (h.yan@cityu.edu.hk).

means none of the object labels are known in advance and the objects are classified according to their similarities. Unsupervised classification is also called clustering.

Clustering has wide applications in many fields [33], such as image segmentation [19, 23], bioinformatics [18, 37], object recognition [8], data mining [14], spatial data analysis [16, 35], data reduction [21, 17, 15], etc. For example, a large number of diseases have a unique gene expression profile which can be detected by microarray technology. In microarray analysis, clustering methods are used to improve visual display and interpret experiment performances [10]. Genes which display similar expression patterns are likely to be coagulated and may possess a common function. Therefore, clustering approaches are employed to recognize gene sets with similar expression patterns [39]. Clustering techniques can be categorized according to various criterions, like agglomerative vs. divisive, monothetic vs. polythetic, hard vs. fuzzy, deterministic vs. stochastic, incremental vs. nonincremental, etc. [19]. A rough but usual taxonomy is to classify clustering methods as hierarchical clustering and partitional clustering [13]. Hierarchical clustering algorithms merge small clusters into large ones or split large clusters into small ones, while partitional clustering approaches generate the desired partitions directly.

As an important class of partitional clustering approaches [13], spectral clustering methods cluster items based on spectral graph theory. They are popular in computer vision, VLSI design, and so on [22, 9, 3]. Compared with other partitional clustering algorithms, such as k-means and k-medoids, spectral clustering methods are robust and stable. Also, spectral clustering methods are easy to implement. In fact, for graph clustering problems, the objects are assumed to have pairwise correlations. However, in many application, such as computer vision, text mining, and network analysis [26, 11], the items often have multiway affinities. Therefore, it is natural to promote hypergraph-based spectral clustering approaches to solve such kinds of problems.

A widely used spectral clustering algorithm for hypergraph clustering is the normalized hypergraph cut (NH-Cut) method [40]. In this method, the cost function of the optimization model is a second order polynomial function, which can be linked to a Laplacian matrix of a graph. Benson, Gleich, and Leskovec [2] extended the graph spectral methods to hypergraph cases by using the second left eigenvector of a tensor's one-mode unfolding matrix. Since the aforementioned methods deal with matrices finally, they did not make the best of the higher order relationships contained in the hypergraphs. Chen, Qi, and Zhang [7] defined a normalized Laplacian tensor of an even-uniform hypergraph and generalized the Fiedler vector from matrices to tensors. By utilizing the Z-eigenvector associated with the second smallest Z-eigenvalue of the normalized Laplacian tensor, all vertices of the corresponding hypergraph are partitioned into two sets. Numerical experiments show that this approach is efficient for solving bipartitioning problems. How to extend the spectral hypergraph method for bipartitioning problems in [7] to multiclass hypergraph clustering problems is an important problem to investigate.

Our contributions. In this paper we propose a novel heuristic tensor optimization model to solve the hypergraph clustering problem. I. We study the case when the order of the tensor equals the order of the hypergraph. This tensor is defined as the Laplacian tensor of the hypergraph. We establish connections between the Laplacian tensor and the hypergraph connectivity and relate the optimization model for bipartitioning problem to the spectrum

of the Laplacian tensor. II. The optimization model covers a class of spectral clustering approaches. Especially when the order of the tensor degenerates to 2, it becomes the matrix spectral method [40]. The theoretical analysis applies to the general case when the order of the tensor is an ordinary even number. III. Numerical experiments show that our method works well for multiclustering problems in image processing.

The paper is organized as follows. In section 2, we introduce some preliminaries about tensors and hypergraphs. In section 3, we propose the hypergraph clustering model. We show that the Laplacian tensor is symmetric and positive semidefinite, and the smallest Z-eigenvalue of the proposed Laplacian tensor is zero. We prove that the second smallest Z-eigenvalue of the Laplacian tensor is positive if and only if the even-uniform hypergraph is connected. Also, we prove that a Cheeger inequality holds, which presents the relation of inequality between the second smallest Z-eigenvalue of the Laplacian tensor and the Cheeger ratio of a bipartition cut. In section 4, we apply the algorithm for optimization on a Stiefel manifold proposed by Jiang and Dai [20] to our problem and analyze the convergence property of the iterative sequences. In section 5, we show the numerical performance of our method for solving image segmentation and motion segmentation problems. Finally, we conclude this paper in section 6.

2. Preliminaries. Unless otherwise specified, we use a lowercase letter $x, y, \ldots$ for scalars, bold lowercase letter $\mathbf{x}, \mathbf{y}, \ldots$ for vectors, capital letter $X, Y, \ldots$ for matrices, and calligraphic font $\mathcal{A}, \mathcal{L}, \ldots$ for tensors. The symbol $\|A\|_F$ is the Frobenius norm of a matrix A. Denote $\langle A, B\rangle = \mathrm{tr}(A^\top B)$ as the inner product for two matrices $A, B \in \mathbb{R}^{n\times k}$.

A tensor $\mathcal{T}$ is a multiway array with entries $t_{i_1,\ldots,i_r} \in \mathbf{F}$ for $i_j = 1, \ldots, n_j$, and $j = 1, \ldots, r$. When $n_1 = n_2 = \cdots = n_r = n$, we say $\mathcal{T}$ is an rth order n-dimensional tensor. If $\mathbf{F} = \mathbb{R}$, the set of rth order n-dimensional tensor forms a space and is denoted as $\mathbb{R}_{r,n}$. If $\mathcal{T} \in \mathbb{R}_{r,n}$ and $t_{i_1,\ldots,i_r}$ is invariant under any permutations of its indices, then $\mathcal{T}$ is an rth order n-dimensional real symmetric tensor. Given a tensor $\mathcal{A} = (a_{i_1,\ldots,i_r}) \in \mathbb{R}_{r,n}$ and a vector $\mathbf{x} = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$, the products $\mathcal{A}\mathbf{x}^r$, $\mathcal{A}\mathbf{x}^{r-1}$, and $\mathcal{A}\mathbf{x}^{r-2}$ are defined, respectively, as follows:

$$\mathcal{A}\mathbf{x}^r = \sum_{i_1,\ldots,i_r=1}^{n} a_{i_1,\ldots,i_r} x_{i_1} \cancel{x_{i_r}} \in \mathbb{R},$$

$$\mathcal{A}\mathbf{x}^{r-1} = \left(\sum_{i_2,\ldots,i_r=1}^{n} a_{i_1 i_2,\ldots,i_r} x_{i_2} \cdots x_{i_r} \right)_{i_1=1,2,\ldots,n} \in \mathbb{R}^n,$$

$$\mathcal{A}\mathbf{x}^{r-2} = \left(\sum_{i_3,\ldots,i_r=1}^{n} a_{i_1 i_2 i_3,\ldots,i_r} x_{i_3} \cancel{x_{i_r}} \right)_{i_1,i_2=1,2,\ldots,n} \in \mathbb{R}^{n\times n}.$$

Let $X \in \mathbb{R}^{n\times k}$. We define a new tensor $\mathcal{B} = \mathcal{A}X^r \in \mathbb{R}_{r,k}$ such that

$$b_{j_1 j_2,\ldots,j_r} = \sum_{i_1,i_2,\ldots,i_r} a_{i_1 i_2,\ldots,i_r} X_{i_1 j_1} X_{i_2 j_2} \cdots X_{i_r j_r}.$$

In addition, $\mathrm{tr}(\mathcal{A}) = \sum_{i=1}^{n} a_{ii,\ldots,i}$ means the trace of a tensor $\mathcal{A} \in \mathbb{R}_{r,n}$.

The conceptions of tensor eigenvalues and eigenvectors are proposed independently by Qi [31] and Lim [28].

Definition 2.1 (see [31]). *Let $\mathcal{A} \in \mathbb{R}_{r,n}$. If there exist $\lambda \in \mathbb{R}$ and $\mathbf{x} \in \mathbb{R}^n$ such that*

$$\mathcal{A}\mathbf{x}^{r-1} = \lambda \mathbf{x}, \qquad \mathbf{x}^\top \mathbf{x} = 1,$$

then λ is called a Z-eigenvalue of $\mathcal{A}$ and $\mathbf{x}$ is a Z-eigenvector of $\mathcal{A}$ associated with λ.

Suppose that λ is a Z-eigenvalue of tensor $\mathcal{A}$. The geometric multiplicity of λ is defined as the maximum number of linearly independent Z-eigenvectors associated to λ [6]. If the geometric multiplicity of λ is 1, we say λ is geometrically simple.

A hypergraph is an extension of an ordinary graph. While all edges of a graph have two vertices, the edge of a hypergraph can join any number of vertices. Let $V = \{1, 2, \ldots, n\}$ be the vertex set, and $E = \{e_1, e_2, \ldots, e_m\}$ be the edge set for $e_p \subset V, p = 1, \ldots, m$. Then $G = (V, E)$ is a hypergraph. If there are positive numbers s_e associated with the edges $e \in E$ of the hyperpragh, then this hypergraph is a weighted hypergraph with s_e being the weight linked with the edge e. When the weight of each edge is 1, a weighted hypergraph is an ordinary hypergraph. If the length of each edge of the hypergraph is the same, i.e., $|e_p| = r$ for $p = 1, \ldots, m$, then the hypergraph G is called a uniform hypergraph or an r-graph for short. Here, r is called the order of the hypergraph. When $r = 2$, a hypergraph is in fact an ordinary graph. For any vertex $i \in V$, the degree of i is denoted as

$$d(i) = \text{sum}\{s_e : i \in e, e \in E\}.$$

Two vertices a and b are connected if there is a finite path $\{a, i_1, i_2, \ldots, i_l, b\}$ such that any adjacent vertices in this path are included in at least one edge. If any pair of vertices in a hypergraph is connected, then this hypergraph is connected.

3. The hypergraph clustering model.

The multiclass clustering problem for a given hypergraph G is to divide the vertex set of G into k classes (groups), such that vertices with strong correlations are grouped together and vertices with low connections are partitioned. The cluster number $k \geq 2$ is far less than the vertex number n. We call this problem k-clustering for short. It mainly involves two subproblems:

- how to build a reasonable mathematical model for the partitioning cost,
- how to find the optimal solution effectively.

These two subproblems are associated. It is meaningless if a model describes the partitioning cost well but is difficult to solve. On the other hand, if the model does not match the partitioning problem well, even the exact global solution for this optimization model cannot provide a good partitioning. We have to consider the problem from both aspects and to make a tradeoff between the accuracy of the model and the efficiency of the computation.

For the purpose of establishing a mathematical model for clustering vertices of a connected hypergraph, we try to cut some edges of the hypergraph. Once an edge of the hypergraph is cut and vertices therein are merged into different clusters, the cutting cost arises. Then we start by generating the cutting cost of edges in the edge set and summing them together. Intuitively, the cutting cost should satisfies four requirements: (1) If the vertices of an edge are divided into different clusters, the cost of this edge is positive. (2) If vertices of an edge are fully included in one cluster, the cutting cost is zero without considering its weight. (3) The larger the weight of an edge is, the higher the cutting cost is. (4) The more clusters

vertices of an edge belong to, the higher the cutting cost is. Based on these rough criteria, we demonstrate our preparatory cutting cost of an edge as follows.

3.1. A new Laplacian tensor for 2-clustering of a hypergraph. The Laplacian matrix plays an important role in spectral graph clustering. On the other hand, a tensor is a natural generalization of a matrix when a graph extends to a hypergraph and the spectral hypergraph theory is studied via tensors [32, Chapter 4]. Therefore, we employ the Laplacian tensor to solve the hypergraph clustering problem. Since a symmetric tensor corresponds to a multivariate homogeneous polynomial [32], we construct the Laplacian tensor first by introducing the following rth order polynomial.

At the beginning, we suppose that all vertices are grouped into $k = 2$ subsets A and B. Given the indicator $\mathbf{x} \in \mathbb{R}^n$ with entries

$$x_i = \begin{cases} \frac{1}{2} & \text{if } v_i \in A, \\ -\frac{1}{2} & \text{otherwise} \end{cases} \tag{3.1}$$

for $i = 1, \ldots, n$, we define the cutting cost function of 2-clustering for an r-uniform hypergraph with m edges as follows:

$$f(\mathbf{x}) = \sum_{e \in E} s_e \sum_{i,j \in e} |x_i - x_j|^r. \tag{3.2}$$

This cutting cost function meets well the requirements (1)–(3) listed above.

However, if we minimize this function directly, it may tend to cut edges with small weights and separate isolated vertices. Such cases are well discussed for graph clustering in [24, 34]. To avoid such bias, a normalized cutting cost is proposed for graph partitioning in [34]. By utilizing this normalized technique, we improve (3.2) and obtain the new cutting cost function as

$$f(\mathbf{x}) = \sum_{e \in E} s_e \sum_{i,j \in e} \left| \frac{x_i}{\sqrt[r]{d_i}} - \frac{x_j}{\sqrt[r]{d_j}} \right|^r, \tag{3.3}$$

where d_i and d_j are degrees of vertices i and j, respectively. A vertex with a large degree seems like a superstar, which has more social relation. So it is hard to separate the superstar vertex using simply cost function (3.2). By introducing the normalization, the superstar vertex is divided by its large degree and processes like ordinary vertices in some sense.

We give the expression of tensor $\mathcal{L}$ arising from (3.3) in the following proposition.

Proposition 3.1. *For an even uniform weighted hypergraph $G = (V, E, s)$, we define the normalized Laplacian tensor*

$$\mathcal{L} = \sum_{e \in E} s_e \sum_{i,j \in e} \underbrace{\boldsymbol{u}_{ij} \circ \boldsymbol{u}_{ij} \circ \cdots \circ \boldsymbol{u}_{ij}}_{r \quad times}, \tag{3.4}$$

where $\circ$ denotes the tensor outer product, $\boldsymbol{u}_{ij} = \frac{\boldsymbol{e}_i}{\sqrt[r]{d_i}} - \frac{\boldsymbol{e}_j}{\sqrt[r]{d_j}}$, and $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$ are column vectors with all elements being zero except the ith and jth entries are one, respectively. Then, the function in (3.3) can be rewritten as

$$f(\boldsymbol{x}) = \mathcal{L}\boldsymbol{x}^r. \tag{3.5}$$

Proof. From (3.4), for the vector $\mathbf{x} \in \mathbb{R}^n$ we have

$$\begin{aligned}
\mathcal{L}\mathbf{x}^r &= \sum_{e\in E} s_e \sum_{i,j\in e} \sum_{i_1,i_2,\ldots,i_r=1}^{n} [\mathbf{u}_{ij} \circ \mathbf{u}_{ij} \circ \cdots \circ \mathbf{u}_{ij}]_{i_1,i_2,\ldots,i_r} (\mathbf{x})_{i_1} (\mathbf{x})_{i_2} \cdots (\mathbf{x})_{i_r} \\
&= \sum_{e\in E} s_e \sum_{i,j\in e} \sum_{i_1,i_2,\ldots,i_r=1}^{n} (\mathbf{u}_{ij})_{i_1} (\mathbf{x})_{i_1} (\mathbf{u}_{ij})_{i_2} (\mathbf{x})_{i_2} \cdots (\mathbf{u}_{ij})_{i_r} (\mathbf{x})_{i_r} \\
&= \sum_{e\in E} s_e \sum_{i,j\in e} \left[\sum_{i_1=1}^{n} (\mathbf{u}_{ij})_{i_1} (\mathbf{x})_{i_1} \sum_{i_2=1}^{n} (\mathbf{u}_{ij})_{i_2} (\mathbf{x})_{i_2} \cdots \sum_{i_r=1}^{n} (\mathbf{u}_{ij})_{i_r} (\mathbf{x}_h)_{i_r} \right] \\
&= \sum_{e\in E} s_e \sum_{i,j\in e} (\mathbf{u}_{ij}^\top \mathbf{x})^r \qquad (3.6) \\
&= \sum_{e\in E} s_e \sum_{i,j\in e} \left(\frac{\mathbf{x}_i}{\sqrt[r]{d_i}} - \frac{\mathbf{x}_j}{\sqrt[r]{d_j}} \right)^r .
\end{aligned}$$

It is then obvious that the objective function in (3.5) is equivalent to (3.3). ∎

The next theorem shows the relationship between the smallest Z-eigenvalue of the normalized Laplacian tensor $\mathcal{L}$ and the connectivity of its corresponding hypergraph.

Theorem 3.2. *In terms of the normalized even order Laplacian tensor $\mathcal{L}$, we sort all its Z-eigenvalues as*

$$\lambda_0 \le \lambda_1 \le \cdots . \qquad (3.7)$$

Then the following conclusions hold.

(1) *$\mathcal{L}$ is symmetric and positive semidefinite.*
(2) *$\lambda_0 = 0$ with one of its corresponding Z-eigenvector being $\boldsymbol{t}_0 = \frac{\bar{\boldsymbol{t}}}{\|\bar{\boldsymbol{t}}\|}$, where $\bar{\boldsymbol{t}} = (\sqrt[r]{d_1}, \ldots, \sqrt[r]{d_n})^\top$.*
(3) *The number of connected components of G equals the geometric multiplicity of the Z-eigenvalue 0.*
(4) *The hypergraph G is connected if and only if λ_0 is geometrically simple.*

Proof. (1) Since r is even, it can be deduced from (3.3) that $\mathcal{L}\mathbf{x}^r \ge 0$ for any $\mathbf{x} \in \mathbb{R}^n$.

(2) When $\mathbf{x} = \mathbf{t}_0$, it is straightforward to calculate that $\mathcal{L}\mathbf{x}^{r-1} = \mathbf{0}$. Hence, the Laplacian tensor $\mathcal{L}$ has a Z-eigenvalue 0. On the other hand, because $\mathcal{L}$ is positive semidefinite, all Z-eigenvalues of $\mathcal{L}$ are nonnegative. Hence, $\lambda_0 = 0$ is the smallest Z-eigenvalues of $\mathcal{L}$ with an associated Z-eigenvector $\mathbf{t}_0$.

(3) The conclusion can be proved in a similar way to the proof of Theorem 5.1(3) in [27]. Suppose the connected components of G are $\{V^1, \ldots, V^l\}$. Denote l vectors $\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_l$ such that

$$(\bar{\mathbf{y}}_h)_i = \begin{cases} \sqrt[r]{d_i} & \text{if} \quad v_i \in V^h, \\ 0 & \text{otherwise} \end{cases} \qquad (3.8)$$

for $i = 1, \ldots, n, h = 1, \ldots, l$. If G has l connected components, from (3.4) we get the pth element of $\mathcal{L}\bar{\mathbf{y}}_h^{r-1}$

$$\left(\mathcal{L}\bar{\mathbf{y}}_h^{r-1}\right)_p = \sum_{e\in E} s_e \sum_{i,j\in e} \mathbf{u}_{ij}^p \left(\frac{(\bar{\mathbf{y}}_h)_i}{\sqrt[r]{d_i}} - \frac{(\bar{\mathbf{y}}_h)_j}{\sqrt[r]{d_j}}\right)^{r-1} = 0.$$

Denote

$$\mathbf{y}_h = \frac{\bar{\mathbf{y}}_h}{\|\bar{\mathbf{y}}_h\|}$$

for $h = 1, \ldots, l$. Hence, vectors $\mathbf{y}_1, \ldots, \mathbf{y}_l$ are Z-eigenvectors corresponding to the smallest Z-eigenvalue 0. Since $\mathbf{y}_1, \ldots, \mathbf{y}_l$ are orthogonal, the geometric multiplicity of 0 is at least the number of connected components l.

If there is another Z-eigenvector $\tilde{\mathbf{y}} = (\tilde{y}_i)$ associated with 0, we have

$$\mathcal{L}\tilde{\mathbf{y}}^r = \sum_{e\in E} s_e \sum_{i,j\in e} \left(\frac{\tilde{y}_i}{\sqrt[r]{d_i}} - \frac{\tilde{y}_j}{\sqrt[r]{d_j}}\right)^r = 0,$$

which means

$$\frac{\tilde{y}_i}{\sqrt[r]{d_i}} - \frac{\tilde{y}_j}{\sqrt[r]{d_j}} = 0 \tag{3.9}$$

for any two vertices v_i and v_j contained in one edge. Then for $i, j \in V^h$, $h = 1, \ldots, n$, (3.9) holds and

$$\tilde{y}_i = \alpha_h \sqrt[r]{d_i} \quad \text{for} \quad v_i \in V^h, \quad h = 1, \ldots, l,$$

where α_h is a constant. The eigenvector $\tilde{\mathbf{y}}$ can be expressed as $\tilde{\mathbf{y}} = \sum_{h=1}^{l} \frac{\alpha_h}{\|\bar{\mathbf{y}}_h\|}\mathbf{y}_h$. Therefore the geometric multiplicity of 0 is at most l. The conclusion holds.

(4) It can be deduced from (3) directly. ∎

Next, we focus on the second smallest Z-eigenvalue of the normalized Laplacian tensor $\mathcal{L}$ and get the following theorem.

Theorem 3.3. *Let λ_1 be the second smallest Z-eigenvalue of the normalized Laplacian tensor $\mathcal{L}$ of an even uniform connected hypergraph. Then, we have*

$$\begin{aligned} \lambda_1 = \quad & \min \quad \mathcal{L}\boldsymbol{x}^r \\ & \text{s.t.} \quad \boldsymbol{x}^\top \boldsymbol{x} = 1, \boldsymbol{x}^\top \boldsymbol{t}_0 = 0, \end{aligned} \tag{3.10}$$

where $\boldsymbol{t}_0$ is the Z-eigenvector corresponding to the smallest Z-eigenvalue $\lambda_0 = 0$ of $\mathcal{L}$.

Proof. In the first step, we prove that the global minimizer $\mathbf{x}_*$ of optimization (3.10) is a Z-eigenvector of $\mathcal{L}$. From the KKT condition of optimization, there are multipliers μ_1 and μ_2 such that

$$\begin{aligned} r\mathcal{L}\mathbf{x}_*^{r-1} - 2\mu_1\mathbf{x}_* - \mu_2\mathbf{t}_0 = 0, \\ \mathbf{x}_*^\top \mathbf{x}_* = 1, \qquad \mathbf{x}_*^\top \mathbf{t}_0 = 0. \end{aligned} \tag{3.11}$$

Taking an inner product between $\mathbf{t}_0$ and (3.11), we get $\mu_2\|\mathbf{t}_0\|^2 = r\mathbf{t}_0^\top(\mathcal{L}\mathbf{x}_*^{r-1})$. Owing to the definition of $\mathbf{u}_{ij}$ below (3.4), we have $\mathbf{u}_{ij}^\top\mathbf{t}_0 = 0$ for all $i, j \in e \in E$ and hence

$$\mathbf{t}_0^\top(\mathcal{L}\mathbf{x}_*^{r-1}) = \sum_{e\in E} s_e \sum_{i,j\in e} (\mathbf{u}_{ij}^\top\mathbf{x}_*)^{r-1}(\mathbf{u}_{ij}^\top\mathbf{t}_0) = 0. \tag{3.12}$$

Then, we get $\mu_2 = 0$. The equality (3.11) reduces to $\mathcal{L}\mathbf{x}_*^{r-1} = \frac{2\mu_1}{r}\mathbf{x}_*$, which means that $\mathbf{x}_*$ is a Z-eigenvector of $\mathcal{L}$ associated with the Z-eigenvalue $\mathcal{L}\mathbf{x}_*^r$.

Second, let $\lambda > 0$ be a Z-eigenvalue of $\mathcal{L}$ and $\mathbf{x}$ be the associated Z-eigenvector, i.e., we have $\mathcal{L}\mathbf{x}^{r-1} = \lambda\mathbf{x}$ and $\mathbf{x}^\top\mathbf{x} = 1$. By a similar discussion of (3.12), we obtain

$$\mathbf{x}^\top\mathbf{t}_0 = \lambda^{-1}\mathbf{t}_0^\top(\lambda\mathbf{x}) = \lambda^{-1}\mathbf{t}_0^\top(\mathcal{L}\mathbf{x}^{r-1}) = 0.$$

Thus, $\mathbf{x}$ is a feasible solution of the optimization (3.10), which means $\lambda = \mathcal{L}\mathbf{x}^r \geq \mathcal{L}\mathbf{x}_*^r$. That is, $\lambda_1 = \mathcal{L}\mathbf{x}_*^r$ and this theorem holds. ∎

Suppose that (A, B) is a bipartition of the vertex set V of the even-uniform connected hypergraph. Let $\mathrm{vol}A = \sum_{i\in A} d_i$ be the volume of A and let $\partial A = \{e \in E : e\cap A \neq \emptyset, e\cap B \neq \emptyset\}$ be the boundary of A. The cut between A and B is $\mathrm{cut}(A) = \sum_{e\in\partial A} s_e$. Clearly, $\partial B = \partial A$ and $\mathrm{cut}(A) = \mathrm{cut}(B)$. The Cheeger ratio of A is defined as

$$h(A) = \frac{\mathrm{cut}(A)}{\min(\mathrm{vol}A, \mathrm{vol}B)}.$$

Hence, $h(A) = h(B)$. The Cheeger constant of a hypergraph is the smallest Cheeger ratio:

$$h_G = \min_{A\subseteq V} h(A).$$

The Cheeger constant reveals the connectivity of a hypergraph and is an important structure parameter of the hypergraph. In the upcoming theorem, we establish a relationship between the Cheeger constant and the second smallest Z-eigenvalue of the proposed normalized Laplacian tensor for a hypergraph.

Theorem 3.4. *Let λ_1 be the second smallest Z-eigenvalue of the normalized Laplacian tensor $\mathcal{L}$ of an even uniform connected hypergraph. Then, we have*

$$\lambda_1 \leq \kappa 2^{r/2} h_G, \tag{3.13}$$

where $\kappa = r^2/4$ is a constant.

Proof. Given any nonempty and proper subset A of V, we define

$$\alpha := \left(\sum_{i\in A}\left(\sqrt[r]{d_i}\right)^2\right)^{1/2}, \quad \beta := \left(\sum_{i\in B}\left(\sqrt[r]{d_i}\right)^2\right)^{1/2}, \quad \text{and } \gamma := \sqrt{\alpha^2+\beta^2}.$$

Then, we introduce a vector $\mathbf{x} \in \mathbb{R}^n$ with entries

$$x_i := \begin{cases} \dfrac{\beta}{\alpha\gamma}\sqrt[r]{d_i} & \text{if } i \in A, \\ \dfrac{-\alpha}{\beta\gamma}\sqrt[r]{d_i} & \text{if } i \in B. \end{cases}$$

From this definition, it is straightforward to check $\mathbf{x}^\top\mathbf{x} = 1$ and $\mathbf{x}^\top\mathbf{t}_0 = 0$. Recalling (3.10), we know

$$\begin{aligned}\lambda_1 \le \mathcal{L}\mathbf{x}^r &= \sum_{e\in E} s_e \sum_{i,j\in e}\left(\frac{x_i}{\sqrt[r]{d_i}} - \frac{x_j}{\sqrt[r]{d_j}}\right)^r \\ &= \sum_{e\in \partial A} s_e \sum_{i,j\in e}\left(\frac{x_i}{\sqrt[r]{d_i}} - \frac{x_j}{\sqrt[r]{d_j}}\right)^r.\end{aligned}$$

Denote the cardinality $|e\cap A| = q_e \in \{1,\dots,r-1\}$. Then $|e\cap B| = r - q_e$. By some calculations, it yields that

$$\begin{aligned}\lambda_1(G) &\le \sum_{e\in\partial A} s_e \left(\frac{\alpha^2+\beta^2}{\alpha\beta\gamma}\right)^r q_e(r-q_e) \\ &\le \frac{r^2}{4}\left(\frac{\alpha^2+\beta^2}{\alpha^2\beta^2}\right)^{r/2} \sum_{e_p\in\partial A} s_e \\ &= \frac{r^2}{4}\left(\frac{1}{\alpha^2}+\frac{1}{\beta^2}\right)^{r/2} \mathrm{cut}(A).\end{aligned}$$

For p-norms, we know $\|x\|_p \le \|x\|_q$ if $p > q > 1$. Consider a vector $\mathbf{z}\in\mathbb{R}^n$ with entries $z_i = \sqrt[r]{d_i}$ if $i\in A$ and $z_i = 0$ if $i\in B$. Then, by $r\ge 2$, we obtain

$$\alpha = \|\mathbf{z}\| \ge \|\mathbf{z}\|_r = \left(\sum_{i\in A} d_i\right)^{1/r} = (\mathrm{vol}A)^{1/r}.$$

In a similar way, we get $\beta \ge (\mathrm{vol}B)^{1/r}$. Without loss of generality, we suppose $\mathrm{vol}A \le \mathrm{vol}B$. Then, we get

$$\begin{aligned}\lambda_1(G) &\le \frac{r^2}{4}\left(\frac{1}{(\mathrm{vol}A)^{2/r}} + \frac{1}{(\mathrm{vol}B)^{2/r}}\right)^{r/2} \mathrm{cut}(A) \\ &\le \frac{r^2}{4}\frac{2^{r/2}}{\mathrm{vol}A}\mathrm{cut}(A) \\ &= \frac{r^2}{4}2^{r/2}h(A).\end{aligned}$$

Owing to the arbitrariness of $A\subseteq V$, we complete the proof of this theorem. ∎

In [7], Chen, Qi, and Zhang considered the bipartitioning problem of an even order uniform hypergraph. They proposed a homogeneous polynomial

$$\sigma\sum_{e\in E} s_e \sum_{i\in e}\left(\frac{x_i}{\sqrt[r]{d_i}} - \frac{1}{r}\sum_{j\in e}\frac{x_j}{\sqrt[r]{d_j}}\right)^r$$

to determine the cost of a bipartition of the hypergraph, where σ is a positive constant. Indeed, this polynomial measures the cost of the star expansion of the hypergraph, where the

star expansion means each hyperedge of the hypergraph is approximated by a star. From this viewpoint, the cost (3.3) corresponds to the clique expansion of the hypergraph, i.e., each hypergraph is approximated by a clique. Based on the star expansion, they defined a Laplacian tensor, which is different from our Laplacian tensor associated with the clique expansion in Proposition 3.1. The Laplacian tensor in [7] also enjoys the properties in Theorem 3.2. While a Cheeger inequality in [7] was given, the coefficient of the Cheeger constant in the Cheeger inequality in [7] is different from the coefficient in our Cheeger inequality in Theorem 3.4. In the context of hypergraph clustering, clique expansion and star expansion of a hypergraph generate two different approaches.

3.2. An optimization model for k-clustering of a hypergraph. Now, we consider the hypergraph clustering problem which will generate $k > 2$ clusters. At the beginning, we revisit the case that an even-uniform hypergraph G has multiple connected components. From the proof in Theorem 3.2 we can see that if a hypergraph has l connected components, the orthogonal Z-eigenspace corresponding to the smallest Z-eigenvalue 0 has dimension l, and the l orthogonal Z-eigenvectors are

$$\left\{ \frac{\mathbf{y}_1}{\|\mathbf{y}_1\|_2}, \frac{\mathbf{y}_2}{\|\mathbf{y}_2\|_2}, \ldots, \frac{\mathbf{y}_l}{\|\mathbf{y}_l\|_2} \right\}.$$

Hereafter, we suppose that the hypergraph G is connected. By Theorem 3.3 for the 2-clustering problem, we see that the Z-eigenvector corresponding to the second smallest Z-eigenvalue λ_1 is orthogonal to $\mathbf{t}_0$ which is the Z-eigenvector corresponding to the smallest Z-eigenvalue λ_0. In these two cases, orthogonality plays crucial roles for hypergraph clustering. Now, we are going on to explore the orthogonality further.

Suppose that vertices of the hypergraph G are grouped into $k \geq 2$ subsets $\{C_1, \ldots, C_k\}$. We define the assignment matrix $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k] \in \mathbb{R}^{n \times k}$ as

$$X_{ih} = \begin{cases} 1 & \text{if } v_i \in C_h, \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

for $i = 1, \ldots, n$ and $h = 1, \ldots, k$. Each column vector $\mathbf{x}_h$ is in fact the indicator vector of the hth group C_h. The cost function of a k-clustering for an r-uniform hypergraph with m edges is given as

$$f(X) = \sum_{h=1}^{k} \sum_{e \in E} s_e \sum_{i,j \in e} \left(\frac{X_{ih}}{\sqrt[r]{d_i}} - \frac{X_{jh}}{\sqrt[r]{d_j}} \right)^r. \tag{3.15}$$

This cost function meets well requirements (1)–(4) listed at the beginning of this section.

Up to now we have gotten a relatively reasonable clustering cost function (3.15) with constraint (3.14). We note that this problem is NP-complete [25]. Nevertheless, just as we have mentioned, the model is required to be not only reasonable but also convenient for calculation. In terms of computation, the optimization problem (3.15) can be solved by a 0–1 integer programming. Hence we have to sacrifice the veracity of the model slightly to enhance its convenience of computation. By relaxing the 0–1 constraint and limiting the order r being

even, we finally design the optimization model for solving the k-clustering problem of an even order uniform hypergraph as

$$\begin{aligned} \min & f(X) = \sum_{h=1}^{k} \sum_{e\in E} s_e \sum_{i,j\in e} \left(\frac{X_{ih}}{\sqrt[r]{d_i}} - \frac{X_{jh}}{\sqrt[r]{d_j}} \right)^r \\ \text{s.t. } & X^\top X = I. \end{aligned} \tag{3.16}$$

We use the k-means method to divide row vectors of optimal solution X in (3.16) into k classes and obtain explicit k clusters of the vertex set of the hypergraph. Hence the clustering results are acquired.

In a word, for an even uniform hypergraph $G = (V, E, s)$, the model (3.16) could be rewritten as

$$\begin{aligned} \min \quad & f(X) = \operatorname{tr}(\mathcal{L}X^r) = \sum_{h=1}^{k} \mathcal{L}\mathbf{x}_h^r \\ \text{s.t. } & X^\top X = I, \end{aligned} \tag{3.17}$$

where $\mathcal{L}$ is the normalized Laplacian tensor introduced in Proposition 3.1. The necessity of the normalization is illustrated in section 5.1.

Remark. Zhou, Huang, and Schölkopf [40] set the parameter $r = 2$ in (3.16). Then, the optimal value of (3.16) with $r = 2$ is the sum of k smallest eigenvalues of a Laplacian matrix and the optimal solution is associated k eigenvectors. Here, we establish (3.16) from tensor-based spectral hypergraph theory, where the parameter r coincides with the order of the hypergraph. In fact, theoretical analysis in this section applies to the general case when the order of the tensor is an ordinary even number.

4. Computation of the optimization model. The feasible set in (3.17) is called the Stiefel manifold. We denoted it as

$$\mathbb{S}_{n,k} := \{X \in \mathbb{R}^{n\times k} : X^\top X = I_k\}.$$

The gradient of the objective function f with respect to X is

$$G|_X := \mathcal{D}f(X) = \left(\frac{\partial f(X)}{\partial X_{ij}} \right) = r\left(\mathcal{L}\mathbf{x}_1^{r-1}, \mathcal{L}\mathbf{x}_2^{r-1}, \ldots, \mathcal{L}\mathbf{x}_k^{r-1}\right)^\top \in \mathbb{R}^{n\times k}, \tag{4.1}$$

where the tth entry of $\mathcal{L}\mathbf{x}_h^{r-1}$ is

$$\begin{aligned} \left(\mathcal{L}\mathbf{x}_h^{r-1}\right)_t &= \left[\Big(\sum_{e\in E} s_e \sum_{i,j\in e} \mathbf{u}_{ij} \circ \mathbf{u}_{ij} \circ \cdots \circ \mathbf{u}_{ij} \Big) \mathbf{x}_h^{r-1} \right]_t \\ &= \sum_{e\in E} s_e \sum_{i,j\in e} \sum_{i_2,\ldots,i_r=1}^{n} (\mathbf{u}_{ij} \circ \mathbf{u}_{ij} \circ \cdots \circ \mathbf{u}_{ij})_{t,i_2,\ldots,i_r} (\mathbf{x}_h)_{i_2} \cdots (\mathbf{x}_h)_{i_r} \\ &= \sum_{e\in E} s_e \sum_{i,j\in e} \left[(\mathbf{u}_{ij})_t \sum_{i_2=1}^{n} (\mathbf{u}_{ij})_{i_2} (\mathbf{x}_h)_{i_2} \sum_{i_3=1}^{n} (\mathbf{u}_{ij})_{i_3} (\mathbf{x}_h)_{i_3} \cdots \sum_{i_r=1}^{n} (\mathbf{u}_{ij})_{i_r} (\mathbf{x}_h)_{i_r} \right] \\ &= \sum_{e\in E} s_e \sum_{i,j\in e} (\mathbf{u}_{ij})_t (\mathbf{u}_{ij}^\top \mathbf{x}_h)^{r-1}. \end{aligned} \tag{4.2}$$

We use G as an abbreviation for $G|_X$ if there is no confusion.

The first order necessary condition for the Stiefel manifold constrained optimization problem is given in [38, Lemma 1] and [20, Lemma 2.1]. It is also applicable to our model.

Lemma 4.1 (see [38], [20]). *Define the gradient of $f(X)$ in the tangent space of the Stiefel manifold $\mathbb{S}_{n,k}$ at X as*

$$\nabla f = G - XG^\top X, \tag{4.3}$$

where G is the gradient $G|_X$ in (4.1). If X is a local minimizer of problem (3.17), it satisfies that

$$\nabla f = 0.$$

4.1. The optimization algorithm on Stiefel manifold. Jiang and Dai [20] proposed a framework to preserve the orthogonal constraint for the optimization algorithm on the Stiefel manifold. If X is the current feasible point, we employ the update scheme in [20] to generate a new iterative point $X_{new} = Y(\tau; X)$ for the hypergraph clustering model (3.17) in the following way:

$$\begin{cases} E = (I_n - (1-2\rho)XX^\top)\nabla f, \\ W = -(I_n - XX^\top)E, \\ J(\tau) = I_k + \frac{\tau^2}{4}W^\top W + \frac{\tau}{2}X^\top E, \\ Y(\tau; X) = (2X + \tau W)J(\tau)^{-1} - X, \end{cases} \tag{4.4}$$

where ρ is a positive parameter and τ is the step size in the curvilinear search. We note that $\nabla f = 0$ if and only if $E = 0$, where E is defined in the first line of (4.4).

For the sake of convenience, we use $Y(\tau)$ and $Y_i(\tau)$ to stand for $Y(\tau; X)$ and $Y(\tau; X_i)$, respectively. For any given feasible point X, it is proved in [20, Lemma 3.1] that

$$Y(\tau)^\top Y(\tau) = I_k, \tag{4.5}$$

$$Y^{'}(0) = -E, \tag{4.6}$$

$$f_\tau^{'}(Y(0)) = \frac{\partial f(Y(\tau))}{\partial \tau}\,|_{\tau=0} = -\langle G, E\rangle \le -\min\{\rho, 1\}\|\nabla f\|_F^2. \tag{4.7}$$

The function $f(Y(\tau))$ is continuously differentiable and bounded due to the fact that the Stiefel manifold is compact. The symbol $f_\tau'(Y_i(0)$ means the derivative of $f(Y(\tau))$ with respect to τ at zero. Thus, from (4.7), we can see that there exists a positive τ_i such that the following Armijo condition holds:

$$f(Y_i(\tau_i)) \le f(Y_i(0)) + \delta\tau_i f_\tau'(Y_i(0)), \tag{4.8}$$

where δ is a positive parameter.

We employ the update scheme (4.4) to keep the iterative points on the Stiefel manifold. The step size τ in (4.4) is chosen such that the Armijo condition (4.8) holds. The method for solving the hypergraph clustering model (3.17) (MSHC) is then obtained.

Algorithm MSHC Method for solving the hypergraph clustering model.

Given $X_0 \in \mathbb{S}_{n,k}$, $0 < \rho$, $0 < \delta, \beta, \epsilon < 1$, and $\varepsilon_{\max} > 0$.
while the sequence of iterates does not converge **do**
  Calculate $f(Y_i(0))$ and $f'_\tau(Y(0))$ according to (3.17) and (4.7), respectively;
  Choose the smallest nonnegative integer l such that $\tau = \beta^l \tau_i^0$ with $\tau_i^0 = \min\{\frac{1}{2}, \frac{\varepsilon_{\max}}{\|E_i\|_F}\}$ satisfy (4.8);
  Let $\tau_i = \tau$ and update $X_{i+1} = Y(\tau_i; X_i)$ by (4.4);
  Compute $\nabla f(X_{i+1})$ by (4.3);
  $i \leftarrow i + 1$.
end while

4.2. Convergent results. It can be seen from (3.17), (4.1), and (4.3) that $f(X)$ is twice continuously differentiable.

Lemma 4.2. *Since the set $\mathbb{S}_{n,k}$ is compact, we can find a constant $M > 1$ such that*

$$|f(X)| < M, \quad \|\mathcal{D}f(X)\|_F < M, \quad \|\nabla f(X)\|_F < M, \tag{4.9}$$

and the maximum absolute value of the elements of the second derivative of $f(X)$ is bounded up by M. Also $\mathcal{D}f(X)$ is Lipschitz continuous, which means

$$\|\mathcal{D}f(X) - \mathcal{D}f(\tilde{X})\|_F \le L\|X - \tilde{X}\|_F$$

with L being a positive constant for any $X, \tilde{X} \in \mathbb{S}_{n,k}$.

The next lemma shows that the step size produced in Algorithm MSHC is bounded below. It can be proved in a similar way with the proof of Lemma 4.5 in the prepublication version of [20].[1]

Lemma 4.3. *If τ_i satisfies (4.8), then when $\|\nabla f(X_i)\|_F \neq 0$ we have*

$$\tau_i \ge c,$$

where $c = \min(\frac{2\beta(1-\delta)\min\{\rho,1\}}{c_1 \max\{2\rho,1\}^2}, \min\{\frac{1}{2}, \frac{\varepsilon_{\max}}{\max\{2\rho,1\}M}\})$ and $c_1 = \frac{M(4+\varepsilon_{\max})}{2} + \frac{L(2+\varepsilon_{\max})}{2}$.

Based on the above analysis, the convergent result of MSHC is obtained. The proof is analogous to that of Theorems 4.2 and 4.4 in [5].

Theorem 4.4. *Suppose MSHC generates an infinite sequence of iterates $\{X_i\}$ and function values $\{f(X_i)\}$. Then*

(1) *there exists a constant y^* such that $\lim_{i\to\infty} f(X_i) = y^*$,*

(2) $\lim_{i\to\infty} \|\nabla f(X_i)\|_F = 0$.

Proof. (1) Because $f(X_i)$ satisfies the Armijo condition (4.8), the sequence $\{f(X_i)\}$ is monotone decreasing. Thus it can be deduced from the fact $|f(X)| \le M$ that $\{f(X_i)\}$ converges.

[1] https://arxiv.org/abs/arXiv:1301.0172v1.

(2) From (4.8) and (4.7) we have

$$f(X_i) - f(X_{i+1}) \geq -\delta\tau_i f'_\tau(X_i) \geq \min\{\rho, 1\}\delta\tau_i \|\nabla f\|_F^2.$$

Thus,

$$\begin{aligned} 2M &\geq f(X_1) - y^* \\ &\geq \sum_{i=1}^{\infty} [f(X_i) - f(X_{i+1})] \\ &\geq \sum_{i=1}^{\infty} \min\{\rho, 1\}\delta\tau_i \|\nabla f(X_i)\|_F^2, \end{aligned}$$

which means

$$\begin{aligned} \sum_{i=1}^{\infty} \|\nabla f(X_i)\|_F^2 &\leq \frac{2M}{\min\{\rho, 1\}\delta \min\{\tau_i\}} \\ &\leq \frac{2M}{\min\{\rho, 1\}\delta c} \quad [\text{from Lemma 4.3}] \\ &< +\infty. \end{aligned}$$

The conclusion holds. ∎

5. Numerical experiments. In this section, we demonstrate the numerical performance of the hypergraph clustering methods (3.17) for real-world problems. Algorithm MSHC is implemented in MATLAB R2016a on a Thinkpad laptop with an Intel dual core CPU at 2.40GHZ×2 and 8GB of RAM. Let $\text{tol}_i^X := \frac{\|X_i\|_F - \|X_{i-1}\|_F}{\sqrt{n}}$, $\text{tol}_i^f := \frac{|f(X_i) - f(X_{i-1})|}{|f(X_i)|+1}$, and MaxIter be the maximal iteration number. MSHC terminates when one of the following conditions holds:

$$\text{① } \text{tol}_i^X \leq \epsilon_X, \quad \text{② } \text{tol}_i^f \leq \epsilon_f, \quad \text{③ } \|\mathcal{D}f(X_i)\|_F \leq \epsilon \|\mathcal{D}f(X_0)\|_F.$$

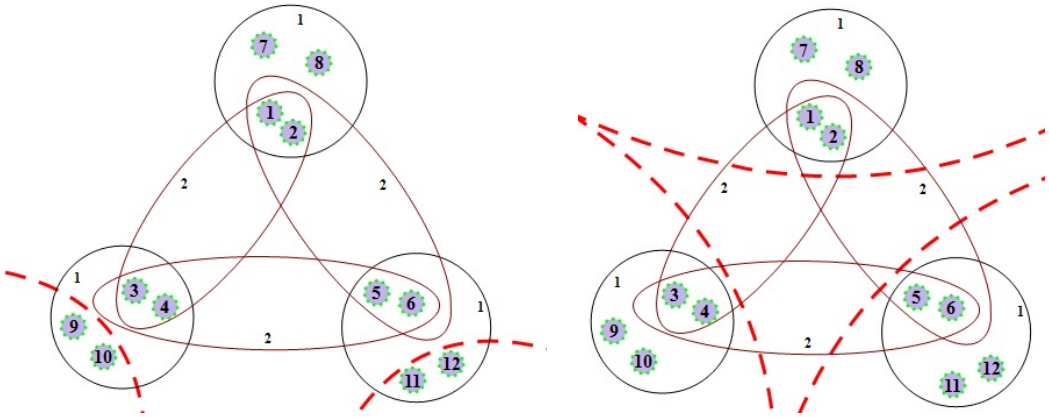
The parameters in Algorithm MSHC are set as

$$\epsilon = \epsilon_X = \epsilon_f = 10^{-8}, \rho = 0.25, \beta = 0.5, \delta = 10^{-3}, \varepsilon_{\max} = 10^{10}, \text{MaxIter} = 500.$$

5.1. A simple example for illustering the necessity of normalization. We consider a 4-graph $G = (V, E, s)$ with twelve vertices $V = \{1, 2, \ldots, 12\}$, six edges $E = \{\{1, 2, 3, 4\}, \{1, 2, 5, 6\}, \{3, 4, 5, 6\}, \{1, 2, 7, 8\}, \{3, 4, 9, 10\}, \{5, 6, 11, 12\}\}$, and associated weights of edges $s = (2, 2, 2, 1, 1, 1)$. See Figure 1 for the hypergraph. Here, vertices $1, 2, \ldots, 6$ are critical vertices and vertices $7, 8, \ldots, 12$ are regarded as outliers. Moreover, weight of edges containing only critical vertices are large; otherwise the weight is small. The question is how to divide vertices into $k = 3$ classes.

For the purpose of comparison, we first solve an unnormalized optimization model:

$$\begin{aligned} &\min f(X) = \sum_{h=1}^{k} \sum_{e \in E} s_e \sum_{i,j \in e} (X_{ih} - X_{jh})^r, \\ &\text{s.t. } X^\top X = I. \end{aligned}$$



(a) Unnormalized approach (b) Normalized approach

Figure 1. *A simple example for illustrating the necessity of normalization.*

Recalling (3.17), we discard information of degrees in the cost function. Using the MSHC algorithm and the k-means method, we obtain three clusters of vertices:

$$\{1,2,3,4,5,6,7,8\}, \quad \{9,10\}, \quad \{11,12\}.$$

This is illustrated in Figure 1(a). Although two edges with small weight are cut, there is indeed a supercluster, which contains eight vertices and is greater than the remaining parts. Roughly speaking, the unnormalized approach does nothing besides finding outliers.

Now, let us see the normalized one, i.e., the optimization model (3.17). By applying the MSHC algorithm and the k-means method, we obtain three clusters:

$$\{1,2,7,8\}, \quad \{3,4,9,10\}, \quad \{5,6,11,12\}.$$

See Figure 1(b). These classes have the same size. The main vertices are divided into three clusters. It is clear that the second clustering result is much more balanced and reasonable than the first one. Hence, we recommend using the normalized Laplacian tensor for hypergraph clustering.

5.2. Image segmentation. In this subsection, we apply the MSHC method to image segmentation problems. All images except the one in Figure 2, in this subsection come from the web Unsplash.com.[2] The original picture in Figure 2 is taken by the author. For a given original image, we use the SLIC technique [1] to generate its superpixels. These superpixels constitute the vertex set of a hypergraph. We now modify the three step approach in [7] and construct edges of the hypergraph from superpixels as follows. First, we generate a complete graph, which means each pair of vertices is connected in an edge. The graph is then extended to a kth even order hypergraph by stuffing each edge of the graph with $k-2$ vertices. We repeat this step for $\binom{n-2}{k-2}$ times to get a complete intermediate hypergraph. Finally, for each edge in the original graph, we choose one of its augmented edges in the intermediate

[2]Unsplash.com license gives viewers the right to copy, modify, distribute and use the photos for free.

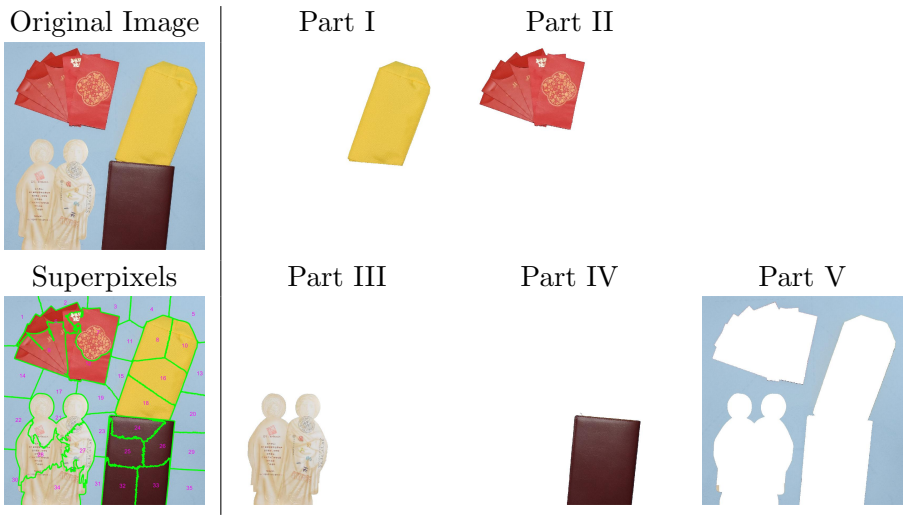


Figure 2. *Image segmentation by 5-clustering. LapMatrix and MSHC produce the same segmentation results.*

hypergraph and add a certain percent of the remaining edges whose weights are larger than others in the intermediate hypergraph. The percentage of the remaining edges we choose in the following examples is 0.1 except the one in the dog background separation experiment, which is 0.15. All edges in the hypergraph are selected without repetition. The weight of each edge is proportional to the similarity of color distributions of superpixels and is inversely correlated with the star distance among superpixels in an edge. We adopt the formulations in [7] to compute the similarity of color distributions of superpixels and the star distance among superpixels. We use the same hypergraph generated from the original image for different methods to solve the same problem.

We compare the hypergraph clustering method in [7] and [40] with our MSHC algorithm for solving foreground and background image segmentation problems. The methods in [7] and [40] are denoted as Fielder and LapMatrix in the following discussions. The bipartitioning results given by the three approaches look the same. In Figure 3, the original images are shown in the first column, and the superpixel clustering results given by SLIC are listed in the second column. The foreground and background images are illustrated in the last two columns. It can be seen that all of the three methods segment the images well. However, the LapMatrix method in [40] runs faster than the other two. The reason is that we only need to solve a matrix eigenvalue problem when using LapMatrix, while the MSHC and the Fielder approaches both solve the tensor eigenvalue problems.

A heuristic method for k-way clustering is to use the eigenvectors of the first k smallest eigenvalues of the Laplacian matrix proposed in [40]. Next, we compare the MSHC algorithm with the method for solving multi-image segmentation problems. For 4- and 5-clustering problems, the results of these two methods look the same, and we demonstrate the segmented images of 4- and 5-clustering in Figures 4 and 2, respectively. Both LapMatrix and MSHC work well for these examples. The results of 3-clustering are compared in Figure 5. We can see that the MSHC algorithm performs better than the LapMatrix method. Our method segments the original images well.

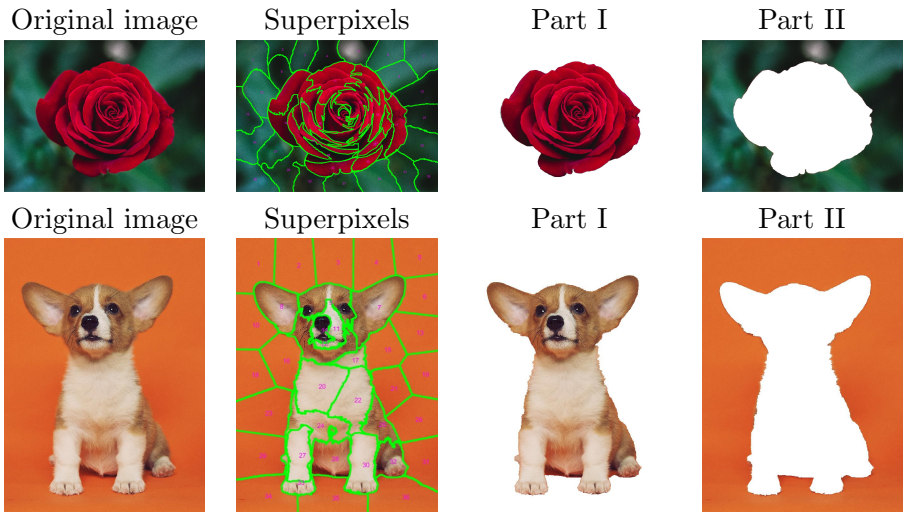


Figure 3. *Image segmentation by 2-clustering. Fiedler, LapMatrix, and MSHC obtain the same segmentation results.*

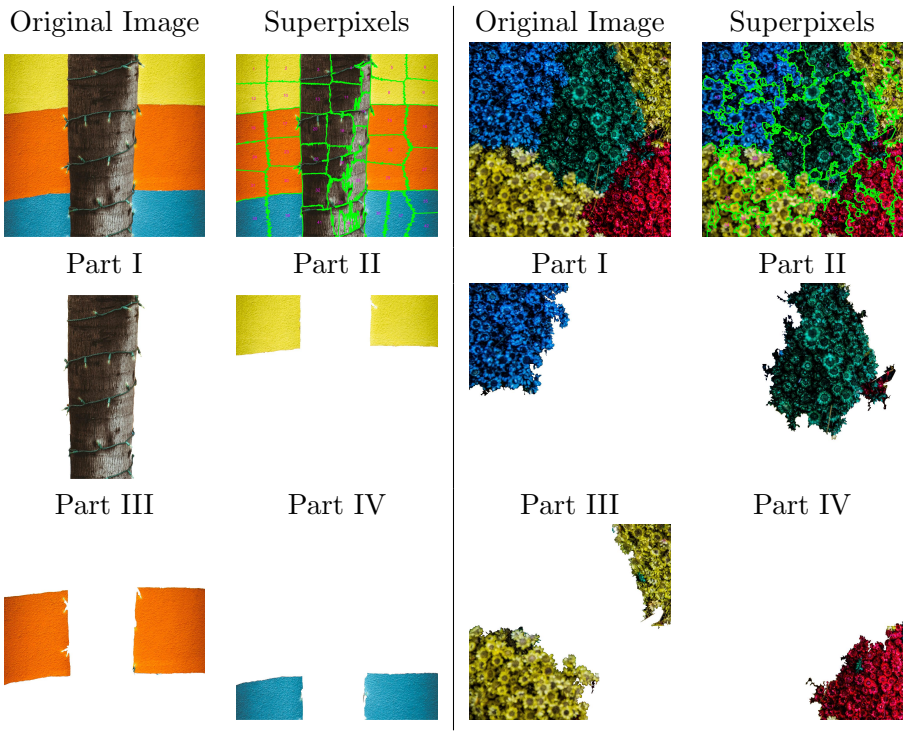


Figure 4. *Image segmentation by 4-clustering. LapMatrix and MSHC produce the same segmentation results.*

5.3. Motion segmentation. The motion segmentation problem aims to segment regions or features in a video sequence to several subsets according to different motions. It is an important topic in computer vision, which can be applied in object recognition, video surveillance,

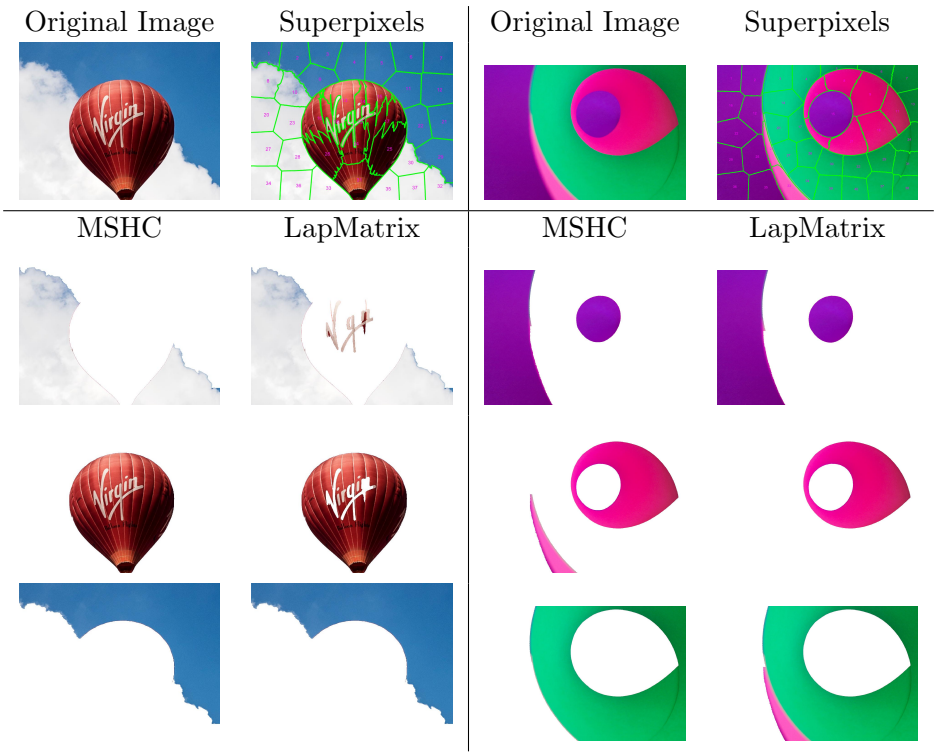


Figure 5. *Image segmentation by 3-clustering. MSHC performs better than LapMatrix.*

three-dimensional mapping, and so on. For example, with the aid of motion segmentation approach, the motoring system is able to recognize illegal intrusions and send alerts automatically. A benchmark called Hopkins155 for motion segmentation comparison was proposed in [36]. There are 50 video sequences of indoor and outdoor scenes, with 15 video sequences containing two motions and 35 video sequences containing three motions. The feature points of each frame are extracted and tracked in the data set. By considering the feature points of two groups in the three motion sequence, three sequences with two motions are generated from each video sequence with three motions. Thus, there are 120 motion sequences with two motions and 35 motion sequences with three motions. In light of the content of the video and the type of motion, there are three categories which are checkerboard sequences, traffic sequences, and articulated/non-rigid sequences. Sample frames from three corresponding motion sequences with extracted feature points are shown in Figure 6.

We compare the performance of our MSHC algorithm with K-means [29], K-flats [4], and TTM [12] methods for the Hopkins155 database. For the two hypergraph clustering algorithms, MSHC and TTM, the hypergraph is constructed by regarding the image point trajectories as vertices and employing the nearest subspace neighbor (NSN) approach in [30] to determine the similarity among every two vertices. NSN finds neighbor points most likely to be on the same subspace in a greedy fashion. Each edge of the hypergraph contains 10 vertices which are given by an key image point trajectory as well as its 9 nearest points. The key image point trajectories are chosen from the trajectories one by one without repetition

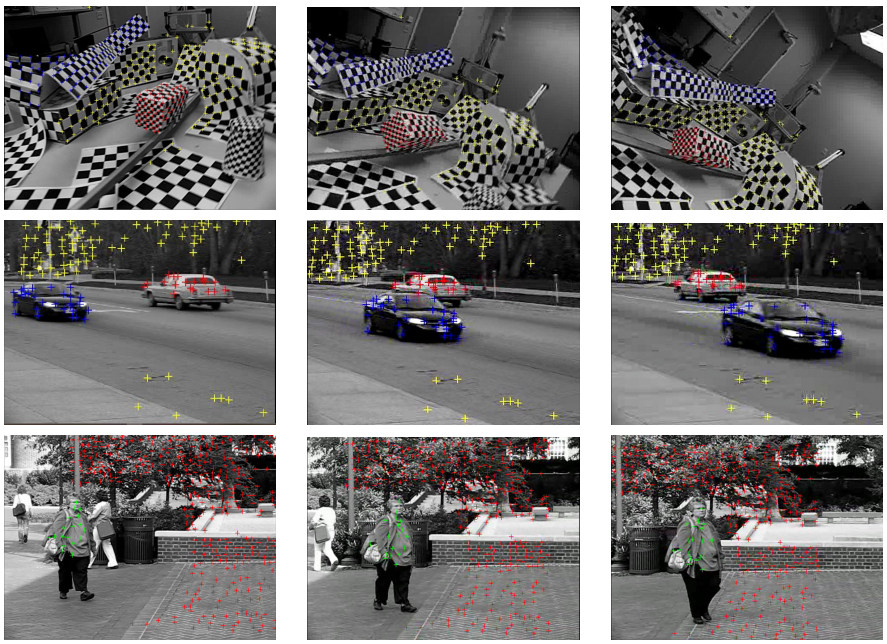


Figure 6. *Sample frames from Hopkins* 155 *database.*

and deletion. The weight of an edge is given by the summation of the pairwise similarity of vertices contained in this edge. Based on the ground-truth provided in the database, we calculate the classification error CE, which is the quotient of the number of misclassified points and the total number of points.

In Table 1, we show the mean, median, and maximum classification error of the aforementioned algorithms for computing the two motion and three motion problems in the database, respectively. It can be seen that our MSHC approach performs well in most of the cases. Figure 7 demonstrates the mean, median, and maximum classification error for all 155 sequences in the Hopskin database. The new MSHC method classifies the motions more accurately than K-means and K-flats and is comparable to TTM.

6. Conclusions. We studied multiclass clustering problems by exploiting a new normalized Laplacian tensor arising from an weighted hypergraph. For an even-uniform hypergraph, the normalized Laplacian tensor is symmetric and positive semidefinite. The smallest Z-eigenvalue of the normalized Laplacian tensor is zero, whose geometric multiplicity equals the connected components of the hypergraph. For a connected hypergraph, the smallest positive Z-eigenvalues provides a lower bound for the Cheeger constant, which is closely related to the connectivity of hypergraphs. To obtain multiple clusters of the vertex set of a hypergraph, we established a heuristic mathematical optimization model with an orthogonal constraint, which was solved by a gradient method on the Stiefel manifold. Numerical experiments illustrated that our clustering method is effective for image segmentation and motion segmentation.

When a hypergraph has more vertices, the size of the associated normalized Laplacian tensor (3.4) increases quickly. How to design a fast computational method is relevant for

Table 1
Clustering error statistics for two motion and three motion sequences.

Sequences	CE	Two motion sequences				Three motion sequences			
		MSHC	K-means	K-flats	TTM	MSHC	K-means	K-flats	TTM
Check.	Mean	0.1024	0.2034	0.1384	0.0549	0.1284	0.2743	0.1670	0.2131
	Median	0.0272	0.1925	0.1335	0.0130	0.0717	0.2041	0.1737	0.2680
	Max	0.4670	0.4979	0.3987	0.4906	0.4413	0.5744	0.4659	0.4481
Traffic	Mean	0.0992	0.2011	0.1171	0.0140	0.1195	0.2766	0.0577	0.1078
	Median	0.0251	0.1487	0.0638	0	0.0243	0.3182	0.0650	0.0081
	Max	0.4422	0.4895	0.4145	0.3750	0.3140	0.4762	0.0943	0.3645
Articul.	Mean	0.2269	0.1242	0.1057	0.2208	0.2187	0.0212	0.1789	0.2916
	Median	0.1558	0.0779	0.0808	0.1558	0.2187	0.0212	0.1789	0.2916
	Max	0.4841	0.4146	0.2468	0.4940	0.4042	0.0426	0.3511	0.3298
All	Mean	0.1005	0.1956	0.1312	0.0703	0.1464	0.2599	0.1536	0.1881
	Median	0.0388	0.1792	0.0949	0.0153	0.0917	0.2048	0.1742	0.2145
	Max	0.4841	0.4979	0.4276	0.4939	0.4480	0.5743	0.4033	0.4162

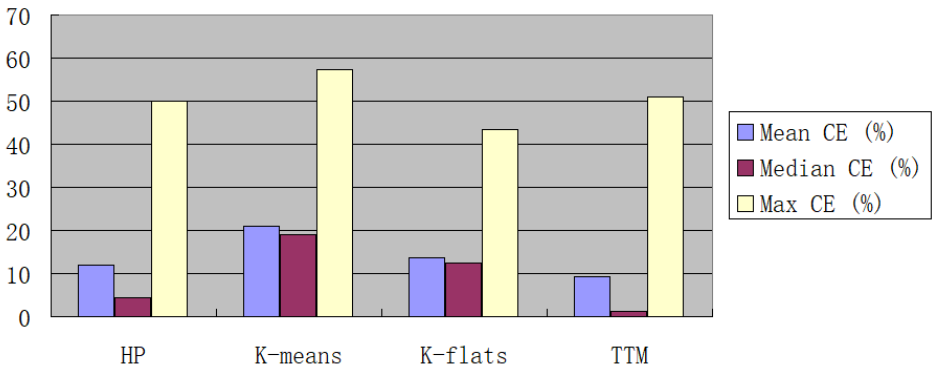


Figure 7. *Clustering error statistics for all* 155 *sequences*

hypergraph clustering. The reproducing kernel Hilbert space (RKHS) method [23] is effective for supervised segmentation of graphs and support vector machines, since RKHS has the benefit of fast computations. One of our further research topics is to apply the RKHS method to hypergraph partitioning and clustering.

Acknowledgments. The authors are grateful to the associate editor and the anonymous reviewers for their comments, which helped us to improve our paper significantly.

REFERENCES

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, *SLIC superpixels compared to state-of-the-art superpixel methods*, IEEE Trans. Pattern Anal. Mach. Intell., 34 (2012), pp. 2274–2282.

[2] A. R. Benson, D. F. Gleich, and J. Leskovec, *Tensor spectral clustering for partitioning higher-order network structures*, in Proceedings of the 2015 SIAM International Conference on Data Mining, SIAM, Philadelphia, 2015, pp. 118–126.

[3] R. Bergmann and D. Tenbrinck, *A graph framework for manifold-valued data*, SIAM J. Imaging Sci., 11 (2018), pp. 325–360, https://doi.org/10.1137/17M1118567.

[4] P. S. Bradley and O. L. Mangasarian, *K-plane clustering*, J. Global Optim., 16 (2000), pp. 23–32.
[5] J. Chang, Y. Chen, and L. Qi, *Computing eigenvalues of large scale sparse tensors arising from a hypergraph*, SIAM J. Sci. Comput., 38 (2016), pp. A3618–A3643, https://doi.org/10.1137/16M1060224.
[6] K.-C. Chang, K. Pearson, and T. Zhang, *Perron-Frobenius theorem for nonnegative tensors*, Commun. Math. Sci., 6 (2008), pp. 507–520.
[7] Y. Chen, L. Qi, and X. Zhang, *The Fiedler vector of a Laplcian tensor for hypergraph partitioning*, SIAM J. Sci. Comput., 39 (2017), pp. A2508–A2537.
[8] C. Dorai and A. K. Jain, *Shape spectra based view grouping for free-form objects*, in Proceedings of the International Conference on Image Processing, Vol. 3, IEEE, 1995, pp. 340–343.
[9] A. Elmoataz, M. Toutain, and D. Tenbrinck, *On the p-Laplacian and ∞-Laplacian on graphs with applications in image and data processing*, SIAM J. Imaging Sci., 8 (2015), pp. 2412–2451, https://doi.org/10.1137/15M1022793.
[10] V. Elyasigomari, M. Mirjafari, H. R. Screen, and M. H. Shaheed, *Cancer classification using a novel gene selection approach by means of shuffling based on data clustering with optimization*, Appl. Soft Computing, 35 (2015), pp. 43–51.
[11] D. Ghoshdastidar and A. Dukkipati, *A provable generalized tensor spectral method for uniform hypergraph partitioning*, in Proceedings of the International Conference on Machine Learning, 2015, pp. 400–409.
[12] D. Ghoshdastidar and A. Dukkipati, *Uniform hypergraph partitioning: Provable tensor methods and sampling techniques*, J. Mach. Learn. Res., 18 (2017), pp. 1638–1678.
[13] N. Grira, M. Crucianu, and N. Boujemaa, *Unsupervised and semi-supervised clustering: A brief survey*, A Review of Machine Learning Techniques for Processing Multimedia Content, 1 (2004), pp. 9–16.
[14] J. Han, J. Pei, and M. Kamber, *Data Mining: Concepts and Techniques*, Elsevier, New York, 2011.
[15] A. Hinneburg and D. A. Keim, *An efficient approach to clustering in large multimedia databases with noise*, in Proceedings of KDD, 1998, pp. 58–65.
[16] M. Honarkhah and J. Caers, *Stochastic simulation of patterns using distance-based pattern modeling*, Math. Geosci., 42 (2010), pp. 487–517.
[17] Z. Huang, *A fast clustering algorithm to cluster very large categorical data sets in data mining*, Data Min. Knowl. Discov., 3 (1997), pp. 34–39.
[18] International Human Genome Sequencing Consortium, *Initial sequencing and analysis of the human genome*, Nature, 409 (2001), pp. 860–921.
[19] A. K. Jain, M. N. Murty, and P. J. Flynn, *Data clustering: A review*, ACM Comput. Surv., 31 (1999), pp. 264–323.
[20] B. Jiang and Y. Dai, *A framework of constraint preserving update schemes for optimization on Stiefel manifold*, Math. Program., 153 (2015), pp. 535–575.
[21] D. Jiang, G. Chen, B. C. Ooi, K.-L. Tan, and S. Wu, *epiC: An extensible and scalable system for processing big data*, VLDB, 7 (2014), pp. 541–552.
[22] J. Malik, S. Belongie, T. Leung, and J. Shi, *Contour and texture analysis for image segmentation*, Int. J. Comput. Vis., 43 (2001), pp. 7–27.
[23] S. H. Kang, B. Shafei, and G. Steidl, *Supervised and transductive multi-class segmentation using p-Laplacians and RKHs methods*, J. Vis. Commun. Image Represent., 25 (2014), pp. 1136–1148.
[24] R. Kannan, S. Vempala, and A. Vetta, *On clusterings: Good, bad and spectral*, J. ACM, 51 (2004), pp. 497–515, https://doi.org/10.1145/990308.990313.
[25] R. M. Karp, *Reducibility Among Combinatorial Problems*, Plenum, New York, 1972, pp. 85–103.
[26] S. Klamt, U.-U. Haus, and F. Theis, *Hypergraphs and cellular networks*, PLoS Comput. Biol., 5 (2009), e1000385.
[27] G. Li, L. Qi, and G. Yu, *The Z-eigenvalues of a symmetric tensor and its application to spectral hypergraph theory*, Numer. Linear Algebra Appl., 20 (2013), pp. 1001–1029.
[28] L.-H. Lim, *Singular values and eigenvalues of tensors: A variational approach*, in Proceedings of the 1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, 2005, IEEE, 2005, pp. 129–132.
[29] J. MacQueen, *Some methods for classification and analysis of multivariate observations*, in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Oakland, CA, 1967, pp. 281–297.

[30] D. Park, C. Caramanis, and S. Sanghavi, *Greedy subspace clustering*, in Proceedings of Advances in Neural Information Processing Systems, 2014, pp. 2753–2761.
[31] L. Qi, *Eigenvalues of a real supersymmetric tensor*, J. Symbolic Comput., 40 (2005), pp. 1302–1324.
[32] L. Qi and Z. Luo, *Tensor Analysis: Spectral Theory and Special Tensors*, SIAM, Philadephia, 2017.
[33] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, M. J. Er, W. Ding, and C.-T. Lin, *A review of clustering techniques and developments*, Neurocomputing, 267 (2017), pp. 664–681.
[34] J. Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE Trans. Pattern Anal. Mach. Intell., 22 (2000), pp. 888–905.
[35] P. Tahmasebi, A. Hezarkhani, and M. Sahimi, *Multiple-point geostatistical modeling based on the cross-correlation functions*, Comput. Geosci., 16 (2012), pp. 779–797.
[36] R. Tron and R. Vidal, *A benchmark for the comparison of 3-d motion segmentation algorithms*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2007, pp. 1–8.
[37] J. C. Venter, M. D. Adams, E. W. Myers, et al., *The sequence of the human genome*, Science, 291 (2001), pp. 1304–1351.
[38] Z. Wen and W. Yin, *A feasible method for optimization with orthogonality constraints*, Math. Program., 142 (2013), pp. 397–434, https://doi.org/10.1007/s10107-012-0584-1.
[39] S. Wu, A.-C. Liew, H. Yan, and M. Yang, *Cluster analysis of gene expression data based on self-splitting and merging competitive learning*, IEEE Trans. Inform. Tech. Biomed., 8 (2004), pp. 5–15.
[40] D. Zhou, J. Huang, and B. Schölkopf, *Learning with hypergraphs: Clustering, classification, and embedding*, in Proceedings of Advances in Neural Information Processing Systems, 2007, pp. 1601–1608.