Tensor LRR and Sparse Coding-Based Subspace Clustering

Yifan Fu, Junbin Gao, David Tien, Zhouchen Lin, *Senior Member, IEEE*, and Xia Hong

***Abstract*— Subspace clustering groups a set of samples from a union of several linear subspaces into clusters, so that the samples in the same cluster are drawn from the same linear subspace. In the majority of the existing work on subspace clustering, clusters are built based on feature information, while sample correlations in their original spatial structure are simply ignored. Besides, original high-dimensional feature vector contains noisy/redundant information, and the time complexity grows exponentially with the number of dimensions. To address these issues, we propose a tensor low-rank representation (TLRR) and sparse coding-based (TLRRSC) subspace clustering method by simultaneously considering feature information and spatial structures. TLRR seeks the lowest rank representation over original spatial structures along all spatial directions. Sparse coding learns a dictionary along feature spaces, so that each sample can be represented by a few atoms of the learned dictionary. The affinity matrix used for spectral clustering is built from the joint similarities in both spatial and feature spaces. TLRRSC can well capture the global structure and inherent feature information of data and provide a robust subspace segmentation from corrupted data. Experimental results on both synthetic and real-world data sets show that TLRRSC outperforms several established state-of-the-art methods.**

***Index Terms*— Dictionary learning, sparse coding (SC), subspace clustering, tensor low-rank representation (TLRR).**

I. Introduction

IN RECENT years, we have witnessed a huge growth of multidimensional data due to technical advances in sensing, networking, data storage, and communication technologies. This prompts the development of a low-dimensional representation that best fits a set of samples in a high-dimensional space. Linear subspace learning is a type of traditional dimensionality reduction technique that finds an optimal linear mapping to a lower dimensional space. For example, principle component analysis (PCA) [40] is essentially based on the hypothesis that the data are drawn from a low-dimensional subspace. However, in practice, a data set is not often well described by a single subspace. Therefore, it is more reasonable to consider data residing on a union of multiple low-dimensional subspaces, with each subspace fitting a subgroup of data. The objective of the subspace clustering is to assign data to their relevant subspace clusters based on, for example, assumed models. In the last decade, subspace clustering has been widely applied to many real-world applications, including motion segmentation [14], [20], social community identification [9], and image clustering [3]. A famous survey on subspace clustering [44] classifies most existing subspace clustering algorithms into three categories: statistical methods [19], algebraic methods [38], [49], and spectral clustering-based methods [14], [29].

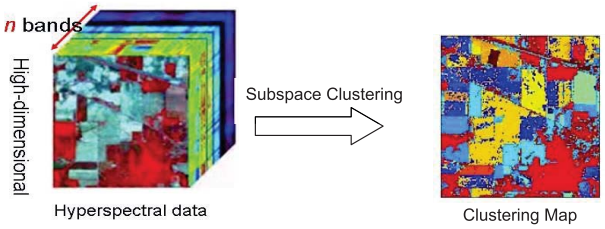


Fig. 1. Illustration of subspace clustering with high-dimensional data.

In the existing traditional subspace clustering algorithms [44], one usually uses an unfolding process to rearrange samples into a list of individual vectors, represented by a matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$, with each sample $\mathbf{x}_i$ $(1 \leq i \leq N)$ being denoted by a column vector. However, in many applications, samples may have multidimensional spatial structural forms, such as 2-D/mode hyperspectral images. In the 3-D hyperspectral image case, one wishes to cluster all the pixels, each of which is represented as a spectrum vector consisting of many bands, as shown in Fig. 1. As a result, the performance of traditional subspace clustering algorithms may be compromised in practical applications for two reasons: 1) they do not consider the inherent structure and correlations in the original data and 2) building a model based on original high-dimensional features is not effective to filter the noisy/redundant information in the original feature spaces, and the time complexity grows exponentially with the number of dimensions.

Manuscript received November 2, 2014; revised February 16, 2016; accepted April 3, 2016. Date of publication April 27, 2016; date of current version September 15, 2016. This work was supported by the Australian Research Council through the Discovery Project under Grant DP130100364. The work of Z. Lin was supported in part by the National Basic Research Program of China (973 Program) under Grant 2015CB352502, in part by the National Natural Science Foundation of China under Grant 61231002 and Grant 61272341, and in part by Microsoft Research Asia through the Collaborative Research Program. *(Corresponding author: Zhouchen Lin.)*

Y. Fu and D. Tien are with the School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW 2795, Australia (e-mail: fuyf939@gmail.com; dtien@csu.edu.au).

J. Gao is with the Discipline of Business Analytics, The University of Sydney Business School, The University of Sydney, NSW 2006, Australia (e-mail: junbin.gao@sydney.edu.au).

Z. Lin is with the Key Laboratory of Machine Perception (Ministry of Education), School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China, and also with the Cooperative Medianet Innovation Center, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zlin@pku.edu.cn).

X. Hong is with the Department of Computer Science, School of Mathematical and Physical Sciences, University of Reading, Reading RG6 6AY, U.K. (e-mail: x.hong@reading.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2553155

For the first issue, tensor is a suitable representation for such multidimensional data, such as hyperspectral images, in a format of a multiway array. The order of a tensor is the number of dimensions, also known as ways or modes. Thus, a set of hyperspectral images with a 2-D spatial structure can be denoted by an order-3 tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, with mode-$i$ ($1 \leq i \leq 2$) denoting the sample's position along its two spatial directions, and the mode-3 denoting the sample feature direction, e.g., a range of wavelengths in the spectral dimension. Fu *et al.* [50] proposed a novel subspace clustering method called tensor low-rank representation (TLRR), where the input data are represented in their original structural form as a tensor. It finds a lowest rank representation for the input tensor, which can be further used to build an affinity matrix. The affinity matrix used for spectral clustering [54] records pairwise similarity along the row and column directions.

For the second issue, finding low-dimensional inherent feature spaces is a promising solution. Dictionary learning [51] is commonly used to seek the lowest rank [29], [52], [53] or sparse representation [14], [55] with respect to a given dictionary, which is often the data matrix $\mathbf{X}$ itself. LRR and sparse representation/sparse coding (SC) take sparsity into account in different ways. The former defines a holistic sparsity on a whole data representation matrix, while the latter finds the sparsest representation of each data vector individually. SC has been widely used in numerous signal processing tasks, such as imaging denoising, texture synthesis, and image classification [26], [36], [48]. Nevertheless, the performance of SC deteriorates when data are corrupted. Therefore, it is highly desirable to integrate spatial information into SC to improve the clustering performance and reduce the computational complexity as well.

Against this background, we propose a novel subspace clustering method, where the input data are represented in their original structural form as a tensor. Our model finds the lowest rank representation for each spatial mode of the input tensor, and a sparse representation with respect to a learned dictionary in the feature mode. The combination of similarities in spatial and feature spaces is used to build an affinity matrix for spectral clustering. In summary, the contribution of this paper is threefold.

1) We propose a TLRR to explore spatial correlations among samples. Unlike previous work that merely considers sample feature similarities and reorders original data into a matrix, our model takes sample spatial structure and correlations into account. Specifically, our method directly seeks an LRR of natural structural form—a high-order tensor.
2) This paper integrates dictionary learning for sparse representation in the feature mode of tensor. This setting fits each individual sample with its sparsest representation with respect to the learned dictionary, consequently resolving exponential complexity and memory usage issues of some classical subspace clustering methods (e.g., statistical- and algebraic-based methods) effectively.
3) The new subspace clustering algorithm based on our model is robust and capable of handling noise in the data. Since our model considers both feature and spatial similarities among the samples, even if data are severely corrupted, it can still maintain a good performance, since the spatial correlation information is utilized in order to cluster data into their respective subspaces correctly.

II. Related Work

Vidal [44] classifies the existing subspace clustering algorithms into three categories: statistical methods, algebraic methods, and spectral clustering-based methods.

Statistical models assume that mixed data are formed by a set of independent samples from a mixture of a certain distribution, such as Gaussian. Each Gaussian distribution can be considered as a single subspace, and then subspace clustering is transformed into a mixture of Gaussian model estimation problems. This estimation can be obtained by the expectation maximization algorithm in the mixture of probabilistic PCA [41], or serial subspace searching in random sample consensus (RANSAC) [17]. Unfortunately, these solutions are sensitive to noise and outliers. Some efforts have been made to improve algorithm robustness. For example, agglomerative lossy compression [31] finds the optimal segmentation that minimizes the overall coding length of the segmented data, subject to each subspace being modeled as a degenerate Gaussian. However, the optimization difficulty is still a bottleneck in solving this problem.

Generalized PCA (GPCA) [45] is an algebraic-based method to estimate a mixture of linear subspaces from the sample data. It factorizes a homogeneous polynomial whose degree is the number of subspaces and the factors (roots) represent normal vectors of each subspace. GPCA has no restriction on subspaces and works well under certain conditions. Nevertheless, the performance of the algebraic-based methods in the presence of noise deteriorates as the number of subspaces increases. Robust algebraic segmentation [38] is proposed to improve its robustness, but the complexity issue still exists. Iterative methods improve the performance of the algebraic-based algorithms to handle noisy data in a repeated refinement. The k-subspace method [6], [19] extends the k-means clustering algorithm from the data distributed around cluster centers to data drawn from the subspaces of any dimensions. It alternates between assigning samples to subspaces and re-estimating subspaces. The k-subspace method can converge to a local optimum in a finite number of iterations. Nevertheless, the final solution depends on good initialization and is sensitive to outliers.

The works in [14], [29], and [38] are representative of spectral clustering-based methods. They aim to find a linear representation, $\mathbf{Z}$, for all the samples in terms of all other samples, which is solved by finding the optimal solution to the following objective function:

$$\begin{aligned} &\min_{\mathbf{Z}} \|\mathbf{Z}\|_b + \frac{\lambda}{2}\|\mathbf{E}\|_q \\ &\text{s.t. } \mathbf{X} = \mathbf{X}\mathbf{Z} + \mathbf{E} \end{aligned} \tag{1}$$

where $\|\cdot\|_q$ and $\|\cdot\|_b$ denote the norms for error and the new representation matrix $\mathbf{Z}$, respectively, and λ is the parameter to

balance the two terms. Using the resulting matrix $\mathbf{Z}$, an affinity matrix $|\mathbf{Z}| + |\mathbf{Z}^T|$ is built and used for spectral clustering. The sparse subspace clustering (SSC) [14] uses the l_1-norm $\|\mathbf{Z}\|_1$ in favor of a sparse representation, with the expectation that within-cluster affinities are sparse (but not zero) and between-cluster affinities shrink to zero. However, this method is not designed to accurately capture the global structure of data and may not be robust to noise in data. The LRR [29] employs the nuclear norm $\|\mathbf{Z}\|_*$ to guarantee a low-rank structure, and the $l_{2,1}$-norm is used in the error term to make it robust to outliers.

Dictionary learning for sparse representation aims at learning a dictionary $\mathbf{D}$, such that each sample in the data set can be represented as a sparse linear combination of the atoms of $\mathbf{D}$. The problem of dictionary learning is formulated as

$$\min_{\mathbf{D},\mathbf{z}_i(i=1,2,\ldots,N)} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{D}\mathbf{z}_i\|_2^2 \quad \text{s.t. } \|\mathbf{z}_i\|_0 = r \tag{2}$$

where $\|\cdot\|_2$ denotes the l_2-norm, $\|\mathbf{z}_i\|_0$ is the l_0-norm of the coefficient vector $\mathbf{z}_i$, which is defined as the number of nonzero elements, and r is a predefined sparsity integer for each sample. The optimization is carried out using an iterative algorithm that is formed by two alternative steps: 1) the SC by fixing the dictionary $\mathbf{D}$ and 2) the dictionary update with a fixed sparse representation.

With regard to SC for a given dictionary, the existing algorithms are divided into three categories: optimization methods, greedy methods, and thresholding-based methods. Basis pursuit (BP) is a commonly used optimization method that uses a convex optimization method to minimize the l_1-norm $\|\mathbf{z}_i\|_1$ subject to the constraint $\mathbf{x}_i = \mathbf{D}\mathbf{z}_i$, if the vector $\mathbf{z}_i$ is sparse enough and the matrix $\mathbf{D}$ has sufficiently low coherence [12], [42]. The computational complexity of BP is very high, thus it is not suitable for large-scale problems. In comparison, the greedy algorithm matching pursuit (MP) has a significantly smaller complexity than BP, especially when the sparsity level is low [43]. A popular extension of MP is orthogonal MP (OMP) [33], [34], which iteratively refines a sparse representation by successively identifying one component at a time that yields the greatest improvement in quality until an expected sparsity level is achieved or the approximation error is below the given threshold. The thresholding-based methods contain algorithms that do not require an estimation of the sparsity. In such algorithms, the hard thresholding operator gives way to a soft thresholding operator with a positive threshold, such as the iterative hard thresholding algorithm [5] and the hard thresholding pursuit [18]. Another important method for SC is the message-passing algorithm studied in [11].

The main differences in dictionary update algorithms are in the ways they update the dictionary. The Sparsenet [35] and the method of optimal directions (MODs) [15] perform the dictionary update with fixed values of coefficients. Sparsenet updates each atom of dictionary iteratively with a projected fixed step gradient descent. MOD updates the whole dictionary in one step by finding a closed-form solution of an unconstrained least-square problem. Different from the above two algorithms, K-singular value decomposition (SVD) [1] updates each dictionary atom and the values of its nonzero sparse coefficient simultaneously. The atom update problem then becomes a PCA problem. The K-SVD algorithm is flexible and can work with any pursuit methods.

III. Notations and Problem Formulation

A. Definition and Notations

Before formulating the subspace-clustering problem, we first introduce some tensor fundamentals and notations. Please refer to [22] for more detailed definitions and notations.

Definition 1 (Tensor Matricization): Matricization is the operation of rearranging the entries of a tensor so that it can be represented as a matrix. Let $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ be a tensor of order-N, the mode-n matricization of $\mathcal{X}$ reorders the mode-n vectors into columns of the resulting matrix, denoted by $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times (I_{n+1} I_{n+2} \ldots I_N I_1 I_2 \ldots I_{n-1})}$.

Definition 2 (Kronecker Product [22]): The Kronecker product of matrices $\mathbf{A} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} \in \mathbb{R}^{P \times L}$, denoted by $\mathbf{A} \otimes \mathbf{B}$, is a matrix of size $(IP) \times (JL)$ defined by

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \cdots & a_{IJ}\mathbf{B} \end{bmatrix}. \tag{3}$$

Definition 3 (n-Mode Product): The n-mode product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ by a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$, denoted by $\mathcal{X} \times_n \mathbf{U}$, is a tensor with entries

$$(\mathcal{X} \times_n \mathbf{U})_{i_1,\ldots,i_{n-1},j,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \ldots i_N} u_{j i_n}. \tag{4}$$

The n-mode product is also denoted by each mode-n vector multiplied by the matrix $\mathbf{U}$. Thus, it can be expressed in terms of tensor matricization as well

$$\mathcal{Y} = \mathcal{X} \times_n \mathbf{U} \Leftrightarrow \mathbf{Y}_{(n)} = \mathbf{U}\mathbf{X}_{(n)}. \tag{5}$$

Definition 4 (Tucker Decomposition): Given an order-N tensor $\mathcal{X}$, its Tucker decomposition is an approximated tensor defined by

$$\begin{aligned} \hat{\mathcal{X}} &\equiv [\![\mathcal{G}; \mathbf{U}_1, \ldots, \mathbf{U}_N]\!] = \mathcal{G} \times_1 \mathbf{U}_1 \times_2 \ldots \times_N \mathbf{U}_N \\ &= \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \cdots \sum_{r_N=1}^{R_N} g_{r_1 r_2 \ldots r_N} \mathbf{u}_{r_1}^1 \circ \mathbf{u}_{r_2}^2 \ldots \circ \mathbf{u}_{r_N}^N \end{aligned} \tag{6}$$

where $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times \ldots R_N}$ is called a core tensor, $\mathbf{U}_n = [\mathbf{u}_1^n, \mathbf{u}_2^n, \ldots, \mathbf{u}_{R_n}^n] \in \mathbb{R}^{I_n \times R_n}$ $(1 \le n \le N)$ are the factor matrices, and the symbol $\circ$ represents the vector outer product.

For ease of presentation, key symbols used in this paper are shown in Table I.

B. Tensorial Data Sets

Given an order-N tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, we consider a data set of all the I_N-dimensional vectors/features along $\mathcal{X}$'s N-mode (also called N-mode fibers). The size of the data set is $(I_1 \times I_2 \times \cdots \times I_{N-1})$. Assume that these samples are drawn from a union of K independent subspaces $\{S_k\}_{k=1}^K$ of unknown

TABLE I
NOTATIONS USED IN THIS PAPER

$\mathcal{X}$ and $\mathcal{E}$	an order-Ninput tensor and an order-Nerror tensor
$\mathbf{X}_{(n)}$ and $\mathbf{E}_{(n)}$	the mode-n matricization of tensors $\mathcal{X}$ and $\mathcal{E}$
S_k and $\mathcal{L}_k(k = 1, 2, ., , , K)$	the kth subspace and its corresponding orthogonal basis
$\mathbf{X}_k$ and Z_k	samples drawn from subspace S_k and corresponding low-dimensional representation
$\mathbf{S} = [S_1, S_2, \ldots, S_K]$	a set of multiple learnt subspaces
$\mathbf{Z} = [Z_1, Z_2, \ldots, Z_K]$	the low-dimensional representations of $\mathcal{X}$ with respect to $\mathbf{S}$
$\mathbf{U}_n \in \mathbb{R}^{I_n \times R_n}(1 \leq n \leq N)$	the factor matrices along mode-n
$\mathbf{D}$ and $\mathbf{A}$	a learned dictionary and a sparse representation on $\mathbf{X}_{(N)}$
r and R	the maximum number of non-zero elements in each instance and the matrix $\mathbf{A}$
$\Phi_{(I_1,\ldots,I_{N-1},I_N)}$	a transformation of inverse matricization
$\mathbf{W}$	the structured coefficient matrix
λ, Y_n and $\mu_n > 0$	a balance parameter, the Lagrange multiplier and a penalty parameter, respectively
$\|\cdot\|_1$, $\|\cdot\|_0$, $\|\cdot\|_{2,1}$, $\|\cdot\|_*$ and $\|\cdot\|_F$	the l_1 norm, l_0 norm , $l_{2,1}$ norm, the nuclear and Frobenius norm

dimensions, i.e., $\sum_{k=1}^{K} S_k = \bigoplus_{k=1}^{K} S_k$, where $\bigoplus$ is the direct sum. Our purpose is to cluster all the I_N-dimensional vectors from the tensor $\mathcal{X}$ into K subspaces by incorporating their relevant spatial and feature information in the tensor.

IV. SUBSPACE CLUSTERING VIA TENSOR LOW-RANK REPRESENTATION AND SPARSE CODING

A. Tensor Low-Rank Representation on Spatial Modes

The new approach LRR [29] is very successful in subspace clustering for even highly corrupted data, outliers, or missing entries. Inspired by the idea used in LRR, we consider a model of LRR for an input tensor $\mathcal{X}$ similar to problem (1). Specifically, we decompose the input tensor $\mathcal{X}$ into a Tucker decomposition, in which the core tensor $\mathcal{G}$ is the input tensor itself along with a factor matrix $\mathbf{U}_n$ at each mode $n \leq N$. That is, the proposed data representation model is

$$\mathcal{X} = [\![\mathcal{X}; \mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_N]\!] + \mathcal{E}. \tag{7}$$

Here, we are particularly interested in the case, where $\mathbf{U}_N = \mathbf{I}_{I_N}$ (the identity matrix of order I_N). If we define $\mathbf{Z} = \mathbf{U}_{N-1} \otimes \mathbf{U}_{N-2} \otimes \cdots \otimes \mathbf{U}_1$, then based on the above multiple linear model, we may interpret the entries of $\mathbf{Z}$ as the similarities between the pairs of all the vectors along the N-mode of the data tensor $\mathcal{X}$. These similarities are calculated based on the similarities along all the $N-1$ spatial modes through the factor matrices $\mathbf{U}_n$ $(n = 1, \ldots, N-1)$, each of which measures the similarity at the nth spatial mode.

As in LRR, model (7) uses the data to represent itself; therefore, we can expect low-rank factor matrices $\mathbf{U}_n$. It is well known that it is very hard to solve an optimization problem with matrix rank constraints. A common practice is to relax the rank constraint by replacing it with the nuclear norm [32] as suggested by the matrix completion methods [8], [21]. Thus, we finally formulate our model as follows:

$$\begin{aligned} &\min_{\mathbf{U}_1,\ldots,\mathbf{U}_{N-1}} \sum_{n=1}^{N-1} \|\mathbf{U}_n\|_* + \frac{\lambda}{2}\|\mathcal{E}\|_F^2 \\ &\text{s.t. } \mathcal{X} = [\![\mathcal{X}; \mathbf{U}_1, \ldots, \mathbf{U}_{N-1}, \mathbf{I}_{I_N}]\!] + \mathcal{E} \end{aligned} \tag{8}$$

where $\|\cdot\|_*$ denotes the nuclear norm of a matrix, defined as the sum of singular values of the matrix, $\|\cdot\|_F$ denotes the Frobenius norm of a tensor, i.e., the square root of the sum of the squares of all its entries, and $\lambda > 0$ is a parameter to balance the two terms, which can be tuned empirically. That is, TLRR seeks optimal low-rank solutions $\mathbf{U}_n(1 \leq n < N)$ of the structured data $\mathcal{X}$ using itself as a dictionary [50].

Remark 1: There is a clear link between the LRR and the TLRR in (8). If we consider the mode-N matricization in (8), we will see that it can be converted into an LRR model with $\mathbf{Z} = \mathbf{U}_{N-1} \otimes \mathbf{U}_{N-2} \otimes \cdots \otimes \mathbf{U}_1$. However, in the standard LRR, such an explicit Kronecker structure in $\mathbf{Z}$ has been ignored, so the number of unknown parameters in $\mathbf{Z}$ is $(I_1 \times I_2 \times \cdots I_{N-1})^2$. Such a huge number cause difficulty in the LRR algorithm when doing SVD. However, TLRR exploits the Kronecker structure with the number of unknown parameters reduced to $I_1^2 + I_2^2 + \cdots + I_{N-1}^2$. Our experiments demonstrate that the TLRR is much faster than the LRR.

B. Dictionary Learning for Sparse Representation on Feature Mode

Dictionary learning for sparse representation has been proved to be very effective in machine learning, neuroscience, signal processing, and statistics [13], [25], [37]. Similar ideas have been proposed for subspace clustering. SSC algorithm [14] is an inspiring approach that uses data itself as a given dictionary, sparsely representing each sample as a linear combination of the rest of the data. However, such representation is computationally expensive for large-scale data. In addition, it is well known that such SC techniques strongly rely on the internal coherence of the dictionary, and their performance may drop grossly as the number of clusters grows. In contrast, our sparse modeling framework exploits a sparse representation in the design of an optimization procedure dedicated to the problem of dictionary learning, with comparable memory consumption and a lower computational cost than SSC.

Based on the model in (8), we consider a dictionary learning model for sparse representation along the N-mode (feature mode) of $\mathcal{X}$. To be specific, we approximate the mode-N matricization of tensor $\mathbf{X}_{(N)}$ with a dictionary to be learned $\mathbf{D} \in \mathbb{R}^{I_N \times m}$ and a sparse representation $\mathbf{A} \in \mathbb{R}^{m \times (I_1 \times I_2 \times \cdots \times I_{N-1})}$ on feature spaces over all the samples, so that the feature vectors of each sample can be represented by a few atoms of $\mathbf{D}$ (i.e., $||\mathbf{a}_i||_0 < r$ for $1 \leq i \leq I_1 \times I_2 \times \ldots \times I_{N-1}$). Thus, our SC model in the

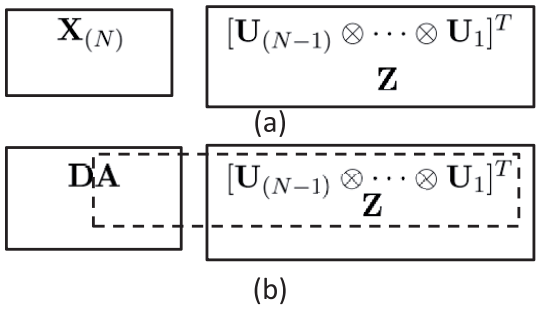


Fig. 2. Mode-N matricization of $\mathcal{X}$ decomposition for TLRR model and our algorithm. (a) Tensor LRR model. (b) Our model.

feature direction has a similar formulation to problem (2)

$$\min_{\mathbf{D},\mathbf{A}} \|\mathbf{X}_{(N)} - \mathbf{D}\mathbf{A}\|_2^2 \quad \text{s.t. } \|\mathbf{A}\|_0 = R \tag{9}$$

where $R = r \times (I_1 \times I_2 \times \cdots \times I_{N-1})$ is the maximum number of nonzero elements in the matrix $\mathbf{A}$. By solving problem (9), we can obtain an optimal dictionary on the feature space of the input tensor, and sparse factors for each sample with respect to the learned dictionary.

C. Tensor Spatial Low-Rank Representation and Feature Sparse Coding

By integrating the advantages of LRR and SC, we propose a spectral-based subspace clustering method, which simultaneously considers sample feature information and spatial structures. More specifically, we first define a transformation of inverse matricization $\Phi_{(I_1,\ldots,I_{N-1},I_N)}$, which converts a matrix $\mathbf{M} \in \mathbb{R}^{I_N \times (I_1 I_2 \cdots I_{N-1})}$ back into an order-N tensor $\mathcal{M} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_{N-1} \times I_N}$, i.e., $\Phi_{I_1,\ldots,I_{N-1},I_N}(M) = \mathcal{M}$ [2].

Now, we replace the fixed data tensor core in (8) with a tensor reconverted from a matrix $\mathbf{DA}$ by applying the inverse matricization operator Φ, i.e., the model is

$$\mathcal{X} = [\![\Phi_{(I_1,I_2,\ldots,I_{N-1},I_N)}(\mathbf{DA}); \mathbf{U}_1, \ldots, \mathbf{U}_{N-1}, \mathbf{I}_{I_N}]\!] + \mathcal{E} \tag{10}$$

where $\mathbf{D} \in \mathbb{R}^{I_N \times m}$ is the feature mode dictionary to be learned, and $\mathbf{A} \in \mathbb{R}^{m \times (I_1 I_2 \cdots I_{N-1})}$ is the sparse representation coefficient matrix. Finally, our proposed learning model can be formulated as follows:

$$\begin{aligned} &\min_{\mathbf{U}_1,\ldots,\mathbf{U}_{N-1},\mathbf{D},\mathbf{A}} \sum_{n=1}^{N-1} \|\mathbf{U}_n\|_* + \frac{\lambda}{2} \| \mathcal{E} \|_F^2 \\ &\text{s.t. } \mathcal{X} = [\![\Phi_{(I_1,I_2,\ldots,I_{N-1},I_N)}(\mathbf{DA}); \mathbf{U}_1, \ldots, \mathbf{U}_{N-1}, \mathbf{I}_{I_N}]\!] + \mathcal{E} \\ &\|\mathbf{A}\|_0 = R \end{aligned} \tag{11}$$

where R is a given sparsity. Thus, our model (11) aims to find the lowest rank representations along all the spatial modes, and learn a dictionary with its sparse representation over the samples on the feature mode at the same time.

Unlike the TLRR model in [50] merely considers multiple space information, our model takes both spatial and feature information into consideration. The advantage of our model over the TLRR model is shown in Fig. 2. Taking the mode-N matricization of $\mathcal{X}$ as an example, the TLRR model uses $\mathbf{X}_{(N)}$ as a dictionary, the coefficient matrix $\mathbf{Z} = [\mathbf{U}_{(N-1)} \otimes \cdots \otimes \mathbf{U}_1]^T$. However, our model learns a dictionary $\mathbf{D}$ on the feature space, which makes

$$\mathbf{X}_{(N)} = \mathbf{DA}[\mathbf{U}_{(N-1)} \otimes \cdots \otimes \mathbf{U}_1]^T$$

as shown in Fig. 2(b). Accordingly, the dictionary representation of data $\mathbf{X}_{(N)}$ is given by the structured coefficient matrix $\mathbf{Z} = \mathbf{A}[\mathbf{U}_{(N-1)} \otimes \cdots \otimes \mathbf{U}_1]^T$. Thus, both spatial information encoded in $\mathbf{U}_n$'s and the feature relations encoded in $\mathbf{A}$ make a contribution to the final dictionary representation.

Remark 2: Problem (11) is ill-posed due to the scaling between variables $\mathbf{D}, \mathbf{A}$, and $\mathbf{U}$. To make a well-posed problem, we also require extra constraints; for example, the columns of D are the unit vectors and the largest entry of $\mathbf{A}$ to be 1.

Remark 3: Using the Frobenius norm for the error, we are dealing with Gaussian noises in the tensor data. If based on some domain knowledge, we know some noise patterns along a particular mode, for example, in multispectral imaging data, noises in some spectral bands are significant, we may adapt the so-called robust noise models, such as $l_{2,1}$-norm [10] instead.

Remark 4: As pointed out in Remark 1, TLRR improves the computational cost of the original LRR by introducing the Kronecker structure into the expression matrix. Although the new model looks more complicated than the traditional dictionary model, the computational complexity would not blow up due to the Kronecker structure. The cost added over the traditional dictionary learning is the overhead in handling low-rank constraints over spatial modes. This is the price we have to pay for incorporating the spatial information of the data.

D. Solving the Optimization Problem

Optimizing (11) can be carried out using an iterative approach that solves the following two subproblems.

1) *Solve TLRR Problem:* Fix $\mathbf{D}$ and $\mathbf{A}$ to update $\mathbf{U}_n$, where $1 \le n < N$ by

$$\begin{aligned} &\min_{\mathbf{U}_1,\ldots,\mathbf{U}_{N-1}} \sum_{n=1}^{N-1} \|\mathbf{U}_n\|_* + \frac{\lambda}{2} \| \mathcal{E}\|_F^2 \\ &\text{s.t. } \mathcal{X} = [\![\Phi(\mathbf{DA}); \mathbf{U}_1, \ldots, \mathbf{U}_{N-1}, \mathbf{I}_{I_N}]\!] + \mathcal{E}. \end{aligned} \tag{12}$$

2) *Solve Dictionary Learning for SC Problem:* Fix $\mathbf{U}_n (1 \le n < N)$ to update $\mathbf{D}$ and $\mathbf{A}$ by

$$\begin{aligned} &\min_{\mathbf{D},\mathbf{A}} \frac{\lambda}{2} \|\mathcal{X} - [\![\Phi(\mathbf{DA}); \mathbf{U}_1, \ldots, \mathbf{U}_{N-1}, \mathbf{I}_{I_N}]\!]\|_F^2 \\ &\text{s.t. } ||\mathbf{A}||_0 = R. \end{aligned} \tag{13}$$

We will employ the block coordinate descent (BCD) [4] to solve the optimization problem (12) by fixing all the other mode variables to solve one variable at a time alternatively. For instance, TLRR fixes $\mathbf{U}_1, \ldots, \mathbf{U}_{n-1}, \mathbf{U}_{n+1}, \ldots, \mathbf{U}_{N-1}$ to minimize (12) with respect to the variable $\mathbf{U}_n (n = 1, 2, \ldots, N-1)$, which is equivalent to solve the following optimization subproblem:

$$\begin{aligned} &\min_{\mathbf{U}_n} \|\mathbf{U}_n\|_* + \frac{\lambda}{2} \| \mathcal{E}\|^2 \\ &\text{s.t. } \mathcal{X} = [\![\Phi(\mathbf{DA}); \mathbf{U}_1, \ldots, \mathbf{U}_{N-1}, \mathbf{I}_{I_N}]\!] + \mathcal{E}. \end{aligned} \tag{14}$$

Algorithm 1 Solving Problem (12) by BCD

Input: data tensor $\mathcal{X}$, dictionary $\mathbf{D}$ and $\mathbf{A}$ and the parameter λ
Output: factor matrices $\mathbf{U}_n$ $(n = 1, 2, \ldots, N-1)$

randomly initialize $\mathbf{U}_n \in \mathbb{R}^{I_n \times R_n}$ for $n = 1, \ldots, N-1$
for $n = 1, \ldots, N-1$ **do**
  $\mathbf{X}_{(n)} \leftarrow$ the mode-n matricization of the tensor $\mathcal{X}$
  $\Phi(\mathbf{DA})_{(n)} \leftarrow$ the mode-n matricization the tensor $\Phi(\mathbf{DA})$
end for
while reach maximum iterations or converge to stop **do**
  for $n = 1, \ldots, N-1$ **do**
    $\mathbf{B}_{(n)} \leftarrow \Phi(\mathbf{DA})_{(n)}(\mathbf{I} \otimes \mathbf{U}_{N-1} \otimes \cdots \mathbf{U}_{n+1} \otimes \mathbf{U}_{n-1} \otimes \cdots \otimes \mathbf{U}_1)^T$
    $\mathbf{U}_n \leftarrow$ solve the subproblem (15)
  end for
end while

Using tensorial matricization, problem (14) can be rewritten in terms of matrices as follows:

$$\min_{\mathbf{U}_n} \|\mathbf{U}_n\|_* + \frac{\lambda}{2}\|\mathbf{E}_{(n)}\|_F^2 \quad \text{s.t. } \mathbf{X}_{(n)} = \mathbf{U}_n\mathbf{B}_{(n)} + \mathbf{E}_{(n)} \tag{15}$$

where $\mathbf{B}_{(n)} = \Phi(\mathbf{DA})_{(n)}(\mathbf{I} \otimes \mathbf{U}_{N-1} \otimes \cdots \mathbf{U}_{n+1} \otimes \mathbf{U}_{n-1} \otimes \cdots \otimes \mathbf{U}_1)^T$.

Based on (15), each matrix $\mathbf{U}_n (1 \le n < N)$ is optimized alternatively, while the other matrices are held fixed. All the matrices update iteratively until the change in fit drops below a threshold or when the number of iterations reaches a maximum, whichever comes first. The general process of the BCD is shown in Algorithm 1.

We use the linearized alternating direction method (LADM) [30] to solve the constrained optimization problem (15).

First of all, the augmented Lagrange function of (15) can be written as

$$L(\mathbf{E}_{(n)}, \mathbf{U}_n, \mathbf{Y}_n) = \|\mathbf{U}_n\|_* + \frac{\lambda}{2}\|\mathbf{E}_{(n)}\|_F^2 + \operatorname{tr}\left[\mathbf{Y}_n^T(\mathbf{X}_{(n)} - \mathbf{U}_n\mathbf{B}_{(n)} - \mathbf{E}_{(n)})\right] + \frac{\mu_n}{2}\|\mathbf{X}_{(n)} - \mathbf{U}_n\mathbf{B}_{(n)} - \mathbf{E}_{(n)}\|_F^2 \tag{16}$$

where Y_n is the Lagrange multiplier and $\mu_n > 0$ is a penalty parameter.

Then, the variables are updated by minimizing the augmented Lagrangian function L alternately, i.e., minimizing one variable at a time, while the other variables are fixed. The Lagrange multiplier is updated according to the feasibility error. More specifically, the iterations of LADM go as follows.

1) Fix all others to update $\mathbf{E}_{(n)}$ by

$$\min_{\mathbf{E}_{(n)}} \left\|\mathbf{E}_{(n)} - \left(\mathbf{X}_{(n)} - \mathbf{U}_n\mathbf{B}_{(n)} + \frac{\mathbf{Y}_n}{\mu_n}\right)\right\|_F^2 + \frac{\lambda}{\mu_n}\|\mathbf{E}_{(n)}\|_F^2 \tag{17}$$

which is equivalent to a least-square problem. The solution is given by

$$\mathbf{E}_n = \frac{\lambda}{\lambda + \mu_n}\left(\mathbf{X}_{(n)} - \mathbf{U}_n\mathbf{B}_{(n)} + \frac{\mathbf{Y}_n}{\mu_n}\right). \tag{18}$$

Algorithm 2 Solving Problem (15) by LADM

Input: matrices $\mathbf{X}_{(n)}$ and $\mathbf{B}_{(n)}$, parameter λ
Output: : factor matrices $\mathbf{U}_n$

initialize: $\mathbf{U}_n = 0, \mathbf{E}_{(n)} = 0, \mathbf{Y}_n = 0, \mu_n = 10^{-6}, max_u = 10^{10}, \rho = 1.1, \varepsilon = 10^{-8}$ and $\eta_n = \|\mathbf{B}_{(n)}\|^2$.
while $\|\mathbf{X}_{(n)} - \mathbf{U}_n\mathbf{B}_{(n)} - \mathbf{E}_{(n)}\|_\infty \ge \varepsilon$ **do**
  $\mathbf{E}_{(n)} \leftarrow$ the solution (18) to the subproblem (17);
  $\mathbf{U}_n \leftarrow$ the iterative solution by (21) by for example five iterations;
  $\mathbf{Y}_n \leftarrow \mathbf{Y}_n + \mu_n(\mathbf{X}_{(n)} - \mathbf{U}_n\mathbf{B}_{(n)} - \mathbf{E}_{(n)})$
  $\mu_n \leftarrow \min(\rho\mu_n, max_u)$
end while

2) Fix all others to update $\mathbf{U}_n$ by

$$\min_{\mathbf{U}_n} \|\mathbf{U}_n\|_* - \operatorname{tr}\left(\mathbf{Y}_n^T\mathbf{U}_n\mathbf{B}_{(n)}\right) + \frac{\mu_n}{2}\|(\mathbf{X}_{(n)} - \mathbf{E}_{(n)}) - \mathbf{U}_n\mathbf{B}_{(n)}\|_F^2. \tag{19}$$

3) Fix all others to update $\mathbf{Y}_n$ by

$$\mathbf{Y}_n \leftarrow \mathbf{Y}_n + \mu_n(\mathbf{X}_{(n)} - \mathbf{U}_n\mathbf{B}_{(n)} - \mathbf{E}_{(n)}). \tag{20}$$

However, there is no closed-form solution to problem (19) because of the coefficient $\mathbf{B}_{(n)}$ in the third term. We propose to use the linearized approximation with an added proximal term to approximate the objective in (19), as described in [27]. Suppose that $\mathbf{U}_{(n)}^k$ is the current approximated solution to (19) and the sum of the last two terms is denoted by L, then the first-order Taylor expansion at $\mathbf{U}_{(n)}^k$ plus a proximal term is given by

$$L \approx \mu_n\left\langle\left(\mathbf{U}_n^k\mathbf{B}_{(n)} + \mathbf{E}_n - \mathbf{X}_{(n)} - \frac{\mathbf{Y}_n}{\mu_n}\right)\mathbf{B}_{(n)}^T, \mathbf{U}_n - \mathbf{U}_n^k\right\rangle + \frac{\mu_n\eta_n}{2}\left\|\mathbf{U}_n - \mathbf{U}_n^k\right\|_F^2 + \text{consts}.$$

Thus, solving (19) can be converted to iteratively solve the following problem:

$$\min_{\mathbf{U}_n} \|\mathbf{U}_n\|_* + \frac{\mu_n\eta_n}{2}\left\|\mathbf{U}_n - \mathbf{U}_n^k + \mathbf{P}_n\right\|_F^2$$

where $\mathbf{P}_n = (1/\eta_n)(\mathbf{U}_n^k\mathbf{B}_{(n)} + \mathbf{E}_n - \mathbf{X}_{(n)} - (\mathbf{Y}_n/\mu_n))\mathbf{B}_{(n)}^T$. The above problem can be solved by applying the SVD thresholding operator to $\mathbf{M}_n = \mathbf{U}_n^k - \mathbf{P}_n$. Take SVD for $\mathbf{M}_n = \mathbf{W}_n\Sigma_n\mathbf{V}_n^T$, then the new iteration is given by

$$\mathbf{U}_n^{k+1} = \mathbf{W}_n\text{soft}(\Sigma_n, \eta_n\mu_n)\mathbf{V}_n^T \tag{21}$$

where $\text{soft}(\Sigma, \sigma) = \max\{0, (\Sigma)_{ii} - 1/\sigma\}$ is the soft thresholding operator for a diagonal matrix [7].

Now, we consider solving dictionary learning for SC problem (13). Using tensorial matricization, problem (13) can be equivalently written in terms of matrices as follows:

$$\min_{\mathbf{D},\mathbf{A}} \frac{\lambda}{2}\|\mathbf{E}_{(N)}\|_F^2 \quad \text{s.t. } \mathbf{X}_{(N)} = \mathbf{DAC}^T + \mathbf{E}_{(N)} \quad \|\mathbf{A}\|_0 = R \tag{22}$$

where $\mathbf{C} = (\mathbf{U}_{N-1} \otimes \cdots \otimes \mathbf{U}_1)$. Problem (22) can be solved by using a two-phase BCD approach. In the first phase, we

Algorithm 3 Solving Problem (13) by BCD

Input: matrices: $\mathbf{X}_{(N)}$ and $\mathbf{C}$
Output: dictionary $\mathbf{D}$ and sparse representation matrix $\mathbf{A}$
1: initialize the dictionary $\mathbf{D}$ with a random strategy.
2: **while** reach maximum iterations **do**
3: sparse representation $\mathbf{A} \leftarrow$ solve the problem (22) with fixed $\mathbf{D}$;
4: dictionary $\mathbf{D} \leftarrow$ solve the problem (23) with K-SVD strategy;
5: **end while**

optimize $\mathbf{A}$ by fixing $\mathbf{D}$; in the second phase, we update $\mathbf{D}$ by fixing $\mathbf{A}$. The process repeats until some stop criterion is satisfied.

When the dictionary $\mathbf{D}$ is given, the sparse representation $\mathbf{A}$ can be obtained by solving (22) with fixed $\mathbf{D}$.

The resulting problem becomes a 2-D SC problem, which can be solved by the 2-D-OMP [16].

Remark 5: The 2-D-OMP is in fact equivalent to 1-D-OMP, with exactly the same results. However, the memory usage of the 2-D-OMP is much lower than the 1-D-OMP. Note that the 2-D-OMP only needs the memory usage of size $I_N \times m + (I_1 \times I_2 \cdots \times I_{(N-1)})^2$. However, the 1-D-OMP needs $(I_1 \times I_2 \cdots \times I_{(N)}) \times (m \times I_1 \times I_2 \cdots \times I_{(N-1)})$.

Given the sparse coefficient matrix $\mathbf{A}$, we define $\mathbf{F} = \mathbf{A}\mathbf{C}^T$, then the dictionary $\mathbf{D}$ can be updated by

$$\min_{\mathbf{D}} \frac{\lambda}{2}\|\mathbf{E}_{(N)}\|_F^2 \quad \text{s.t. } \mathbf{X}_{(N)} = \mathbf{D}\mathbf{F} + \mathbf{E}_{(N)}. \tag{23}$$

Actually, (23) is a least-square problem. As it is large scale, a direct closed-form solution will cost too much overhead. Here, we propose an iterative way alternatively on the columns of $\mathbf{D}$ based on the spare structures in $\mathbf{F}$. Let us consider only one column $\mathbf{d}_j$ in the dictionary and its corresponding coefficients, the jth row in $\mathbf{F}$, denoted by $\mathbf{f}^j$. Equation (23) can be rewritten as

$$\begin{aligned}\|\mathbf{E}_{(N)}\|_F^2 &= \left\|\mathbf{X}_{(N)} - \sum_{j=1}^{m} \mathbf{d}_j \mathbf{f}^j\right\|_F^2 \\ &= \left\|(\mathbf{X}_{(N)} - \sum_{j\neq l} \mathbf{d}_j \mathbf{f}^j) - \mathbf{d}_l \mathbf{f}^l\right\|_F^2 \\ &= \left\|\mathbf{E}_{(N)}^l - \mathbf{d}_l \mathbf{f}^l\right\|_F^2. \end{aligned} \tag{24}$$

We have decomposed the multiplication $\mathbf{DF}$ into the sum of m rank-1 matrices, where m is the number of atoms in $\mathbf{D}$. The matrix $\mathbf{E}_{(N)}^l$ represents the error for all the m examples when the lth atom is removed. Indeed, we are using K-SVD strategy [1] to update each atom $\mathbf{d}_l$ and $\mathbf{f}^l$ $(1 \le l \le m)$ by fixing all the other terms. However, the sparsity constraint is enforced in such an update strategy.

The general process of dictionary learning for SC is shown in Algorithm 3.

Algorithm 4 Subspace Clustering by TLRRSC

Input: structured data: tensor $\mathcal{X}$, number of subspaces K
Output: : the cluster indicator vector $\mathbf{l}$ with terms of all samples
1: **while** reach maximum iterations or converge to stop **do**
2: lowest-rank representation $\mathbf{U}_n(n = 1, 2, \ldots, N-1) \leftarrow$ solve the problem (12)
3: sparse representation $\mathbf{A}$ and the dictionary $\mathbf{D} \leftarrow$ solve the problem (13)
4: **end while**
5: $\mathbf{Z}^s \leftarrow \mathbf{U}_{N-1} \otimes \mathbf{U}_{N-2} \otimes \cdots \otimes \mathbf{U}_1$
6: $\mathbf{l} \leftarrow$ Normalized Cuts$(|\mathbf{Z}^s| + |\mathbf{Z}^{sT}| + |\mathbf{A}^T\mathbf{A}|)$

E. Complete Subspace Clustering Algorithm

After iteratively solving two subproblems (12) and (13), we finally obtain the low-rank and sparse representations given by $\mathbf{U}_i(i = 1, 2, \ldots, N-1))$ and $\mathbf{A}$ for the data $\mathcal{X}$. We create a similarity matrix on the spatial spaces $\mathbf{Z}^s = \mathbf{U}_{N-1} \otimes \mathbf{U}_{N-2} \otimes \cdots \otimes U_1$. The affinity matrix is then defined by $|\mathbf{Z}^s| + |\mathbf{Z}^{sT}| + |\mathbf{A}^T\mathbf{A}|$.[1] Each element of the affinity matrix is the joint similarity between a pair of mode-N vectorial samples across all the $N-1$ spatial modes/directions and the Nth feature mode. Finally, we employ the normalized cuts clustering method [39] to divide the samples into their respective subspaces. Algorithm 4 outlines the whole subspace clustering method of TLRRSC.

F. Computational Complexity

The TLRRSC algorithm composes of two iterative updating parameter steps followed by an normalized cut on an affinity matrix. Assuming the iteration time is t, $I_N = N$, low rank value is r, and $I_n = d$, $(1 \le n \le N-1)$.

In the process of updating lowest rank representation $\mathbf{U}_n(n = 1, 2, \ldots, N-1)$, the complexity of computing $\mathbf{DA}$ is $O(N^2 d^{N-1})$, and the computational costs regarding updating $\mathbf{B}_n$, $\mathbf{U}_n$, and $\mathbf{Y_n}$ to solve problem (15) are $O(N^2 d^{N-1})$, $O(Nrd^2)$, and $O(N^2 d^{N-1})$, respectively. Accordingly, the computational complexity of $\mathbf{U}_n(n = 1, 2, \ldots, N-1)$ is approximately $\mathrm{O}(N^2 d^{N-1}) + \mathrm{O}(Nrd^2)$.

In the dictionary learning process, the costs of updating $\mathbf{A}$ and $\mathbf{D}$ are $O(md^{N-1})$ and $O(N(k^2 m + 2Nd^{N-1}))$, respectively, where k is the sparsity value in the K-SVD algorithm.

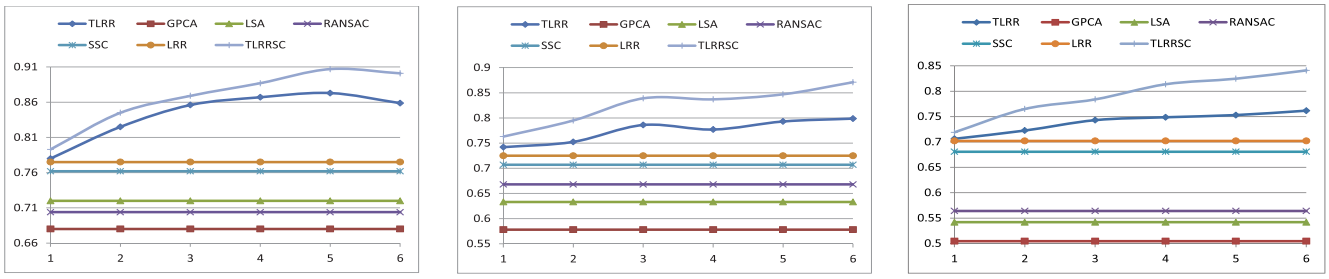
After obtaining the final optimal $\mathbf{U}_n(n = 1, 2, \ldots, N-1)$, $\mathbf{A}$, and $\mathbf{D}$, the time complexity of creating an affinity matrix is $O(d^{N-1})$. With the affinity matrix, the normalized cut can be solved with a complexity of $O(N \log N + d^{2(N-1)})$.

With the above analysis, the total complexity of TLRRSC is

$$\begin{aligned} O(N^2 d^{N-1}) + O(Nrd^2) + O(md^{N-1}) + O(N(k^2 m + 2Nd^{N-1})) \\ + O(d^{N-1}) + O(N \log N + d^{2(N-1)}). \end{aligned} \tag{25}$$

As $k, r \ll d$, therefore, the approximate complexity is $O((N^2 + m)d^{N-1}) + O(N \log N + d^{2(N-1)})$.

[1] To maintain scaling, we may use $|(A^T A)^{1/2}|$, but the experiments show that the simple definition $|\mathbf{Z}^s| + |\mathbf{Z}^{sT}| + |\mathbf{A}^T\mathbf{A}|$ works well. Other possible choices are $|Z^T Z| + |A^T A|$ and $(|Z| + |Z^T|) \odot |A^T A|$.



(a) (b) (c)

Fig. 3. Accuracy comparisons with respect to different orders of a tensor. (a) High-order tensorial data with 5-D feature spaces. (b) High-order tensorial data with 10-D feature spaces. (c) High-order tensorial data with 20-D feature spaces.

V. Experimental Results

In this section, we present a set of experimental results on some synthetic and real data sets with multidimensional spatial structures. The intention of these experiments is to demonstrate our new method TLRRSC's superiority over the state-of-the-art subspace clustering methods in prediction accuracy, computation complexity, memory usage, and noise robustness. To analyze the clustering performance, the Hungarian algorithm [23] is applied to measure the accuracy by comparing the predicted clustering results with the ground truth.

A. Baseline Methods

Because our proposed method is closely related to LRR and SSC, we choose LRR, TLRR, and SSC methods as the baselines. Moreover, some previous subspace clustering methods are also considered.

1) LRR: The LRR methods have been successfully applied to subspace clustering for even highly corrupted data, outliers, or missing entries. In this paper, we consider an LRR method introduced in [29], which is based on minimizing

$$\min_{\mathbf{Z}} \|\mathbf{Z}\|_* + \frac{\lambda}{2}\|\mathbf{E}\|_{2,1} \quad \text{s.t. } \mathbf{X} = \mathbf{XZ} + \mathbf{E}. \tag{26}$$

However, this method conducts subspace clustering on a rearranged matrix, ignoring data spatial correlations. Thus, the entries of affinity matrix $|\mathbf{Z}| + |\mathbf{Z}^T|$ denote the pairwise similarity in the low-dimensional feature spaces.

2) TLRR: As an improvement over LRR, TLRR finds an LRR for an input tensor by exploring factors along each spatial dimension/mode, which aims to solve the problem (8). An affinity matrix built for spectral clustering records the pairwise similarity along all the spatial modes.

3) SSC: SSC has a similar formulation to LRR, except for the employment of the l_1-norm $\|\mathbf{Z}\|_1$ in favor of a sparse representation. For fair comparisons, we implement two versions of SSC, i.e., SSC_1 is a l_1-norm version [$q = 2$ and $b = 1$ in (1)] and $\text{SSC}_{2,1}$ is a $l_{2,1}$-norm version [$q = 2, 1$ and $b = 1$ in (1)]. SSC denotes SSC_1 if not specified in the following experiments.

4) Some Other Methods: We also consider some previous subspace clustering methods for comparison, including GPCA [45], local subspace analysis (LSA) [47], and RANSAC [17].

In the following experiments, the parameter setting is as follows: a balance parameter $\lambda = 0.1$, a penalty parameter $\mu_n = 10^{-6}$, and the convergence threshold $\varepsilon = 10^{-8}$.

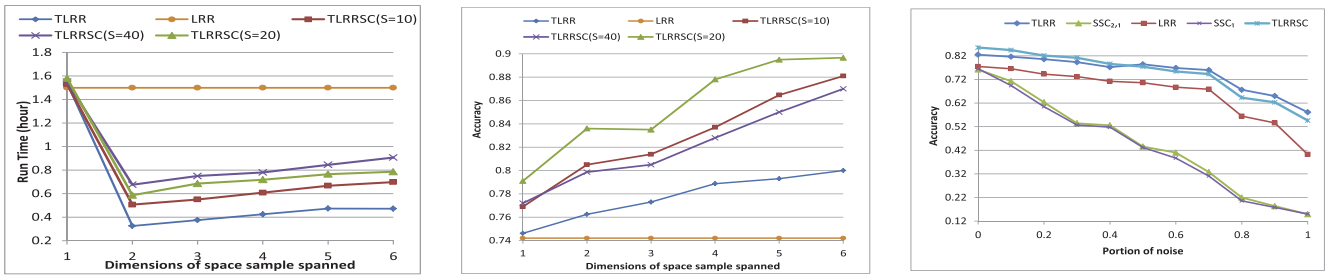
B. Results on Synthetic Data Sets

In this section, we evaluate TLRRSC against the state-of-the-art subspace clustering methods on synthetic data sets. We use three synthetic data sets containing three subspaces, each of which is formed by N_k samples of d dimension feature, respectively, where $d \in \{5, 10, 20\}$, $k \in \{1, 2, 3\}$, $N_1 = 30$, $N_2 = 24$, and $N_3 = 10$. The generation process is as follows.

1) Select three cluster center points $c_i \in \mathbb{R}^d$ for the above subspaces, respectively, which are far from each other.
2) Generate a matrix $\mathbf{C}^k \in \mathbb{R}^{d \times N_k}$, each column of which is drawn from a Gaussian distribution $\mathcal{N}(\cdot|c_k, \Sigma^k)$, where $\Sigma^k \in \mathbb{R}^{d \times d}$ is a diagonal matrix, such that the kth element is 0.01 and others are 1s. This setting guarantees that each cluster lies roughly in a $d - 1$ dimension subspace.
3) Combine samples in each subspace to form an entire data set $\mathbf{X} = \cup \mathbf{C}^k$.

1) Performance With High-Order Tensorial Data: To show the TLRRSC's advantage of handling high-order tensorial data over other baseline methods, we create five other synthetic data sets from the above data $\mathbf{X}$ by reshaping it into a higher j-mode tensor ($3 \le j \le 7$). Since all other baseline methods except TLRR conduct subspace clustering on an input matrix, i.e., a two-mode tensor, we use $\mathbf{X}$ on all these baseline methods for the purpose of fair comparisons. Fig. 3 shows the results on all the baseline methods with different dimensions of feature spaces.

As we can see, TLRRSC and TLRR perform much better than other methods in the higher mode of tensor. This observation suggests that incorporating data structure information into subspace clustering can boost clustering performance, while the performance of other methods always stays still, because these methods treat each sample independently, ignoring inherent data spatial structure information. TLRRSC is always superior to TLRR, which demonstrates that incorporating



(a) (b) (c)

Fig. 4. Time and accuracy comparison on the synthetic data sets with 20-D feature spaces. (a) Run time comparisons with respect to different orders of a tensor. (b) Accuracy comparisons with respect to different orders of a tensor. (c) Accuracy comparisons with respect to different potions of noisy samples.

feature similarity can further boost the clustering performance. Another interesting observation is that the performance gap between TLRR and TLRRSC is enlarged with the growth of feature dimensions, which suggests that seeking the inherent sparse representation in the high-dimensional feature space does help improve the clustering performance by filtering redundant information.

To compare the accuracy and running time among LRR-based algorithms, we create a matrix $\widetilde{\mathbf{X}} \in \mathbb{R}^{200\times 640}$ containing three subspaces, each of which is formed by containing three subspaces, each of which is formed by N_k samples of 200-D features, where $k \in \{1, 2, 3\}$, $N_1 = 300$, $N_2 = 240$, and $N_3 = 100$. The generation process is similar to the construction of $\mathbf{X}$. Then, we create five other synthetic data sets from the new data matrix $\widetilde{\mathbf{X}}$ by reshaping it into a higher j-mode tensor ($3 \leq j \leq 7$). Fig. 4(a) and (b) compares the accuracy and running time among LRR-based algorithms on the data set with 200-D feature spaces. To investigate the proposed algorithm's performance with different sparsity values S used in the dictionary learning along the feature direction, we use three sparsity values $S \in \{10, 20, 40\}$. We observe that as the order of tensor increases, the running time of TLRR and TLRRSC is significantly reduced compared with LRR (as shown in Fig. 4), and the clustering accuracy of TLRR and TLRRSC is superior to its vectorized counterpart LRR (as shown in Fig. 4). These observations suggest that the structural information has an important impact on speeding up the subspace clustering process and improving clustering accuracy.

As TLRRSC needs extra time to solve the sparse representation along the feature mode, the time cost of TLRRSC is a little more expensive than TLRR. Moreover, when the sparsity value is 20, TLRRSC performs best compared with other sparsity values, which suggests that our method can accurately cluster data with a small sparsity value. To sum up, our new method TLRRSC can achieve better performance with a comparable time cost in the higher mode of tensor.

2) Performance With Different Portions of Noisy Samples: Consider the cases where there exist noisy samples in the data. We randomly choose 0%, 10%, ..., 100% of the samples of the above $\mathbf{C}^k$, respectively, and add Gaussian noises $\mathcal{N}(\cdot|c_k, 0.3\Sigma^k)$ to these samples. Then, a noisy data set $\mathbf{X}'$ is generated by combining the corrupted $\mathbf{C}^k$ to one. The performances on $\text{SSC}_{2,1}$, SSC_1, LRR, TLRR, and TLRRSC are shown in Fig. 4. Obviously, LRR-based subspace clustering methods, such as TLRRSC, TLRR, and LRR, maintain their accuracies even though 70% of samples are corrupted by noise. Moreover, three LRR-based methods significantly outperform both $\text{SSC}_{2,1}$ and SSC_1, as shown in Fig. 4, which suggests that LRR is good at handling noisy data, while SSC is not, because it solves the columns of the representation matrix independently. For low-rank-based methods, LRR method is inferior to the structure-based TLRR and TLRRSC. This is mainly because TLRR and TLRRSC integrate data spatial information into subspace clustering, resulting in a good performance even when 90% of data are corrupted. Another interesting result is that TLRRSC is marginally superior to TLRR when the noise rate is <50%, but its performance becomes inferior to TLRR, as the noise rate continually increases to 100%. This again proves that SC is sensitive to noise. Although TLRRSC maintains a good performance by exploring the spatial correlations among samples, sparse representation along the feature spaces induces more noises as the noise portion increases. Therefore, the clustering performance depresses with noisy feature similarities integrated in the affinity matrix.

3) Performance With Dictionary Learning for Sparse Coding: Like LRR and SSC, our model TLRRSC considers sparsity regarding low-dimensional representation on the feature space. In contrast to LRR and SSC, using the input data as a dictionary, TLRRSC learns a dictionary and its corresponding sparse representation. In this section, we compare the performances of different sparse strategies.

First of all, we create a matrix $\widehat{\mathbf{X}} \in \mathbb{R}^{30\times 64}$ containing three subspaces, each of which is formed by N_k samples of 30 dimensions, where $k \in \{1, 2, 3\}$, $N_1 = 30$, $N_2 = 24$, and $N_3 = 10$. The generation process is similar to the construction of $\mathbf{X}$, except each cluster center points $c_k \in \mathbb{R}^{30}$ and the last 20 diagonal elements in $\Sigma^k \in \mathbb{R}^{30\times 30}$ are 0.01 and others are 1s. This setting guarantees that each cluster lies roughly in a 10-D subspace. Fig. 5 shows the evaluated mean of each band on the reconstructed data matrix denoted by the product of a dictionary and its sparse representation. Obviously, the evaluated mean of our model TLRRSC is the closest to the true value, compared with LRR and SSC. This suggests that the TLRRSC finds a better dictionary to fit the data, instead of a fixed dictionary $\widehat{\mathbf{X}}$ in the other two methods. Moreover, Fig. 6 shows the sparse representations obtained for $\widehat{\mathbf{X}}$. In our algorithm, we learn a dictionary $\mathbf{D}^{30\times 200}$ in the feature space

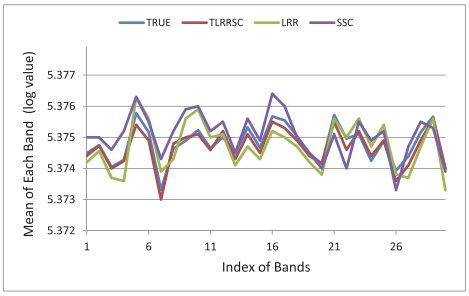


Fig. 5. Comparison between the true band mean and evaluated one on the synthetic data.

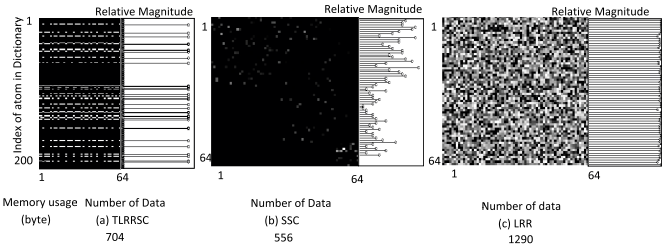
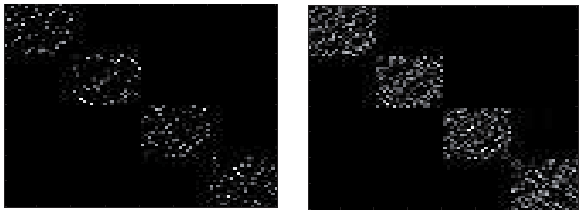


Fig. 6. Sparse coefficients, their relative magnitudes, and memory usage of TLRRSC, LRR, and SSC on the synthetic data.

with 200 atoms, while the other two models use given data as a dictionary, the corresponding sparse representation under the dictionary for baselines is shown in the black blocks of Fig. 6. Each line in the white block statistics of the total number of each atom used in the new sparse representation of the given data set (i.e., the relative magnitude). For LRR algorithm, each atom in the dictionary is activated with almost the same relative magnitude, whereas in Fig. 6(b), far fewer atoms are activated with a higher magnitudes. This is mainly because LRR uses a holistic sparsity defined by low rank, where in SSC, sparsity is represented individually. The original high-dimensional data matrix $\widehat{\mathbf{X}}$ needs 1290-B memory spaces, while all sparsity involved methods reduce space costs to some extent, as shown in Fig. 6. In Fig. 6(a), our sparse representation only activates a few atoms with almost the same high magnitude. We can clearly see that the number of lines in the white part of Fig. 6(a) is fewer than that of Fig. 6(b). Although the memory usage of our model TLRRSC is 26% more than SSC, our sparse representation activates a far fewer number of atoms [Fig. 6(a)] than for SSC [Fig. 6(b)].

4) Performance Comparisons With Other LRR + SC Subspace Clustering Algorithm: We compare our algorithm's performance with another state-of-the-art algorithm LRSSC in [28], which also takes the advantage of SC and LRR. LRSSC minimizes a weighted sum of nuclear norm and vector 1-norm of the representation matrix simultaneously, so as to preserve the properties of interclass separation and intraclass connectivity at the same time. Therefore, it works well in the matrices where data distribution is skewed and subspaces are not independent. Unlike LRSSC explicitly satisfies LRR and SC property simultaneously, our model updates the parameters for LRR and SC alternatively, and our model focuses on multidimensional data with a high-dimensional feature space.



(a) (b)

Fig. 7. Subspace clustering algorithm performance comparisons. (a) LRSSC. (a) TLRRS.

In the experiments, we randomly generate four disjoint subspaces of dimension 10 from $\mathbb{R}^{50}$, each sampled 20 data points. The 50 unit length random samples are drawn from each subspace and we concatenate into a $\mathbb{R}^{50\times 80}$ data matrix. The clustering results are shown in Fig. 7. As we can see, our algorithm performs better than LRSSC, and this is maybe because the alterative update LRR parameters and SC parameters in the iterations can help find better solution for each setting.

C. Results on Real Data Sets

We evaluate our model on a clean data set called the Indian Pines [24] and a corrupted data set called Pavia University database [46].

The Indian Pines data set is gathered by airborne visible infrared imaging spectrometer sensor over the Indian Pines test site in north-western Indiana, and consists of 145×145 pixels and 224 spectral reflectance bands in the wavelength range 0.4–2.5 μm. The whole data set is formed by 16 different classes having an available ground truth. In our experiments, 24 bands covering the region of water absorption are discarded. The task is to group pixels into clusters according to their spectral reflectance band information.

The Pavia University database is acquired by the reflective optics system imaging spectrometer sensor with a geometric resolution of 1.3 m, during a flight campaign over Pavia, northern Italy. Pavia University consists of 610×340 pixels, each of which has 103 spectral bands covering 0.43–0.86 μm. The data set contains nine different classes with available ground truths. We examine the noise robustness of the proposed model by adding Gaussian white noises with intensities ranging from 20 to 60 with a step of 20 to the whole database.

1) Subspace Clustering Performance: In this section, we show TLRRSC's performance in subspace clustering with the subspace number given. Table II shows the results of all baseline methods on both data sets. Clearly, our method TLRRSC outperforms the other six baselines on this data set. The advantage of TLRRSC mainly comes from its ability to incorporate 2-D data structure information and 200-D bands information into the LRR and sparse representation.

Besides, the efficiency (in terms of running time) of TLRRSC is comparable with TLRR, GPCA, and RANSAC methods. TLRR costs more computational time, because its optimization procedure needs more iterations than GPCA and RANSAC to converge. The results regarding time cost on TLRR and LRR are consistent with Remark 1, which shows

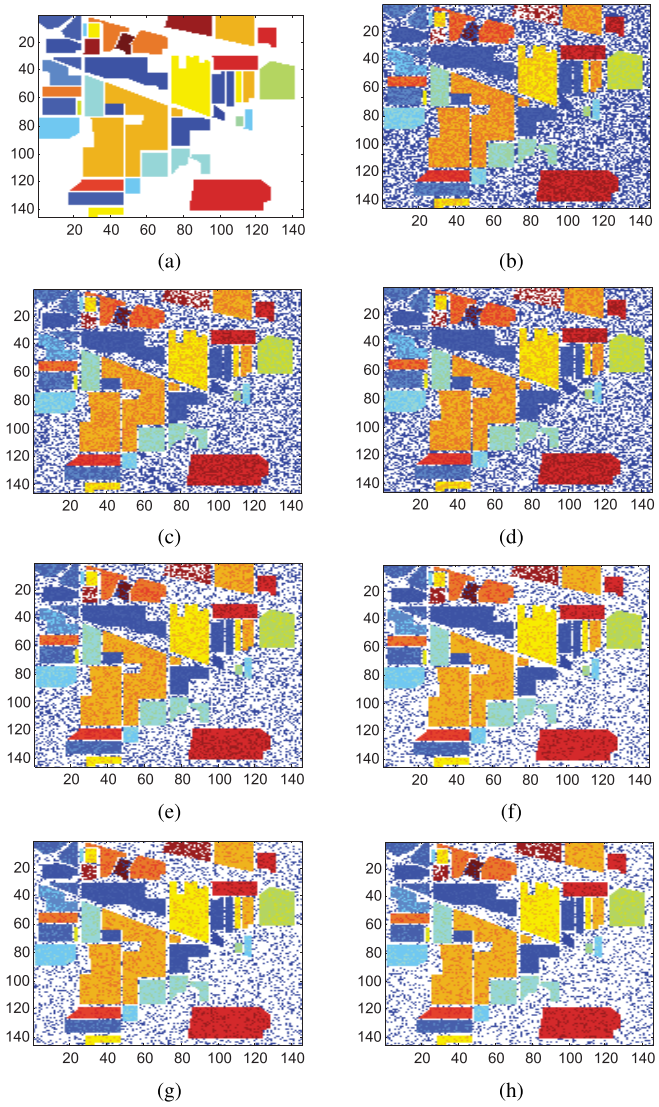


Fig. 8. Subspace clustering results on Indian Pines, each color represents a class. (a) Ground truth. (b) GPCA. (c) LSA. (d) RANSAC. (e) SSC. (f) LRR. (g) TLRR. (h) TLRRSC.

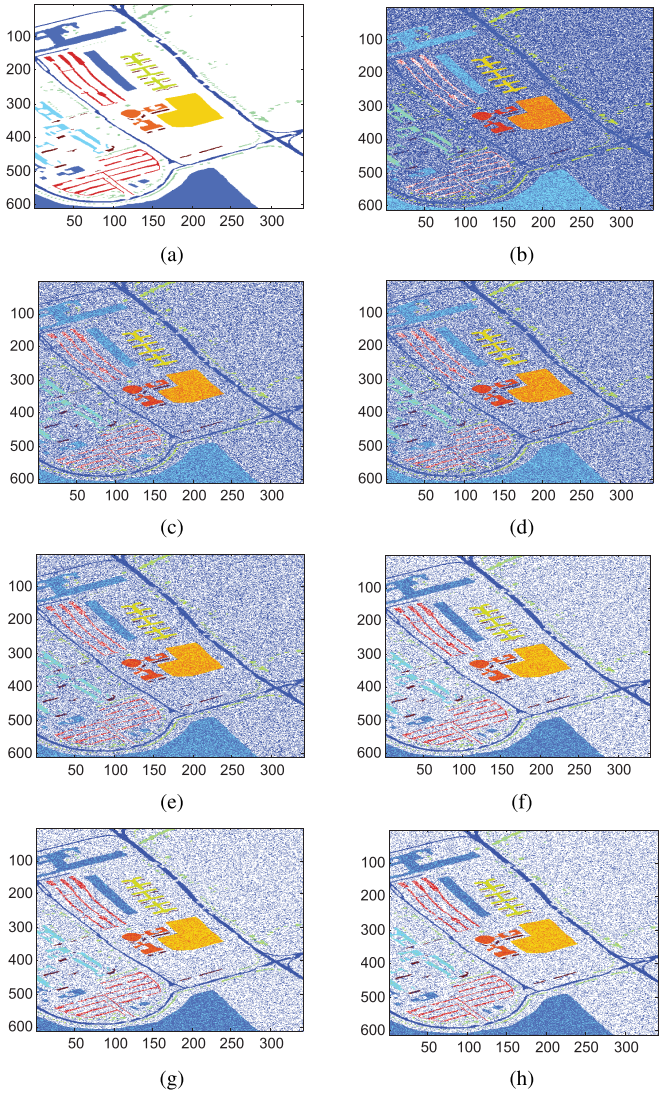


Fig. 9. Subspace clustering results on Pavia University (noise intensity = 60), each color represents a class. (a) Ground truth. (b) GPCA. (c) LSA. (d) RANSAC. (e) SSC. (f) LRR. (g) TLRR. (h) TLRRSC.

that TLRR significantly reduces time cost by exploiting the Kronecker structure along each space dimension. Although GPCA and RANSAC are faster than LRR and TLRR, their accuracy is much lower than those of LRR and TLRR. Even though TLRRSC uses 22 more minutes than TLRR for a dictionary-learning task in the feature space, its overall performance is better than TLRR.

When data are corrupted, the performance of SSC is inferior to all LRR-based methods, which shows that sparse representation is not good at handling corrupted data, such as LRR. Although our model employs a sparse representation on the feature space, our model TLRRSC still performs best among all the methods on the corrupted data set. This is because TLRRSC explores data spatial correlation information with an LRR, which guarantees accurately clustering data into different subgroups. Figs. 8 and 9 show the subspace clustering results on both data sets. In Figs. 8 and 9, each cluster is represented by a particular color. Obviously, we can see many blue points scattered in Figs. 8 and 9, which originally belonging to other classes that are wrongly classified into the blue class. Similarly, there are a few other colors scattered in the green class in Fig. 8 and the orange class in Fig. 9. Accordingly, it is easy to see that the clustering results on TLRRSC are closest to the ground truths.

2) Choosing the Parameter λ: The parameter $\lambda > 0$ is used to balance the effects of the two parts in problem (11). Generally speaking, the choice of this parameter depends on the prior knowledge of the error level of data. When the errors are slight, a relatively larger λ should be used, while when the errors are heavy, we should set a smaller value. Fig. 10 (blue curve) is the evaluation results on the Indian Pines data set. While λ ranges from 0.04 to 0.2, the clustering accuracy slightly varies from 80.34% to 81.98%. This phenomenon is mainly because TLRRSC employs LRR representation to explore data structure information. It has been proved that LRR works well on clean data (the Indian Pines is a clean data set), and there is an invariance in LRR

TABLE II
SUBSPACE CLUSTERING RESULTS ON THE REAL DATA SETS

			Subspace clustering accuracy(%)						
			GPCA	LSA	RANSAC	SSC	LRR	TLRR	TLRRSC
Indianpines		Mean	47.6	58.3	53.2	69.8	77.6	78.6	**80.5**
		Std.	10.45	10.56	9.98	7.02	5.45	4.67	**4.08**
		Max	70.9	81.5	78.3	80.7	85.4	89.7	**92.4**
		Time (min.)	6.87	177.84	**5.90**	745.73	380.07	51.23	73.48
Pavia University	Intensity=20	Mean	30.2	51.65	46.7	61.9	72.3	74.6	**76.8**
		Std.	12.83	10.69	8.76	8.95	5.38	4.54	**3.97**
		Max	62.5	70.1	73.8	80.6	85.7	87.6	**91.2**
		Time (hr.)	1.84	27.31	**0.76**	95.17	41.02	7.69	11.05
	Intensity=40	Mean	25.68	48.7	44.2	57.7	69.1	70.4	**73.5**
		Std.	11.79	13.69	7.68	9.75	6.74	6.01	**3.08**
		Max	60.2	65.17	69.8	76.8	80.1	82.5	**87.6**
		Time (hr.)	1.69	28.76	**0.79**	96.12	40.87	7.15	10.98
	Intensity=60	Mean	23.2	44.12	42.07	52.78	66.8	67.8	**69.8**
		Std.	10.68	15.09	8.67	8.99	4.96	4.02	**3.17**
		Max	54.2	60.2	63.8	71.6	78.7	79.3	**85.9**
		Time (hr.)	1.58	29.33	**0.97**	94.15	43.07	7.05	9.99

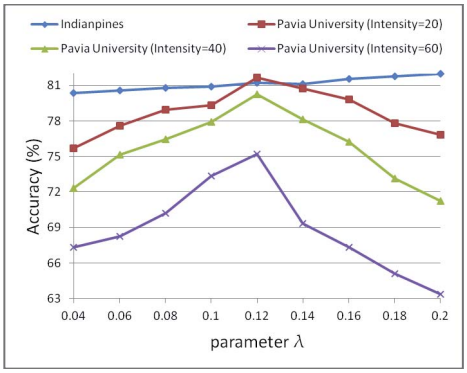


Fig. 10. Influences of the parameter λ of TLRRSC. These results are collected from the Indian Pines data set and the Pavia University data set.

that implies that it can be partially stable while λ varies (for the proof of this property, see [29, Th. 4.3]). Notice that TLRRSC is more sensitive to λ on the Pavia University data set than on the Indian Pines data set. This is because the samples in the Indian Pines data set are clean, whereas the Pavia University data set contains some corrupted information. The more heavily data are corrupted, and the performance of our new method is more influenced by the λ value.

3) Memory Usage With Respect to Different Sparsity Strategies: In TLRRSC, we learn a sparse representation on the feature mode (i.e., the third mode) through a dictionary-learning task. In this section, we compare the memory usage among TLRRSC, SSC, and LRR on the mode-3 matricization of the Indian Pine database and the Pavia University database. For an order-3 tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, the memory complexity of our models TLRRSC and SSC is $O(r \times (I_1 \times I_2))$ at most, where r is the maximum sparsity for each instance, while LRR requires $O(m \times (I_1 \times I_2))$, where m is the rank of the sparse representation. Usually $(m \gg r)$. As shown in the first row of Table III, SSC has the least memory cost, and TLRRSC takes the second place. The reason behind this phenomenon is that our dictionary-learning model based on both spatial information and feature relation involves more structured sparse representation than feature-based SSC model. However,

TABLE III
MEMORY USAGE COMPARISON ON REAL DATA SETS

	Memory Usage for Sparse Representation (MB)			
	Original	TLRRSC	SSC	LRR
Indianpines	4.20	1.54	**1.21**	2.48
Pavia University	21.36	7.83	**6.18**	11.87

TLRRSC has an advantage over SSC in accuracy and running time, as shown in Table II. Accordingly, our model maintains good performance with a comparable memory cost. The second row in Table III shows the memory cost of TLRRSC, LRR, and SSC on Pavia University. The memory usage of SSC is the lowest, which is consistent with the result on Indian Pines. However, the performance of SSC depresses on the corrupted data set, as shown in Table II, while LRR is very effective in handling noise. Moreover, our model's memory usage is comparable. Therefore, we assert that TLRRSC is a noise robust method with low memory usage.

VI. CONCLUSION

In this paper, we propose TLRR and SC for subspace clustering in this paper. Unlike the existing subspace clustering methods work on an unfolded matrix, TLRRSC builds a model on data original structure form (i.e., tensor) and explores data similarities along all spatial dimensions and feature dimensions. On the synthetic higher mode tensorial data sets, we show that our model considering the data structure maintains a good performance. Moreover, the experimental results with different noise rates show that our model maintains a good performance on highly corrupted data. On the real-world data set, our method shows promising results, with low computation gains and memory usage. Moreover, our model is robust to noises, and capable of recovering corrupted data.

REFERENCES

[1] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.
[2] B. W. Bader and T. G. Kolda, "Efficient MATLAB computations with sparse and factored tensors," *SIAM J. Sci. Comput.*, vol. 30, no. 1, pp. 205–231, Dec. 2007.

[3] R. Basri and D. W. Jacobs, "Lambertian reflectance and linear subspaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 2, pp. 218–233, Feb. 2003.

[4] M. Blondel, K. Seki, and K. Uehara, "Block coordinate descent algorithms for large-scale sparse multiclass classification," *Mach. Learn.*, vol. 93, no. 1, pp. 31–52, 2013. [Online]. Available: http://dx.doi.org/10.1007/s10994-013-5367-2

[5] T. Blumensath and M. E. Davies, "Normalized iterative hard thresholding: Guaranteed stability and performance," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 2, pp. 298–309, Apr. 2010.

[6] M. Bouguessa, S. Wang, and Q. Jiang, "A K-means-based algorithm for projective clustering," in *Proc. ICPR*, vol. 1. 2006, pp. 888–891.

[7] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM J. Optim.*, vol. 20, no. 4, pp. 1956–1982, 2010. [Online]. Available: http://dx.doi.org/10.1137/080738970

[8] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Commun. ACM*, vol. 55, no. 6, pp. 111–119, 2012. [Online]. Available: http://doi.acm.org/10.1145/2184319.2184343

[9] A. Jalali, Y. Chen, S. Sanghavi, and H. Xu, "Clustering partially observed graphs via convex optimization," in *Proc. 28th ICML*, 2011, pp. 1001–1008.

[10] C. Ding, D. Zhou, X. He, and H. Zha, "R_1-PCA: Rotational invariant L_1-norm principal component analysis for robust subspace factorization," in *Proc. 23rd ICML*, 2006, pp. 281–288.

[11] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing: I. Motivation and construction," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Jan. 2010, pp. 1–5.

[12] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization," in *Proc. Nat. Acad. Sci. USA*, vol. 100, no. 5, pp. 2197–2202, 2003.

[13] M. Elad, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 54, no. 12, pp. 3736–3745, Dec. 2006.

[14] E. Elhamifar and R. Vidal, "Sparse subspace clustering," in *Proc. IEEE Conf. CVPR*, Jun. 2009, pp. 2790–2797.

[15] K. Engan, S. O. Aase, and J. H. Husoy, "Method of optimal directions for frame design," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 1. Mar. 1999, pp. 2443–2446.

[16] Y. Fang, J. Wu, and B. Huang, "2D sparse signal recovery via 2D orthogonal matching pursuit," *Sci. China Inf. Sci.*, vol. 55, no. 4, pp. 889–897, 2012.

[17] M. A. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981. [Online]. Available: http://doi.acm.org/10.1145/358669.358692

[18] S. Foucart, "Hard thresholding pursuit: An algorithm for compressive sensing," *SIAM J. Numer. Anal.*, vol. 49, no. 6, pp. 2543–2563, 2011.

[19] J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, and D. Kriegman, "Clustering appearances of objects under varying illumination conditions," in *Proc. IEEE Comput. Soc. Conf. CVPR*, vol. 1. Jun. 2003, pp. I-11–I-18.

[20] K. Kanatani, "Motion segmentation by subspace separation and model selection," in *Proc. 8th IEEE ICCV*, vol. 2. Jul. 2001, pp. 586–591.

[21] R. H. Keshavan, A. Montanari, and S. Oh, "Matrix completion from noisy entries," *J. Mach. Learn. Res.*, vol. 11, pp. 2057–2078, Mar. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1859920

[22] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.

[23] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, Mar. 1955.

[24] D. Landgrebe, "Multispectral data analysis: A signal theory perspective," School Elect. Eng., Purdue Univ., West Lafayette, IN, USA, Tech. Rep., 1998.

[25] G. Montavon, G. Orr, and K.-R. Müller Eds., *Neural Networks: Tricks of the Trade*. Germany: Springer, 1998.

[26] S. Li, "Non-negative sparse coding shrinkage for image denoising using normal inverse Gaussian density model," *Image Vis. Comput.*, vol. 26, no. 8, pp. 1137–1147, Aug. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.imavis.2007.12.006

[27] Z. Lin, R. Liu, and Z. Su, "Linearized alternating direction method with adaptive penalty for low-rank representation," in *Proc. NIPS*, 2011, pp. 612–620.

[28] Y.-X. Wang, H. Xu, and C. Leng, "Provable subspace clustering: When LRR meets SSC," in *Proc. NIPS*, 2013, pp. 64–72.

[29] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171–184, Jan. 2013.

[30] R. Liu, Z. Lin, and Z. Su, "Linearized alternating direction method with parallel splitting and adaptive penalty for separable convex programs in machine learning," in *Proc. ACML*, 2013, pp. 116–132.

[31] Y. Ma, H. Derksen, W. Hong, and J. Wright, "Segmentation of multivariate mixed data via lossy data coding and compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 9, pp. 1546–1562, Sep. 2007.

[32] M. Jaggi and M. Sulovský, "A simple algorithm for nuclear norm regularized problems," in *Proc. 27th ICML*, 2010, pp. 471–478. [Online]. Available: http://www.icml2010.org/papers/196.pdf

[33] D. Needell, J. A. Tropp, and R. Vershynin, "Greedy signal recovery review," in *Proc. 42nd Asilomar Conf. Signals, Syst. Comput.*, 2008, pp. 1048–1050.

[34] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Commun. ACM*, vol. 53, no. 12, pp. 93–100, Dec. 2010. [Online]. Available: http://doi.acm.org/10.1145/1859204.1859229

[35] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 2, pp. 607–609, 1996.

[36] G. Peyré, "Sparse modeling of textures," *J. Math. Imag. Vis.*, vol. 34, no. 1, pp. 17–31, May 2009. [Online]. Available: http://dx.doi.org/10.1007/s10851-008-0120-3

[37] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, 2007, pp. 759–766.

[38] S. R. Rao, A. Y. Yang, S. S. Sastry, and Y. Ma, "Robust algebraic segmentation of mixed rigid-body and planar motions from two views," *Int. J. Comput. Vis.*, vol. 88, no. 3, pp. 425–446, 2010. [Online]. Available: http://dx.doi.org/10.1007/s11263-009-0314-1

[39] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[40] J. Shlens, "A tutorial on principal component analysis," Syst. Neurobiol. Lab., Salk Inst. Biol. Stud., La Jolla, CA, USA, Tech. Rep., 2005.

[41] M. E. Tipping and C. M. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural Comput.*, vol. 11, no. 2, pp. 443–482, Feb. 1999. [Online]. Available: http://dx.doi.org/10.1162/089976699300016728

[42] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.

[43] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[44] R. Vidal, "Subspace clustering," *IEEE Signal Process. Mag.*, vol. 28, no. 2, pp. 52–68, Mar. 2011.

[45] R. Vidal, Y. Ma, and S. S. Sastry, "Generalized principal component analysis (GPCA)," in *Proc. IEEE Comput. Soc. Conf. CVPR*, vol. 1. Jun. 2003, pp. I-621–I-628.

[46] L. Wei, S. Li, M. Zhang, Y. Wu, S.-Z. Su, and R. Ji, "Spectral-spatial classification of hyperspectral imagery based on random forests," in *Proc. 5th Int. Conf. Internet Multimedia Comput. Service*, New York, NY, USA, 2013, pp. 163–168. [Online]. Available: http://doi.acm.org/10.1145/2499788.2499853

[47] J. Yan and M. Pollefeys, "A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate," in *Proc. 9th ECCV*, 2006, pp. 94–106.

[48] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 1794–1801.

[49] T. Zhang, A. Szlam, and G. Lerman, "Median K-flats for hybrid linear modeling with many outliers," in *Proc. IEEE 12th ICCV*, Sep./Oct. 2009, pp. 234–241.

[50] Y. Fu, J. Gao, D. Tien, and Z. Lin, "Tensor LRR based subspace clustering," in *Proc. IJCNN*, 2014, pp. 1877–1884.

[51] L. Jing, M. K. Ng, and T. Zeng, "Dictionary learning-based subspace structure identification in spectral clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 8, pp. 1188–1199, Aug. 2013.

[52] K. Tang, R. Liu, Z. Su, and J. Zhang, "Structure-constrained low-rank representation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2167–2179, Dec. 2014.

[53] Y. Deng, Q. Dai, R. Liu, Z. Zhang, and S. Hu, "Low-rank structure learning via nonconvex heuristic recovery," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 3, pp. 383–396, Mar. 2013.
[54] Y. Wang, Y. Jiang, Y. Wu, and Z.-H. Zhou, "Spectral clustering on multiple manifolds," *IEEE Trans. Neural Netw.*, vol. 22, no. 7, pp. 1149–1161, Jul. 2011.
[55] R. He, W.-S. Zheng, B.-G. Hu, and X.-W. Kong, "Two-stage nonnegative sparse representation for large-scale face recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 1, pp. 35–46, Jan. 2013.

Yifan Fu received the M.E. degree in software engineering from Northeast Normal University, Changchun, China, in 2009, and the Ph.D. degree in computer science from the University of Technology Sydney, Sydney, NSW, Australia, in 2013.

She has been a Research Associate with the School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW, Australia, since 2013. Her current research interests include machine learning and data mining, more specifically, active learning, ensemble methods, graph mining, and tensor decomposition.

Junbin Gao received the B.Sc. degree in computational mathematics from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1982, and the Ph.D. degree from the Dalian University of Technology, Dalian, China, in 1991.

He was an Associate Lecturer, a Lecturer, an Associate Professor, and a Professor with the Department of Mathematics, HUST, from 1982 to 2001. He was a Professor of Computer Science with the School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW, Australia. He was a Senior Lecturer and Lecturer in Computer Science with the University of New England, Armidale, NSW, Australia, from 2001 to 2005. He is currently a Professor of Big Data Analytics with the University of Sydney Business School, The University of Sydney, Sydney, NSW, Australia. His current research interests include machine learning, data analytics, Bayesian learning and inference, and image analysis.

David Tien received the bachelor's degree in computer science from the Chinese Academy of Sciences, Heiloingjiang University, Harbin, China, the master's degree in pure mathematics from Ohio State University, Columbus, OH, USA, and the Ph.D. degree in electrical engineering from The University of Sydney, Sydney, NSW, Australia.

He is currently teaching computer science with Charles Sturt University, Bathurst, NSW, Australia. His current research interests include image and signal processing, artificial intelligence, telecommunication coding theory, and biomedical engineering.

Dr. Tien serves as the Chairman of the IEEE NSW Computer Chapter.

Zhouchen Lin (M'00–SM'08) received the Ph.D. degree in applied mathematics from Peking University, Beijing, China, in 2000.

He was a Lead Researcher with the Visual Computing Group, Microsoft Research Asia, Beijing, before 2012. He is currently a Professor with the Key Laboratory of Machine Perception, School of Electronics Engineering and Computer Science, Peking University. His current research interests include computer vision, image processing, computer graphics, machine learning, pattern recognition, and numerical computation and optimization.

Prof. Lin is an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and the *International Journal of Computer Vision*, and an Area Chair of Computer Vision and Pattern Recognition (CVPR) 2014, International Conference on Computer Vision 2015, Neural Information Processing Systems 2015, AAAI 2016, CVPR 2016, and International Joint Conference on Artificial Intelligence 2016.

Xia Hong received the B.Sc. and M.Sc. degrees from the National University of Defense Technology, Changsha, China, in 1984 and 1987, respectively, and the Ph.D. degree from the University of Sheffield, Sheffield, U.K., in 1998, all in automatic control.

She was a Research Assistant with the Beijing Institute of Systems Engineering, Beijing, China, from 1987 to 1993. She was also a Research Fellow with the Department of Electronics and Computer Science, University of Southampton, Southampton, U.K., from 1997 to 2001. She is currently a Professor with the Department of Computer Science, School of Mathematical and Physical Sciences, University of Reading, Reading, U.K. She is actively engaged in research into nonlinear systems identification, data modeling, estimation and intelligent control, neural networks, pattern recognition, learning theory, and their applications. She has authored over 100 research papers, and co-authored a research book.

Dr. Hong received the Donald Julius Groen Prize by IMechE in 1999.