Information Sciences 612 (2022) 87–106

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A structure noise-aware tensor dictionary learning method for high-dimensional data clustering

Jing-Hua Yang [a], Chuan Chen [b,*], Hong-Ning Dai [c,*], Le-Le Fu [d], Zibin Zheng [b]

[a] The Faculty of Information Technology, Macau University of Science and Technology, Macau, China
[b] The School of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou, China
[c] The Department of Computer Science, Hong Kong Baptist University, Hong Kong, China
[d] The School of Systems Science and Engineering, Sun Yat-Sen University, Guangzhou, China

ARTICLE INFO

Article history:
Received 29 December 2021
Received in revised form 19 August 2022
Accepted 24 August 2022
Available online 27 August 2022

Keyword:
High-dimensional data clustering
Tensor dictionary learning
Structure noise
Tensor low-rank representation
Structural sparsity
Proximal alternating minimization

ABSTRACT

With the development of data acquisition technology, high-dimensional data clustering is an important yet challenging task in data mining. Despite advances achieved by current clustering methods, they can be further improved. First, many of them usually unfold the high-dimensional data into a large matrix, consequently resulting in destroying the intrinsic structural property. Second, some methods assume that the noise in the dataset conforms to a predefined distribution (e.g., the Gaussian or Laplacian distribution), which violates real-world applications and eventually decreases the clustering performance. To address these issues, in this paper, we propose a novel tensor dictionary learning method for clustering high-dimensional data with the coexistence of structure noise. We adopt tensors, the natural and powerful tools for the generalizations of vectors and matrices, to characterize high-dimensional data. Meanwhile, to depict the noise accurately, we decompose the observed data into clean data, structure noise, and Gaussian noise. Furthermore, we use low-rank tensor modeling to characterize the inherent correlations of clean data and adopt tensor dictionary learning to adaptively and accurately describe the structure noise instead of using the predefined distribution. We design the proximal alternating minimization algorithm to solve the proposed model with the theoretical convergence guarantee. Experimental results on both simulated and real datasets show that the proposed method outperforms the compared methods for high-dimensional data clustering.

1. Introduction

Data clustering is a basic and important topic in computer vision and machine learning [1–4]. The main goal is to group (or segment) the data with similar structures into clusters. A number of data clustering methods have emerged, such as spectral clustering-based methods [5,6], matrix decomposition-based clustering methods [7,8], and deep learning-based clustering methods [1,9]. With the growth of the amount of high-dimensional data, dimensionality reduction and feature extraction methods based on sparse and low-rank representations have been widely studied and applied in practical applications.

* Corresponding authors
E-mail addresses: yangjinghua110@126.com (J.-H. Yang), chenchuan@mail.sysu.edu.cn (C. Chen), hndai@ieee.org (H.-N. Dai), lawrencefzu@gmail.com (L.-L. Fu), zhzibin@mail.sysu.edu.cn (Z. Zheng).

https://doi.org/10.1016/j.ins.2022.08.081
0020-0255/© 2022 Elsevier Inc. All rights reserved.

The sparse/low-rank representation-based clustering methods [10–13] usually obtain the coefficient matrices first and then exploit the spectral clustering tool on the coefficient matrices to get the final clustering results. For instance, Elhamifar et al. [10] used the sparse optimization program to cluster data points lying in a low-dimensional subspace. Liu et al. [11] introduced the low-rank representation to segment the data into their respective subspaces. Moreover, many variants of these methods have also been studied. In particular, Brbic et al. [14] combined the advantages of both sparse and low-rank representations for subspace clustering. Lu et al. [15] studied robust subspace segmentation with the help of least square regression. Tang et al. [16] proposed the structure-constrained low-rank representation for disjoint subspace clustering. Although the above clustering methods derived by sparse and low-rank representations achieve good performance, the data representation matrices obtained by the original data are fixed, thereby being inflexible to various applications.

Owing to the promising performance in sparse representation and low-rank modeling, *dictionary learning* has received widespread attention in data clustering [6,17,18]. Dictionary learning assumes that a signal can be described as a linear combination of a few elements (aka atoms). There exists a number of dictionary learning-based methods for handling different noises, e.g., Gaussian noise [19–21], Laplacian noise [22,23], and mixed noise [24]. Furthermore, Zhou et al. [17] used the dictionary learning to adaptively learn the complex mixed noise distribution in real applications instead of using predefined distributions. The above methods are suitable for 1D data but are ineffective for the high-dimensional data. When dealing with 2D or 3D data, they require transforming the high-dimensional data into vectors, while this vectorization operator breaks the intrinsic structure of high-dimensional data and limits the performance of dictionary learning.

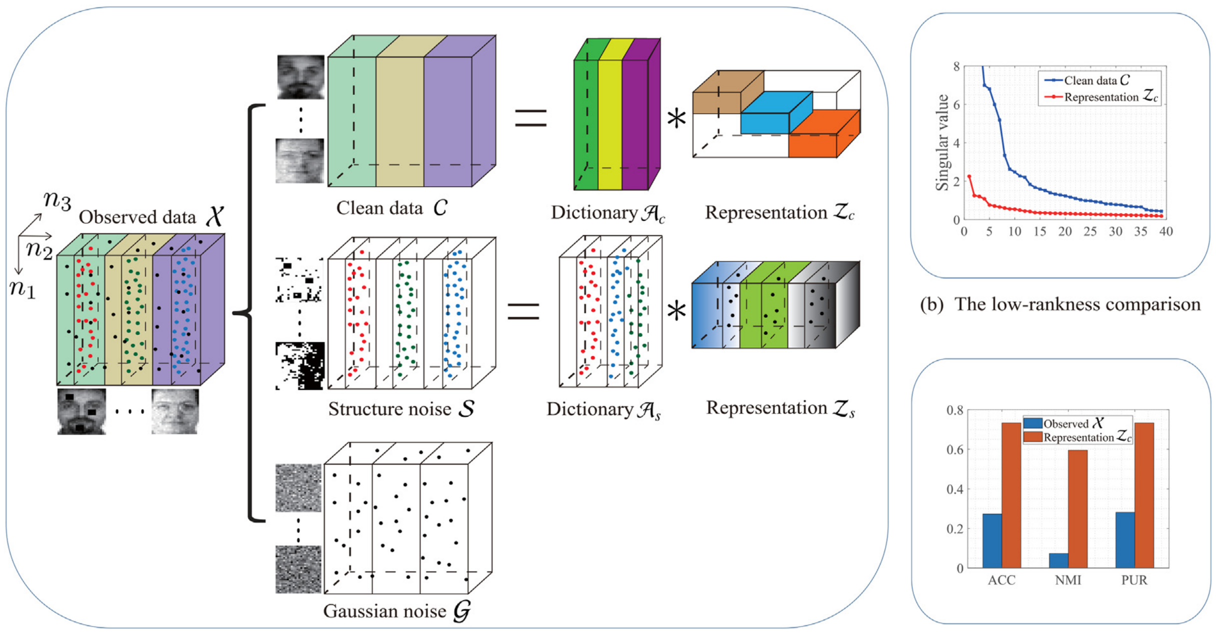
To deal with the high-dimensional data, based on different tensor decompositions, some tensor dictionary learning techniques were developed [25–27]. Despite merits in improving the representation ability of data, these tensor dictionary learning methods still have the following limitations. One main concern lies in the common assumption that the noise contained in the data follows the Gaussian distribution, thereby constraining it with the Frobenius norm. However, in practical applications, there often exists extremely complex noise while matching the noise in actual data with a predefined distribution often fails; this inevitably affects the performance of tensor dictionary learning. Moreover, without the full utilization of the learning capability of data, the fixed dictionary is not adaptive for all datasets.

To address the above limitations, we propose a novel structure noise-aware tensor dictionary learning method for high-dimensional data clustering in this work. Fig. 1(a) depicts the framework of the proposed method. To preserve the intrinsic structure of data, each sample is replaced into the lateral slice of the tensor $\mathscr{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ to construct the observed data. Then, we decompose the observed data into the sum of clean data $\mathscr{C}$, structure noise $\mathscr{S}$, and Gaussian noise $\mathscr{G}$. For clean data $\mathscr{C}$, since the data usually lie in several low-dimensional subspaces, we use the tensor low-rank constraint to project the original data into the potential low-dimensional subspaces. Specifically, based on the idea of subspace clustering, we apply the tensor nuclear norm (TNN) to characterize the low-rankness of the representation tensor and explore the global correlations of the underlying data. Moreover, the singular value comparisons in Fig. 1(b) show that applying the low-rank constraint on the representation $\mathscr{Z}$ is more effective than the clean data $\mathscr{C}$. For the structure noise $\mathscr{S}$, since different data contain different types of structure noise, we adaptively learn the structure noise dictionary instead of a universal dictionary and the structure noise representation tensor at the same time. More importantly, since the structure noise is distributed along the second dimension of the tensor, it is structurally sparse. Therefore, we use $l_{2,1,2}$-norm to constrain the structural sparsity of the representation tensor. Meanwhile, we use F-norm to constrain the general Gaussian noise. As a result, the proposed model can be rewritten as

$$\begin{aligned} \min_{\mathscr{A}_c,\mathscr{A}_s,\mathscr{Z}_c,\mathscr{Z}_s} \quad & \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c * \mathscr{Z}_c - \mathscr{A}_s * \mathscr{Z}_s\|_F^2 + \lambda_1\|\mathscr{Z}_c\|_* + \lambda_2\|\mathscr{Z}_s\|_{2,1,2}, \\ \text{s.t.} \quad & \mathscr{A}_c \in \Omega, \Omega = \left\{\mathscr{A}_c : \|\mathscr{A}_{c(j_1)}\|_F^2 \leqslant 1, j_1 = 1,\cdots,k_c\right\}, \\ & \mathscr{A}_s \in \Theta, \Theta = \left\{\mathscr{A}_s : \|\mathscr{A}_{s(j_2)}\|_F^2 \leqslant 1, j_2 = 1,\cdots,k_s\right\}, \end{aligned} \tag{1}$$

where λ_1 and λ_2 are regularization parameters, $\mathscr{A}_{c(j_1)}$ and $\mathscr{A}_{s(j_2)}$ denote the j_1-th and j_2-th frontal slices of dictionaries $\mathscr{A}_c$ and $\mathscr{A}_s$, respectively, and each frontal slice represents a dictionary atom. In addition, we design an effective proximal alternating minimization (PAM) algorithm to solve the proposed model and establish the theoretical convergence guarantee of the proposed algorithm. After that, we exploit the spectral clustering tool (Ncut) [28] on the representation tensor of clean data to obtain clustering results. Extensive experiments on different datasets show the effectiveness and superiority of the proposed method. Fig. 1(c) shows that the clustering performance obtained by the representation tensor $\mathscr{Z}_c$ is better than that obtained by directly using the observed data $\mathscr{X}$. This implies that the proposed tensor decomposition strategy is effective for improving the clustering performance. In summary, the main contributions of this work are summarized as follows.

- We propose a novel tensor dictionary learning method for clustering data corrupted by structured complex noise. Compared with the matrix-type dictionary learning methods, the proposed tensor dictionary learning method can preserve the intrinsic structure of high-dimensional data and fully explore the global correlations embedded in the observed data.
- The proposed method can adaptively learn the dictionary and representation coefficient of structure noise jointly, rather than presupposing it to follow a specific distribution. Moreover, we introduce $l_{2,1,2}$-norm to explore the structural sparsity of structure noise.


Observed data $\mathcal{X}$
n_3
n_2
n_1
Clean data $\mathcal{C}$
Dictionary $\mathcal{A}_c$
Representation $\mathcal{Z}_c$
Structure noise $\mathcal{S}$
Dictionary $\mathcal{A}_s$
Representation $\mathcal{Z}_s$
Gaussian noise $\mathcal{G}$
Singular value
Clean data $\mathcal{C}$
Representation $\mathcal{Z}_c$
Observed $\mathcal{X}$
ACC
NMI
PUR

(a) The decomposition strategy

(b) The low-rankness comparison

(c) Clustering comparison

Fig. 1. The overview of the proposed method. (a) The decomposition strategy of the proposed method for structure noise data. Clearly, the representation tensor $\mathcal{Z}_c$ of clean data is low-rank and the representation tensor $\mathcal{Z}_s$ of structure noise is structurally sparse along the second dimension. (b) The low-rankness comparison between clean data $\mathcal{C}$ and representation tensor $\mathcal{Z}_c$. (c) The clustering performance of the observed data $\mathcal{X}$ and decomposed representation tensor $\mathcal{Z}_c$ (in order to clearly see the decomposed structure noise, we display the data after the linear transformation, where black pixels represent structure noise)..

- We develop a PAM-based algorithm for solving the proposed optimization problem and establish the theoretical convergence. Extensive experiments on both simulated and real datasets show the effectiveness of the proposed method on high-dimensional data clustering.

The rest of this work is organized as follows. Section 2 reviews the related dictionary learning and clustering methods. Section 3 presents the notations used in this work. Section 4 introduces the proposed model. Section 5 designs an effective algorithm to calculate the proposed model with theoretical convergence analysis. Section 6 presents the experimental results to show the effectiveness of the proposed method. Section 7 discusses some details of the proposed method. Section 8 summarizes this work.

2. Related Works

In this section, we briefly review some related dictionary learning and clustering methods.

2.1. Matrix Dictionary Learning

Dictionary learning [17] describes a sample as a linear combination of a few atoms. Mathematically, a sample $\mathbf{x} \in \mathbb{R}^n$ can be approximated by

$$\mathbf{x} = A\mathbf{z} + \mathbf{e}, \tag{2}$$

where $A \in \mathbb{R}^{n\times d}$ denotes the dictionary with each column termed as an atom, $\mathbf{z} \in \mathbb{R}^d$ denotes the representation coefficient of data $\mathbf{x}$, and $\mathbf{e} \in \mathbb{R}^n$ refers to the additive Gaussian noise. The goal of dictionary learning is to find the sparse representation $\mathbf{z}$, which has much fewer than n nonzero entries, i.e., $\|\mathbf{z}\|_0 \ll n$, under dictionary A.

To handle the image problem, matrix-based dictionary learning is proposed to represent the image data by a set of basis vectors spanning a low-dimensional space. The representative K-singular value decomposition (KSVD) algorithm [19] learns the dictionary $A \in \mathbb{R}^{n_1\times d}$ and the sparse representation $Z \in \mathbb{R}^{d\times n_2}$ from the image $X \in \mathbb{R}^{n_1\times n_2}$ by the following optimization problem:

$$\begin{aligned} \min_{A,Z} \quad & \|X - AZ\|_F^2, \\ \text{s.t.} \quad & \|A_{(j_1)}\|_2^2 \leqslant 1, j_1 \in \{1,2,\cdots,d\}, \\ & \|Z_{(j_2)}\|_0 \leqslant T, j_2 \in \{1,2,\cdots,n_2\}, \end{aligned} \tag{3}$$

where $A_{(j_1)}$ denotes the j_1-th column of A, and each column denotes a dictionary atom, d is the number of atoms in the dictionary, and $\|Z_{(j_2)}\|_0 \leqslant T$ means that the representation vector of j_2-th sample has fewer than T nonzero entries. For different tasks, various variants based on model (3) have been developed [21,29]. For example, Jiang et al. [21] used label information and KSVD for recognition. Furthermore, Feng et al. [29] proposed integrating dimensionality reduction with dictionary learning for face recognition. These methods have an implicit assumption that the noise in the data follows a Gaussian distribution. Moreover, the works in [22,23] assumed that both large-scale corruptions and outliers can be approximated from the Laplacian distribution. In addition, many self-expression-based subspace clustering methods [30,31,25] have been proposed and choose the observed data as the dictionary. However, these assumptions may not accurately model the data, especially, when the observed data are heavily corrupted by the structure noise. To accurately characterize the noise, Zhou et al. [17] decomposed the noise into two parts: structure noise and Gaussian noise. They then used dictionary learning to adaptively estimate the noise:

$$\begin{aligned} \min_{A_c,Z_c,A_s,Z_s} \quad & \|X - A_cZ_c - A_sZ_s\|_F^2 + \lambda_1\|Z_c\|_* + \lambda_2\|Z_s\|_1, \\ \text{s.t.} \quad & \|A_{c(j_1)}\|_2^2 \leqslant 1, j_1 \in \{1,2,\cdots,d_c\}, \|A_{s(j_2)}\|_2^2 \leqslant 1, j_2 \in \{1,2,\cdots,d_s\}, \end{aligned} \tag{4}$$

where $\|Z_c\|_*$ is the matrix nuclear norm, A_c and A_s respectively denote the dictionaries of clean data and structure noise, and Z_c and Z_s are two corresponding coefficient matrices. When dealing with high-dimensional data, the above matrix-based dictionary learning methods require to transform high-dimensional data into vectors, where the vectorization operator breaks the intrinsic structure of high-dimensional data, thereby limiting their performance.

2.2. Tensor Dictionary Learning

Tensor dictionary learning techniques [26,27] based on different tensor decompositions were developed to deal with high-dimensional data. Specifically, the work in [26] used the tensor singular value decomposition [32] and proposed the following K-tensor singular value decomposition (KTSVD) model:

$$\min_{\mathscr{Z}} \|\mathscr{X} - \mathscr{A} * \mathscr{Z}\|_F^2 + \lambda\|\mathscr{Z}\|_{TS}, \tag{5}$$

where $*$ denotes the tensor-tensor product (see Definition 1), $\mathscr{X} \in \mathbb{R}^{n_1\times n_2\times n_3}$ denotes the observed tensor, $\mathscr{A} \in \mathbb{R}^{n_1\times k\times n_3}$ is a learned tensor dictionary, and $\|\mathscr{Z}\|_{TS}$ denotes the tensor tubal sparsity defined as the number of nonzero tubes of $\mathscr{Z}$ in the third dimension. The work in [26] assumed that the noise contained in the data follows a Gaussian distribution, thereby constraining it with the Frobenius norm. Taking the input data as the dictionary, Zhou et al. [2] introduced TNN to learn the tensor low-rank representation and applied the l_1-norm to constrain the sparse noise. However, the complex noise in practical applications cannot be accurately described with a predefined distribution.

In Table 1, we compare the related studies with our work from data representation, noise distribution, and dictionary learning.

3. Notations and Preliminaries

In this section, we list main notations and definitions used in this work.

3.1. Notations

We use the calligraphy letter $\mathscr{Z}$ to denote the tensor, the upper case letter Z to denote the matrix, the bold lower case letter $\mathbf{z}$ to denote the vector, and the lower case letter z to denote the scalar. For a third-order tensor $\mathscr{Z} \in \mathbb{R}^{n_1\times n_2\times n_3}$, we use $\mathscr{Z}_{i,j,k}$ to denote its (i,j,k)-th entry and the MATLAB notation $\mathscr{Z}(i,:,:)$, $\mathscr{Z}(:,j,:)$, and $\mathscr{Z}(:,:,k)$ to denote the i-th horizontal, j-th lateral, and k-th frontal slice, respectively. For convenience, we denote the frontal slice $\mathscr{Z}(:,:,k)$ by $\mathscr{Z}^{(k)}$, the lateral slice $\mathscr{Z}(:,j,:)$ by $\mathscr{Z}_{(j)}$, and the tube by $\mathscr{Z}(i,j,:)$. For a matrix $Z \in \mathbb{R}^{n_1\times n_2}$, the (i,j)-th entry and the j-th column are denoted as z_{ij} and $Z_{(j)}$, respectively.

Now, we show some norms used in this work. For the third-order tensor $\mathscr{Z} \in \mathbb{R}^{n_1\times n_2\times n_3}$, the l_1-norm is defined as $\|\mathscr{Z}\|_1 = \sum_{i,j,k}|\mathscr{Z}_{i,j,k}|$, the Frobenius norm is $\|\mathscr{Z}\|_F = \sqrt{\sum_{i,j,k}|\mathscr{Z}_{i,j,k}|^2}$, and the infinity norm is $\|\mathscr{Z}\|_\infty = \max_{i,j,k}|\mathscr{Z}_{i,j,k}|$. For a matrix, its nuclear norm is $\|Z\|_* = \sum_i\sigma_i(Z)$, and $\sigma_i(Z)$ denotes the i-th singular value of Z. The l_2-norm of a vector $\mathbf{z} \in \mathbb{R}^n$ is defined as $\|\mathbf{z}\|_2 = \sqrt{\sum_i\mathbf{z}_i^2}$, and $\mathbf{z}_i$ is the i-th element of $\mathbf{z}$. The main symbols and their meanings are summarized in Table 2.

Table 1
Summary of some related works.

Methods	Data representation	Noise distribution	Dictionary learning
K-singular value decomposition [19]	Matrix	Gaussian	Adaptive
Dictionary learning based impulse noise [22]	Matrix	Laplacian	Adaptive
Robust non-negative dictionary learning [23]	Matrix	Laplacian	Adaptive
Dictionary learning with structure noise [17]	Matrix	Structure and Gaussian	Adaptive
Latent multi-view subspace clustering [30]	Matrix	Sparse	Adaptive
Robust subspace segmentation [33]	Matrix	Gaussian	Predefined
Robust kernel low-rank representation [31]	Matrix	Sparse	Predefined
K-tensor singular value decomposition [26]	Tensor	Gaussian	Adaptive
Tensor factorization for dictionary learning [34]	Tensor	Gaussian	Adaptive
Tensor low-rank representation [2]	Tensor	Sparse	Predefined
Structure noise-aware tensor dictionary learning (this paper)	Tensor	Structure and Gaussian	Adaptive

Table 2
The summary of symbols used in our work.

Symbols	Meanings
$\mathscr{Z}, Z, \mathbf{z}$, and z	tensor, matrix, vector, and scalar
$\mathscr{Z}^{(k)}$ and $\mathscr{Z}_{(j)}$	the k-th frontal slice and the j-th lateral slice of $\mathscr{Z}$
$\mathscr{Z}_{i,j,k}$	the (i,j,k)-th element of $\mathscr{Z}$
$\|\mathscr{Z}\|_1, \|\mathscr{Z}\|_F$, and $\|\mathscr{Z}\|_\infty$	l_1-norm, Frobenius norm, and infinity norm of $\mathscr{Z}$
$\|\mathscr{Z}\|_*$ and $\|\mathscr{Z}\|_{2,1,2}$	tensor nuclear norm and $l_{2,1,2}$-norm of $\mathscr{Z}$
$\|Z\|_*$	matrix nuclear norm of Z
$\|\mathbf{z}\|_2$	the l_2-norm of the vector $\mathbf{z}$
n_2 and $n_1 \times n_3$	the number and size of samples
$\mathscr{X}, \mathscr{C}, \mathscr{S}$, and $\mathscr{N} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$	observed data, underlying data, structure noise, and Gaussian noise
$\mathscr{A}_c \in \mathbb{R}^{n_1 \times k_c \times n_3}$	the dictionary of the clean data
$\mathscr{Z}_c \in \mathbb{R}^{k_c \times n_2 \times n_3}$	the representation tensor of the clean data
$\mathscr{A}_s \in \mathbb{R}^{n_1 \times k_s \times n_3}$	the dictionary of the structure noise
$\mathscr{Z}_s \in \mathbb{R}^{k_s \times n_2 \times n_3}$	the representation tensor of the structure noise
k_c and k_s	the number of atoms in the dictionaries $\mathscr{A}_c$ and $\mathscr{A}_s$
$\mathscr{A}_{c(j_1)} \in \mathbb{R}^{n_1 \times n_3}$	the j_1-th atom of the dictionary $\mathscr{A}_c$
$\mathscr{A}_{s(j_2)} \in \mathbb{R}^{n_1 \times n_3}$	the j_2-th atom of the dictionary $\mathscr{A}_s$

3.2. Preliminaries

We introduce some related definitions used in our work.

For $\mathscr{Z} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, we denote the discrete Fourier transform (DFT) of $\mathscr{Z}$ along the third dimension by $\bar{\mathscr{Z}}$, i.e., $\bar{\mathscr{Z}} = \mathrm{fft}(\mathscr{Z}, [], 3)$. Moreover, we can obtain $\mathscr{Z}$ by the inverse DFT, i.e., $\mathscr{Z} = \mathrm{ifft}(\bar{\mathscr{Z}}, [], 3)$. Specifically, the block diagonal matrix $\overline{Z} \in \mathbb{R}^{n_1 n_3 \times n_2 n_3}$ of $\bar{\mathscr{Z}}$ has the following form:

$$\overline{Z} = \mathrm{bdiag}(\bar{\mathscr{Z}}) = \begin{pmatrix} \bar{\mathscr{Z}}^{(1)} & & \\ & \ddots & \\ & & \bar{\mathscr{Z}}^{(n_3)} \end{pmatrix}, \tag{6}$$

where $\bar{\mathscr{Z}}^{(k)}$ denotes the k-th frontal slice of $\bar{\mathscr{Z}}$. And the block-circulant matrix of $\mathscr{Z} \in \mathbb{R}^{n_1 n_3 \times n_2 n_3}$ is

$$\mathrm{bcirc}(\mathscr{Z}) = \begin{pmatrix} \mathscr{Z}^{(1)} & \mathscr{Z}^{(n_3)} & \cdots & \mathscr{Z}^{(2)} \\ \mathscr{Z}^{(2)} & \mathscr{Z}^{(1)} & \cdots & \mathscr{Z}^{(3)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathscr{Z}^{(n_3)} & \mathscr{Z}^{(n_3-1)} & \cdots & \mathscr{Z}^{(1)} \end{pmatrix}. \tag{7}$$

Moreover, we define the unfold and fold operators as

$$\mathrm{unfold}(\mathscr{Z}) = \begin{pmatrix} \mathscr{Z}^{(1)} \\ \mathscr{Z}^{(2)} \\ \vdots \\ \mathscr{Z}^{(n_3)} \end{pmatrix}, \mathrm{fold}(\mathrm{unfold}(\mathscr{Z})) = \mathscr{Z}. \tag{8}$$

Definition 1. (*Tensor-tensor product*) The tensor-tensor product of $\mathscr{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\mathscr{B} \in \mathbb{R}^{n_2 \times l \times n_3}$ is defined as

$$\mathscr{C} = \mathscr{A} * \mathscr{B} = fold(bcirc(\mathscr{A}) \times unfold(\mathscr{B})),$$

where $\mathscr{C} \in \mathbb{R}^{n_1 \times l \times n_3}$ and $\times$ denotes the matrix product.

Definition 2 (*Identity tensor*). If the first frontal slice of a tensor $\mathscr{I} \in \mathbb{R}^{n_1 \times n_1 \times n_3}$ is an $n_1 \times n_1$ identity matrix and other frontal slices are zero matrix, then it is the identity tensor.

Definition 3 (*Transpose tensor*). For a tensor $\mathscr{Z} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, by transposing each of the frontal slice, the transpose tensor $\mathscr{Z}^\top \in \mathbb{R}^{n_2 \times n_1 \times n_3}$ can be obtained by reversing the order of transposed frontal slices 2 through n_3.

Definition 4 (*Orthogonal tensor*). If the tensor $\mathscr{Z} \in \mathbb{R}^{n_1 \times n_1 \times n_3}$ satisfies

$$\mathscr{Z}^\top * \mathscr{Z} = \mathscr{Z} * \mathscr{Z}^\top = \mathscr{I},$$

then it is orthogonal.

Definition 5 (*TSVD*). Let $\mathscr{Z} \in \mathbb{R}^{n_1 \times n_2 \times n_3}, \mathscr{Z}$ has the following tensor singular value decomposition (TSVD)

$$\mathscr{Z} = \mathscr{U} * \mathscr{S} * \mathscr{V}^\top, \tag{9}$$

where $\mathscr{S} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is a diagonal tensor, and $\mathscr{U} \in \mathbb{R}^{n_1 \times n_1 \times n_3}$ and $\mathscr{V} \in \mathbb{R}^{n_2 \times n_2 \times n_3}$ are orthogonal.

Definition 6 (*TNN*). The tensor nuclear norm (TNN) of the tensor $\mathscr{Z} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is

$$\|\mathscr{Z}\|_* = \sum_{i=1}^{n_3} \|\bar{\mathscr{Z}}^{(i)}\|_*,$$

where $\|\bar{\mathscr{Z}}^{(i)}\|_*$ is the nuclear norm of $\bar{\mathscr{Z}}^{(i)}$.

Definition 7 (*Tensor tubal rank*). The tubal rank rank($\mathscr{Z}$) is defined as the number of nonzero singular tubes of $\mathscr{S}$, where $\mathscr{S}$ is from the TSVD of $\mathscr{Z} = \mathscr{U} * \mathscr{S} * \mathscr{V}^\top$, i.e.,

$$\text{rank}(\mathscr{Z}) = \#\{i : \mathscr{S}(i,j,:) \neq 0\}, \tag{10}$$

where $\#$ denotes the cardinality of a set.

4. The Proposed Model

In this section, we present the proposed tensor dictionary learning model with structure noise for high-dimensional data clustering. The research significance of the work lies in the fact that structure noise is very common in real applications due to equipment or environmental factors [17]. Structure noise seriously hinders the tasks of data mining and applications. Therefore, we focus on improving the data mining algorithm. For convenience, we denote the dictionaries of the clean data and the structure noise by $\mathscr{A}_c \in \mathbb{R}^{n_1 \times k_c \times n_3}$ and $\mathscr{A}_s \in \mathbb{R}^{n_1 \times k_s \times n_3}$, respectively and denote the representation tensors of the clean data and the structure noise by $\mathscr{Z}_c \in \mathbb{R}^{k_c \times n_2 \times n_3}$ and $\mathscr{Z}_s \in \mathbb{R}^{k_s \times n_2 \times n_3}$, respectively.

In Fig. 1, we present the proposed framework and decompose the observed data into three parts: clean data $\mathscr{C}$, structure noise $\mathscr{S}$, and other Gaussian noise $\mathscr{N}$. In practical applications, since the number of samples is generally greater than the number of classes, samples belonging to the same class are usually located in a lower-dimensional subspace [19,26]. Since the dimension of the subspace is same as the rank of the representation tensor, we use the low-rank constraint to characterize the low-rankness of the representation tensor of clean data. For the structure noise, we adaptively learn the dictionary of structure noise for different data and set the corresponding representation tensor to be sparse. Therefore, we propose the following model:

$$\begin{aligned} \min_{\mathscr{A}_c,\mathscr{A}_s,\mathscr{Z}_c,\mathscr{Z}_s} \quad & \tfrac{1}{2}\|\mathscr{X} - \mathscr{A}_c * \mathscr{Z}_c - \mathscr{A}_s * \mathscr{Z}_s\|_F^2 + \lambda_1 \text{rank}(\mathscr{Z}_c) + \lambda_2 \|\mathscr{Z}_s\|_0, \\ \text{s.t.} \quad & \mathscr{A}_c \in \Omega, \Omega = \left\{\mathscr{A}_c : \|\mathscr{A}_{c(j_1)}\|_F^2 \leqslant 1, j_1 = 1,2,\cdots,k_c\right\}, \\ & \mathscr{A}_s \in \Theta, \Theta = \left\{\mathscr{A}_s : \|\mathscr{A}_{s(j_2)}\|_F^2 \leqslant 1, j_2 = 1,2,\cdots,k_s\right\}, \end{aligned} \tag{11}$$

where λ_1 and λ_2 are regularization parameters, $\|\mathscr{A}_{c(j_1)}\|_F^2 \leqslant 1$ and $\|\mathscr{A}_{s(j_2)}\|_F^2 \leqslant 1$ normalize each atom in the dictionary $\mathscr{A}_c$ and $\mathscr{A}_s$, respectively. We use $\text{rank}(\mathscr{Z}_c)$ to characterize the global low-rankness of the representation tensor of clean data and $\|\mathscr{Z}_s\|_0$ to characterize the sparseness of the representation tensor of structure noise. However, since the corresponding minimization problems of $\text{rank}(\cdot)$ and $\|\cdot\|_0$ are NP-hard, we replace $\text{rank}(\mathscr{Z}_c)$ and $\|\mathscr{Z}_s\|_0$ with the corresponding convex relaxations $\|\mathscr{Z}_c\|_*$ and $\|\mathscr{Z}_s\|_{2,1,2}$, respectively.

Finally, we propose the following **T**ensor **D**ictionary **L**earning-based **S**tructure **N**oise model (TDLSN) for high-dimensional data clustering:

$$\begin{aligned} \min_{\mathscr{A}_c,\mathscr{A}_s,\mathscr{Z}_c,\mathscr{Z}_s} \quad & \tfrac{1}{2}\|\mathscr{X} - \mathscr{A}_c * \mathscr{Z}_c - \mathscr{A}_s * \mathscr{Z}_s\|_F^2 + \lambda_1\|\mathscr{Z}_c\|_* + \lambda_2\|\mathscr{Z}_s\|_{2,1,2}, \\ \text{s.t.} \quad & \mathscr{A}_c \in \Omega, \Omega = \left\{\mathscr{A}_c : \|\mathscr{A}_{c(j_1)}\|_F^2 \leqslant 1, j_1 = 1,\cdots,k_c\right\}, \\ & \mathscr{A}_s \in \Theta, \Theta = \left\{\mathscr{A}_s : \|\mathscr{A}_{s(j_2)}\|_F^2 \leqslant 1, j_2 = 1,\cdots,k_s\right\}, \end{aligned} \tag{12}$$

where $\|\mathscr{Z}_c\|_*$ denotes the TNN of the representation tensor $\mathscr{Z}_c$, and $\|\mathscr{Z}_s\|_{2,1,2}$ denotes the structural sparsity of the representation tensor $\mathscr{Z}_s$, which is defined as

$$\|\mathscr{Z}_s\|_{2,1,2} = \sum_j \|\mathscr{Z}_s(:,j,:)\|_F. \tag{13}$$

The proposed model consists of three terms. The first quadratic term characterizes the Gaussian noise. The second term adopts the TNN to describe the global low-rankness of the representation tensor of clean data. The third one uses $l_{2,1,2}$-norm to constrain the structural sparsity of structure noise along the second dimension.

The proposed TDLSN is a more general and flexible framework than other related methods. In particular, the proposed TDLSN (12) becomes KTSVD in [26] when the structure noise is ignored and the learning dictionary representation is fixed; our TDLSN degenerates into tensor low-rank representation (TLRR) clustering [2] when the structure noise is ignored, and the dictionary is set as the input data. In summary, the advantages of the proposed TDLSN are as follows:

- Compared with matrix-based sparse and/or low-rank representation clustering methods, TDLSN can preserve the intrinsic structure of high-dimensional data without the data vectorizing operation. Moreover, the matrix-based methods only obtain the simple (1-D) consistency information between samples, and cannot explore the complex consistency information within samples. By contrast, the proposed tensor method can fully explore the high-order correlation among samples by TNN to obtain the underlying consistent information for clustering high-dimensional data.
- Compared with existing tensor-based sparse or low-rank representation clustering methods, TDLSN decomposes the data into the sum of clean data, structure noise, and Gaussian noise, thereby accurately characterizing noise. To characterize the structure noise distributed in the second dimension of the tensor, we use the $l_{2,1,2}$-norm to depict the structural sparsity.
- Compared with the unified dictionary, TDLSN can adaptively learn the dictionaries of the clean data and the structure noise for different data. Thus, TDLSN can obtain a discriminative data low-dimensional representation to improve the clustering performance.

Remark. We elaborate on the connection of the proposed method with multi-view clustering. Tensor decomposition and its related methods [6,35–37] have also been developed for solving the multi-view clustering problem. The differences between the proposed method and these multi-view clustering methods are summarized as follows:

(1) **The processing of high-dimensional data.** We focus on the single-view clustering of high-dimensional data. To process the data, we directly stack each high-dimensional sample into a tensor in order, which can preserve the intrinsic structure of high-dimensional data. In contrast, the aforementioned methods all deal with the clustering task of multi-view data, i.e., each data has multiple feature representations. However, due to the structural inconsistency of multi-view data, these multi-view clustering methods vectorize the original high-dimensional data and destroy the intrinsic structure of the data.
(2) **The distribution of noise.** In practical applications, the observed data are often corrupted by complex noise. In this work, we decompose the data into the sum of clean data, structure noise, and Gaussian noise. The proposed tensor dictionary learning method can accurately characterize and separate the complex noise. However, these multi-view clustering methods assume that the noise in the datasets all follows one specific predefined distribution, resulting in potentially imprecise separation of clean data.
(3) **The choice of dictionary.** The self-expression-based multi-view clustering methods [35,36] use the observed data itself as the dictionary to learn representation coefficient matrices. However, when the observed data are contaminated, the learned coefficient matrices are imprecise, resulting in poor clustering performance. In contrast, our method adopts the tensor dictionary learning to adaptively learn the dictionary to reduce the effect of the complex noise in the observed data.

5. The Proposed Algorithm

First, we design PAM [38,39] to solve the proposed model (12). Then, we establish the convergence analysis in theory. Finally, we analyze the computational complexity of the proposed algorithm.

5.1. PAM Solver

We define the objective function as

$$F(\mathscr{A}_c,\mathscr{A}_s,\mathscr{Z}_c,\mathscr{Z}_s)=\frac{1}{2}\|\mathscr{X}-\mathscr{A}_c*\mathscr{Z}_c-\mathscr{A}_s*\mathscr{Z}_s\|_F^2+\lambda_1\|\mathscr{Z}_c\|_*+\lambda_2\|\mathscr{Z}_s\|_{2,1,2}+\Psi(\mathscr{A}_c)+\Phi(\mathscr{A}_s), \tag{14}$$

where $\Psi(\mathscr{A}_c)$ and $\Phi(\mathscr{A}_s)$ are indicator functions defined as

$$\Psi(\mathscr{A}_c)=\begin{cases}0, & \text{if } \|\mathscr{A}_{c(j_1)}\|_F^2\leqslant 1,\\ \infty, & \text{otherwise.}\end{cases}\qquad \Phi(\mathscr{A}_s)=\begin{cases}0, & \text{if } \|\mathscr{A}_{s(j_2)}\|_F^2\leqslant 1,\\ \infty, & \text{otherwise.}\end{cases}$$

According to the framework of the PAM algorithm, the proposed optimization problem (12) can be decomposed into the following four subproblems:

$$\begin{cases}\mathscr{A}_c^{t+1}=\arg\min\limits_{\mathscr{A}_c}\Big\{Q_1(\mathscr{A}_c|\mathscr{A}_c^t)=F(\mathscr{A}_c,\mathscr{A}_s^t,\mathscr{Z}_c^t,\mathscr{Z}_s^t)+\frac{\rho}{2}\|\mathscr{A}_c-\mathscr{A}_c^t\|_F^2\Big\},\\ \mathscr{A}_s^{t+1}=\arg\min\limits_{\mathscr{A}_s}\Big\{Q_2(\mathscr{A}_s|\mathscr{A}_s^t)=F(\mathscr{A}_c^{t+1},\mathscr{A}_s,\mathscr{Z}_c^t,\mathscr{Z}_s^t)+\frac{\rho}{2}\|\mathscr{A}_s-\mathscr{A}_s^t\|_F^2\Big\},\\ \mathscr{Z}_c^{t+1}=\arg\min\limits_{\mathscr{Z}_c}\Big\{Q_3(\mathscr{Z}_c|\mathscr{Z}_c^t)=F(\mathscr{A}_c^{t+1},\mathscr{A}_s^{t+1},\mathscr{Z}_c,\mathscr{Z}_s^t)+\frac{\rho}{2}\|\mathscr{Z}_c-\mathscr{Z}_c^t\|_F^2\Big\},\\ \mathscr{Z}_s^{t+1}=\arg\min\limits_{\mathscr{Z}_s}\Big\{Q_4(\mathscr{Z}_s|\mathscr{Z}_s^t)=F(\mathscr{A}_c^{t+1},\mathscr{A}_s^{t+1},\mathscr{Z}_c^{t+1},\mathscr{Z}_s)+\frac{\rho}{2}\|\mathscr{Z}_s-\mathscr{Z}_s^t\|_F^2\Big\},\end{cases} \tag{15}$$

where ρ denotes a positive constant, and the superscript t denotes the iteration indexes. Since the subproblems of $\mathscr{A}_c$ and $\mathscr{A}_s$ are similar to each other, and the subproblems of $\mathscr{Z}_c$ and $\mathscr{Z}_s$ are similar to each other, we only present the details of optimizing the subproblems of $\mathscr{A}_c$ and $\mathscr{Z}_c$. For the subproblems of $\mathscr{A}_s$ and $\mathscr{Z}_s$, please refer to Appendix A for details.

$\mathscr{A}_c$-subproblem. The $\mathscr{A}_c$-subproblem is

$$\mathscr{A}_c^{t+1}=\text{argmin}_{\mathscr{A}_c}\frac{1}{2}\|\mathscr{X}-\mathscr{A}_c*\mathscr{Z}_c^t-\mathscr{A}_s^t*\mathscr{Z}_s^t\|_F^2+\frac{\rho}{2}\|\mathscr{A}_c-\mathscr{A}_c^t\|_F^2+\Psi(\mathscr{A}_c). \tag{16}$$

We use the alternating direction method of multipliers (ADMM) algorithm [40] and covert (16) to the following form by introducing the auxiliary variable $\mathscr{Y}_1$,

$$\begin{aligned}\min_{\mathscr{A}_c,\mathscr{Y}_1}\quad & \tfrac{1}{2}\|\mathscr{X}-\mathscr{A}_c*\mathscr{Z}_c^t-\mathscr{A}_s^t*\mathscr{Z}_s^t\|_F^2+\tfrac{\rho}{2}\|\mathscr{A}_c-\mathscr{A}_c^t\|_F^2+\Psi(\mathscr{Y}_1),\\ \text{s.t.}\quad & \mathscr{Y}_1=\mathscr{A}_c.\end{aligned} \tag{17}$$

And the corresponding augmented Lagrangian function of (17) is

$$\mathscr{L}(\mathscr{A}_c,\mathscr{Y}_1,\mathscr{C}_1)=\frac{1}{2}\|\mathscr{X}-\mathscr{A}_c*\mathscr{Z}_c^t-\mathscr{A}_s^t*\mathscr{Z}_s^t\|_F^2+\frac{\rho}{2}\|\mathscr{A}_c-\mathscr{A}_c^t\|_F^2+\Psi(\mathscr{Y}_1)+\frac{\beta_1}{2}\|\mathscr{Y}_1-\mathscr{A}_c+\frac{\mathscr{C}_1}{\beta}\|_F^2, \tag{18}$$

where β_1 is a penalty parameter, and $\mathscr{C}_1$ denotes the Lagrangian multiplier. Then, ADMM alternately updates the three subproblems $\mathscr{A}_c,\mathscr{Y}_1$, and $\mathscr{C}_1$. We give details of updating each subproblem as follows.

- **Calculation of** $\mathscr{A}_c$. Given $\mathscr{Y}_1$ and $\mathscr{C}_1$, the minimization problem of $\mathscr{A}_c$ is

$$\mathscr{A}_c^{t,k+1}=\text{argmin}_{\mathscr{A}_c}\frac{1}{2}\|\mathscr{X}-\mathscr{A}_c*\mathscr{Z}_c^{t,k}-\mathscr{A}_s^{t,k}*\mathscr{Z}_s^{t,k}\|_F^2+\frac{\rho}{2}\|\mathscr{A}_c-\mathscr{A}_c^t\|_F^2+\frac{\beta_1}{2}\|\mathscr{Y}_1^k-\mathscr{A}_c+\frac{\mathscr{C}_1^k}{\beta}\|_F^2, \tag{19}$$

where the superscript k denotes the inner iteration indices. Its solution satisfies the following equation:

$$\mathscr{A}_c*\left(\mathscr{Z}_c^{t,k}*\left(\mathscr{Z}_c^{t,k}\right)^\top+(\rho+\beta_1)\mathscr{I}\right)=\left(\mathscr{X}-\mathscr{A}_s^{t,k}*\mathscr{Z}_s^{t,k}\right)*\left(\mathscr{Z}_c^{t,k}\right)^\top+\rho\mathscr{A}_c^t+\beta_1\left(\mathscr{Y}_1^k+\frac{\mathscr{C}_1^k}{\beta}\right), \tag{20}$$

which can be solved effectively in the Fourier domain [2].

- **Calculation of** $\mathscr{Y}_1$. Given $\mathscr{A}_c$ and $\mathscr{C}_1$, the minimization problem of $\mathscr{Y}_1$ is

$$\mathscr{Y}_1^{k+1}=\text{argmin}_{\mathscr{Y}_1}\Psi(\mathscr{Y}_1)+\frac{\beta_1}{2}\|\mathscr{Y}_1-\mathscr{A}_c^{t,k+1}+\frac{\mathscr{C}_1^k}{\beta}\|_F^2, \tag{21}$$

which has the closed form solution

$$\mathscr{Y}_{1(j_1)}^{k+1} = \frac{M_{1(j_1)}}{\max\left\{1, \|M_{1(j_1)}\|_2^2\right\}}, j_1 = 1, 2, \ldots, k_c, \tag{22}$$

where $M_{1(j_1)} = \mathscr{A}_{c(j_1)}^{t,k+1} - \mathscr{C}_{1(j_1)}^{k}/\beta_1$.

- **Update multiplier** $\mathscr{C}_1$. The multiplier $\mathscr{C}_1$ can be updated by

$$\mathscr{C}_1^{k+1} = \mathscr{C}_1^{k+1} + \beta_1\left(\mathscr{Y}_1^{k+1} - \mathscr{A}_c^{t,k+1}\right). \tag{23}$$

Following, we present the details of solving $\mathscr{Z}_c$-subproblem.

$\mathscr{Z}_c$-subproblem. The $\mathscr{Z}_c$-subproblem is

$$\mathscr{Z}_c^{t+1} = \text{argmin}_{\mathscr{Z}_c} \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{Z}_c - \mathscr{A}_s^{t+1} * \mathscr{Z}_s^t\|_F^2 + \lambda_1\|\mathscr{Z}_c\|_* + \frac{\rho}{2}\|\mathscr{Z}_c - \mathscr{Z}_c^t\|_F^2. \tag{24}$$

Since the $\mathscr{Z}_c$-subproblem is convex, we also adopt the ADMM solver.

Introducing the variable $\mathscr{P}$, we can convert (24) to the following problem:

$$\begin{aligned} \min_{\mathscr{Z}_c, \mathscr{P}} \quad & \tfrac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{P} - \mathscr{A}_s^{t+1} * \mathscr{Z}_s^t\|_F^2 + \lambda_1\|\mathscr{Z}_c\|_* + \tfrac{\rho}{2}\|\mathscr{Z}_c - \mathscr{Z}_c^t\|_F^2, \\ \text{s.t.} \quad & \mathscr{P} = \mathscr{Z}_c. \end{aligned} \tag{25}$$

The optimization problem (25) matches the ADMM framework [41,42]. Thus, we present the augmented Lagrangian function of (25) as

$$\mathscr{L}(\mathscr{Z}_c, \mathscr{P}, \mathscr{E}) = \tfrac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{P} - \mathscr{A}_s^{t+1} * \mathscr{Z}_s^t\|_F^2 + \lambda_1\|\mathscr{Z}_c\|_* + \tfrac{\rho}{2}\|\mathscr{Z}_c - \mathscr{Z}_c^t\|_F^2 + \tfrac{\beta_2}{2}\|\mathscr{P}^k - \mathscr{Z}_c + \tfrac{\mathscr{E}^k}{\beta}\|_F^2, \tag{26}$$

where β_2 is a penalty parameter, and $\mathscr{E}$ denotes a Lagrangian multiplier. Then, ADMM alternately iterates the three subproblems $\mathscr{Z}_c, \mathscr{P}$, and $\mathscr{E}$. We give details of solving each subproblem as follows.

- **Calculation of** $\mathscr{Z}_c$. Given $\mathscr{P}$ and $\mathscr{E}$, the minimization problem of $\mathscr{Z}_c$ is

$$\mathscr{Z}_c^{t,k+1} = \text{argmin}_{\mathscr{Z}_c} \lambda_1\|\mathscr{Z}_c\|_* + \frac{\beta_2 + \rho}{2}\|\mathscr{Z}_c - \frac{\rho\mathscr{Z}_c^t + \beta_2\left(\mathscr{P}^k + \mathscr{E}^k/\beta\right)}{\beta_2 + \rho}\|_F^2. \tag{27}$$

It can be solved via the tensor singular value thresholding operator [43], i.e.,

$$\mathscr{Z}_c^{t,k+1} = \mathscr{U} * \Gamma_{\frac{\lambda_1}{\beta+\rho}}(\mathscr{S}) * \mathscr{V}^\top, \tag{28}$$

where $\mathscr{U} * \mathscr{S} * \mathscr{V}^\top$ is the TSVD of $\left(\rho\mathscr{Z}_c^t + \beta_2\left(\mathscr{P}^k + \mathscr{E}^k/\beta\right)\right)/(\beta_2 + \rho)$, and $\Gamma_{\frac{\lambda_1}{\beta_2+\rho}}(\mathscr{S}) = \text{ifft}\left(\max\left(\bar{\mathscr{S}} - \frac{\lambda_1}{\beta_2+\rho}, 0\right), [], 3\right)$.

Algorithm 1: The PAM solver for (12).

```
Input: Tensor $\mathscr{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$.
1: Parameters: $\lambda_1, \lambda_2, \beta_1, \beta_2, \beta_3, \beta_4, \rho, \epsilon = 10^{-5}$, outer iteration $t_{\text{out}} = 100$, inner iteration $k_{\text{inner}} = 10$.
2: Initialize: $\mathscr{A}_c, \mathscr{A}_s, \mathscr{Z}_c, \mathscr{Z}_s$.
3: Outer iteration: While $t \leqslant t_{\text{out}}$ and is not converged do
4:     Inner iteration: While $k \leqslant k_{\text{inner}}$ do
5:         Update $\mathscr{A}_c$ by solving (20);
6:         Update $\mathscr{Y}_1$ by solving (22);
7:         Update $\mathscr{C}_1$ by solving (23);
8:     End while
9:     While $k \leqslant k_{\text{inner}}$ do
10:        Update $\mathscr{A}_s$ by solving (40);
11:        Update $\mathscr{Y}_2$ by solving (42);
12:        Update $\mathscr{C}_2$ by solving (43);
13:    End while
14:    While $k \leqslant k_{\text{inner}}$ do
15:        Update $\mathscr{Z}_c$ by solving (28);
16:        Update $\mathscr{P}$ by solving (30);
17:        Update $\mathscr{E}$ by solving (31);
18:    End while
19:    While $k \leqslant k_{\text{inner}}$ do
20:        Update $\mathscr{Z}_s$ by solving (48);
21:        Update $\mathscr{Q}$ by solving (50);
```

(continued on next page)

a (*continued*)

Algorithm 1: The PAM solver for (12).

```
22:         Update $\mathscr{F}$ by solving (51);
23:     End while
24: Check the convergence condition:
        $\max(\|\mathscr{A}_c^{t+1}-\mathscr{A}_c^t\|_\infty, \|\mathscr{A}_s^{t+1}-\mathscr{A}_s^t\|_\infty, \|\mathscr{Z}_c^{t+1}-\mathscr{Z}_c^t\|_\infty, \|\mathscr{Z}_s^{t+1}-\mathscr{Z}_s^t\|_\infty) \leqslant \epsilon$;
25: End while
26: Apply the spectral clustering tool (Ncut) with $\mathscr{Z}_c$.
Output: The clustering results.
```

- **Calculation of $\mathscr{P}$.** Given $\mathscr{Z}_c$ and $\mathscr{E}$, the $\mathscr{P}$ subproblem is

$$\mathscr{P}^{k+1} = \text{argmin}_{\mathscr{P}} \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{P} - \mathscr{A}_s^{t+1} * \mathscr{Z}_s^{t,k}\|_F^2 + \frac{\beta_2}{2}\|\mathscr{P} - \mathscr{Z}_c^{t,k+1} + \frac{\mathscr{E}^k}{\beta}\|_F^2. \tag{29}$$

The minimizer of the $\mathscr{P}$ satisfies the following equation:

$$\left[\left(\mathscr{A}_c^{t+1}\right)^\top * \mathscr{A}_c^{t+1} + \beta_2 I\right] * \mathscr{P} = \left(\mathscr{A}_c^{t+1}\right)^\top * \left(\mathscr{X} - \mathscr{A}_s^{t+1} * \mathscr{Z}_s^{t,k}\right) + \beta_2\left(\mathscr{Z}_c^{t,k+1} - \mathscr{E}^k/\beta\right), \tag{30}$$

which can be calculated in the Fourier domain [2].

- **Update multiplier $\mathscr{E}$.** The multiplier $\mathscr{E}$ can be updated by

$$\mathscr{E}^{k+1} = \mathscr{E}^{k+1} + \beta_2\left(\mathscr{P}^{k+1} - \mathscr{Z}_c^{t,k+1}\right). \tag{31}$$

Finally, we summarize the proposed algorithm in Algorithm 1.

5.2. The Convergence Analysis

Next, we prove the convergence of the proposed algorithm. For convenience, we define the following formulation:

$$\begin{aligned}
F(\mathscr{A}_c, \mathscr{A}_s, \mathscr{Z}_c, \mathscr{Z}_s) &:= \tfrac{1}{2}\|\mathscr{X} - \mathscr{A}_c * \mathscr{Z}_c - \mathscr{A}_s * \mathscr{Z}_s\|_F^2 + \lambda_1\|\mathscr{Z}_c\|_* + \lambda_2\|\mathscr{Z}_s\|_{2,1,2} + \Psi(\mathscr{A}_c) + \Phi(\mathscr{A}_s),\\
W(\mathscr{A}_c, \mathscr{A}_s, \mathscr{Z}_c, \mathscr{Z}_s) &:= \tfrac{1}{2}\|\mathscr{X} - \mathscr{A}_c * \mathscr{Z}_c - \mathscr{A}_s * \mathscr{Z}_s\|_F^2,\\
f_1(\mathscr{Z}_c) &:= \lambda_1\|\mathscr{Z}_c\|_*,\\
f_2(\mathscr{Z}_s) &:= \lambda_2\|\mathscr{Z}_s\|_{2,1,2},
\end{aligned}$$

Theorem 1. *The bounded sequence $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ (i.e., iterations in (15)) obtained by* **Algorithm 1** *globally converges to the critical point of (12).*

The proof of Theorem 1 follows the line of **Theorem 6.2** in [38]. Particularly, we require to show the three conditions:

- F satisfies the KŁ property at each point,
- the sequence $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ satisfies the sufficient decrease condition ((64) in [38]),
- the sequence $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ satisfies the relative error condition ((65)-(66) in [38]).

Before verifying the theorem, we present some related lemmas. We give the detailed proofs of the following Lemmas 1–3 in Appendix B.

First, we prove the KŁ property of $F(\mathscr{A}_c, \mathscr{A}_s, \mathscr{Z}_c, \mathscr{Z}_s)$.

Lemma 1 (*KŁ Lemma*). *Function $F(\mathscr{A}_c, \mathscr{A}_s, \mathscr{Z}_c, \mathscr{Z}_s)$ has the KŁ property at each point.*

Next, we verify that the bounded sequence $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ satisfies the sufficient decrease condition and the relative error condition.

Lemma 2 (*Sufficient decrease Lemma*). *For $\rho > 0$, let $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ be a sequence from* **Algorithm 1**, *then*

$$\begin{aligned}
&F\left(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\right) + \tfrac{\rho}{2}\|\mathscr{A}_c^{t+1} - \mathscr{A}_c^t\|_F^2 \leqslant F\left(\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\right),\\
&F\left(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^t, \mathscr{Z}_s^t\right) + \tfrac{\rho}{2}\|\mathscr{A}_s^{t+1} - \mathscr{A}_s^t\|_F^2 \leqslant F\left(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\right),\\
&F\left(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^t\right) + \tfrac{\rho}{2}\|\mathscr{Z}_c^{t+1} - \mathscr{Z}_c^t\|_F^2 \leqslant F\left(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^t, \mathscr{Z}_s^t\right),\\
&F\left(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}\right) + \tfrac{\rho}{2}\|\mathscr{Z}_c^{t+1} - \mathscr{Z}_c^t\|_F^2 \leqslant F\left(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^t\right).
\end{aligned} \tag{32}$$

Lemma 3 (*Relative error Lemma*). *The sequence* $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ *comes from* **Algorithm 1** *and* $\rho > 0$. *Then, there exist* $V_1^{t+1}, V_2^{t+1}, V_3^{t+1}$, *and* V_4^{t+1} *satisfying*

$$\begin{aligned}
&\|V_1^{t+1} + \nabla_{\mathscr{A}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{A}_c^{t+1} - \mathscr{A}_c^t\|_F,\\
&\|V_2^{t+1} + \nabla_{\mathscr{A}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^t, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{A}_s^{t+1} - \mathscr{A}_s^t\|_F,\\
&\|V_3^{t+1} + \nabla_{\mathscr{Z}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{Z}_c^{t+1} - \mathscr{Z}_c^t\|_F,\\
&\|V_4^{t+1} + \nabla_{\mathscr{Z}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1})\|_F \leqslant \rho\|\mathscr{Z}_s^{t+1} - \mathscr{Z}_s^t\|_F.
\end{aligned} \tag{33}$$

Finally, we give the detailed proof of **Theorem** 1.

Proof of **Theorem** *1*. Lemma 1 show that $F(\mathscr{A}_c, \mathscr{A}_s, \mathscr{Z}_c, \mathscr{Z}_s)$ satisfies the KŁ property at each point.

According to Lemmas 2 and 3, the bounded sequence $\{\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ satisfies the sufficient decrease condition and the relative error condition. Thus, we also verify the (64)-(65)-(66) in [38] by Lemmas 2 and 3. Based on the above analysis, we have completed the proof that conforms to Theorem 6.2 in [38]. Specifically, the sequence $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ globally converges to the critical point of (12).

5.3. The Computational Complexity

Now we analyze the computational complexity of the proposed **Algorithm 1**. The main computational cost depends on the calculation of subproblems $\mathscr{A}_c, \mathscr{A}_s, \mathscr{Z}_c$, and $\mathscr{Z}_s$. For $\mathscr{A}_c$ and $\mathscr{A}_s$ subproblems, the complexity are $\mathscr{O}((k_c + k_s)n_1n_2n_3 + k_cn_2^2n_3)$ and $\mathscr{O}((k_c + k_s)n_1n_2n_3 + k_sn_2^2n_3)$, respectively. For $\mathscr{Z}_c$ subproblem, the main computation cost lies in the FFT and TSVD of the tensor with size $k_c \times n_2 \times n_3$, the per-iteration complexity is $\mathscr{O}(k_cn_2n_3 \log n_3 + k_cn_2^2n_3)$. For $\mathscr{Z}_s$ subproblem, the complexity is $\mathscr{O}((k_c + k_s)n_1n_2n_3)$. Therefore, the total computational complexity is

$$\mathscr{O}\big((t + k)\big((k_c + k_s)n_1n_2n_3 + k_cn_2n_3 \log n_3 + k_cn_2^2n_3\big), \tag{34}$$

where t and k denote the total numbers of outer and inner iterations, respectively.

6. Experiments

In this section, we evaluate the performance of the proposed tensor dictionary learning method on data clustering. All numerical experiments are conducted on a computer with 16 GB of RAM and an Intel(R) Core(TM) i5-6600 M CPU and the installation of MATLAB R2021a.

Initialization and Parameter Settings. For the initialization of four variables $\mathscr{A}_c, \mathscr{A}_s, \mathscr{Z}_c$, and $\mathscr{Z}_s$ in Algorithm 1, we first use the TRPCA [43] method to roughly divide the original data into underlying data and noise. Then, based on the decomposed data and noise, we randomly choose the dictionaries and the representation coefficients from clean data and structure noise, respectively. We empirically set the numbers of atoms for clean data and structure noise to be 4 and 3 for each class in all experiments.

The proposed method involves the following parameters: $\lambda_1, \lambda_2, \beta_1, \beta_2, \beta_3, \beta_4$, and ρ. We empirically adjust the parameter λ_1 within the range $[0, 20]$ with the increment of 2 and the parameter λ_2 in $[0, 0.1]$ with the increment of 0.01. For penalties $\beta_1, \beta_2, \beta_3, \beta_4$, and ρ, we empirically choose β_1 and β_2 from $\{0.01, 0.05, 0.1\}$, $\beta_3 = 1$, and β_4 and ρ from $\{0.1, 0.2, 0.3, 0.5\}$, respectively. For the compared methods, we tune the parameters according to the authors' suggestions in their papers to obtain the best results.

Datasets Description. First, we test the proposed method on simulated datasets, which are contaminated by black blocks, stripes, sparse noise, and Gaussian noise. Then, we evaluate the proposed method on common real-world databases, including face datasets AR, handwritten alphabet dataset Alphadigits[1], object image dataset CIFAR[2], and environment image dataset UCSD[3]. All test data are normalized to $[0, 1]$.

Compared Methods. We compared the proposed TDLSN with several relevant and state-of-the-art approaches, namely, K-means based on Singular Value Decomposition (KSVD) [19], Robust Non-Negative Dictionary Learning (RNNDL) [23], Dictionary Learning with Structure Noise (DLSN) [17], K-means based on Tensor Singular Value Decomposition (KTSVD) [26], Latent Multi-view Subspace Clustering (LMSC) [30], Robust Subspace Segmentation (RSS) [33], Robust Kernel Low-Rank Representation (RKLRR) [31], Tensor Low-Rank Representation (TLRR) [2], Learnable Subspace Clustering (LeaSC) [44], and Deep Subspace Clustering (DeepSC) [45]. Among them, KSVD and RNNDL assume that the noise obeys the Gaussian distribution and Laplacian distribution, respectively. DLSN divides the noise into structure noise and additional Gaussian noise, and then uses the matrix nuclear norm and l_1-norm in their model. KTSVD is a tensor-based dictionary learning method and assumes that the noise follows the Gaussian distribution. The above four comparison methods are representative dictionary learning

[1] http://www.cs.toronto.edu/roweis/data.html.
[2] http://www.cs.utoronto.ca/ kriz/cifar.html.
[3] http://www.svcl.ucsd.edu/projects/backgroundsubtraction/ucsdbgsubdataset.htm.

methods. The four comparison methods LMSC, RSS, RKLRR, and TLRR are self-expressed subspace clustering methods. TLRR applies TNN to characterize the low-rankness of underlying data and uses l_1-norm to depict the sparsity of noise. The remaining LeaSC and DeepSC are deep subspace clustering methods.

Evaluation Metrics. For the quantitative measures of clustering results, we utilize the clustering accuracy (ACC) [12,20], the normalized mutual information (NMI), and the purity (PUR). Higher ACC, NMI, and PUR values indicate more accurate clustering results.

6.1. Simulation Experiments

We first test the performance of the proposed method on simulation experiments. We randomly select five classes of COIL dataset[4] and five classes of YALE dataset [2] as tested data. Each class contains 30 samples. We simulate structure noisy data by setting the following five types of noises:

Case 1 **(Block Structure Noise).** This case adds several 3×3 or 5×5 occlusion blocks to the selected five data of each class, and the locations of the occlusion blocks are randomly determined.

Case 2 **(Block Structure Noise + Gaussian Noise).** The Gaussian noise with zero-mean and 0.01 noise standard deviation is added to each data. The block structure noise is added as described in Case 1.

Case 3 **(Stripe Noise + Gaussian Noise).** Five selected images of each class from COIL and YALE datasets are corrupted by multiple stripes. The location of stripes is randomly determined. Gaussian noise with zero-mean and 0.01 noise standard deviation is added to each image.

Case 4 **(Block Structure Noise + Gaussian Noise + Salt and Pepper Noise).** The Gaussian noise with zero-mean and 0.01 noise standard deviation, and the salt-and-pepper noise with 0.02 noise proportion are added to each image. The block structure noise is also added as presented in Case 1.

Case 5 **(Block Structure Noise + Stripe Noise + Gaussian Noise + Salt and Pepper Noise).** The block structure noise, Gaussian noise, and salt and pepper noise are added as described in Case 4. Furthermore, we randomly select five images of each class and add multiple stripes to form the corrupted data.

In Fig. 2, we present some examples of different structural noisy data in the simulation experiments.

Tables 3 and 4 show the ACC, NMI, and PUR values of the compared methods and our method on the COIL and YALE datasets. For clarity, we use bold to highlight the best clustering results. From two tables, one can see that: (1) The dictionary learning method KTSVD obtains higher ACC, NMI, and PUR values than other dictionary learning methods KSVD, RNNDL, and DLSN. (2) The self-expression subspace clustering method TLRR achieves better clustering performance than vector-based clustering methods LMSC, RSS, and RKLRR. The main reason is that KTSVD and TLRR are tensor-based methods, in which tensors can well preserve the intrinsic structure of the high-dimensional data. Therefore, they can approximately explore the potential subspace of the high-dimensional data well. (3) The proposed method TDLSN obtains the highest ACC, NMI, and PUR values for the structure noise data clustering in most cases. In particular, for **Case 5** with serious structure noise entries, TDLSN also has obvious advantages compared with the sub-optimal method KTSVD. The reason is that KTSVD only considers the clustering of the general noise data, which treats the block structure noise, stripe structure noise, sparse noise, and Gaussian noise as a kind of noise. It also shows that this way of dealing with structure noise is ineffective. In contrast, TDLSN is effective in separating structure noise with semantic information from other noises. (4) Compared with the deep subspace clustering methods LeaSC and DeepSC, our method outperforms them in different cases of structure noise. Since LeaSC and DeepSC only use the l_2 norm to constrain the Gaussian noise, their performance is not satisfied under the influence of complex structure noise. (5) In the last column of each table, we calculate the p-value according to the numerical results of all methods. We observe that the p-values are very small, which implies that the improvement of the proposed method is significant compared with the comparison methods. In summary, the proposed method can adaptively learn dictionaries of different structure noises and can accurately estimate the structure noise to obtain better clustering results.

Table 5 presents the decomposed data obtained by different dictionary learning methods on the object dataset COIL. We observe that TDLSN removes not only the Gaussian noise but also the block structure noise. Clearly, KSVD, RNNDL, and KTSVD cannot separate structure noise from other noises, because they assume that noise only obeys a predefined distribution. Although DLSN can separate structure noise to a certain extent, the global structure of the decomposed underlying data is destroyed due to the vectorization operation in their method. As a comparison, TDLSN can preserve the overall structure of the underlying data. This also demonstrates that the proposed TDLSN can characterize the noise more precisely than the compared methods.

[4] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php.

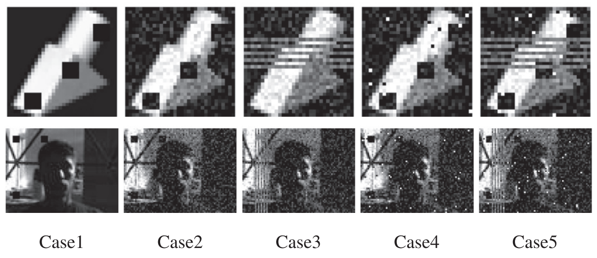


Fig. 2. The examples of different structure noisy data. The first row: COIL dataset. The second row: YALE dataset..

6.2. Real Experiments

In this part, we test the performance of all methods on real data clustering. Four real datasets are considered in our experiments. Some examples of real-world data with structure noise are presented in Fig. 3.

(1) ***AR face dataset***: This dataset has 1300 facial images of 100 subjects under various lighting conditions, where each subject contains 13 images. Most images are corrupted by glasses, sunglasses, scarves, bright lights, and other unknown noise.
(2) ***Alphadigits handwritten dataset***: This dataset contains 390 images of 26 alphabets including various forms, where each alphabet contains 15 images.
(3) ***CIFAR dataset***: This dataset is the object image dataset, which contains 32 by 32 color images in 10 different classes.
(4) ***UCSD dataset***: This dataset contains 1200 different environment images from 18 video sequences, and each image is normalized to 32 by 32.

Table 6 lists the clustering results of different methods on the real datasets. We can observe that the proposed method TDLSN gains higher ACC, NMI, and PUR values than the other competitors. Moreover, the lower p-values of the significance test also show the superiority of the proposed method.

7. Discussions

The effects of structure noise. In this section, we verify the influence of the proposed decomposition technique for handling noise. Specifically, ignoring the dictionary learning of structure noise, we rewrite the model as follows:

$$\begin{aligned} &\min_{\mathscr{A}_c,\mathscr{Z}_c} \ \tfrac{1}{2}\|\mathscr{X}-\mathscr{A}_c*\mathscr{Z}_c\|_F^2+\lambda_1\|\mathscr{Z}_c\|_*, \\ &\text{s.t.} \quad \mathscr{A}_c\in\Omega,\Omega=\left\{\mathscr{A}_c:\|\mathscr{A}c(j_1)\|_2^2\leqslant 1,j_1=1,\cdots,k_c\right\}. \end{aligned} \tag{35}$$

For convenience, we name the above model tensor dictionary learning with noise model as TDLN. Table 7 shows the numerical experiment results of methods TDLN and TDLSN on three real datasets. Obviously, the ACC, NMI, and PUR values of TDLSN are higher than those of TDLN. This demonstrates that the decomposition and characterization of structure noise is crucial for improving the clustering performance.

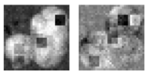
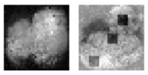
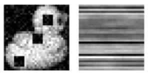
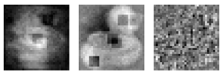
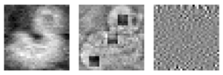
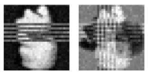
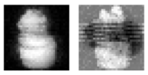
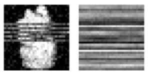
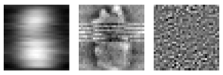
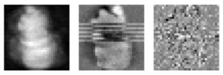
Table 3
The clustering performance (ACC, NMI, and PUR) of different methods on COIL dataset. The best results are highlighted in **bold**.

Dataset	Metrics	KSVD	RNNDL	DLSN	KTSVD	LMSC	RSS	RKLRR	TLRR	LeaSC	DeepSC	TDLSN	p-value
Case 1	ACC	0.4267	0.5180	0.5313	0.6387	0.3107	0.4100	0.5167	0.5093	0.4067	0.4687	**0.7233**	1.05e-05
	NMI	0.2880	0.2849	0.3572	0.5333	0.0649	0.1767	0.3730	0.4369	0.2130	0.3251	**0.6030**	5.99e-05
	PUR	0.4733	0.5180	0.5593	0.6387	0.3153	0.4347	0.5167	0.5453	0.4333	0.4953	**0.7233**	1.57e-05
Case 2	ACC	0.5067	0.5147	0.5367	0.6600	0.3107	0.3180	0.4073	0.5120	0.3140	0.3667	**0.7140**	5.20e-05
	NMI	0.3072	0.4033	0.3300	0.5842	0.0649	0.0787	0.2514	0.3921	0.0764	0.2037	**0.5857**	2.23e-04
	PUR	0.5067	0.5167	0.5367	0.6733	0.3153	0.3333	0.4207	0.5307	0.3313	0.3800	**0.7140**	6.17e-05
Case 3	ACC	0.4780	0.4867	0.5413	0.5327	0.3107	0.3260	0.5247	0.5540	0.3120	0.4800	**0.7333**	9.33e-06
	NMI	0.4262	0.3149	0.3657	0.4999	0.0649	0.0979	0.3382	0.3344	0.0949	0.3200	**0.5949**	1.03e-04
	PUR	0.5067	0.5133	0.5413	0.5533	0.3153	0.3527	0.5287	0.5540	0.3253	0.5067	**0.7333**	1.30e-05
Case 4	ACC	0.5093	0.3087	0.5073	0.6373	0.3107	0.3000	0.3667	0.5207	0.2807	0.3873	**0.8133**	2.66e-06
	NMI	0.3577	0.0687	0.3483	0.5221	0.0649	0.0616	0.1351	0.3586	0.0300	0.2073	**0.6791**	1.22e-05
	PUR	0.5520	0.3160	0.5073	0.6507	0.3153	0.3040	0.3987	0.5340	0.2867	0.4213	**0.8133**	4.94e-06
Case 5	ACC	0.4667	0.3733	0.5320	0.6540	0.3107	0.2980	0.3480	0.5073	0.2873	0.4400	**0.7393**	1.52e-05
	NMI	0.3272	0.2012	0.3665	0.5395	0.0649	0.0527	0.1027	0.2986	0.0381	0.2580	**0.6069**	4.23e-05
	PUR	0.5133	0.3733	0.5720	0.6540	0.3153	0.3113	0.3527	0.5073	0.2880	0.4600	**0.7393**	2.97e-05

Table 4
The clustering performance (ACC, NMI, and PUR) of different methods on YALE dataset. The best results are highlighted in **bold**.

Dataset	Metrics	KSVD	RNNDL	DLSN	KTSVD	LMSC	RSS	RKLRR	TLRR	LeaSC	DeepSC	TDLSN	*p*-value
Case 1	ACC	0.4400	0.5340	0.5007	0.8600	0.3107	0.3640	0.8047	**0.9267**	0.3733	0.7067	0.9133	1.20e-03
	NMI	0.2477	0.4559	0.4134	0.7627	0.0649	0.0922	0.6089	**0.8565**	0.1486	0.5438	0.8110	1.60e-03
	PUR	0.4827	0.5340	0.5840	0.8600	0.3153	0.3640	0.8047	**0.9267**	0.3733	0.7067	0.9133	1.30e-03
Case 2	ACC	0.5593	0.3160	0.4387	0.7133	0.3107	0.2933	0.2713	0.5587	0.6840	0.5467	**0.8400**	5.84e-05
	NMI	0.3563	0.1912	0.3351	0.6406	0.0649	0.0885	0.0445	0.3324	0.4138	0.3412	**0.7331**	2.96e-05
	PUR	0.5733	0.3533	0.5187	0.7133	0.3153	0.3333	0.2740	0.5740	0.6840	0.5467	**0.8400**	6.22e-05
Case 3	ACC	0.5167	0.4447	0.4880	0.6927	0.3107	0.3333	0.3020	0.5687	0.6653	0.5800	**0.8200**	4.30e-05
	NMI	0.3495	0.4124	0.2851	0.5932	0.0649	0.1416	0.0438	0.3544	0.4084	0.4075	**0.7182**	3.76e-05
	PUR	0.5387	0.5493	0.5013	0.6927	0.3153	0.3733	0.3127	0.5753	0.6653	0.5800	**0.8200**	4.87e-05
Case 4	ACC	0.5147	0.5200	0.4627	0.7407	0.3107	0.3400	0.2567	0.5120	0.4000	0.3087	**0.8800**	4.51e-06
	NMI	0.3937	0.5401	0.3795	0.6707	0.0649	0.1206	0.0150	0.2446	0.1776	0.0861	**0.7428**	7.68e-05
	PUR	0.5680	0.5933	0.5427	0.7407	0.3153	0.3400	0.2593	0.5127	0.4220	0.3353	**0.8800**	1.20e-05
Case 5	ACC	0.5200	0.5113	0.4787	0.5940	0.3107	0.3460	0.2547	0.4147	0.3167	0.3953	**0.7866**	1.89e-06
	NMI	0.3938	0.4427	0.2955	0.5266	0.0649	0.0992	0.0145	0.1945	0.0728	0.1269	**0.6258**	5.80e-05
	PUR	0.5600	0.5507	0.5273	0.5940	0.3153	0.3800	0.2580	0.4420	0.3347	0.4087	**0.7866**	5.55e-06

Table 5
The decomposed data obtained by different methods with case 2 and case 3 on COIL dataset.

Dataset	Observed	KSVD	RNNDL
			
	(ACC, NMI, PUR)	(0.5067, 0.3072, 0.5067)	(0.5147, 0.4033, 0.5167)
Case 2	KTSVD	DLSN	TDLSN
			
	(0.6600, 0.5842, 0.6733)	(0.5367, 0.3300, 0.5367)	(**0.7140**, **0.5857**, **0.7140**)
Dataset	Observed	KSVD	RNNDL
			
	(ACC, NMI, PUR)	(0.4780,0.4262,0.5067)	(0.4867,0.3149, 0.5133)
Case 3	KTSVD	DLSN	TDLSN
			
	(0.5327,0.4999, 0.5533)	(0.5413, 0.3657, 0.5413)	(**0.7333**,**0.5949**, **0.7333**)

(a) AR

(b) CIFAR

Fig. 3. Some examples of (a) AR datasets and (b) CIFAR datasets..

Table 6
The clustering performance (ACC, NMI, and PUR) of different methods on COIL dataset. The best results are highlighted in **bold**.

Dataset	Metrics	KSVD	RNNDL	DLSN	KTSVD	LMSC	RSS	RKLRR	TLRR	LeaSC	DeepSC	TDLSN	*p*-value
AR	ACC	0.1730	0.1699	0.2717	0.2732	0.1271	0.3479	0.3721	0.3559	0.3405	0.1760	**0.3900**	1.70e-03
	NMI	0.4167	0.4883	0.5518	0.5730	0.4625	0.5556	0.6198	0.6230	0.5980	0.5023	**0.6499**	6.93e-04
	PUR	0.1852	0.1902	0.2921	0.2933	0.1286	0.3757	0.3999	0.3784	0.3694	0.1868	**0.4105**	2.60e-03
Alphadigits	ACC	0.2141	0.1490	0.2003	0.2949	0.1846	0.1841	0.2587	0.2613	0.2167	0.2026	**0.3392**	9.24e-06
	NMI	0.3123	0.2577	0.3144	0.4341	0.3050	0.2993	0.3932	0.3996	0.3450	0.3111	**0.4819**	1.58e-05
	PUR	0.2303	0.1562	0.2126	0.3156	0.2036	0.1987	0.2723	0.2790	0.2272	0.2200	**0.3607**	9.50e-06
CIFAR	ACC	0.1510	0.1349	0.1767	0.1852	-	0.1363	0.1790	0.2032	0.1748	0.1753	**0.2266**	6.35e-05
	NMI	0.0294	0.0199	0.0475	0.0636	-	0.0173	0.0592	0.0715	0.0540	0.0422	**0.0878**	1.60e-04
	PUR	0.1610	0.1390	0.1891	0.1989	-	0.1438	0.1881	0.2083	0.1808	0.1850	**0.2317**	1.34e-04
UCSD	ACC	0.3031	0.1883	0.7628	0.8469	0.1711	0.4296	0.8341	0.8922	0.6826	0.5050	**0.9153**	3.10e-03
	NMI	0.4483	0.1838	0.8249	0.8913	0.1849	0.5415	0.8784	0.9382	0.7575	0.6392	**0.9356**	7.40e-03
	PUR	0.4257	0.1987	0.7769	0.8557	0.1843	0.5206	0.8537	0.9104	0.7385	0.5759	**0.9192**	4.80e-03

Table 7
Clustering performance (ACC, NMI, and PUR) of the three methods on test datasets.

Method	AR			Alphadigits			CIFAR		
	ACC	NMI	PUR	ACC	NMI	PUR	ACC	NMI	PUR
TDLN	0.3425	0.6231	0.3648	0.3090	0.4476	0.3346	0.1619	0.0318	0.1698
TDLSN	**0.3900**	**0.6499**	**0.4108**	**0.3392**	**0.4819**	**0.3607**	**0.2266**	**0.0878**	**0.2317**

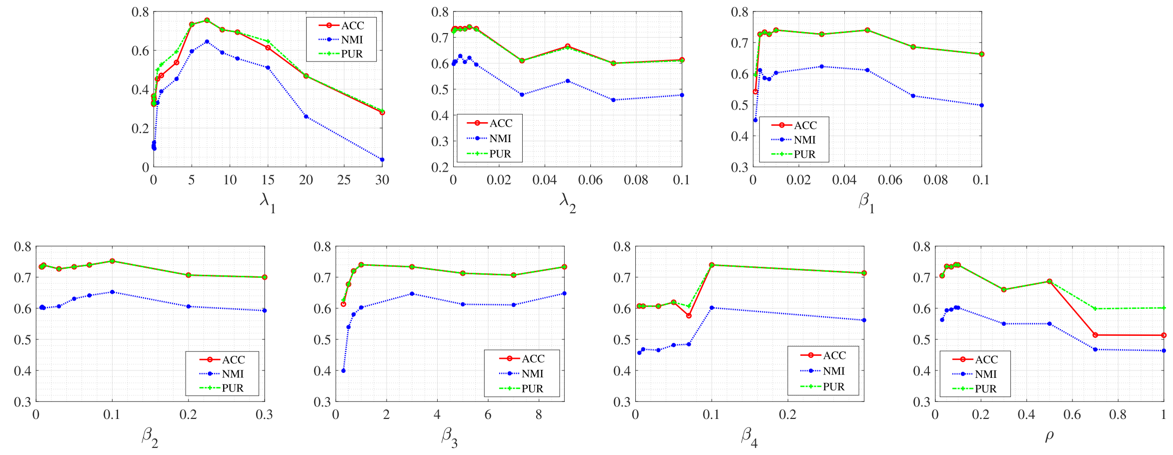


Fig. 4. The parameters analysis of the proposed method on COIL dataset with Case 5..

Table 8
The ACC, NMI, PUR, and running time (in seconds) of all algorithms on COIL and YALE datasets with ***Case*** **5**. The best results are highlighted in **bold**.

Dataset	Metrics	KSVD	RNNDL	DLSN	KTSVD	LMSC	RSS	RKLRR	TLRR	LeaSC	DeepSC	TDLSN
COIL	ACC	0.4667	0.3733	0.5320	0.6540	0.3107	0.2980	0.3480	0.5073	0.2873	0.4400	**0.7393**
	NMI	0.3272	0.2012	0.3665	0.5395	0.0649	0.0527	0.1027	0.2986	0.0381	0.2580	**0.6069**
	PUR	0.5133	0.3733	0.5720	0.6540	0.3153	0.3113	0.3527	0.5073	0.2880	0.4600	**0.7393**
	Time	0.49	4.96	4.34	312.15	2.15	**0.15**	3.76	6.88	0.53	2.94	154.93
YALE	ACC	0.5200	0.5113	0.4787	0.5940	0.3107	0.3460	0.2547	0.4147	0.3167	0.3953	**0.7866**
	NMI	0.3938	0.4427	0.2955	0.5266	0.0649	0.0992	0.0145	0.1945	0.0728	0.1269	**0.6258**
	PUR	0.5600	0.5507	0.5273	0.5940	0.3153	0.3800	0.2580	0.4420	0.3347	0.4087	**0.7866**
	Time	18.42	29.60	37.47	2833.80	20.15	0.67	**0.32**	35.87	0.98	3.18	281.37

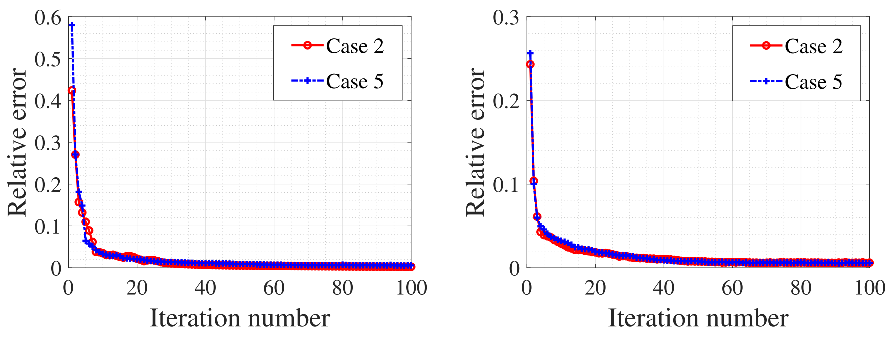


Fig. 5. The relative error curves of the proposed method on COIL and YALE datasets with Case 2 and Case 5..

Parameters analysis. The proposed method includes regularization parameters λ_1 and λ_2 and penalty parameters $\beta_1, \beta_2, \beta_3, \beta_4$, and ρ. Taking the COIL dataset with **Case 5** as an example, we test the effects of these parameters on the performance of the proposed method. Fig. 4 shows the ACC, NMI, and PUR curves obtained by the proposed method under different parameters. We have the following observations: (1) TDLSN is sensitive to different regularization parameters λ_1 and λ_2, and the highest ACC, NMI, and PUR can be achieved by the hand-tuning strategy; (2) TDLSN is robust for penalty parameters β_2 and β_3. According to the above observations, we empirically adjust the parameter λ_1 in the range $[0, 20]$ with an increment of 2 and the parameter λ_2 in $[0, 0.1]$ with an increment of 0.01. We recommend adjusting penalty parameters β_1 and β_2 in $[0, 0.1]$ with an increment of $0.01, \beta_3$ in $[0, 10]$ with an increment of 1, and β_4 and ρ in $[0, 1]$ with an increment of 0.1 for satisfactory clustering results.

The running time. In Table 8, we report the running time of different methods on the COIL and YALE datasets with **Case 5**. One can see that our method achieves the best clustering performance in terms of ACC, NMI, and PUR values although it is slower than other baselines while is still faster than KTSVD. As one of future directions, we will speed up the proposed method.

Convergence analysis. In Theorem 1, we have provided the theoretical convergence of the proposed Algorithm 1. Next, taking **Case 2** and **Case 5** of COIL and YALE datasets as examples, we study the numerical convergence of the proposed algorithm. Fig. 5 plots the relative error curves of the successive representation tensor $\mathscr{Z}_c^t$ and $\mathscr{Z}_c^{t+1}$, i.e., $\|\mathscr{Z}_c^{t+1} - \mathscr{Z}_c^t\|_F^2 / \|\mathscr{Z}_c^t\|_F^2$. We observe that the relative error curves decrease as the iteration number increases, demonstrating that the proposed algorithm also numerically converges.

8. Conclusions

To deal with high-dimensional data clustering with structure noise, we explore the tensor-based dictionary learning in the decomposition method. In particular, we decompose the tensor data into three parts: clean data, structure noise, and Gaussian noise. Meanwhile, we apply the adaptive tensor dictionary learning tool to explore the inherent global low-rankness of underlying data and the structural sparsity of structure noise. An effective algorithm is designed to solve the proposed tensor dictionary learning model with guaranteed convergence analysis. Experimental results show that the proposed method can accurately remove structure noise, thereby significantly improving the performance of data clustering compared with the baselines.

CRediT authorship contribution statement

Jing-Hua Yang: Conceptualization, Formal analysis, Methodology, Visualization, Writing - original draft. **Chuan Chen:** Supervision, Validation, Writing - review & editing. **Hong-Ning Dai:** Supervision, Validation, Writing - review & editing. **Le-Le Fu:** Formal analysis, Data curation, Writing - review & editing. **Zibin Zheng:** Supervision, Validation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (62176269), the Innovative Research Foundation of Ship General Performance (25622112), and Macao Science and Technology Development Fund under Macao Funding Scheme for Key R & D Projects (0025/2019/AKP).

Appendix A

In this section, we present the details of the $\mathscr{A}_s$ and $\mathscr{Z}_s$ subproblems.

1. $\mathscr{A}_s$-subproblem. The $\mathscr{A}_s$-subproblem is

$$\mathscr{A}_s^{t+1} = \mathrm{argmin}_{\mathscr{A}_s} \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{Z}_c^t - \mathscr{A}_s * \mathscr{Z}_s^t\|_F^2 + \frac{\rho}{2}\|\mathscr{A}_s - \mathscr{A}_s^t\|_F^2 + \Phi(\mathscr{A}_s). \tag{36}$$

We introduce the auxiliary variable $\mathscr{Y}_2$,

$$\begin{array}{ll} \mathrm{argmin}_{\mathscr{A}_s} & \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{Z}_c^t - \mathscr{A}_s * \mathscr{Z}_s^t\|_F^2 + \frac{\rho}{2}\|\mathscr{A}_s - \mathscr{A}_s^t\|_F^2 + \Phi(\mathscr{Y}_2), \\ \text{s.t.} & \mathscr{Y}_2 = \mathscr{A}_s. \end{array} \tag{37}$$

We show the corresponding augmented Lagrangian function of (37) as

$$\mathscr{L}(\mathscr{A}_s, \mathscr{Y}_2, \mathscr{C}_2) = \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{Z}_c^t - \mathscr{A}_s * \mathscr{Z}_s^t\|_F^2 + \frac{\rho}{2}\|\mathscr{A}_s - \mathscr{A}_s^t\|_F^2 + \Phi(\mathscr{Y}_2) + \frac{\beta_3}{2}\|\mathscr{Y}_2^k - \mathscr{A}_s + \frac{\mathscr{C}_2^k}{\beta}\|_F^2, \tag{38}$$

where β_3 is a penalty parameter, $\mathscr{C}_2$ denotes the Lagrangian multiplier. We update $\mathscr{A}_s, \mathscr{Y}_2$, and $\mathscr{C}_2$ by the following three subproblems.

• **Calculation of** $\mathscr{A}_s$. Given $\mathscr{Y}_2$ and $\mathscr{C}_2$, the minimization problem of $\mathscr{A}_s$ is

$$\mathscr{A}_s^{t,k+1} = \mathrm{argmin}_{\mathscr{A}_s} \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{Z}_c^t - \mathscr{A}_s * \mathscr{Z}_s^t\|_F^2 + \frac{\rho}{2}\|\mathscr{A}_s - \mathscr{A}_s^t\|_F^2 + \frac{\beta_3}{2}\|\mathscr{Y}_2^k - \mathscr{A}_s + \frac{\mathscr{C}_2^k}{\beta}\|_F^2. \tag{39}$$

The solution satisfies the following equation:

$$\mathscr{A}_s * \left(\mathscr{Z}_s^t * \left(\mathscr{Z}_s^t\right)^T + (\rho + \beta_3)\mathscr{I}\right) = \left(\mathscr{X} - \mathscr{A}_c^t * \mathscr{Z}_c^t\right) * \left(\mathscr{Z}_s^t\right)^T + \rho\mathscr{A}_s^t + \beta_3\left(\mathscr{Y}_2^k + \frac{\mathscr{C}_2^k}{\beta}\right), \tag{40}$$

which can be solved in the Fourier domain.

• **Calculation of** $\mathscr{Y}_2$. Given $\mathscr{A}_s$ and $\mathscr{C}_2$, the minimization problem of $\mathscr{Y}_2$ is

$$\mathscr{Y}_2^{k+1} = \mathrm{argmin}_{\mathscr{Y}_2} \Phi(\mathscr{Y}_2) + \frac{\beta_3}{2}\|\mathscr{Y}_2 - \mathscr{A}_s^{t,k+1} + \frac{\mathscr{C}_2^k}{\beta_3}\|_F^2, \tag{41}$$

which has the closed form solution

$$\mathscr{Y}_{2(j_2)} = \frac{M_{2(j_2)}}{\max\left\{1, \|M_{2(j_2)}\|_2^2\right\}}, j_2 = 1, 2, \ldots, k_s, \tag{42}$$

where $M_{2(j_2)} = \mathscr{A}_{s(j_2)}^{t,k+1} - \mathscr{C}_{2(j_2)}^k/\beta_3$.

• **Update multiplier** $\mathscr{C}_2$. The multiplier $\mathscr{C}_2$ can be updated by

$$\mathscr{C}_2^{k+1} = \mathscr{C}_2 + \beta_3\left(\mathscr{Y}_2^{k+1} - \mathscr{A}_s^{t,k+1}\right). \tag{43}$$

2. $\mathscr{Z}_s$-subproblem. The $\mathscr{Z}_s$-subproblem is

$$\mathscr{Z}_s^{t+1} = \mathrm{argmin}_{\mathscr{Z}_s} \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{Z}_c^{t+1} - \mathscr{A}_s^{t+1} * \mathscr{Z}_s\|_F^2 + \lambda_2\|\mathscr{Z}_s\|_{2,1,2} + \frac{\rho}{2}\|\mathscr{Z}_s - \mathscr{Z}_s^t\|_F^2. \tag{44}$$

Similarly, we employ ADMM to calculate the $\mathscr{Z}_s$-subproblem. By introducing variable $\mathscr{Q}$, we rewritten the subproblem (44) as

$$\begin{array}{ll} \min\limits_{\mathscr{Z}_s, \mathscr{Q}} & \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{Z}_c^{t+1} - \mathscr{A}_s^{t+1} * \mathscr{Q}\|_F^2 + \lambda_2\|\mathscr{Z}_s\|_{2,1,2} + \frac{\rho}{2}\|\mathscr{Z}_s - \mathscr{Z}_s^t\|_F^2, \\ \text{s.t.} & \mathscr{Q} = \mathscr{Z}_s. \end{array} \tag{45}$$

The corresponding augmented Lagrangian function of (45) is

$$\mathscr{L}(\mathscr{X}_s, \mathscr{Q}, \mathscr{F}) = \tfrac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{X}_c^{t+1} - \mathscr{A}_s^{t+1} * \mathscr{Q}\|_F^2 + \lambda_2\|\mathscr{X}_s\|_{2,1,2} + \tfrac{\rho}{2}\|\mathscr{X}_s - \mathscr{X}_s^t\|_F^2 + \tfrac{\beta_4}{2}\|\mathscr{Q}^k - \mathscr{X}_s + \tfrac{\mathscr{F}^k}{\beta_4}\|_F^2, \tag{46}$$

where $\mathscr{F}$ is Lagrangian multiplier and β_4 is penalty parameter. Next, we solve every subproblem involved in the optimization problem (46).

• **Calculation of** $\mathscr{X}_s$. Given $\mathscr{Q}$ and $\mathscr{F}$, the minimization problem of $\mathscr{X}_s$ is

$$\mathscr{X}_s^{t,k+1} = \text{argmin}_{\mathscr{X}_s} \lambda_2\|\mathscr{X}_s\|_{2,1,2} + \frac{\beta_4 + \rho}{2}\|\mathscr{X}_s - \frac{\rho\mathscr{X}_s^t + \beta_4(\mathscr{Q}^k + \mathscr{F}^k/\beta_4)}{\beta_4 + \rho}\|_F^2. \tag{47}$$

The closed-form solution is

$$\mathscr{X}_{s(j)} = \max\left\{1 - \frac{\lambda_2}{(\beta_4 + \rho)\|U(:,j,:)\|_F}, 0\right\}\mathscr{U}(:,j,:), j = 1, 2, \cdots, n_2, \tag{48}$$

where $\mathscr{U} = (\rho\mathscr{X}_s^t + \beta_4(\mathscr{Q}^k + \mathscr{F}^k/\beta_4))/(\beta_4 + \rho)$.

• **Calculation of** $\mathscr{Q}$. Given $\mathscr{X}_s$ and $\mathscr{F}$, the $\mathscr{Q}$ subproblem is

$$\mathscr{Q}^{k+1} = \text{argmin}_{\mathscr{Q}} \frac{1}{2}\|\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{X}_c^{t+1} - \mathscr{A}_s^{t+1} * \mathscr{Q}\|_F^2 + \frac{\beta_4}{2}\|\mathscr{Q} - \mathscr{X}_s^{t,k+1} + \frac{\mathscr{F}^k}{\beta_4}\|_F^2. \tag{49}$$

The minimization problem of $\mathscr{Q}$ satisfies the following equation:

$$\left((\mathscr{A}_s^{t+1})^T * \mathscr{A}_s^{t+1} + \beta_4\mathscr{I}\right) * \mathscr{Q} = (\mathscr{A}_s^{t+1})^T * (\mathscr{X} - \mathscr{A}_c^{t+1} * \mathscr{X}_c^{t+1}) + \beta_4(\mathscr{X}_s^{t,k+1} - \mathscr{F}^k/\beta_4). \tag{50}$$

Similarly, we solve it in the Fourier domain.

• **Update multiplier** $\mathscr{F}$. The multiplier $\mathscr{F}$ can be updated by

$$\mathscr{F}^{k+1} = \mathscr{F} + \beta_4(\mathscr{Q}^{k+1} - \mathscr{X}_s^{t,k+1}). \tag{51}$$

Appendix B

In this section, we present the proofs of Lemmas 1–3. Before we prove Lemmas, we first show an important property of KŁ function [38] in Lemma 4. Please refer to [38] for more details of the KŁ function.

Lemma 4 (*Theorem 3 in [46]*). *If a function f is a semi-algebraic real valued function, then it satisfies KŁ property and is KŁ function.*

Then, we prove Lemmas 1–3. We first prove the KŁ property of $F(\mathscr{A}_c, \mathscr{A}_s, \mathscr{X}_c, \mathscr{X}_s)$.

Lemma 5 (*KŁ Lemma*). *Function $F(\mathscr{A}_c, \mathscr{A}_s, \mathscr{X}_c, \mathscr{X}_s)$ satisfies the KŁ property at each point.*

Proof of Lemma 1. We verify that each term of $F(\mathscr{A}_c, \mathscr{A}_s, \mathscr{X}_c, \mathscr{X}_s)$ satisfies the KŁ property. Since norms $\|\cdot\|_F, \|\cdot\|_1, \|\cdot\|_2$ are semi-algebraic functions [46], $\|\mathscr{X} - \mathscr{A}_c * \mathscr{X}_c - \mathscr{A}_s * \mathscr{X}_s\|_F^2$ and $\|\mathscr{X}_s\|_{2,1,2}$ are semi-algebraic. Similarly, $\lambda_1\|\mathscr{X}_c\|_*$ is semi-algebraic. And $\Psi(\mathscr{A}_c)$ and $\Phi(\mathscr{A}_s)$ are semi-algebraic indicator functions [46]. Since the semi-algebraic function satisfies the KŁ property (see Lemma 4), the function $F(\mathscr{A}_c, \mathscr{A}_s, \mathscr{X}_c, \mathscr{X}_s)$ satisfies the KŁ property.

Second, we prove that the bounded sequence $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{X}_c^t, \mathscr{X}_s^t\}$ satisfies the sufficient decrease condition.

Lemma 6 (*Sufficient decrease Lemma*). *For $\rho > 0$, let $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{X}_c^t, \mathscr{X}_s^t\}$ be a sequence from* **Algorithm 1**, *then*

$$\begin{aligned}
&F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{X}_c^t, \mathscr{X}_s^t) + \tfrac{\rho}{2}\|\mathscr{A}_c^{t+1} - \mathscr{A}_c^t\|_F^2 \leqslant F(\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{X}_c^t, \mathscr{X}_s^t),\\
&F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^t, \mathscr{X}_s^t) + \tfrac{\rho}{2}\|\mathscr{A}_s^{t+1} - \mathscr{A}_s^t\|_F^2 \leqslant F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{X}_c^t, \mathscr{X}_s^t),\\
&F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^{t+1}, \mathscr{X}_s^t) + \tfrac{\rho}{2}\|\mathscr{X}_c^{t+1} - \mathscr{X}_c^t\|_F^2 \leqslant F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^t, \mathscr{X}_s^t),\\
&F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^{t+1}, \mathscr{X}_s^{t+1}) + \tfrac{\rho}{2}\|\mathscr{X}_c^{t+1} - \mathscr{X}_c^t\|_F^2 \leqslant F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^{t+1}, \mathscr{X}_s^t).
\end{aligned} \tag{52}$$

Proof of Lemma 2. According to Q_1, Q_2, Q_3, and Q_4 defined in (15), when $\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^{t+1}$, and $\mathscr{X}_s^{t+1}$ are minimizers of Q_1, Q_2, Q_3, and Q_4, we have

$$\begin{aligned}
&F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{X}_c^t, \mathscr{X}_s^t) + \tfrac{\rho}{2}\|\mathscr{A}_c^{t+1} - \mathscr{A}_c^t\|_F^2 = Q_1(\mathscr{A}_c^{t+1}|\mathscr{A}_c^t) \leqslant Q_1(\mathscr{A}_c^t|\mathscr{A}_c^t) = F(\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{X}_c^t, \mathscr{X}_s^t),\\
&F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^t, \mathscr{X}_s^t) + \tfrac{\rho}{2}\|\mathscr{A}_s^{t+1} - \mathscr{A}_s^t\|_F^2 = Q_2(\mathscr{A}_s^{t+1}|\mathscr{A}_s^t) \leqslant Q_2(\mathscr{A}_s^t|\mathscr{A}_s^t) = F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{X}_c^t, \mathscr{X}_s^t),\\
&F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^{t+1}, \mathscr{X}_s^t) + \tfrac{\rho}{2}\|\mathscr{X}_c^{t+1} - \mathscr{X}_c^t\|_F^2 = Q_3(\mathscr{X}_c^{t+1}|\mathscr{X}_c^t) \leqslant Q_3(\mathscr{X}_c^t|\mathscr{X}_c^t) = F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^t, \mathscr{X}_s^t),\\
&F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^{t+1}, \mathscr{X}_s^{t+1}) + \tfrac{\rho}{2}\|\mathscr{X}_s^{t+1} - \mathscr{X}_s^t\|_F^2 = Q_4(\mathscr{X}_s^{t+1}|\mathscr{X}_s^t) \leqslant Q_4(\mathscr{X}_s^t|\mathscr{X}_s^t)\\
&= F(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{X}_c^{t+1}, \mathscr{X}_s^t).
\end{aligned} \tag{53}$$

Third, we prove that the bounded sequence $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ satisfies the relative error condition.

Lemma 7 (*Relative error Lemma*). *The sequence* $\{\mathscr{A}_c^t, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t\}$ *comes from* **Algorithm 1** *and* $\rho > 0$. *There exist* $V_1^{t+1}, V_2^{t+1}, V_3^{t+1}$, *and* V_4^{t+1} *satisfying*

$$\begin{aligned}
&\|V_1^{t+1} + \nabla_{\mathscr{A}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{A}_c^{t+1} - \mathscr{A}_c^t\|_F,\\
&\|V_2^{t+1} + \nabla_{\mathscr{A}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^t, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{A}_s^{t+1} - \mathscr{A}_s^t\|_F,\\
&\|V_3^{t+1} + \nabla_{\mathscr{Z}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{Z}_c^{t+1} - \mathscr{Z}_c^t\|_F,\\
&\|V_4^{t+1} + \nabla_{\mathscr{Z}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1})\|_F \leqslant \rho\|\mathscr{Z}_s^{t+1} - \mathscr{Z}_s^t\|_F.
\end{aligned} \tag{54}$$

Proof of Lemma 3. Note that $\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}$, and $\mathscr{Z}_s^{t+1}$ are optimal solutions of Q_1, Q_2, Q_3, and Q_4, respectively. For each subproblem, we have

$$\begin{cases}
0 \in \partial_{\mathscr{A}_c}\Psi(\mathscr{A}_c^{t+1}) + \nabla_{\mathscr{A}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}) + \rho(\mathscr{A}_c^{t+1} - \mathscr{A}_c^t),\\
0 \in \partial_{\mathscr{A}_s}\Phi(\mathscr{A}_s^{t+1}) + \nabla_{\mathscr{A}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}) + \rho(\mathscr{A}_s^{t+1} - \mathscr{A}_s^t),\\
0 \in \partial_{\mathscr{Z}_c} f_1(\mathscr{Z}_c^{t+1}) + \nabla_{\mathscr{Z}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}) + \rho(\mathscr{Z}_c^{t+1} - \mathscr{Z}_c^t),\\
0 \in \partial_{\mathscr{Z}_s} f_2(\mathscr{Z}_s^{t+1}) + \nabla_{\mathscr{Z}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}) + \rho(\mathscr{Z}_s^{t+1} - \mathscr{Z}_s^t).
\end{cases} \tag{55}$$

Then, we define $V_1^{t+1}, V_2^{t+1}, V_3^{t+1}$, and V_4^{t+1} as

$$\begin{cases}
V_1^{t+1} := -\nabla_{\mathscr{A}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}) - \rho(\mathscr{A}_c^{t+1} - \mathscr{A}_c^t),\\
V_2^{t+1} := -\nabla_{\mathscr{A}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}) - \rho(\mathscr{A}_s^{t+1} - \mathscr{A}_s^t),\\
V_3^{t+1} := -\nabla_{\mathscr{Z}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}) - \rho(\mathscr{Z}_c^{t+1} - \mathscr{Z}_c^t),\\
V_4^{t+1} := -\nabla_{\mathscr{Z}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1}) - \rho(\mathscr{Z}_s^{t+1} - \mathscr{Z}_s^t).
\end{cases} \tag{56}$$

It is clear that $V_1^{t+1} \in \partial_{\mathscr{A}_c}\Psi(\mathscr{A}_c^{t+1}), V_2^{t+1} \in \partial_{\mathscr{A}_s}\Phi(\mathscr{A}_s^{t+1})$, $V_3^{t+1} \in \partial_{\mathscr{Z}_c} f_1(\mathscr{Z}_c^{t+1})$, and $V_4^{t+1} \in \partial_{\mathscr{Z}_s} f_2(\mathscr{Z}_s^{t+1})$. Therefore, we have

$$\begin{aligned}
&\|V_1^{t+1} + \nabla_{\mathscr{A}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^t, \mathscr{Z}_c^t, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{A}_c^{t+1} - \mathscr{A}_c^t\|_F,\\
&\|V_2^{t+1} + \nabla_{\mathscr{A}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^t, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{A}_s^{t+1} - \mathscr{A}_s^t\|_F,\\
&\|V_3^{t+1} + \nabla_{\mathscr{Z}_c} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^t)\|_F \leqslant \rho\|\mathscr{Z}_c^{t+1} - \mathscr{Z}_c^t\|_F,\\
&\|V_4^{t+1} + \nabla_{\mathscr{Z}_s} W(\mathscr{A}_c^{t+1}, \mathscr{A}_s^{t+1}, \mathscr{Z}_c^{t+1}, \mathscr{Z}_s^{t+1})\|_F \leqslant \rho\|\mathscr{Z}_s^{t+1} - \mathscr{Z}_s^t\|_F.
\end{aligned} \tag{57}$$

References

[1] S. Baek, G. Yoon, J. Song, S.M. Yoon, Self-supervised deep geometric subspace clustering network, Inf. Sci. (2022), https://doi.org/10.1016/j.ins.2022.08.006.
[2] P. Zhou, C.Y. Lu, J.S. Feng, Z.C. Lin, S.C. Yan, Tensor low-rank representation for data recovery and clustering, IEEE Trans. Pattern Anal. Mach. Intell. 43 (5) (2021) 1718–1732.
[3] L. Guo, X. Zhang, Z. Liu, X. Xue, Q. Wang, S. Zheng, Robust subspace clustering based on automatic weighted multiple kernel learning, Inf. Sci. 573 (2021) 453–474.
[4] W.B. Hu, C. Chen, F.H. Ye, Z.B. Zheng, Y.F. Du, Learning deep discriminative representations with pseudo supervision for image clustering, Inf. Sci. 568 (2021) 199–215.
[5] F. Nie, X. Wang, M. Jordan, H. Huang, The constrained Laplacian rank algorithm for graph-based clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2016, pp. 1969–1976.
[6] H.Y. Wang, G.Q. Han, J.Y. Li, B. Zhang, J.Z. Chen, Y. Hu, C. Han, H.M. Cai, Learning task-driving affinity matrix for accurate multi-view clustering through tensor subspace learning, Inf. Sci. 563 (2021) 290–308.
[7] N.D. Buono, G. Pio, Non-negative matrix tri-factorization for co-clustering, Inf. Sci. 301 (2015) 13–26.
[8] G.S. Cui, X.L. Li, Y.S. Dong, Subspace clustering guided convex nonnegative matrix factorization, Neurocomputing 292 (2018) 38–48.
[9] Y.H. Yao, C.J. Wang, Interpretable clustering on dynamic graphs with recurrent graph neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2021, pp. 4608–4616.
[10] E. Elhamifar, R. Vidal, Sparse subspace clustering: Algorithm, theory, and applications, IEEE Trans. Pattern Anal. Mach. Intell. 35 (11) (2013) 2765–2781.
[11] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2013) 171–184.
[12] L. Wang, J. Huang, M. Yin, R. Cai, Z. Hao, Block diagonal representation learning for robust subspace clustering, Inf. Sci. 526 (2020) 54–67.
[13] X. Zhang, Z. Ren, H. Sun, K. Bai, X. Feng, Z. Liu, Multiple kernel low-rank representation-based robust multiview subspace clustering, Inf. Sci. 551 (2021) 324–340.
[14] M. Brbic, I. Kopriva, Multi-view low-rank sparse subspace clustering, Pattern Recognit. 73 (2018) 247–258.
[15] C. Lu, H. Min, Z. Zhao, L. Zhu, D. Huang, S. Yan, Robust and efficient subspace segmentation via least squares regression, in: Proceedings of the European Conference on Computer Vision, 2012, pp. 347–360.
[16] K. Tang, R. Liu, Z. Su, J. Zhang, Structure-constrained low-rank representation, IEEE Trans. Neural Netw. Learn. Syst. 25 (12) (2014) 2167–2179.
[17] P. Zhou, C. Fang, Z.C. Lin, C. Zhang, E.Y. Chang, Dictionary learning with structured noise, Neurocomputing 273 (2018) 414–423.
[18] J. Lv, Z. Kang, B. Wang, L. Ji, Z. Xu, Multi-view subspace clustering via partition fusion, Inf. Sci. 560 (2021) 410–423.
[19] M. Aharon, M. Elad, A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation, IEEE Trans. Signal Process. 54 (11) (2006) 4311–4322.

[20] L.L. Fu, J.H. Yang, C. Chen, C.F. Zhang, Low-rank tensor approximation with local structure for multi-view intrinsic subspace clustering, Inf. Sci. 606 (2022) 877–891.
[21] Z. Jiang, Z. Lin, L. Davis, Label consistent K-SVD: learning a discriminative dictionary for recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (11) (2013) 2651–2664.
[22] S. Wang, Q. Liu, Y. Xia, P. Dong, J. Luo, Q. Huang, D. Feng, Dictionary learning based impulse noise removal via L_1-L_1 minimization, Signal Process. 93 (9) (2013) 2696–2708.
[23] Q. Pan, D. Kong, C. Ding, and B. Luo, Robust non-negative dictionary learning, in Proceedings of the AAAI Conference on Artificial Intelligence, 2014, pp. 2027–2033.
[24] H. Zhu, M.K. Ng, Structured dictionary learning for image denoising under mixed gaussian and impulse noise, IEEE Trans. Image Process. 29 (2020) 6680–6693.
[25] B. Cai, G.F. Lu, Tensor subspace clustering using consensus tensor low-rank representation, Inf. Sci. 609 (2022) 46–59.
[26] Z.M. Zhang, S.C. Aeron, Denoising and completion of 3D data via multidimensional dictionary learning, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2016, pp. 2371–2377.
[27] R.T. Xu, Y. Xu, Y.H. Quan, Factorized tensor dictionary learning for visual tensor data completion, IEEE Trans. Multimedia 23 (2020) 1225–1238.
[28] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[29] Z. Feng, M. Yang, L. Zhang, Y. Liu, D. Zhang, Joint discriminative dimensionality reduction and dictionary learning for face recognition, Pattern Recognit. 46 (8) (2013) 2134–2143.
[30] C.Q. Zhang, Q.H. Hu, H.Z. Fu, P.F. Zhu, X.C. Cao, Latent multi-view subspace clustering, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4333–4341.
[31] S.J. Xiao, M.K. Tan, D. Xu, Z.Y. Dong, Robust kernel low-rank representation, IEEE Trans. Neural Netw. Learn. Syst. 27 (11) (2016) 2268–2281.
[32] K. Braman, Third-order tensors as linear operators on a space of matrices, Linear Algebra Appl. 433 (7) (2010) 1241–1253.
[33] X.J. Guo, Robust subspace segmentation by simultaneously learning data representations and their affinity matrix, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2015, pp. 3547–3553.
[34] S. Soltani, M.E. Kilmer, P.C. Hansen, A tensor-based dictionary learning approach to tomographic image reconstruction, BIT Numer. Math. 56 (2016) 1425–1454.
[35] Y. Xie, W.S. Zhang, Y.Y. Qu, L.Q. Dai, D.C. Tao, Hyper-Laplacian regularized multilinear multi-view self-representation for clustering and semi-supervised learning, IEEE Trans. Cybern. 50 (2) (2020) 572–586.
[36] C.Q. Zhang, H.Z. Fu, J. Wang, W. Li, X.C. Cao, Q.H. Hu, Tensorized multi-view subspace representation learning, Int. J. Comput. Vis. 128 (8) (2020) 2344–2361.
[37] Q.X. Gao, W. Xia, Z.Z. Wan, D.Y. Xie, P. Zhang, Tensor-SVD based graph learning for multi-view subspace clustering, Proceedings of the AAAI Conference on Artificial Intelligence 34 (04) (2020) 3930–3937.
[38] H. Attouch, J. Bolte, B.F. Svaiter, Convergence of descent methods for semi-algebraic and tame poblems: Proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods, Math. Program. 137 (1–2) (2013) 91–129.
[39] H. Zhang, X.L. Zhao, T.X. Jiang, M.K. Ng, T.Z. Huang, Multiscale feature tensor train rank minimization for multidimensional image recovery, IEEE Trans. Cybernetics (2021), https://doi.org/10.1109/TCYB.2021.3108847.
[40] X.L. Zhao, J.H. Yang, T.H. Ma, T.X. Jiang, M.K. Ng, T.Z. Huang, Tensor completion via complementary global, local, and nonlocal priors, IEEE Trans. Image Process. 31 (2022) 984–999.
[41] M. Hong, Z.-Q. Luo, M. Razaviyayn, Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems, SIAM J. Optim. 26 (1) (2016) 337–364.
[42] M. Ding, T.-Z. Huang, X.-L. Zhao, M.K. Ng, T.-H. Ma, Tensor train rank minimization with nonlocal self-similarity for tensor completion, Inverse Probl. Imaging 15 (3) (2021) 475–498.
[43] C.Y. Lu, J.S. Feng, Y.D. Chen, W. Liu, Z.C. Lin, S.C. Yan, Tensor robust principal component analysis with a new tensor nuclear norm, IEEE Trans. Pattern Anal. Mach. Intell. 42 (4) (2020) 925–938.
[44] J. Li, H.F. Liu, Z.Q. Tao, H.D. Zhao, Y. Fu, Learnable subspace clustering, IEEE Trans. Neural Netw. Learn. Syst. 33 (3) (2022) 1119–1133.
[45] X. Peng, J.S. Feng, T.Y. Zhou, Y.J. Lei, S.C. Yan, Deep subspace clustering, IEEE Trans. Neural Netw. Learn. Syst. 31 (12) (2020) 5509–5521.
[46] J. Bolte, S. Sabach, M. Teboulle, Proximal alternating linearized minimization for nonconvex and nonsmooth problems, Math. Program. 46 (1) (2014) 459–494.