Contents lists available at ScienceDirect

Pattern Recognition

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

Low-rank tensor recovery via non-convex regularization, structured factorization and spatio-temporal characteristics

Quan Yu[a], Ming Yang[b,*]

[a] School of Mathematics, Hunan University, Changsha 410082, PR China
[b] Mathematics department of the University of Evansville, Evansville, IN 47722, USA

ARTICLE INFO

Article history:
Received 5 October 2022
Revised 8 December 2022
Accepted 12 January 2023
Available online 15 January 2023

2010 MSC:
15A69
47A80

Keywords:
Tensor completion
Tensor robust principle component analysis
Low-rank approximation
Dynamic background
Spatio-temporal characteristics

ABSTRACT

Recently, the convex low-rank 3rd-order tensor recovery has attracted considerable attention. However, there are some limitations to the convex relaxation approach, which may yield biased estimators. To overcome this disadvantage, we develop a novel non-convex tensor pseudo-norm to replace the weighted sum of the tensor nuclear norm as a tighter rank approximation. Then in tensor robust principle component analysis, we introduce the noise analysis to separate the spare foreground from the dynamic background more accurately. Furthermore, by introducing a spatio-temporal matrix, we can make better use of the inherent spatio-temporal characteristics of the low-rank static background and sparse foreground. Finally, we introduce an incoherent term to constrain the sparse foreground and the dynamic background to improve the separability. Some preliminary numerical examples of color image, video, and face image data sets are presented to illustrate the efficiency of our proposed methods.

1. Introduction

Recently, a great deal of mathematical efforts has been devoted to tensors, which is a high-order extension of the matrix as an important data format for multi-dimensional data applications, such as traffic data imputation [1,2], multi-class learning [3], hyperspectral image denoising [4], color image and gray video recovery [5–7], magnetic resonance imaging (MRI) data recovery [8,9], submodule clustering [10], anomaly detection [11], high dimensional signal processing [12] and multilinear subspace learning [13]. Due to the damage to the collection equipment, the interference of noise, and the difficulty of data collection, the collected data are often incomplete or grossly corrupted. Thus, we are concerned in this paper with the 3rd-order tensor recovery problem, drawing upon recent advances in low-rank tensor completion (LRTC) and tensor robust principal component analysis (TRPCA).

The LRTC problem is to find a low-rank tensor from observed incomplete data. Accordingly, its mathematical model is written as

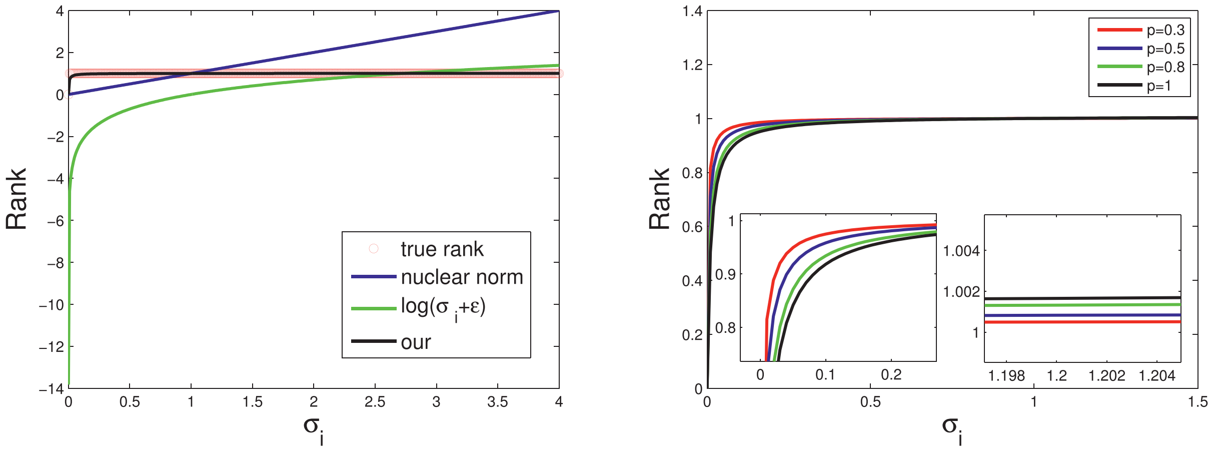
$$\min_{\mathcal{X}} \ \operatorname{rank}(\mathcal{X}), \quad s.t. \quad P_{\Omega}(\mathcal{X}) = P_{\Omega}(\mathcal{M}), \tag{1.1}$$

where $\operatorname{rank}(\cdot)$ is a tensor rank and Ω is an index set locating the observed data. P_{Ω} is a projection operator that keeps the entries of $\mathcal{X}$ in Ω and sets all others to zero.

Different tensor ranks lead to different LRTC models of (1.1) with different methods. With an eye towards application, many researchers have studied the CANDECOMP/PARAFAC (CP) rank and Tucker rank, which corresponds to CP decomposition [14], block term decomposition [15] and Tucker decomposition [16], respectively. The computation of CP rank is an NP-hard problem [17], but the Tucker rank can be obtained directly by unfolding the tensor in matrices to calculate the matrix ranks. Therefore, the LRTC problem is mostly based on Tucker rank. For example, Li et al. [18] have developed tensor nuclear norm-based methods to simultaneously recover both low Tucker rank and sparse tensors from various degraded observations. Li et al. [19] apply the alternating direction method of multipliers (ADMM) based on exact and inexact iteratively reweighted algorithms to solve non-convex ℓ_p-norm relaxation model for low Tucker rank tensor recovery problem. However, the model based Tucker rank needs direct unfolding first, which will destroy the original internal structure of the 3D-array data and lose some important information [20,21]. Recently, based on the tensor-tensor product (t-product) and tensor singular value decomposition (t-SVD) [22], Kilmer et al. [20] proposed the tensor multi-rank and tubal rank definitions. Subsequently, Zhang et al. [23] defined the tensor nuclear norm (TNN) based on t-SVD

[*] Corresponding author.
E-mail addresses: quanyu@hnu.edu.cn (Q. Yu), yangmingmath@gmail.com (M. Yang).

https://doi.org/10.1016/j.patcog.2023.109343
0031-3203/© 2023 Published by Elsevier Ltd.



(a) $p = 0.5, r = 0.01, \varepsilon = 1e\text{-}06$

(b) $r = 0.01$

Fig. 1. The contribution of different functions to the rank with respect to a varying singular value. The true rank is 1 for nonzero σ_i.

and tensor tubal rank to solve the LRTC problem, which maintains the tensor structure more effectively than the direct unfolding.

Recently, some non-convex surrogate functions based on tubal rank also have been used to approximate the rank function. For example, Jiang et al. [24] defined the partial sum of TNN (PSTNN). Cai et al. [25] developed a new t-Gamma tensor quasi-norm. Wang et al. [26] presented the weighted Schatten function. Chen et al. [27] and Yang et al. [28] proposed the weighted Schatten-p function. However, they only considered the tubal rank of mode-3 and ignored the tubal ranks of mode-1 and mode-2. Zheng et al. [29] generalize the TNN as a weighted sum of the tensor nuclear norm (WSTNN) in a balanced way, which considers all modes of tensor together. But there is a gap between the rank function and the nuclear norm, especially when the singular value is large, see Fig. 1. So the adoption of WSTNN usually leads to the approximation of the corresponding tensor N-tubal rank being insufficient.

Another typical tensor recovery problem is the TRPCA problem, which aims to recover the low-rank component and sparse component from observations. More specifically, TRPCA based on t-SVD [30] aims to recover the low tubal rank component $\mathcal{X}$ and to remove the hidden $\mathcal{E}$ resulted from the noisy observations $\mathcal{O} = \mathcal{X} + \mathcal{E}$ via the following optimization

$$\min_{\mathcal{X},\mathcal{E}} \quad \|\mathcal{X}\|_{\text{TNN}} + \lambda\|\mathcal{E}\|_1, \quad \text{s.t.} \quad \mathcal{O} = \mathcal{X} + \mathcal{E}, \tag{1.2}$$

where λ is a balancing parameter, and the sparsity of $\mathcal{E}$ is characterized by the tensor ℓ_1 norm. The TRPCA model [31–33] achieves the detection of moving targets by decomposing the video tensor $\mathcal{O}$ into a low-rank background tensor $\mathcal{X}$ and a sparse moving foreground target tensor $\mathcal{E}$. However, in most cases, a video sequence is always captured with a complex background in which the foreground objects may blend into the background [34], such as wind-blowing leaves, waves, swaying vegetation, fountains, changes in light, ripples on the water, flags flying in the wind and so on. Because the background is not completely static (that is, the background also contains dynamic components), the performance of the foreground detection method will be affected by the dynamic pixel components in the background. It is easy to misjudge the dynamic background as the foreground moving target, resulting in an incomplete and empty edge of the foreground moving object detection [35].

As has been said, we are concerned in this paper with some novel models for LRTC and TRPCA problems. For the LRTC problem, we extend the WSTNN and define a new tensor ℓ_r^p pseudo-norm, which better approximates the rank of a 3rd-order tensor, see Fig. 1. For the TRPCA problem, traditional TRPCA is very prone to voids in the process of background/foreground separation of complex scene videos and easy to misjudge the dynamic background as a moving target, which makes the separation effect not ideal. In order to address this problem, we introduce noise analysis and decompose the visual frequency sequence into three terms, low-rank static background, sparse foreground, and dynamic background. In order to make better use of their own characteristics in the low-rank static background and sparse foreground (the pixels of adjacent two frontal slices of low-rank static background are basically the same, and the pixels of adjacent two horizontal slices and lateral slices of the sparse foreground are very close), we introduce temporal and spatial matrix. At the same time, in order to more accurately separate the sparse foreground from the dynamic background and prevent the moving objects from appearing in both sparse foreground and dynamic background, we introduce an incoherent term to constrain sparse foreground and dynamic background so as to improve the separability. Below is a summary of our main contributions:

(1) For the LRTC problem, we define a novel non-convex tensor pseudo-norm to replace the WSTNN as a tighter rank approximation. Compared with WSTNN, the new tensor pseudo-norm approximates the tensor N-tubal rank better than WSTNN.
(2) For the TRPCA problem, we first introduce the noise analysis and decompose the visual frequency sequence into three terms, which are low-rank static background, sparse foreground, and dynamic background. This is beneficial to better extract foreground objects in complex scenes with a dynamic background. Then, we introduce the spatio-temporal matrix to make better use of the inherent spatio-temporal characteristics of low-rank static background and sparse foreground. Finally, we introduce an incoherent term to constrain the sparse foreground and the dynamic background to improve the separability. It will help us more accurately separate the sparse foreground from the dynamic background and prevent the moving objects from appearing in both the sparse foreground and the dynamic background.

The present paper is built up as follows. Section 2 reviews notations, basic concepts and introduces the tensor ℓ_r^p pseudo-norm. In Section 3, based on the new tensor ℓ_r^p pseudo-norm, a new model of LRTC is introduced and an alternating minimization method is proposed, where any accumulating point of the generated sequence is a Karush-Kuhn-Tucker (KKT) point. In Section 4, we improve the TRPCA model via ℓ_r^p pseudo-norm and other novel methods. Moreover, some numerical experiments on colorful image recovery, gray video recovery, face image shadow removal, and background modeling are reported in Sections 5 and 6, which illustrate the validity of our proposed models. We have also discussed the convergence behavior of our algorithms. Finally, the paper ends with concluding remarks in Section 7.

2. Preliminaries

In this section, we first summarize some notations and propose a new 3rd-order tensor pseudo-norm.

2.1. Notations

Throughout the paper, $\aleph(n) = \{1, 2, \dots, n\}$ for positive integer n. We denote scalars by lowercase letters, e.g. $a, b, c, \dots$; vectors by boldface lowercase letters, e.g. $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \dots$; and matrices by uppercase letters, e.g. $A, B, C, \dots$. 3rd-order tensors are denoted as calligraphic letters $\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots$. For a 3rd-order tensor $\mathcal{A}$, we use the Matlab notation $\mathcal{A}(i, :, :), \mathcal{A}(:, j, :)$ and $\mathcal{A}(:, :, k)$ to denote its ith horizontal, jth lateral and kth frontal slice, respectively. Let $\mathcal{A} = (\mathcal{A}_{ijk}) \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, then $(A_1^{(i)})_{jk} = (A_2^{(j)})_{ik} = (A_3^{(k)})_{ij} = \mathcal{A}_{ijk}$ for all $i \in \aleph(n_1)$, $j \in \aleph(n_2)$ and $k \in \aleph(n_3)$. The inner product of two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is the sum of the products of their entries, i.e.

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} \mathcal{A}_{ijk} \mathcal{B}_{ijk}.$$

The corresponding (Frobenius-) norm is $\|\mathcal{A}\|_F = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$. $\|\mathcal{A}\|_\infty$ represents the maximum absolute value in $\mathcal{A}$. A^T and A^{-1} represent the conjugate transpose and the inverse of A, respectively. I represents the identity matrix. For any $u \in \aleph(3)$, the u-mode matrix product of a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ with a matrix $M_u \in \mathbb{R}^{J \times n_u}$ is denoted by $\mathcal{A} \times_u M_u$ with its entries

$$(\mathcal{A} \times_1 M_1)_{i_1 jk} = \sum_{i_1=1}^{n_1} \mathcal{A}_{ijk} (M_1)_{ii_1},$$

$$(\mathcal{A} \times_2 M_2)_{ij_1 k} = \sum_{j_1=1}^{n_2} \mathcal{A}_{ijk} (M_2)_{jj_1},$$

$$(\mathcal{A} \times_3 M_3)_{ijk_1} = \sum_{k_1=1}^{n_3} \mathcal{A}_{ijk} (M_3)_{kk_1}.$$

2.2. Basic concepts

Now we review the Discrete Fourier Transformation (DFT), which plays a core role in the tensor-tensor product. For $\mathcal{A} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ and $u \in \aleph(3)$, let $\bar{\mathcal{A}}_u \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ be the result of DFT of $\mathcal{A}$ along the uth mode. Specifically, let $F_{n_u} = [\boldsymbol{f_1}, \dots, \boldsymbol{f_{n_u}}] \in \mathbb{R}^{n_u \times n_u}$, where

$$\boldsymbol{f_l} = \left[\omega^{0 \times (l-1)}; \omega^{1 \times (l-1)}; \dots; \omega^{(n_u - 1) \times (l-1)}\right] \in \mathbb{R}^{n_u},$$

$\omega = e^{-\frac{2\pi \mathfrak{b}}{n_u}}$ and $\mathfrak{b} = \sqrt{-1}$. Then

$$\bar{\mathcal{A}}_1(:, j, k) = F_{n_1} \mathcal{A}(:, j, k),$$
$$\bar{\mathcal{A}}_2(i, :, k) = F_{n_2} \mathcal{A}(i, :, k),$$
$$\bar{\mathcal{A}}_3(i, j, :) = F_{n_3} \mathcal{A}(i, j, :),$$

which can be computed by Matlab command "$\bar{\mathcal{A}}_u = \mathrm{fft}(\mathcal{A}, [\ ], u)$". Furthermore, $\mathcal{A}$ can be computed by $\bar{\mathcal{A}}_u$ with the inverse DFT $\mathcal{A} = \mathrm{ifft}(\bar{\mathcal{A}}_u, [\ ], u)$. We further define matrices $\bar{A}_1 \in \mathbb{C}^{n_1 n_2 \times n_1 n_3}, \bar{A}_2 \in \mathbb{C}^{n_1 n_2 \times n_2 n_3}, \bar{A}_3 \in \mathbb{C}^{n_1 n_3 \times n_2 n_3}$ as

$$\bar{A}_u = bdiag_u\left(\bar{\mathcal{A}}_u\right) = \begin{bmatrix} \bar{A}_u^{(1)} & & & \\ & \bar{A}_u^{(2)} & & \\ & & \ddots & \\ & & & \bar{A}_u^{(n_u)} \end{bmatrix}, \forall u \in \aleph(3).$$

Here, $bdiag_u(\cdot)$ is an operator which maps the tensor $\bar{\mathcal{A}}_u$ to the block diagonal matrix $\bar{A}_u$.

Based on these notations, mode-k t-product was introduced in [36].

Definition 2.1. For $\mathcal{A}_1 \in \mathbb{R}^{n_1 \times n_2 \times r_1}$ and $\mathcal{B}_1 \in \mathbb{R}^{n_1 \times r_1 \times n_3}$, define

$$\mathcal{A}_1 *_1 \mathcal{B}_1 := fold_1(bcirc_1(\mathcal{A}_1) \ \cdot unfold_1(\mathcal{B}_1)) \in \mathbb{R}^{n_1 \times n_2 \times n_3}.$$

For $\mathcal{A}_2 \in \mathbb{R}^{n_1 \times n_2 \times r_2}$ and $\mathcal{B}_2 \in \mathbb{R}^{r_2 \times n_2 \times n_3}$, define

$$\mathcal{A}_2 *_2 \mathcal{B}_2 := fold_2(bcirc_2(\mathcal{A}_2) \ \cdot unfold_2(\mathcal{B}_2)) \in \mathbb{R}^{n_1 \times n_2 \times n_3}.$$

For $\mathcal{A}_3 \in \mathbb{R}^{n_1 \times r_3 \times n_3}$ and $\mathcal{B}_3 \in \mathbb{R}^{r_3 \times n_2 \times n_3}$, define

$$\mathcal{A}_3 *_3 \mathcal{B}_3 := fold_3(bcirc_3(\mathcal{A}_3) \ \cdot unfold_3(\mathcal{B}_3)) \in \mathbb{R}^{n_1 \times n_2 \times n_3}.$$

Here $unfold_u(\mathcal{B}_u) = [B_u^{(1)}; B_u^{(2)}; \dots; B_u^{(n_u)}]$, and its inverse operator "$fold_u$" is defined by $fold_u(unfold_u(\mathcal{B}_u)) = \mathcal{B}_u$.

Lemma 2.1 ([22]). *Suppose that $\mathcal{A}$, $\mathcal{B}$ are tensors such that $\mathcal{F} := \mathcal{A} *_u \mathcal{B}$ $(u \in \aleph(3))$ is well defined as in Definition 2.1. Then*

(1) $\|\mathcal{A}\|_F^2 = \frac{1}{n_u} \|\bar{A}_u\|_F^2$;
(2) $\mathcal{F} = \mathcal{A} *_u \mathcal{B}$ *and* $\bar{F}_u = \bar{A}_u \bar{B}_u$ *are equivalent.*

From Lemma 2.1, we can assert that the generalized tensor factorization can be computed by matrix factorization, which is computable.

Definition 2.2 (Matrix. ℓ_r^ppseudo-norm) Given $X \in \mathbb{R}^{m \times n}$ and $m \leq n$, the matrix ℓ_r^p pseudo-norm is defined as

$$\|X\|_{\ell_r^p} := \|X\|_r^p = \sum_{i=1}^{m} \left(\frac{(1+r)\sigma_i(X)}{r + \sigma_i(X)} \right)^p, \tag{2.1}$$

where $p \in (0, 1)$ and $r > 0$.

Note that $\|X\|_r^p \to rank(X)$ as $p \to 0$ and $\gamma \to 0$, then $\|X\|_r^p$ is a non-convex approximation of rank function $rank(X)$. Fig. 1 shows several rank relaxations in the literature. Among them, the logarithmic function $\log(x + \epsilon)$ has been well studied [37]. When $p = 1$, ℓ_r^p degenerates to Kang et al. [38]. It can be seen from Fig. 1 that the matrix pseudo-norm we defined is very consistent with the real rank, and the nuclear norm deviates greatly when the singular value deviates from 1. Therefore, the proposed ℓ_r^p pseudo-norm overcomes the imbalance penalty of different singular values in the convex nuclear norm.

Based on the DFT and the ℓ_r^p pseudo-norm of the matrix, we naturally define the ℓ_r^p pseudo-norm of the tensor.

Definition 2.3 (Tensor. ℓ_r^ppseudo-norm) The tensor ℓ_r^p pseudo-norm of $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is defined as

$$\|\mathcal{A}\|_{\ell_r^p} = \sum_{u=1}^{3} \frac{1}{n_u} \left\| \bar{A}_u \right\|_{\ell_r^p} = \sum_{u=1}^{3} \frac{1}{n_u} \sum_{l=1}^{n_u} \left\| \bar{A}_u^{(l)} \right\|_{\ell_r^p} = \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\| \bar{A}_u^{(l)} \right\|_r^p. \tag{2.2}$$

Theorem 2.2 ([39]). *Let $Y \in \mathbb{R}^{m \times n}$, $m \leq n$ has the singular value decomposition (SVD) $Y = U_Y \operatorname{Diag}(\boldsymbol{\gamma}) V_Y^\top$. Then for each vector $\boldsymbol{w} \in \mathbb{R}^m$ and a scalar $\alpha > 0$, $X^* := S_v(Y, \boldsymbol{w}) = U_Y \operatorname{Diag}(s_\alpha(\boldsymbol{\gamma}, \boldsymbol{w})) V_Y^\top$ is an optimal solution of*

$$\min_{X \in \mathbb{R}^{m \times n}} f(X) := \alpha \sum_{i=1}^{m} w_i \sigma_i(X) + \frac{1}{2} \|X - Y\|_F^2,$$

where $s_\alpha(\boldsymbol{\gamma}, \boldsymbol{w}) = sign(\boldsymbol{\gamma}) \max\{0, ||\boldsymbol{\gamma}| - \alpha \boldsymbol{w}\}$.

Theorem 2.3 ([30]). *For each* $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $\bar{A}_u \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ *satisfies*

$$\begin{cases} \bar{A}_u^{(1)} \in \mathbb{C}^{n_{u_1} \times n_{u_2}}, \\ conj\left(\bar{A}_u^{(l)}\right) = \bar{A}_u^{(n_u - l + 2)}, l = 2, \ldots, \left\lfloor \frac{n_u+1}{2} \right\rfloor. \end{cases}$$

Here u, u_1, $u_2 \in \aleph(3)$, $u_1 < u_2$ *and* $u_1, u_2 \neq u$.

3. Enhanced LRTC model via ℓ_r^p pseudo-norm

In this section, we establish a new tensor completion model based on tensor ℓ_r^p pseudo-norm.

3.1. Non-convex model of tensor completion

Based on the tensor ℓ_r^p pseudo-norm, the LRTC problem reads:

$$\min_{\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}} \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\| \bar{X}_u^{(l)} \right\|_r^p, \quad s.t. \quad P_\Omega(\mathcal{X} - \mathcal{M}) = 0. \tag{3.1}$$

As the low-rank prior may not be sufficient to recover the original tensor accurately [40], we introduce a sparse prior into the tensor completion task. That is to solve the following optimization problem:

$$\min_{\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}} \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\| \bar{X}_u^{(l)} \right\|_r^p + \mu \|\mathcal{X}\|_1, \quad s.t. \quad P_\Omega(\mathcal{X} - \mathcal{M}) = 0. \tag{3.2}$$

(3.2) is difficult to solve due to the interdependent norms. Therefore, by introducing $\mathcal{Y}_1, \mathcal{Y}_2, \mathcal{Y}_3$ such that $\bar{Y}_u^{(l)} = \bar{X}_u^{(l)}, u \in \aleph(3), l \in \aleph(n_u)$, we can rephrase (3.2) as

$$\begin{aligned} &\min_{\mathcal{X}, \mathcal{Y}_u} \quad \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\| \bar{Y}_u^{(l)} \right\|_r^p + \mu \|\mathcal{X}\|_1 \\ &s.t. \quad P_\Omega(\mathcal{X} - \mathcal{M}) = 0, \bar{Y}_u^{(l)} = \bar{X}_u^{(l)}, u \in \aleph(3), l \in \aleph(n_u). \end{aligned} \tag{3.3}$$

For any $u \in \aleph(3), l \in \aleph(n_u)$, denote $\bar{K}_u^{(l)}$ as the Lagrange multiplier corresponding to the linear equality constraint $\bar{Y}_u^{(l)} = \bar{X}_u^{(l)}$, then the augmented Lagrangian of (3.3) is given as

$$\begin{aligned} &\mathcal{L}(\mathcal{X}, \mathcal{Y}_1, \mathcal{Y}_2, \mathcal{Y}_3, \mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \beta) \\ &\quad = \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\| \bar{Y}_u^{(l)} \right\|_r^p + \mu \|\mathcal{X}\|_1 + \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\langle \bar{K}_u^{(l)}, \bar{Y}_u^{(l)} - \bar{X}_u^{(l)} \right\rangle \\ &\qquad + \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{\beta}{2n_u} \left\| \bar{Y}_u^{(l)} - \bar{X}_u^{(l)} \right\|_F^2. \end{aligned} \tag{3.4}$$

Here $\beta > 0$ is the penalty parameter, and $\mathcal{K}_u$, $u \in \aleph(3)$ as the Lagrange multiplier.

Now we are ready to update $\mathcal{X}$, $\mathcal{Y}_u$ and $\mathcal{K}_u$ for all $u \in \aleph(3)$. Note that

$$\begin{aligned} &\sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\langle \bar{K}_u^{(l)}, \bar{Y}_u^{(l)} - \bar{X}_u^{(l)} \right\rangle + \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{\beta}{2n_u} \left\| \bar{Y}_u^{(l)} - \bar{X}_u^{(l)} \right\|_F^2 \\ &\quad = \sum_{u=1}^{3} \frac{1}{n_u} \left\langle \bar{K}_u, \bar{Y}_u - \bar{X}_u \right\rangle + \sum_{u=1}^{3} \frac{\beta}{2n_u} \left\| \bar{Y}_u - \bar{X}_u \right\|_F^2 \\ &\quad = \sum_{u=1}^{3} \langle \mathcal{K}_u, \mathcal{Y}_u - \mathcal{X} \rangle + \sum_{u=1}^{3} \frac{\beta}{2} \|\mathcal{Y}_u - \mathcal{X}\|_F^2. \end{aligned} \tag{3.5}$$

Thus, the sub-problem to update $\mathcal{X}$ is:

$$\begin{aligned} \mathcal{X}^* &= \mathop{arg\min}_{P_\Omega(\mathcal{X} - \mathcal{M}) = 0} \mu \|\mathcal{X}\|_1 + \sum_{u=1}^{3} \langle \mathcal{K}_u, \mathcal{Y}_u - \mathcal{X} \rangle + \sum_{u=1}^{3} \frac{\beta}{2} \|\mathcal{Y}_u - \mathcal{X}\|_F^2 \\ &= \mathop{arg\min}_{P_\Omega(\mathcal{X} - \mathcal{M}) = 0} \mu \|\mathcal{X}\|_1 + \frac{3\beta}{2} \|\mathcal{X} - \frac{1}{3} \sum_{u=1}^{3} \left(\mathcal{Y}_u + \frac{\mathcal{K}_u}{\beta} \right) \|_F^2 \\ &= P_\Omega(\mathcal{M}) + P_{\Omega^c}\left(s_{\frac{\mu}{3\beta}}(\mathcal{B}, \mathbf{1}) \right), \end{aligned} \tag{3.6}$$

where $\mathcal{B} = \frac{1}{3} \sum_{u=1}^{3} \left(\mathcal{Y}_u + \frac{\mathcal{K}_u}{\beta} \right)$ and **1** represents a vector whose elements are all 1.

Then, we update $\bar{Y}_u^{(l)}$ for all $u \in \aleph(3)$ and $l \in \aleph(n_u)$, that is

$$\min_{\bar{Y}_u^{(l)}} \left\| \bar{Y}_u^{(l)} \right\|_r^p + \left\langle \bar{K}_u^{(l)}, \bar{Y}_u^{(l)} - \bar{X}_u^{(l)} \right\rangle + \frac{\beta}{2} \left\| \bar{Y}_u^{(l)} - \bar{X}_u^{(l)} \right\|_F^2, \tag{3.7}$$

which is equivalent to

$$\min_{\bar{Y}_u^{(l)}} \left\| \bar{Y}_u^{(l)} \right\|_r^p + \frac{\beta}{2} \left\| \bar{Y}_u^{(l)} - \bar{X}_u^{(l)} + \frac{1}{\beta} \bar{K}_u^{(l)} \right\|_F^2. \tag{3.8}$$

Since for any given $p \in (0, 1)$ and $r > 0$, $\left(\frac{(1+r)\sigma_s(\bar{Y}_u^{(l)})}{r + \sigma_s(\bar{Y}_u^{(l)})}\right)^p$ is a concave function about $\sigma_s(\bar{Y}_u^{(l)})$, we can deduce that

$$\begin{aligned} \|\bar{Y}_u^{(l)}\|_r^p \leq \|\bar{X}_u^{(l)} - \frac{1}{\beta} \bar{K}_u^{(l)}\|_r^p \\ + pr(1+r)^p \sum_{s=1}^{n_{\bar{u}}} \frac{(w_s)^{p-1}}{(w_s + r)^{p+1}} \left(\sigma_s\left(\bar{Y}_u^{(l)}\right) - w_s \right), \end{aligned}$$

where $\bar{Y}_u^{(l)} \in \mathbb{R}^{n_{u_1} \times n_{u_2}}$, $n_{\bar{u}} = \min\left\{n_{u_1}, n_{u_2}\right\}$ and $w_s = \sigma_s(\bar{X}_u^{(l)} - \frac{1}{\beta} \bar{K}_u^{(l)})$.

Instead of minimizing (3.8), we obtain $\bar{Y}_u^{(l)}$ by minimizing the first order approximation of (3.8), i.e.,

$$\min_{\bar{Y}_u^{(l)}} pr(1+r)^p \sum_{s=1}^{n_{\bar{u}}} \frac{(w_s)^{p-1}}{(w_s + r)^{p+1}} \sigma_s\left(\bar{Y}_u^{(l)}\right) + \frac{\beta}{2} \|\bar{Y}_u^{(l)} - \bar{X}_u^{(l)} + \frac{1}{\beta} \bar{K}_u^{(l)}\|_F^2.$$

By Lemma 2.2, we can update $\bar{Y}_u^{(l)}$ by

$$\bar{Y}_u^{(l)*} = S_{\frac{pr(1+r)^p}{\beta}} \left(\bar{X}_u^{(l)} - \frac{1}{\beta} \bar{K}_u^{(l)}, \frac{\boldsymbol{w}^{p-1}}{(\boldsymbol{w} + r)^{p+1}} \right), u \in \aleph(3), l \in \aleph(n_u). \tag{3.9}$$

By using (3.9) and Lemma 2.3, we can update $\bar{Y}_u^{(l)}$ by

$$\bar{Y}_u^{(l)*} = \begin{cases} S_{\frac{pr(1+r)^p}{\beta}} \left(\bar{X}_u^{(l)} - \frac{1}{\beta} \bar{K}_u^{(l)}, \frac{\boldsymbol{w}^{p-1}}{(\boldsymbol{w}+r)^{p+1}} \right), & l = 1, \ldots, \left\lceil \frac{n_u+1}{2} \right\rceil, \\ conj\left(\bar{Y}_u^{(n_u - l + 2)}\right), & l = \left\lceil \frac{n_u+1}{2} \right\rceil + 1, \ldots, n_u. \end{cases} \tag{3.10}$$

Based on the above discussions, the ADMM method can be outlined as Algorithm 3.1.

3.2. Complexity analysis

The computational complexity of the proposed Algorithm 3.1 is shown as follows: $\mathcal{X}$ and $\mathcal{Y}_u$, $u \in \aleph(3)$ are both of size $n_1 \times n_2 \times n_3$. Updating $\mathcal{Y}_u$ needs to perform the fast Fourier transformation (FFT), inverse FFT, and singular value decomposition with the cost of $\mathcal{O}\left(n_1 n_2 n_3 \sum_{u=1}^{3} (\log(n_u) + \frac{1}{2} \min\{n_{u_1}, n_{u_2}\})\right)$. Here u_1, $u_2 \in \aleph(3)$, $u_1 < u_2$ and $u_1, u_2 \neq u$. Moreover, it takes $\mathcal{O}(n_1 n_2 n_3)$ to update $\mathcal{X}$. Thus, the overall computational complexity of LPRN is $\mathcal{O}\left(n_1 n_2 n_3 \sum_{u=1}^{3} (\log(n_u) + \frac{1}{2} \min\{n_{u_1}, n_{u_2}\})\right)$.

Algorithm 3.1 Solve the non-convex LRTC model (LPRN).

```
Input: The tensor data $\mathcal{M} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the observed set$\Omega$, $p \in (0,1)$, $r > 0$.
Initialize: $\mathcal{X}^0,\ \mathcal{Y}_u^0,\ \mathcal{K}_u^0,\ u \in \aleph(3)$.
While not converge do
    1. Compute $\mathcal{X}^{t+1}$ by (3.6).
    2. Compute each frontal slice of $\bar{\mathcal{Y}}_u^{t+1}$ by (3.10).
    3. Compute $\mathcal{Y}_u^{t+1} = \mathrm{ifft}(\bar{\mathcal{Y}}_u^{t+1}, [\ ], u)$.
    4. Update $\mathcal{K}_u^{t+1}$ by $\mathcal{K}_u^{t+1} = \mathcal{K}_u^t - \beta^t \left(\mathcal{X}^{t+1} - \mathcal{Y}_u^{t+1}\right)$
    5. Update $\beta^{t+1}$ by $\beta^{t+1} = \kappa \beta^t$.
    6. Check the convergence condition $\left\|\mathcal{X}^{t+1} - \mathcal{X}^t\right\|_\infty < \varepsilon$, $\left\|\mathcal{X}^{t+1} - \mathcal{Y}_u^{t+1}\right\|_\infty < \varepsilon$.
    7. $t \leftarrow t+1$.
end while
Output: $\mathcal{X}^{t+1}$.
```

3.3. Convergence analysis

We are now turning to show the convergence of Algorithm 3.1. By using the sub-differential [41], we first present the following KKT conditions for (3.3):

$$\begin{cases} P_{\Omega^c}\left(\mu \partial \|\mathcal{X}\|_1 - \sum\limits_{i=1}^{3} \mathcal{K}_u\right) = 0, \\ P_{\Omega}\left(\mu \partial \|\mathcal{X}\|_1 - \sum\limits_{i=1}^{3} \mathcal{K}_u\right) + \mathcal{Q} = 0, \\ \partial\left(\sum\limits_{l=1}^{n_u} \frac{1}{n_u} \left\|\bar{Y}_u^{(l)}\right\|_r^p\right) + \mathcal{K}_u = 0,\ u \in \aleph(3), \\ \mathcal{X} = \mathcal{Y}_u,\ u \in \aleph(3), \\ P_\Omega(\mathcal{X} - \mathcal{M}) = 0, \end{cases}$$

where $\mathcal{Q}$ and $\mathcal{K}_u$ are Lagrange multipliers. It thus can be used to determine the stop conditions for Algorithm 3.1, which are $\left\|\mathcal{X}^{t+1} - \mathcal{X}^t\right\|_\infty < \varepsilon$, $\left\|\mathcal{X}^{t+1} - \mathcal{Y}_u^{t+1}\right\|_\infty < \varepsilon$.

For convenience, we denote $g_{u,l}(\mathcal{X}) = \frac{1}{n_u}\left\|\bar{X}_u^{(l)}\right\|_r^p$, then (3.2) and (3.3) can be rewritten as

$$\min_{\mathcal{X}} \sum_{u=1}^{3} \sum_{l=1}^{n_u} g_{u,l}(\mathcal{X}) + \mu \|\mathcal{X}\|_1, \quad s.t. \quad P_\Omega(\mathcal{X} - \mathcal{M}) = 0$$

and

$$\min_{\mathcal{X},\mathcal{Y}_u} \sum_{u=1}^{3} \sum_{l=1}^{n_u} g_{u,l}(\mathcal{Y}_u) + \mu \|\mathcal{X}\|_1,$$

$$s.t. \quad P_\Omega(\mathcal{X} - \mathcal{M}) = 0,\ \bar{Y}_u^{(l)} = \bar{X}_u^{(l)},\ u \in \aleph(3),\ l \in \aleph(n_u), \tag{3.11}$$

which can be viewed as the 3rd-order tensor case for the non-convex consensus problem discussed in Hong et al. [42] with $h(\boldsymbol{x}) = \mu\|\boldsymbol{x}\|_1$:

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}) := \sum_{i=1}^{K} g_i(\boldsymbol{x}) + h(\boldsymbol{x}), \quad s.t. \quad \boldsymbol{x} \in X$$

and

$$\min_{\boldsymbol{x},\boldsymbol{x_i} \in \mathbb{R}^n} \sum_{i=1}^{K} g_i(\boldsymbol{x_i}) + h(\boldsymbol{x}), \quad s.t. \quad \boldsymbol{x_i} = \boldsymbol{x},\ \forall i \in \aleph(K),\ \boldsymbol{x} \in X. \tag{3.12}$$

Here $g_i : \mathbb{R}^n \to \mathbb{R}$ is non-convex and smooth for $i \in \aleph(K)$, $h : \mathbb{R}^n \to \mathbb{R}$ is convex but non-smooth. It is worth mentioning that Algorithm 3.1 proposed here can be viewed as a generation of classical proximal ADMM proposed by Hong et al. [42] for solving (3.12) from vector case to tensor case. Similar to the proof in Section 2.3 of [42] for the convergence results of classical proximal ADMM, the convergence results of Algorithm 3.1 is: If β^t is sufficient large, the accumulation point $\left(\mathcal{X}^*, \mathcal{Y}_1^*, \mathcal{Y}_2^*, \mathcal{Y}_3^*, \mathcal{K}_1^*, \mathcal{K}_2^*, \mathcal{K}_3^*\right)$ of sequence $\left\{\mathcal{X}^t, \mathcal{Y}_1^t, \mathcal{Y}_2^t, \mathcal{Y}_3^t, \mathcal{K}_1^t, \mathcal{K}_2^t, \mathcal{K}_3^t\right\}$ generated by Algorithm 3.1 is a KKT point of (3.11).

4. Improved TRPCA model via ℓ_r^p pseudo-norm and other novel methods

4.1. Non-convex model of TRPCA

Another typical tensor recovery problem is the TRPCA, which aims to recover the low-rank component $\mathcal{X}$ and sparse component $\mathcal{E}$ from observations $\mathcal{O} = \mathcal{X} + \mathcal{E} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$. Adopting the ℓ_r^p pseudo-norm to characterize the low-rank part, our TRPCA model is formulated as

$$\min_{\mathcal{X},\mathcal{E}} \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\|\bar{X}_u^{(l)}\right\|_r^p + \lambda \|\mathcal{E}\|_1, \quad s.t. \quad \mathcal{O} = \mathcal{X} + \mathcal{E}. \tag{4.1}$$

In order to better extract the foreground objects in complex scenes with dynamic backgrounds, we introduce the noise analysis and decompose the visual frequency sequence $\mathcal{O}$ into three terms, i.e., $\mathcal{O} = \mathcal{X} + \mathcal{E} + \mathcal{V}$, where $\mathcal{V}$ is the dynamic background. Then we can establish the following improved TRPCA model:

$$\min_{\mathcal{X},\mathcal{E},\mathcal{V}} \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\|\bar{X}_u^{(l)}\right\|_r^p + \lambda \|\mathcal{E}\|_1 + \rho \|\mathcal{V}\|_F^2, \quad s.t. \quad \mathcal{O} = \mathcal{X} + \mathcal{E} + \mathcal{V}. \tag{4.2}$$

For TRPCA, there are some own characteristics in low-rank tensor $\mathcal{X}$ and sparse tensor $\mathcal{E}$. For example, Fig. 2 shows that tensor $\mathcal{O}$ is decomposed into low-rank tensor $\mathcal{X}$ and sparse tensor $\mathcal{E}$, from which we can see the pixels of adjacent two frontal slices of low-rank tensor $\mathcal{X}$ are basically the same, and the pixels of adjacent two horizontal slices and lateral slices of sparse tensor $\mathcal{E}$ are very close. Hence, we introduce three matrices $F \in \mathbb{R}^{n_1 \times n_1}$, $G \in \mathbb{R}^{n_2 \times n_2}$ and $H \in \mathbb{R}^{n_3 \times n_3}$ to adapt the data set. At the same time, in order to more accurately separate the sparse foreground from the dynamic background and prevent the moving objects from appearing in both $\mathcal{E}$ and $\mathcal{V}$, we introduce an incoherent term $\varpi(\mathcal{E}, \mathcal{V}) := -\|\mathcal{E} - \mathcal{V}\|_F^2$ to constrain $\mathcal{E}$ and $\mathcal{V}$ so as to improve the separability. Notice that $\varpi(\mathcal{E}, \mathcal{V}) = -\|\mathcal{E} - \mathcal{V}\|_F^2 = 2\langle \mathcal{E}, \mathcal{V}\rangle - \|\mathcal{E}\|_F^2 - \|\mathcal{V}\|_F^2$. Since $\|\mathcal{E}\|_1$ and $\|\mathcal{V}\|_F^2$ appear in the objective function of (4.2), and the equivalence of $\|\cdot\|_1$ and $\|\cdot\|_F$, it suffices to adopt $\langle \mathcal{E}, \mathcal{V}\rangle$ to measure $\varpi(\mathcal{E}, \mathcal{V})$.

Then the following improved TRPCA model reads:

$$\begin{aligned} \min_{\mathcal{X},\mathcal{E},\mathcal{V}} \quad & \sum_{u=1}^{3} \sum_{l=1}^{n_u} \frac{1}{n_u} \left\|\bar{X}_u^{(l)}\right\|_r^p + \lambda \|\mathcal{E}\|_1 + \rho \|\mathcal{V}\|_F^2 + \delta \langle \mathcal{E}, \mathcal{V}\rangle \\ & + \frac{\eta_1}{2} \|\mathcal{E} \times_1 F\|_F^2 + \frac{\eta_2}{2} \|\mathcal{E} \times_2 G\|_F^2 + \frac{\eta_3}{2} \|\mathcal{X} \times_3 H\|_F^2 \\ s.t. \quad & \mathcal{O} = \mathcal{X} + \mathcal{E} + \mathcal{V}. \end{aligned} \tag{4.3}$$

For simplicity, the following lemma is introduced.

Lemma 4.1 ([36]). *Suppose that* $\mathcal{C} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $F \in \mathbb{R}^{n_1 \times n_1}$, $G \in \mathbb{R}^{n_2 \times n_2}$ *and* $H \in \mathbb{R}^{n_3 \times n_3}$. *Let* $\mathcal{F} \in \mathbb{R}^{n_1 \times n_2 \times n_1}$, $\mathcal{G} \in \mathbb{R}^{n_2 \times n_2 \times n_3}$, $\mathcal{H} \in \mathbb{R}^{n_1 \times n_3 \times n_3}$ *with their slices*

$$F_2^{(1)} = F,\ F_2^{(2)} = \ldots = F_2^{(n_2)} = 0,$$

$$G_3^{(1)} = G^T,\ G_3^{(2)} = \ldots = G_3^{(n_3)} = 0,$$

$$H_1^{(1)} = H^T,\ H_1^{(2)} = \ldots = H_1^{(n_1)} = 0.$$

We now see

$$\mathcal{F} *_2 \mathcal{C} = \mathcal{C} \times_1 F, \quad \mathcal{C} *_3 \mathcal{G} = \mathcal{C} \times_2 G, \quad \mathcal{C} *_1 \mathcal{H} = \mathcal{C} \times_3 H.$$

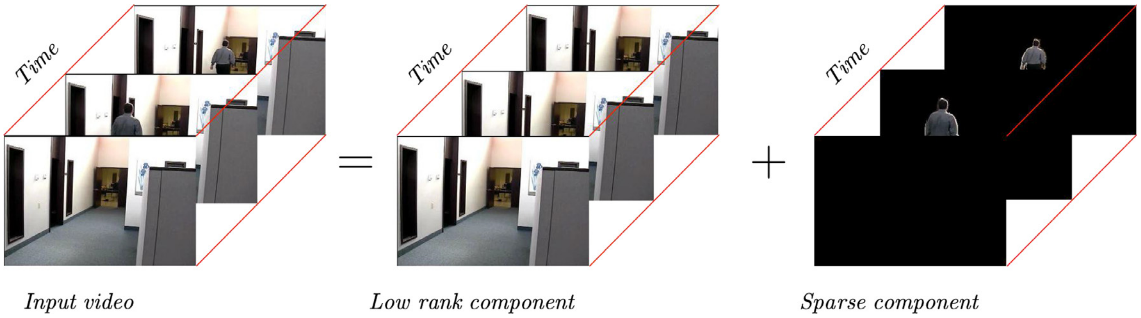


Fig. 2. Description of the temporal and spatial characteristics of tensors.

By using Lemma 4.1, (4.3) can be rewritten as

$$\begin{aligned}\min_{\mathcal{X},\mathcal{E},\mathcal{V}} \quad & \sum_{u=1}^{3}\sum_{l=1}^{n_u}\frac{1}{n_u}\left\|\bar{X}_u^{(l)}\right\|_r^p+\lambda\|\mathcal{E}\|_1+\rho\|\mathcal{V}\|_F^2+\delta\langle\mathcal{E},\mathcal{V}\rangle \\ & +\frac{\eta_1}{2}\|\mathcal{F}*_2\mathcal{E}\|_F^2+\frac{\eta_2}{2}\|\mathcal{E}*_3\mathcal{G}\|_F^2+\frac{\eta_3}{2}\|\mathcal{X}*_1\mathcal{H}\|_F^2 \\ s.t. \quad & \mathcal{O}=\mathcal{X}+\mathcal{E}+\mathcal{V}.\end{aligned} \tag{4.4}$$

From Lemma 2.1, we can rephrase (4.4) as

$$\begin{aligned}\min_{\mathcal{X},\mathcal{E},\mathcal{V}} \quad & \sum_{u=1}^{3}\sum_{l=1}^{n_u}\frac{1}{n_u}\left\|\bar{X}_u^{(l)}\right\|_r^p+\lambda\|\mathcal{E}\|_1+\rho\|\mathcal{V}\|_F^2+\delta\langle\mathcal{E},\mathcal{V}\rangle \\ & +\frac{\eta_1}{2n_2}\sum_{j=1}^{n_2}\left\|\bar{F}_2^{(j)}\bar{E}_2^{(j)}\right\|_F^2+\frac{\eta_2}{2n_3}\sum_{k=1}^{n_3}\left\|\bar{E}_3^{(k)}\bar{G}_3^{(k)}\right\|_F^2 \\ & +\frac{\eta_3}{2n_1}\sum_{i=1}^{n_1}\left\|\bar{X}_1^{(i)}\bar{H}_1^{(i)}\right\|_F^2 \\ s.t. \quad & \mathcal{O}=\mathcal{X}+\mathcal{E}+\mathcal{V}.\end{aligned} \tag{4.5}$$

Eq. (4.5) is difficult to solve due to the interdependent norms. Therefore, we introduce $\mathcal{Y}_u,\mathcal{Z}_u,u\in\aleph(3)$ such that

$$\bar{Y}_u^{(l)}=\bar{X}_u^{(l)},u\in\aleph(3),l\in\aleph(n_u),\ \bar{Z}_1^{(i)}=\bar{X}_1^{(i)},i\in\aleph(n_1),$$
$$\bar{Z}_u^{(l)}=\bar{E}_u^{(l)},u=2,3,l\in\aleph(n_u).$$

We can rephrase (4.5) as

$$\begin{aligned}\min_{\mathcal{X},\mathcal{Y}_u,\mathcal{Z}_u,\mathcal{E},\mathcal{V}} \quad & \sum_{u=1}^{3}\sum_{l=1}^{n_u}\frac{1}{n_u}\left\|\bar{Y}_u^{(l)}\right\|_r^p+\lambda\|\mathcal{E}\|_1+\rho\|\mathcal{V}\|_F^2+\delta\langle\mathcal{E},\mathcal{V}\rangle \\ & +\frac{\eta_1}{2n_2}\sum_{j=1}^{n_2}\left\|\bar{F}_2^{(j)}\bar{Z}_2^{(j)}\right\|_F^2+\frac{\eta_2}{2n_3}\sum_{k=1}^{n_3}\left\|\bar{Z}_3^{(k)}\bar{G}_3^{(k)}\right\|_F^2 \\ & +\frac{\eta_3}{2n_1}\sum_{i=1}^{n_1}\left\|\bar{Z}_1^{(i)}\bar{H}_1^{(i)}\right\|_F^2 \\ s.t. \quad & \mathcal{O}=\mathcal{X}+\mathcal{E}+\mathcal{V},\bar{Y}_u^{(l)}=\bar{X}_u^{(l)},u\in\aleph(3),l\in\aleph(n_u), \\ & \bar{Z}_1^{(i)}=\bar{X}_1^{(i)},i\in[n_1],\ \bar{Z}_u^{(l)}=\bar{E}_u^{(l)},u=2,3,l\in\aleph(n_u).\end{aligned} \tag{4.6}$$

For any $u\in\aleph(3),l\in\aleph(n_u)$, let us denote $\mathcal{S},\bar{K}_u^{(l)},\bar{W}_u^{(l)}$ as the Lagrange multipliers correspondingly. Then the augmented Lagrangian of (4.6) is given as

$$\begin{aligned}&\mathcal{L}(\mathcal{X},\mathcal{E},\mathcal{V},\mathcal{Y}_1,\mathcal{Y}_2,\mathcal{Y}_3,\mathcal{Z}_1,\mathcal{Z}_2,\mathcal{Z}_3,\mathcal{S},\mathcal{K}_1,\mathcal{K}_2,\mathcal{K}_3,\mathcal{W}_1,\mathcal{W}_2,\mathcal{W}_3,\beta) \\ &=\sum_{u=1}^{3}\sum_{l=1}^{n_u}\frac{1}{n_u}\left\|\bar{Y}_u^{(l)}\right\|_r^p+\lambda\|\mathcal{E}\|_1+\rho\|\mathcal{V}\|_F^2+\delta\langle\mathcal{E},\mathcal{V}\rangle \\ &\quad+\frac{\eta_1}{2n_2}\sum_{j=1}^{n_2}\left\|\bar{F}_2^{(j)}\bar{Z}_2^{(j)}\right\|_F^2+\frac{\eta_2}{2n_3}\sum_{k=1}^{n_3}\left\|\bar{Z}_3^{(k)}\bar{G}_3^{(k)}\right\|_F^2 \\ &\quad+\frac{\eta_3}{2n_1}\sum_{i=1}^{n_1}\left\|\bar{Z}_1^{(i)}\bar{H}_1^{(i)}\right\|_F^2+\frac{\beta}{2}\left\|\mathcal{X}+\mathcal{E}+\mathcal{V}-\mathcal{O}+\frac{\mathcal{S}}{\beta}\right\|_F^2 \\ &\quad+\sum_{u=1}^{3}\sum_{l=1}^{n_u}\frac{\beta}{2n_u}\left\|\bar{Y}_u^{(l)}-\bar{X}_u^{(l)}+\frac{\bar{K}_u^{(l)}}{\beta}\right\|_F^2 \\ &\quad+\sum_{i=1}^{n_1}\frac{\beta}{2n_1}\left\|\bar{Z}_1^{(i)}-\bar{X}_1^{(i)}+\frac{\bar{W}_1^{(i)}}{\beta}\right\|_F^2 \\ &\quad+\sum_{u=2}^{3}\sum_{l=1}^{n_u}\frac{\beta}{2n_u}\left\|\bar{Z}_u^{(l)}-\bar{E}_u^{(l)}+\frac{\bar{W}_u^{(l)}}{\beta}\right\|_F^2.\end{aligned} \tag{4.7}$$

Here $\beta>0$ is the penalty parameter, and $\mathcal{S},\mathcal{K}_u,\mathcal{W}_u,\ u\in\aleph(3)$ as the Lagrange multipliers respectively.

Clearly, $\mathcal{Y}_u,u\in\aleph(3)$ can be updated by (3.10). Now, we consider how to update $\mathcal{X},\mathcal{E},\mathcal{S},\mathcal{Z}_u$ and $\mathcal{W}_u$ for $u\in\aleph(3)$. Similar to (3.5), $\mathcal{X}$ can be updated by

$$\begin{aligned}\mathcal{X}^* &= arg\min_{\mathcal{X}}\ \frac{5\beta}{2}\|\mathcal{X}-\frac{1}{5}\sum_{u=1}^{3}\left(\mathcal{Y}_u+\frac{\mathcal{K}_u}{\beta}\right) \\ &\quad-\frac{1}{5}\left(\mathcal{O}-\mathcal{E}-\mathcal{V}-\frac{\mathcal{S}}{\beta}+\mathcal{Z}_1+\frac{\mathcal{W}_1}{\beta}\right)\|_F^2 \\ &=\frac{1}{5}\sum_{u=1}^{3}\left(\mathcal{Y}_u+\frac{\mathcal{K}_u}{\beta}\right)+\frac{1}{5}\left(\mathcal{O}-\mathcal{E}-\mathcal{V}-\frac{\mathcal{S}}{\beta}+\mathcal{Z}_1+\frac{\mathcal{W}_1}{\beta}\right).\end{aligned} \tag{4.8}$$

The sub-problem to update $\mathcal{E}$ is:

$$\mathcal{E}^*=\arg\min_{\mathcal{E}}\ \lambda\|\mathcal{E}\|_1+\frac{3\beta}{2}\|\mathcal{E}-\mathcal{D}\|_F^2=s_{\frac{\lambda}{3\beta}}(\mathcal{D}), \tag{4.9}$$

where

$$\mathcal{D}=\frac{1}{3}\left(\mathcal{O}-\mathcal{X}-\frac{\beta+\delta}{\beta}\mathcal{V}-\frac{\mathcal{S}}{\beta}\right)+\frac{1}{3}\sum_{u=2}^{3}\left(\mathcal{Z}_u+\frac{\mathcal{W}_u}{\beta}\right).$$

The sub-problem to update $\mathcal{V}$ is:

$$\begin{aligned}\mathcal{V}^* &= \arg\min_{\mathcal{V}}\ \rho\|\mathcal{V}\|_F^2+\frac{\beta}{2}\left\|\mathcal{X}+\mathcal{E}+\mathcal{V}-\mathcal{O}+\frac{\mathcal{S}}{\beta}\right\|_F^2+\delta\langle\mathcal{E},\mathcal{V}\rangle \\ &=\frac{\beta}{2\rho+\beta}\left(\mathcal{O}-\mathcal{X}-\mathcal{E}-\frac{\mathcal{S}+\delta\mathcal{E}}{\beta}\right).\end{aligned} \tag{4.10}$$

For any $i\in\aleph(n_1)$, $\bar{Z}_1^{(i)}$ is updated by

$$\bar{Z}_1^{(i)*}=\left(\beta\bar{X}_1^{(i)}-\bar{W}_1^{(i)}\right)\left(\eta_3\bar{H}_1^{(i)}\left(\bar{H}_1^{(i)}\right)^T+\beta I\right)^{-1}. \tag{4.11}$$

For any $j\in\aleph(n_2)$, $\bar{Z}_2^{(j)}$ is updated by

$$\bar{Z}_2^{(j)*}=\left(\eta_1\left(\bar{F}_2^{(j)}\right)^T\bar{F}_2^{(j)}+\beta I\right)^{-1}\left(\beta\bar{E}_2^{(j)}-\bar{W}_2^{(j)}\right). \tag{4.12}$$

For any $k\in\aleph(n_3)$, $\bar{Z}_3^{(k)}$ is updated by

$$\bar{Z}_3^{(k)*}=\left(\beta\bar{E}_3^{(k)}-\bar{W}_3^{(k)}\right)\left(\eta_2\bar{G}_3^{(k)}\left(\bar{G}_3^{(k)}\right)^T+\beta I\right)^{-1}. \tag{4.13}$$

According to (4.11), (4.12), (4.13) and Lemma 2.3, we can update $\bar{Z}_u^{(l)}$ by

$$\bar{Z}_u^{(l)*} = \begin{cases} \bar{Z}_u^{(l)}, & l = 1, \ldots, \lceil \frac{n_u+1}{2} \rceil, \\ conj\left(\bar{Z}_u^{(n_u-l+2)}\right), & l = \lceil \frac{n_u+1}{2} \rceil + 1, \ldots, n_u. \end{cases} \tag{4.14}$$

Then the algorithm can be outlined as Algorithm 4.1:

4.2. Complexity analysis

The main computational cost of Algorithm 4.1 in each iteration rests is to compute $\mathcal{E}, \mathcal{Y}_u, \mathcal{Z}_u$ for $u \in \aleph(3)$. Updating $\mathcal{Y}_u$ needs to perform the FFT, inverse FFT, and singular value decomposition with the cost of $\mathcal{O}\left(n_1 n_2 n_3 \sum_{u=1}^{3} (\log(n_u) + \frac{1}{2} \min\{n_{u_1}, n_{u_2}\})\right)$. Here $u_1, u_2 \in \aleph(3)$, $u_1 < u_2$ and $u_1, u_2 \neq u$. Moreover, it takes $\mathcal{O}(n_1 n_2 n_3)$ to update $\mathcal{E}$ and updating $\mathcal{Z}_u$ costs $\mathcal{O}\left(n_1 n_2 n_3 \sum_{u=1}^{3} n_u + \sum_{u=1}^{3} n_u^3 n_{u+1}\right)$ with $n_4 := n_1$. Thus, the overall computational complexity of LPRN is $\mathcal{O}\left(n_1 n_2 n_3 \sum_{u=1}^{3} n_u + \sum_{u=1}^{3} n_u^3 n_{u+1}\right)$.

The convergence analysis of Algorithms 3.1 and 4.1 are quite similar, for simplicity, we only conduct the convergence analysis of Algorithm 3.1. Experimental discussion of the convergence behavior of Algorithms 3.1 and 4.1 will be in Sections 5 and 6, respectively.

Algorithm 4.1 Solve the non-convex TRPCA model (LPRN).

```
Input: The tensor data $\mathcal{O} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $p \in (0, 1)$, $r > 0$.
Initialize: $\mathcal{S}^0, \mathcal{X}_u^0, \mathcal{Y}_u^0, \mathcal{Z}_u^0, \mathcal{K}_u^0, \mathcal{W}_u^0,\ u \in \aleph(3)$.
While not converge do
    1. Compute $\mathcal{E}^{t+1}$ by (4.9).
    2. Compute $\mathcal{V}^{t+1}$ by (4.10).
    3. Compute $\mathcal{X}^{t+1}$ by (4.8).
    4. Compute each frontal slice of $\bar{\mathcal{Y}}_u^{t+1}$ by (3.10).
    5. Compute $\mathcal{Y}_u^{t+1} = \text{ifft}(\bar{\mathcal{Y}}_u^{t+1}, [\ ], u)$.
    6. Compute each frontal slice of $\bar{\mathcal{Z}}_u^{t+1}$ by (4.14).
    7. Compute $\mathcal{Z}_u^{t+1} = \text{ifft}(\bar{\mathcal{Z}}_u^{t+1}, [\ ], u)$.
    8. Update $\mathcal{S}^{t+1}$ by $\mathcal{S}^{t+1} = \mathcal{S}^t - \beta^t\left(\mathcal{O} - \mathcal{X}^t - \mathcal{E}^{t+1} - \mathcal{V}^{t+1}\right)$.
    9. Update $\mathcal{K}_u^{t+1}$ by $\mathcal{K}_u^{t+1} = \mathcal{K}_u^t - \beta^t\left(\mathcal{X}^{t+1} - \mathcal{Y}_u^{t+1}\right)$
    10. Update $\mathcal{W}_u^{t+1}$ by $\mathcal{W}_u^{t+1} = \mathcal{W}_u^t - \beta^t\left(\mathcal{X}^{t+1} - \mathcal{Z}_u^{t+1}\right)$
    11. Update $\beta^{t+1}$ by $\beta^{t+1} = \kappa\beta^t$.
    12. Check the convergence condition $\left\|\mathcal{X}^{t+1} - \mathcal{X}^t\right\|_\infty < \varepsilon$,
        $\|\mathcal{X}^{t+1} - \mathcal{Y}_u^{t+1}\|_\infty < \varepsilon$,
        $\|\mathcal{X}^{t+1} - \mathcal{Z}_1^{t+1}\|_\infty < \varepsilon$.
    13. $t \leftarrow t + 1$.
end while
Output: $\mathcal{X}^{t+1}, \mathcal{E}^{t+1}, \mathcal{V}^{t+1}$.
```

5. Experimental results for LRTC

In this section, we report some numerical examples to demonstrate the validity of our LPRN-based tensor completion method. We employ the peak signal-to-noise ratio (PSNR), the structural similarity index (SSIM) [43] and the feature similarity index (FSIM) [44] to evaluate the performance of each algorithm. We conduct extensive experiments to evaluate our method, and then, compare it with some existing methods, including TNN [23], WSTNN [29], PSTNN [24], ADMM-ilR [19], IR-t-TNN [26] and t-$S_{w,p}$ [28]. Higher PSNR, SSIM and FSIM usually indicate better performance. All experiments are implemented in Matlab R2020b under Windows 11 on a desktop of a 2.50 GHz CPU and 16 GB memory.

Experimental Data Settings: Two data sets are used for LRTC. The details of these data sets are described as follows:

Berkeley Segmentation database It is composed of 12,000 images segmented manually. The images collected from other dataset contains 30 human subjects. The dataset is a combination of RGB images and grayscale images.

YUV Video Sequences It is a widely used video data set, including 26 videos, and each sequence contains at least 150 frames. The videos in the data set include People such as “Akiyo” and Landscapes such as “Flower”, and so on.

Experimental Parameter Settings: The parameters in the comparison algorithms are adopted from the setting suggested by the original papers. In our method, the parameters p and r are set to 0.6 and 100 in LPRN. The maximum iteration maxIter and the convergence tolerance ε are chosen as $(\text{maxIter}, \varepsilon) = (200, 10^{-3})$.

5.1. Image inpainting

In this part, to evaluate our methods, we conduct tensor completion on Berkeley Segmentation database[1] for image inpainting. We randomly select three images, which are of size 321 × 481 × 3, from this database for testing. The entries are missing at random by sampling ratio $SR = 0.3, 0.5, 0.7$.

The image inpainting results of the three tested images are given in Table 1 and Fig. 3. Table 1 lists the values of the PSNR, SSIM, FSIM, and running time by different methods, and we highlight the best performance in bold. As observed, the proposed method can significantly outperform the compared methods in terms of all evaluation indices. To illustrate the visual quality, in Fig. 3, we show three tested images recovered by different methods with $SR = 0.5$. The proposed method is evidently superior to the compared ones in the recovery of both abundant shape structure and texture information. ADMM-ilR expands the tensor data (3D-array) directly into matrices and applies matrix nuclear norm to preserve the low-rank structure of the tensor, which may destroy multi-data structures and cause performance degradation [20,23]. Based on recent results of tensor factorization, LPRN, TNN, WSTNN, PSTNN, IR-t-TNN, and t-$S_{w,p}$ avoid the loss of tensor structure information [20,23] to obtain better recovery results. However, a key issue for the study of tensor factorization is that TNN, PSTNN, IR-t-TNN, and t-$S_{w,p}$ only consider the rank related to mode-3, but ignore the ranks related to mode-1 and mode-2. WSTNN usually leads to a biased solution since it tends to over-shrink the rank components and treats each rank component equally. Based on the reported results, LPRN outperforms other methods in all images.

5.2. Video inpainting

We conduct our tensor completion method on the widely used YUV Video Sequences.[2] In the experiments, we test our method and other approaches on two videos, in which frame sizes are 288 × 352 pixels. Due to the computational limitation, we only use the first 30 frames of the two sequences. The sampling rates SR are set to 0.2, 0.3, and 0.4. As shown in Fig. 4, we display the 8th slice of the two testing videos with $SR = 0.2$, respectively. Based on the recovery results, our method fills the missing values of the two testing sequences more effectively. It can deal with the details of the frames better. Table 2 shows the quantitative assessments of the results recovered by different methods. On the PSNR, SSIM and FSIM metrics, our method also achieves the best, consistent with the observation in Fig. 4. Taking into account the running time, our method is the fastest method. Video inpainting results

[1] https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/

[2] http://trace.eas.asu.edu/yuv/

Table 1
Color image inpainting performance comparison: PSNR, SSIM, FSIM and running time.

Image	Methods	$SR = 0.3$				$SR = 0.5$				$SR = 0.7$			
		PSNR	SSIM	FSIM	Time	PSNR	SSIM	FSIM	Time	PSNR	SSIM	FSIM	Time
Airplane	LPRN	**29.633**	0.791	**0.893**	**5.039**	**35.419**	0.931	**0.966**	**4.950**	**42.623**	**0.986**	**0.992**	**5.480**
	TNN	27.140	0.697	0.840	30.130	32.477	0.884	0.937	28.911	39.774	0.975	0.985	84.337
	WSTNN	28.854	**0.841**	**0.893**	8.706	34.045	**0.941**	0.959	8.883	40.583	0.985	0.988	26.940
	PSTNN	27.312	0.707	0.843	5.196	32.544	0.887	0.937	5.209	39.990	0.978	0.986	16.764
	ADMM-ilR	26.888	0.650	0.823	20.282	31.665	0.868	0.929	20.533	36.354	0.950	0.973	47.934
	IR-t-TNN	27.843	0.711	0.852	18.183	33.508	0.896	0.947	17.133	41.691	0.982	0.990	35.830
	t-$S_{w,p}$	27.711	0.683	0.846	7.052	33.266	0.871	0.943	7.442	41.469	0.978	0.990	23.736
House	LPRN	**26.145**	0.765	**0.900**	**5.064**	**32.084**	0.923	**0.968**	**5.138**	**39.464**	**0.986**	**0.994**	**5.701**
	TNN	23.781	0.679	0.856	57.133	29.701	0.888	0.948	55.241	37.580	0.979	0.990	46.312
	WSTNN	24.543	**0.811**	0.886	9.062	30.223	**0.940**	0.958	24.255	36.575	0.984	0.988	12.097
	PSTNN	24.095	0.689	0.861	5.156	29.859	0.890	0.949	5.160	37.731	0.981	0.990	5.876
	ADMM-ilR	23.551	0.626	0.840	7.572	28.679	0.830	0.930	26.768	33.148	0.944	0.973	42.049
	IR-t-TNN	24.773	0.704	0.872	37.685	30.885	0.900	0.958	58.714	39.012	0.984	0.993	52.794
	t-$S_{w,p}$	24.676	0.685	0.870	7.030	30.697	0.883	0.956	20.091	38.805	0.980	0.993	22.614
Tiger	LPRN	**28.501**	0.838	**0.914**	**5.006**	**33.979**	**0.948**	**0.970**	**4.972**	**41.437**	**0.989**	**0.993**	**5.459**
	TNN	25.806	0.761	0.870	82.017	31.022	0.915	0.948	87.862	38.500	0.981	0.987	82.525
	WSTNN	26.590	**0.857**	0.894	26.321	31.210	0.946	0.954	28.746	37.661	0.986	0.986	28.629
	PSTNN	26.192	0.768	0.875	5.348	31.232	0.917	0.950	5.408	38.709	0.983	0.988	16.591
	ADMM-ilR	26.077	0.731	0.866	67.490	30.681	0.900	0.943	68.033	35.626	0.965	0.979	56.890
	IR-t-TNN	26.733	0.775	0.883	68.925	32.283	0.925	0.958	44.734	40.507	0.987	0.992	33.525
	t-$S_{w,p}$	26.621	0.761	0.881	24.176	32.202	0.911	0.956	22.288	40.463	0.983	0.991	7.581

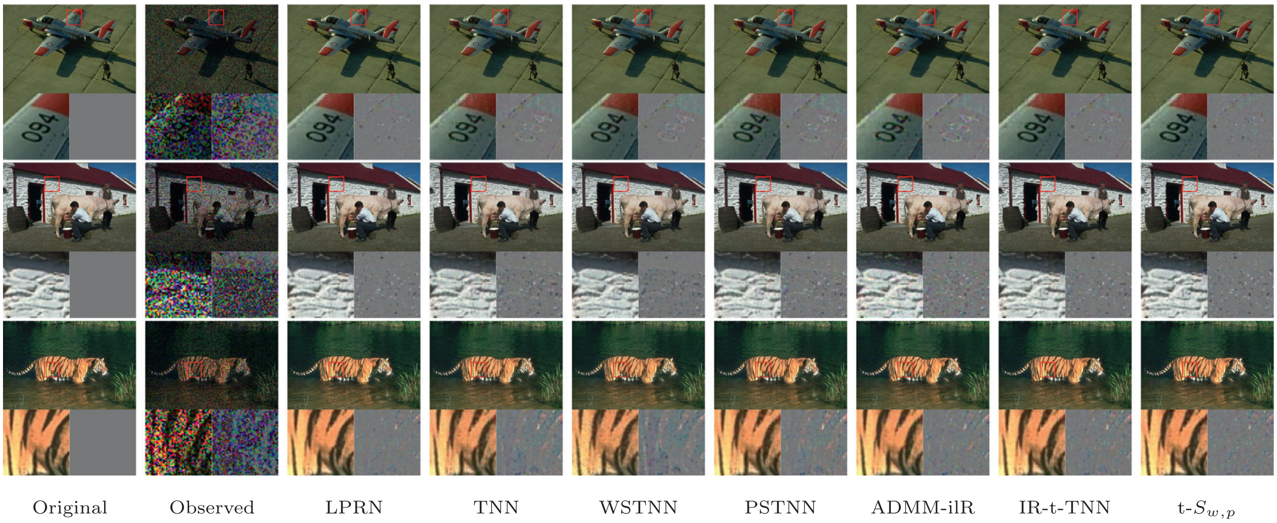


Fig. 3. Examples of color image inpainting with $SR = 0.5$. From top to bottom are respectively corresponding “Airplane”, “House” and “Tiger”. Under each image, we show enlargements of a demarcated patch and the corresponding error map (difference from the Original). Error maps with less color information indicate better restoration performance.

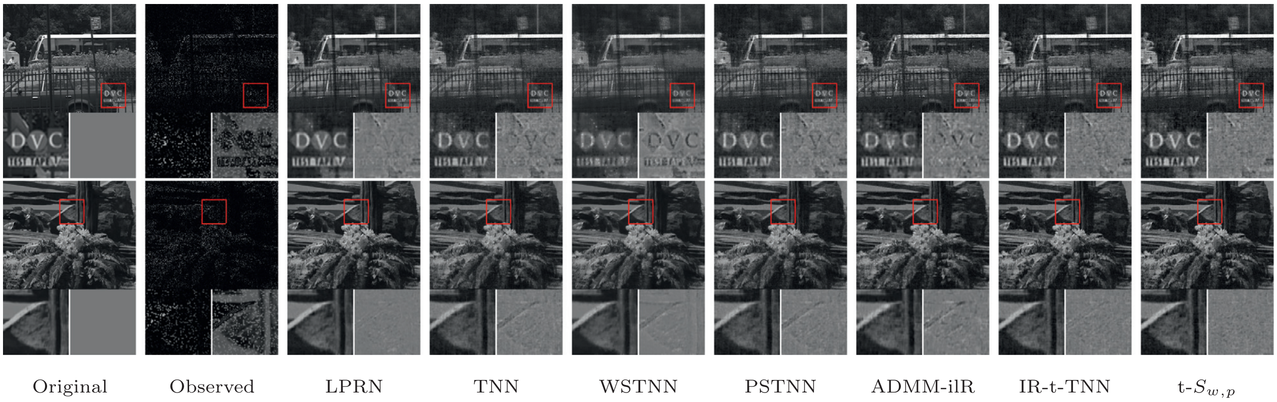


Fig. 4. Examples of video inpainting with $SR = 0.2$. From top to bottom are respectively corresponding “Bus” and “Tempete” (8th slice). Under each image, we show enlargements of a demarcated patch and the corresponding error map (difference from the Original). Error maps with less color information indicate better restoration performance.

Table 2
Video inpainting performance comparison: PSNR, SSIM, FSIM and running time.

Video	Methods	$SR = 0.2$				$SR = 0.3$				$SR = 0.4$			
		PSNR	SSIM	FSIM	Time	PSNR	SSIM	FSIM	Time	PSNR	SSIM	FSIM	Time
Bus	LPRN	**24.336**	**0.727**	**0.871**	**19.786**	**26.398**	**0.815**	**0.892**	**26.213**	**28.370**	**0.868**	**0.924**	**20.814**
	TNN	21.777	0.556	0.781	145.174	23.638	0.669	0.835	243.963	25.438	0.758	0.877	148.861
	WSTNN	21.793	0.634	0.771	43.411	23.585	0.739	0.833	45.095	25.248	0.813	0.875	45.817
	PSTNN	21.968	0.567	0.787	42.144	23.732	0.674	0.838	126.120	25.496	0.760	0.879	75.889
	ADMM-ilR	21.734	0.618	0.786	43.683	24.280	0.766	0.850	39.532	26.714	0.858	0.896	37.421
	IR-t-TNN	22.078	0.551	0.791	309.711	24.070	0.669	0.843	239.745	25.929	0.758	0.883	502.139
	t-$S_{w,p}$	21.553	0.514	0.779	50.769	23.436	0.625	0.829	52.158	25.279	0.716	0.869	162.758
Tempete	LPRN	**26.936**	**0.849**	**0.916**	**22.087**	**29.111**	**0.904**	**0.945**	**14.487**	**31.164**	**0.933**	**0.962**	**32.219**
	TNN	24.089	0.699	0.847	320.624	26.130	0.795	0.893	161.861	28.089	0.861	0.925	156.861
	WSTNN	25.374	0.818	0.890	87.329	27.238	0.880	0.926	50.338	28.964	0.917	0.948	51.540
	PSTNN	24.266	0.709	0.851	127.960	26.226	0.799	0.894	48.928	28.144	0.862	0.926	49.179
	ADMM-ilR	24.180	0.707	0.844	95.220	26.040	0.805	0.891	37.276	27.744	0.869	0.923	32.597
	IR-t-TNN	24.563	0.709	0.854	731.770	26.689	0.803	0.898	230.595	28.689	0.866	0.928	195.270
	t-$S_{w,p}$	24.090	0.682	0.844	162.785	26.203	0.775	0.886	58.306	28.213	0.841	0.918	58.103

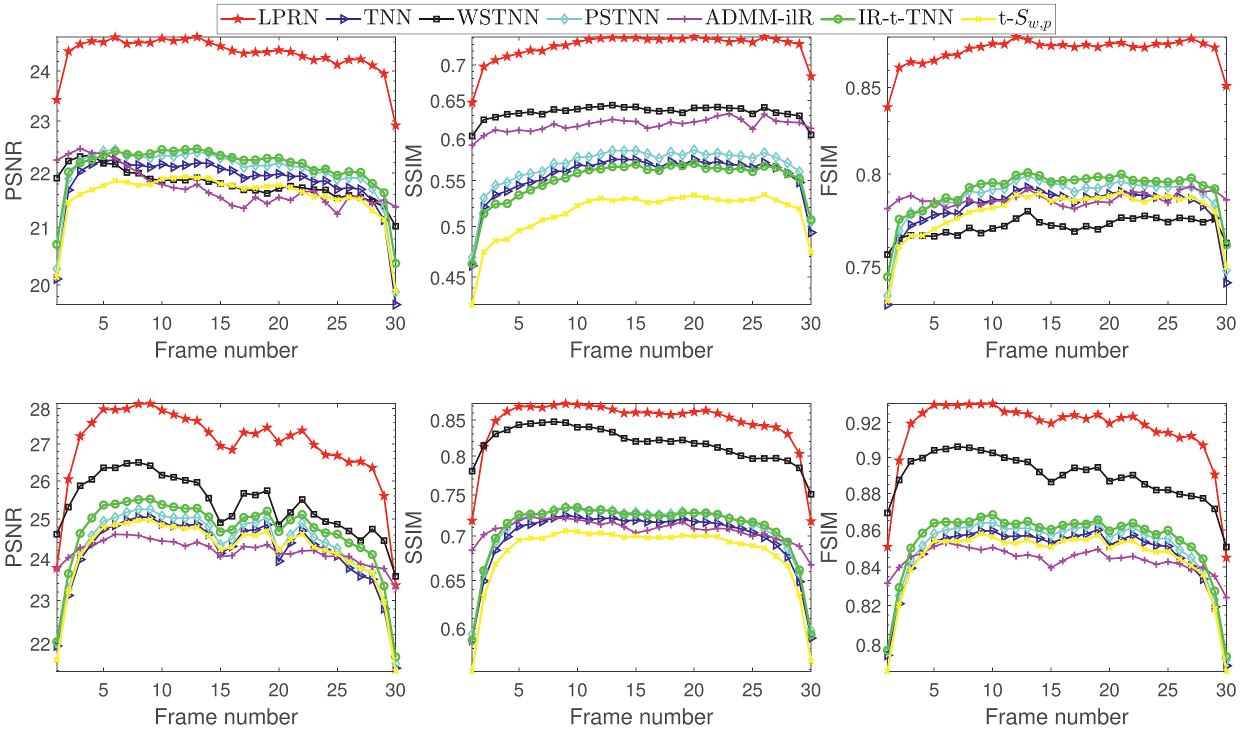


Fig. 5. All frontal slices obtained by different methods on the video "Bus" and "Tempete" with $SR = 0.2$.

are also consistent with the image inpainting results, demonstrating that our method performs tensor completion better and more efficiently.

In addition, Fig. 5 displays the PSNR, SSIM, and FSIM values of each frontal slice of video image "Bus" and "Tempete". As can be seen, in almost all the frontal slices, the PSNR, SSIM and FSIM metrics of the proposed LPRN are much higher than those of the other compared methods, illustrating the superiority of our method.

5.3. Discussions

In this section, we discuss the influence of different parameters in ℓ_r^p pseudo-norm, the effects of sparse prior, and the convergence of the proposed LPRN in the proposed LRTC problem. All tests are based on the video data-"Bus".

5.3.1. ℓ_r^p pseudo-norm setting

We set the SR to 0.2 in the completion tests. Now, we evaluate the performance of the proposed method with different ℓ_r^p pseudo-norm settings. Firstly, p is selected from 0.1 to 0.9 with an increment of 0.1. Meanwhile, r is chosen from $\{1, 10, 100, 1000, 10,000\}$. The quantitative metrics of the results obtained by the proposed method with different ℓ_r^p pseudo-norm settings are reported in Table 3. From Table 3, we can find that $r = 100,\ p = 0.6$ is the best choice.

5.3.2. Effectiveness of sparse prior

We further illustrate the effectiveness of sparse prior in the proposed framework. We set the SR to 0.2 in the completion tests and the results are presented in Fig. 6. $\mu \neq 0$ represents a sparse prior, and $\mu = 0$ represents no sparse prior. In order to better see the difference between the two, we only give three index transformation trends after 15 steps of iteration. From the figure, we can see that in the previous iterations, $\mu = 0$ has more advantages, and as the iteration increases, $\mu \neq 0$ has more advantages. Therefore, the sparse prior has certain advantages for video recovery.

5.3.3. Convergence behavior

Now the aim is to illustrate the convergence of our Algorithm 3.1, and we take the completion with sampling rates

Table 3
The PSNR, SSIM, FSIM and Time with different r and p of ℓ_r^p.

$r\backslash\ell_r^p\backslash p$	Index	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
1	PSNR	9.743	9.768	9.795	9.824	9.854	9.885	9.917	9.951	9.987
	SSIM	0.068	0.069	0.071	0.072	0.073	0.075	0.076	0.077	0.079
	FSIM	0.635	0.636	0.636	0.637	0.638	0.638	0.639	0.640	0.640
	Time	1.580	2.342	2.383	3.203	3.371	3.313	3.268	3.310	3.277
10	PSNR	9.863	10.035	10.293	10.565	11.102	11.905	12.861	22.324	24.345
	SSIM	0.074	0.082	0.090	0.100	0.119	0.149	0.187	0.609	0.709
	FSIM	0.638	0.641	0.644	0.648	0.655	0.668	0.685	0.823	0.866
	Time	3.263	3.172	4.761	4.784	7.368	9.609	10.589	31.845	29.909
100	PSNR	10.062	10.598	11.683	13.624	24.240	**24.336**	24.168	23.815	23.744
	SSIM	0.084	0.106	0.148	0.232	0.713	0.727	**0.719**	0.707	0.705
	FSIM	0.642	0.651	0.669	0.704	**0.869**	0.871	0.868	0.859	0.857
	Time	3.165	3.941	6.475	8.167	24.196	19.786	16.330	18.154	22.877
1000	PSNR	10.137	10.901	12.338	23.509	23.974	23.801	23.402	23.153	22.873
	SSIM	0.087	0.119	0.178	0.684	0.708	0.706	0.694	0.684	0.661
	FSIM	0.644	0.656	0.679	0.849	0.860	0.856	0.846	0.839	0.833
	Time	3.149	4.702	6.542	24.098	19.339	16.898	17.282	24.519	164.259
10,000	PSNR	10.147	10.936	12.617	23.452	23.910	23.710	23.387	22.959	22.346
	SSIM	0.088	0.120	0.191	0.683	0.707	0.704	0.690	0.677	0.593
	FSIM	0.644	0.657	0.685	0.846	0.859	0.856	0.848	0.837	0.829
	Time	3.154	4.714	7.296	23.453	19.900	16.421	19.602	24.392	166.879

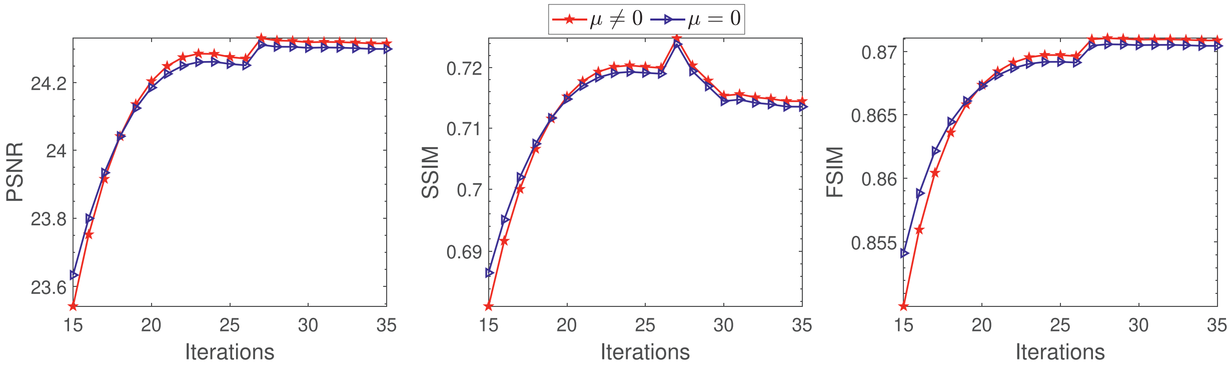


Fig. 6. The PSNR, FSIM, and FSIM values with respect to the iteration for $\mu \neq 0$ and $\mu = 0$.

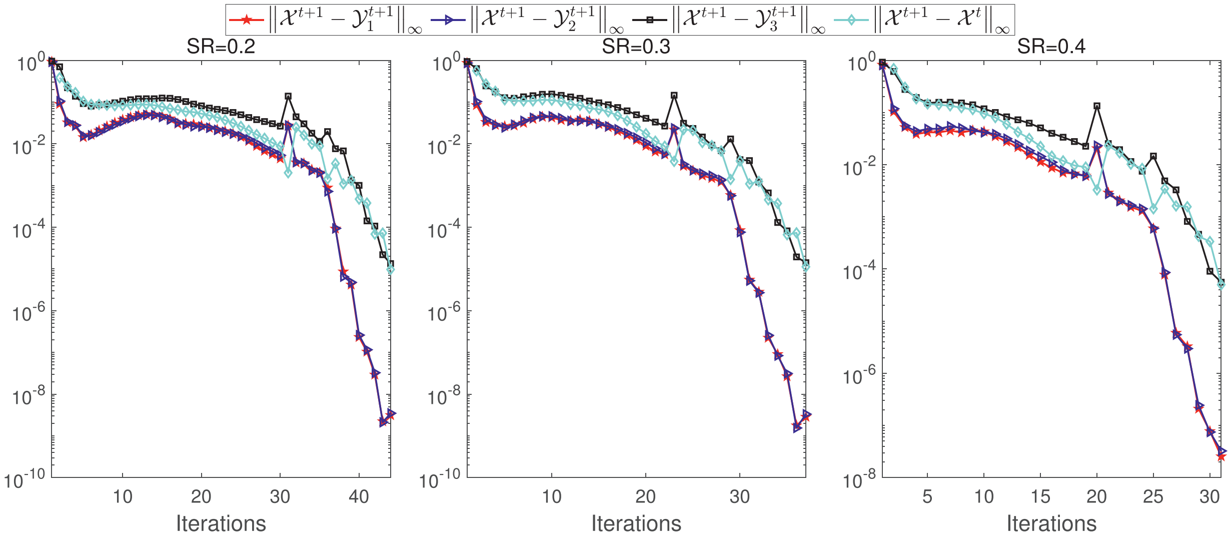


Fig. 7. The convergence behaviors of Algorithm 3.1, with respect to different sampling rates.

$SR = 0.2, 0.3, 0.4$. $\left\|\mathcal{X}^{t+1}-\mathcal{X}^{t}\right\|_\infty$ and $\left\|\mathcal{X}^{t+1}-\mathcal{Y}_u^{t+1}\right\|_\infty$ of each iteration are displayed in Fig. 7. It can be seen from the figure that although the value of $\left\|\mathcal{X}^{t+1}-\mathcal{X}^{t}\right\|_\infty$ and $\left\|\mathcal{X}^{t+1}-\mathcal{Y}_u^{t+1}\right\|_\infty$ fluctuates at some iteration steps, the value of $\left\|\mathcal{X}^{t+1}-\mathcal{X}^{t}\right\|_\infty$ and $\left\|\mathcal{X}^{t+1}-\mathcal{Y}_u^{t+1}\right\|_\infty$ generally decreases as the iteration increases. It is clear that our Algorithm 3.1 steadily converges.

6. Experimental results for TRPCA

In this section, we report some numerical examples to show the validity of our LPRN-based TRPCA method. We conduct extensive experiments to evaluate our method, and then, compare it with some existing methods, including TRPCA [30], LSD [31], DECOLOR [32], IBTSVT [33], ETRPCA [45] and t-$S_{w,p}$ [28].

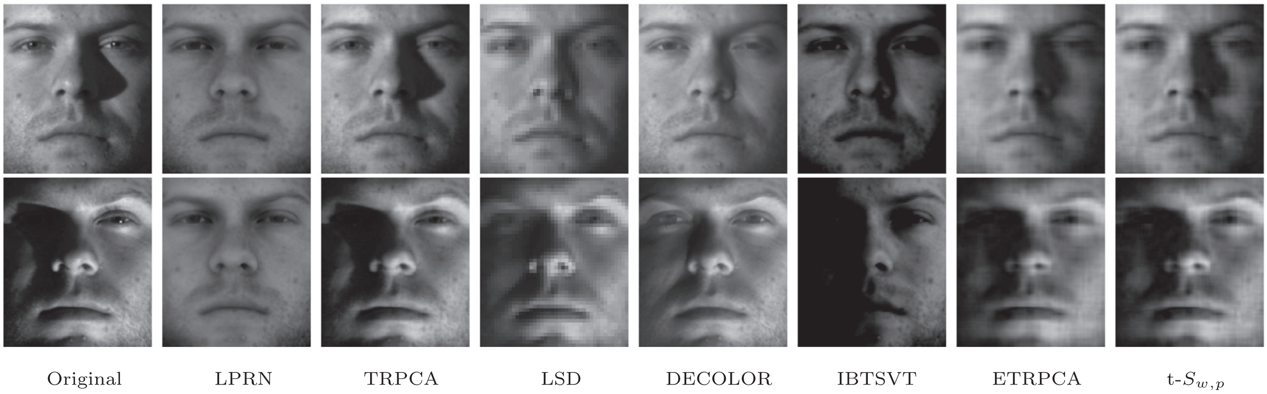


Fig. 8. Seven methods for the face with uneven illumination.

Table 4
The accuracy ratios of eyes, noses and mouth detection by Viola–Jones algorithm.

	Original	LPRN	TRPCA	LSD	DECOLOR	IBTSVT	ETRPCA	t-$S_{w,p}$
Eye detection	0.797	1.000	0.797	0.703	0.891	0.313	0.594	0.531
Nose detection	0.578	1.000	0.563	0.484	0.656	0.313	0.563	0.547
Mouth detection	0.500	1.000	0.516	0.531	0.703	0.313	0.688	0.750

Experimental Data Settings: Two data sets are used for TRPCA. The details of these data sets are described as follows:

Yale B face database It contains 16,128 images of 28 human subjects under 9 poses and 64 illumination conditions. Each face image is of size 192 × 168.
CDnet3 dataset It is accompanied by accurate ground-truth segmentation and annotation of change/motion areas for each video frame. The data includes the following challenges: dynamic background, camera jitter, intermittent object motion, shadows, thermal signatures, challenging weather, low frame rate, acquisition at night, PTZ capture, and air turbulence.

Experimental Parameter Settings: We mainly use four parameters: λ, ρ, δ, β^0. For λ, we adopt an adaptive value to different data and set $\lambda = 0.05/\sqrt{\max\{n_1, n_2\}n_3}$. For the penalty parameter β^0, we select $\beta^0 = 10^{-4}$. In order to better separate the dynamic background from the foreground, we set $\rho = 0.05,\ \delta = 10^{-2},\ \eta_1 = 100,\ \eta_2 = 100,\ \eta_3 = 500$ to perform the relevant constraints. The parameters in the comparison algorithms are adopted from the setting suggested by the original papers. The maximum iteration maxIter and the convergence tolerance ε are chosen as $(\text{maxIter}, \varepsilon) = (200, 10^{-3})$.

6.1. Face image shadow removal

In our experiments, we use the Yale B face database [46]. Each face image is of size 192 × 168 with 64 different lighting conditions. Our proposed algorithm is applied to address the shadow removal problem in face images. Fig. 8 shows one of the comparison results. The LPRN method can result in almost shadow-free faces. In contrast, the other six methods can only recover the faces with some shadow.

In order to further illustrate the effect of shadow elimination in the recovered face images, we carry on face detection with the recovered data from different methods. In our experiments, we employ the face detection algorithm Viola–Jones algorithm [47] to detect the eyes, the noses, and the mouth. The Viola–Jones algorithm is a classical algorithm that can be used to detect people's faces, noses, eyes, mouths, and upper bodies. From Table 4 we can see that our algorithm is able to detect eyes, nose, and mouth 100% of the time. And other algorithms can not reach 100% detection rate.

6.2. Background modeling

In this subsection, we implement the TRPCA algorithms on two videos in CDnet[3] dataset [48]. The videos contain some challenging scenes, e.g., a flowing fountain ("Fountain") and waving trees ("Fall").

A threshold criterion is required to get the final foreground mask and we adopt the same threshold strategy[4] as in Gao et al. [49]. To achieve an accurate evaluation of the proposed method, the criteria of recall and precision [50] are employed:

$$R = \frac{TP}{TP + FN}, \quad P = \frac{TP}{TP + FP}. \tag{6.15}$$

In the recall calculation, TP indicates the foreground pixel is correctly marked as foreground, FN indicates that the foreground pixel is wrongly marked as background and FP indicates that the background pixel is wrongly marked as foreground. The F score balances the precision and recall and an overall quantitative evaluation is defined as follows

$$F = 2 \times \frac{R \times P}{R + P}. \tag{6.16}$$

The F-measure value is between 0 and 1 and a larger value indicates a better effect of the video foreground-background separation.

From Table 5, it can be seen that the proposed algorithm LPRN achieves better segmentation performance among the comparison methods of F, P, and R. Some of the frames of the segmentation results are shown in Fig. 9, it can be observed that LPRN is good at separating dynamic background flowing fountain and waving trees from the foreground, whereas other methods cannot. This is mainly because we divide the original video into three terms, low-rank static background, sparse foreground, and dynamic background. And we introduce temporal and spatial matrices to make

[3] www.changedetection.net.

[4] To obtain a threshold, we first identify the likely locations: pixels with magnitudes less than half of the maximum magnitude of $\mathcal{E}$ are considered background, and the difference between $\mathcal{O}$ and $\mathcal{X}$ at those tentatively identified background locations is used to estimate the expected level of noise. Finally, we set the threshold to the mean of the difference values plus three standard deviations and apply it to $\mathcal{E}$.

Table 5
Comparison of R, P and F in two video sequences.

Video	LPRN		TRPCA		LSD		DECOLOR		ETRPCA		t-$S_{w,p}$	
	P	F	P	F	P	F	P	F	P	F	P	F
	R		R		R		R		R		R	
Fountain	**0.831**	**0.778**	0.328	0.351	0.193	0.241	0.738	0.692	0.347	0.381	0.340	0.377
	0.731		0.377		0.321		0.652		0.423		0.424	
Fall	**0.942**	**0.752**	0.429	0.296	0.507	0.442	0.600	0.580	0.437	0.307	0.350	0.248
	0.625		0.226		0.392		0.562		0.236		0.192	

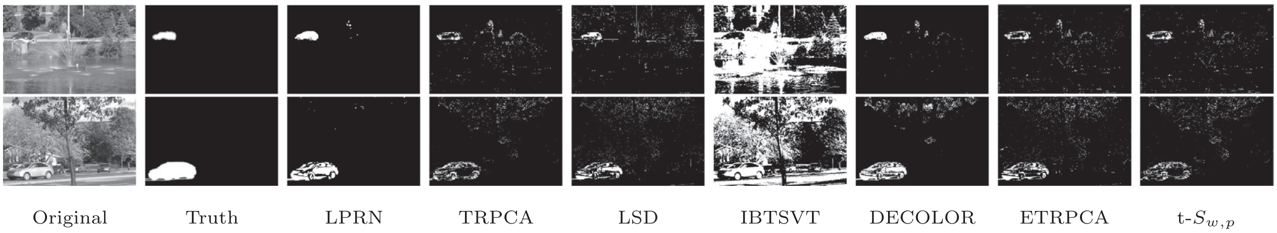


Fig. 9. Detected foreground results by seven different methods.

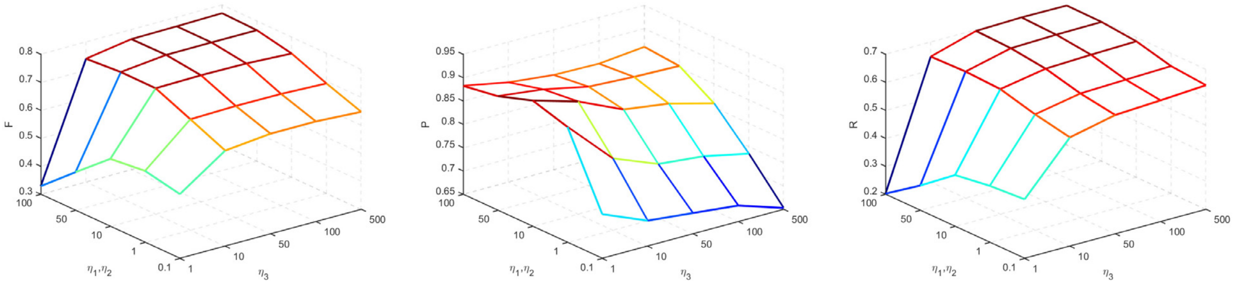


Fig. 10. The F, P and R with different η_1, η_2 and η_3.

better use of their own characteristics in low-rank static backgrounds and sparse foregrounds. At the same time, we introduce an incoherent term to constrain sparse foreground and dynamic background so as to improve the feasibility.

6.3. *Discussions*

In this section, we discuss the influence of different parameters in the dynamic background, spatio-temporal features and threshold strategy, and the convergence of the proposed LPRN in the proposed TRPCA problem. All tests are based on the video data-“Fountain”.

6.3.1. *Dynamic background setting*

We evaluate the performance of the proposed method with different ρ and δ. We set ρ from 0.01 to 1 and δ from 10^{-5} to 10^{-1}. The quantitative metrics of the results obtained by the proposed method with different ρ and δ settings are given in Table 6. The best and the second-best performing methods in each image are highlighted in red and bold, respectively. From Table 6, we can find that the results are relatively stable when $\delta \in \left[10^{-5}, 10^{-2}\right]$. Although certain fluctuations would happen in terms of efficacy, $\rho = 0.05$, $\delta = 10^{-2}$ is suggested as the first choice since it performs well in all our experiments.

6.3.2. *Spatio-temporal features setting*

We evaluate the performance of the proposed method with different spatio-temporal feature parameter settings. η_1, η_2 and η_3 are set within $[0.1, 100]$ and $[1, 500]$, respectively. Fig. 10 illustrates the sensitivity for spatio-temporal features parameter η_1, η_2, η_3. It can be observed that the algorithms can work well across a wide range of values of η_1, η_2, η_3. In particular, when $\eta_1, \eta_2, \eta_3 \geq 10$, the values of F increase as η_1, η_2, η_3 grow, but the value of F is not strongly affected by choosing the different value of η_1, η_2, η_3.

Table 6
The F, P and R with different ρ and δ.

$\rho\backslash\delta$	Index	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}
0.01	F	0.725	0.726	0.728	0.026	0.027
	P	0.948	0.947	0.944	0.013	0.014
	R	0.587	0.588	0.593	1.000	0.198
0.05	F	**0.771**	**0.771**	**0.771**	0.778	0.026
	P	0.860	0.859	0.857	0.831	0.013
	R	0.698	0.699	0.700	0.731	1.000
0.1	F	0.766	0.766	0.766	0.768	0.036
	P	0.825	0.825	0.823	0.810	0.018
	R	0.716	0.716	0.717	0.731	0.891
0.5	F	0.760	0.760	0.760	0.760	0.077
	P	0.800	0.800	0.800	0.797	0.041
	R	0.724	0.724	0.724	0.727	0.744
1	F	0.760	0.760	0.760	0.760	0.061
	P	0.799	0.799	0.799	0.798	0.032
	R	0.725	0.725	0.725	0.726	0.820

6.3.3. *Threshold sensitivity analysis*

In order to analyze the influence of threshold strategy on the experimental results in Gao et al. [49], we give the trend of the experimental results corresponding to the seven methods under different standard deviations. It can be seen from Fig. 11 that with the increase of standard deviations, except for IBTSVD, the corresponding P of each algorithm gradually increases, R gradually decreases, and F first increases and then decreases, reaching the maximum when standard deviations are 3. Regardless of the standard deviations, our algorithm corresponds to the highest F, P, R. With the

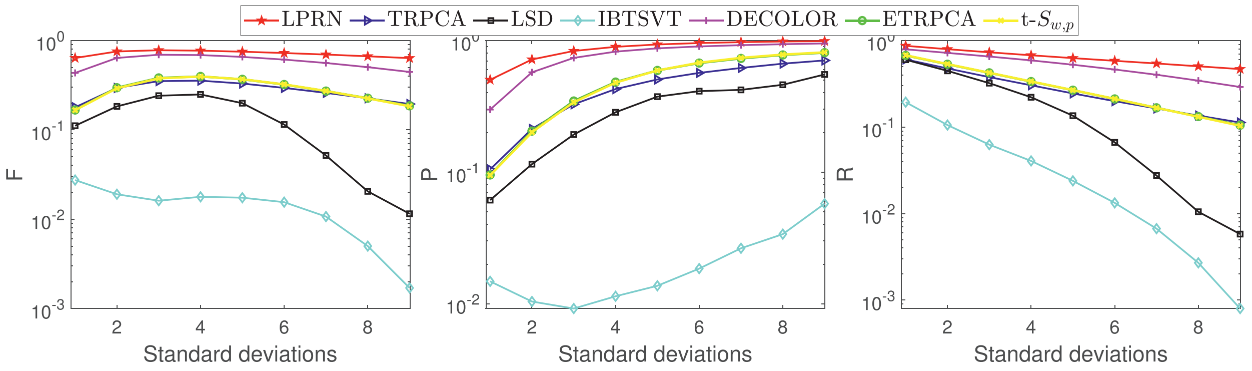


Fig. 11. The F, P, and R with different standard deviations by seven different methods.

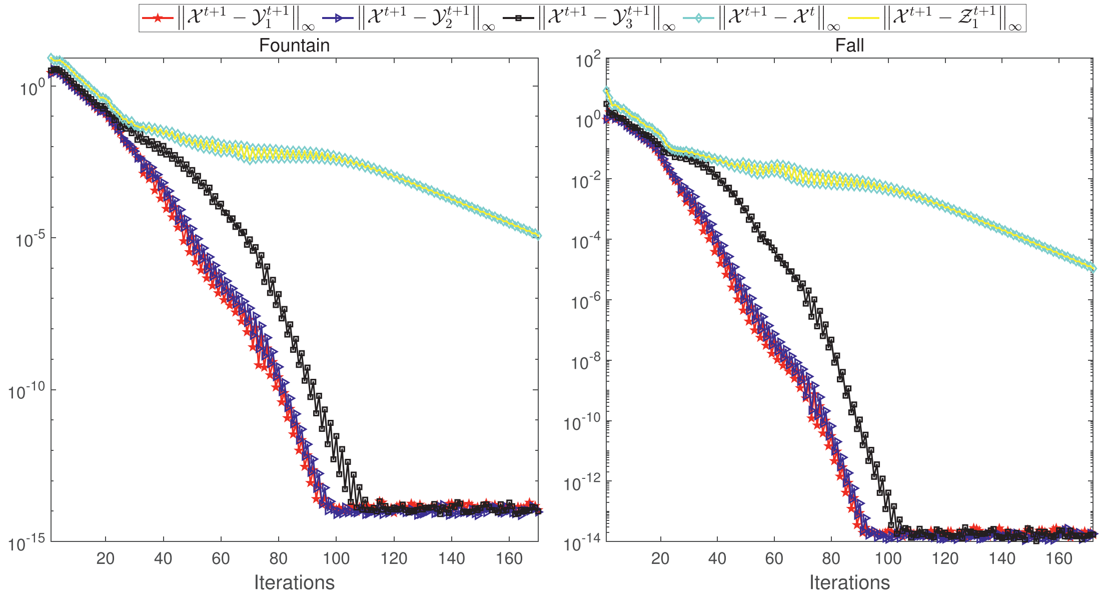


Fig. 12. The convergence behaviors of Algorithm 4.1, with respect to video “Fountain” and “Fall”.

change of standard deviations, the changes of F, P, and R corresponding to our algorithm are the least.

6.3.4. Convergence behavior

To evaluate the numerical convergence of Algorithm 4.1 in Fig. 12, let us take the video “Fountain” and “Fall” as examples. Then Fig. 12 shows a strong convergence behavior of the proposed LPRN solver, which can reach a low relative error of 10^{-2} after 30 iterations. Furthermore, the error decreases as the number of iterations increases, demonstrating the numerical stability and convergence of the proposed LPRN solver.

7. Conclusion

For the LRTC problem, we extended the WSTNN to a new tensor ℓ_r^p pseudo-norm, which better approximates the rank of a 3rd-order tensor. Based on the ℓ_r^p pseudo-norm, we introduced new non-convex tensor recovery models and proposed an alternating minimization method to solve the corresponding optimization problem. In addition, we introduce the noise analysis and decompose the visual frequency sequence into three terms, low-rank static background, sparse foreground, and dynamic background in TRPCA. We introduce temporal and spatial matrices to make better use of their own characteristics in low-rank static backgrounds and sparse foregrounds. At the same time, we introduce an incoherent term to constrain sparse foreground and dynamic background so as to improve the separability. Experimental results showed that the performance of our proposed methods was significantly better than existing methods in the literature.

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.

Data availability

Data will be made available on request.

References

[1] B. Ran, H. Tan, J. Feng, Y. Liu, W. Wang, Traffic speed data imputation method based on tensor completion, Comput. Intell. Neurosci. 2015 (2015) 1–9, doi:10.1155/2015/364089.

[2] H. Tan, Y. Wu, B. Shen, P.J. Jin, B. Ran, Short-term traffic prediction based on dynamic tensor completion, IEEE Trans. Intell. Transp. Syst. 17 (8) (2016) 2123–2133, doi:10.1109/tits.2015.2513411.

[3] G. Obozinski, B. Taskar, M.I. Jordan, Joint covariate selection and joint subspace selection for multiple classification problems, Stat. Comput. 20 (2) (2009) 231–252, doi:10.1007/s11222-008-9111-x.

[4] C. Peng, Y. Liu, K. Kang, Y. Chen, X. Wu, A. Cheng, Z. Kang, C. Chen, Q. Cheng, Hyperspectral image denoising using nonconvex local low-rank and sparse separation with spatial–spectral total variation regularization, IEEE Trans. Geosci. Remote Sens. 60 (2022) 1–17, doi:10.1109/tgrs.2022.3206783.

[5] Q. Yu, X. Zhang, Y. Chen, L. Qi, Low tucker rank tensor completion using a symmetric block coordinate descent method, Numer. Linear Algebra Appl. (2022) e2464, doi:10.1002/nla.2464.

[6] M. Yang, Q. Luo, W. Li, M. Xiao, 3-D array image data completion by tensor decomposition and nonconvex regularization approach, IEEE Trans. Signal Process. 70 (2022) 4291–4304, doi:10.1109/TSP.2022.3201330.

[7] Q. Yu, X. Zhang, T-product factorization based method for matrix and tensor completion problems, Comput. Optim. Appl. (2022), doi:10.1007/s10589-022-00439-y.

[8] S.F. Roohi, D. Zonoobi, A.A. Kassim, J.L. Jaremko, Multi-dimensional low rank plus sparse decomposition for reconstruction of under-sampled dynamic MRI, Pattern Recognit. 63 (2017) 667–679.

[9] M. Li, W. Li, Y. Chen, M. Xiao, The nonconvex tensor robust principal component analysis approximation model via the weighted ℓ_p-norm regularization, J. Sci. Comput. 89 (3) (2021) 1–37.

[10] T. Wu, Graph regularized low-rank representation for submodule clustering, Pattern Recognit. 100 (2020) 107145.

[11] C. Peng, Y. Chen, Z. Kang, C. Chen, Q. Cheng, Robust principal component analysis: afactorization-based approach with linear complexity, Inf. Sci. 513 (2020) 581–599, doi:10.1016/j.ins.2019.09.074.

[12] J. Ying, H. Lu, Q. Wei, J.-F. Cai, D. Guo, J. Wu, Z. Chen, X. Qu, Hankel matrix nuclear norm regularized tensor completion for n-dimensional exponential signals, IEEE Trans. Signal Process. 65 (14) (2017) 3702–3717, doi:10.1109/tsp.2017.2695566.

[13] H. Lu, K.N. Plataniotis, A.N. Venetsanopoulos, A survey of multilinear subspace learning for tensor data, Pattern Recognit. 44 (7) (2011) 1540–1551.

[14] N.D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E.E. Papalexakis, C. Faloutsos, Tensor decomposition for signal processing and machine learning, IEEE Trans. Signal Process. 65 (13) (2017) 3551–3582.

[15] M. Yang, W. Li, M. Xiao, On identifiability of higher order block term tensor decompositions of rank $l_r \otimes$rank-1, Linear Multilinear Algebra 68 (2) (2020) 223–245.

[16] L.R. Tucker, Some mathematical notes on three-mode factor analysis, Psychometrika 31 (3) (1966) 279–311, doi:10.1007/bf02289464.

[17] J.M. Landsberg, Tensors: geometry and applications, number v. 128, Graduate Studies in Mathematics, American Mathematical Society, Providence, R.I, 2012.

[18] X. Li, A. Wang, J. Lu, Z. Tang, Statistical performance of convex low-rank and sparse tensor recovery, Pattern Recognit. 93 (2019) 193–203.

[19] Y.-F. Li, K. Shang, Z.-H. Huang, Low tucker rank tensor recovery via ADMM based on exact and inexact iteratively reweighted algorithms, J. Comput. Appl. Math. 331 (2018) 64–81, doi:10.1016/j.cam.2017.09.029.

[20] M.E. Kilmer, K. Braman, N. Hao, R.C. Hoover, Third-order tensors as operators on matrices: atheoretical and computational framework with applications in imaging, SIAM J. Matrix Anal. Appl. 34 (1) (2013) 148–172, doi:10.1137/110837711.

[21] C.D. Martin, R. Shafer, B. LaRue, An order-p tensor factorization with applications in imaging, SIAM J. Sci. Comput. 35 (1) (2013) A474–A490, doi:10.1137/110841229.

[22] M.E. Kilmer, C.D. Martin, Factorization strategies for third-order tensors, Linear Algebra Appl. 435 (3) (2011) 641–658, doi:10.1016/j.laa.2010.09.020.

[23] Z. Zhang, G. Ely, S. Aeron, N. Hao, M. Kilmer, Novel methods for multilinear data completion and de-noising based on tensor-SVD, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, doi:10.1109/cvpr.2014.485.

[24] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, L.-J. Deng, Multi-dimensional imaging data recovery via minimizing the partial sum of tubal nuclear norm, J. Comput. Appl. Math. 372 (2020) 112680, doi:10.1016/j.cam.2019.112680.

[25] S. Cai, Q. Luo, M. Yang, W. Li, M. Xiao, Tensor robust principal component analysis via non-convex low rank approximation, Appl. Sci. 9 (7) (2019) 1411.

[26] H. Wang, F. Zhang, J. Wang, T. Huang, J. Huang, X. Liu, Generalized nonconvex approach for low-tubal-rank tensor recovery, IEEE Trans. Neural Netw. Learn. Syst. 33 (8) (2022) 3305–3319, doi:10.1109/tnnls.2021.3051650.

[27] Y. Chen, S. Wang, C. Peng, Z. Hua, Y. Zhou, Generalized nonconvex low-rank tensor approximation for multi-view subspace clustering, IEEE Trans. Image Process. 30 (2021) 4022–4035, doi:10.1109/tip.2021.3068646.

[28] M. Yang, Q. Luo, W. Li, M. Xiao, Nonconvex 3D array image data recovery and pattern recognition under tensor framework, Pattern Recognit. 122 (2022) 108311.

[29] Y.-B. Zheng, T.-Z. Huang, X.-L. Zhao, T.-X. Jiang, T.-Y. Ji, T.-H. Ma, Tensor N-tubal rank and its convex relaxation for low-rank tensor recovery, Inf. Sci. 532 (2020) 170–189.

[30] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, S. Yan, Tensor robust principal component analysis with a new tensor nuclear norm, IEEE Trans. Pattern Anal. Mach. Intell. 42 (4) (2019) 925–938.

[31] X. Liu, G. Zhao, J. Yao, C. Qi, Background subtraction based on low-rank and structured sparse decomposition, IEEE Trans. Image Process. 24 (8) (2015) 2502–2514, doi:10.1109/tip.2015.2419084.

[32] X. Zhou, C. Yang, W. Yu, Moving object detection by detecting contiguous outliers in the low-rank representation, IEEE Trans. Pattern Anal. Mach. Intell. 35 (3) (2013) 597–610, doi:10.1109/tpami.2012.132.

[33] L. Chen, Y. Liu, C. Zhu, Iterative block tensor singular value thresholding for extraction of lowrank component of image data, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2017, doi:10.1109/icassp.2017.7952479.

[34] B. Garcia-Garcia, T. Bouwmans, A.J.R. Silva, Background subtraction in real applications: challenges, current models and future directions, Comput. Sci. Rev. 35 (2020) 100204, doi:10.1016/j.cosrev.2019.100204.

[35] Y. Wang, H. Wei, X. Ding, J. Tao, Video background/foreground separation model based on non-convex rank approximation RPCA and superpixel motion detection, IEEE Access 8 (2020) 157493–157503, doi:10.1109/access.2020.3018705.

[36] Q. Yu, X. Zhang, Z.-H. Huang, Multi-tubal rank of third order tensor and related low rank tensor completion problem (2020).

[37] M. Fazel, H. Hindi, S.P. Boyd, Log-det heuristic for matrix rank minimization with applications to Hankel and euclidean distance matrices, in: Proceedings of the 2003 American Control Conference, IEEE, 2003, doi:10.1109/acc.2003.1243393.

[38] Z. Kang, C. Peng, Q. Cheng, Robust PCA via nonconvex rank approximation, in: 2015 IEEE International Conference on Data Mining, IEEE, 2015, doi:10.1109/icdm.2015.15.

[39] Q. Gao, W. Xia, Z. Wan, D. Xie, P. Zhang, Tensor-SVD based graph learning for multi-view subspace clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 3930–3937.

[40] X. Liang, X. Ren, Z. Zhang, Y. Ma, Repairing sparse low-rank texture, in: Computer Vision – ECCV 2012, Springer Berlin Heidelberg, 2012, pp. 482–495, doi:10.1007/978-3-642-33715-4_35.

[41] Z. Lin, M. Chen, Y. Ma, The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices, arXiv preprint arXiv:1009.5055(2010).

[42] M. Hong, Z.-Q. Luo, M. Razaviyayn, Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems, in: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2015, doi:10.1109/icassp.2015.7178689.

[43] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process. 13 (4) (2004) 600–612, doi:10.1109/tip.2003.819861.

[44] L. Zhang, L. Zhang, X. Mou, D. Zhang, FSIM: a feature similarity index for image quality assessment, IEEE Trans. Image Process. 20 (8) (2011) 2378–2386, doi:10.1109/tip.2011.2109730.

[45] Q. Gao, P. Zhang, W. Xia, D. Xie, X. Gao, D. Tao, Enhanced tensor RPCA and its application, IEEE Trans. Pattern Anal. Mach. Intell. 43 (6) (2021) 2133–2140, doi:10.1109/tpami.2020.3017672.

[46] A.S. Georghiades, P.N. Belhumeur, D.J. Kriegman, From few to many: illumination cone models for face recognition under variable lighting and pose, IEEE Trans. Pattern Anal. Mach. Intell. 23 (6) (2001) 643–660, doi:10.1109/34.927464.

[47] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, IEEE Comput. Soc, 2001, doi:10.1109/cvpr.2001.990517.

[48] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, P. Ishwar, Changedetection.net: a new change detection benchmark dataset, in: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, IEEE, 2012, doi:10.1109/cvprw.2012.6238919.

[49] Z. Gao, L.-F. Cheong, M. Shan, Block-sparse RPCA for consistent foreground detection, in: Computer Vision – ECCV 2012, Springer Berlin Heidelberg, 2012, pp. 690–703, doi:10.1007/978-3-642-33715-4_50.

[50] S. Brutzer, B. Hoferlin, G. Heidemann, Evaluation of background subtraction techniques for video surveillance, CVPR 2011, IEEE, 2011, doi:10.1109/cvpr.2011.5995508.

Quan Yu received the master's degree in mathematics from Tianjin University in 2022. He is currently pursuing the Ph.D. degree with the School of Mathematics, Hunan University. His research interests include image processing, low rank matrix optimization and low rank tensor optimization.

Ming Yang received his B.S. in Math from Jilin University, Changchun, China in 2007 and his Ph.D. in Math from Texas A&M University-College Station, USA, in 2012. Currently, he is an assistant professor in the mathematics department of the University of Evansville. His research interests are machine learning, image processing and tensor decomposition. He has published several research papers in top-tier journals, including SIAM Journal on Imaging Sciences, IEEE Signal Processing Letters, IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Image Processing, Journal of Dynamics and Differential Equations, Linear and Multilinear Algebra.