# Robust Low-Tubal-Rank Tensor Completion Based on Tensor Factorization and Maximum Correntopy Criterion

Yicong He and George K. Atia, *Senior Member, IEEE*

***Abstract*— The goal of tensor completion is to recover a tensor from a subset of its entries, often by exploiting its low-rank property. Among several useful definitions of tensor rank, the low tubal rank was shown to give a valuable characterization of the inherent low-rank structure of a tensor. While some low-tubal-rank tensor completion algorithms with favorable performance have been recently proposed, these algorithms utilize second-order statistics to measure the error residual, which may not work well when the observed entries contain large outliers. In this article, we propose a new objective function for low-tubal-rank tensor completion, which uses correntropy as the error measure to mitigate the effect of the outliers. To efficiently optimize the proposed objective, we leverage a half-quadratic minimization technique whereby the optimization is transformed to a weighted low-tubal-rank tensor factorization problem. Subsequently, we propose two simple and efficient algorithms to obtain the solution and provide their convergence and complexity analysis. Numerical results using both synthetic and real data demonstrate the robust and superior performance of the proposed algorithms.**

***Index Terms*— Alternating minimization, correntropy, half-quadratic (HQ), tensor completion, tensor factorization.**

## I. Introduction

HIGH-DIMENSIONAL and multiway data processing have received considerable attention in recent years given the ever-increasing amount of data with diverse modalities generated from different kinds of sensors, networks, and systems. Since tensors are algebraic objects that can be represented as multidimensional arrays (generalizing scalars, vectors, and matrices), they have marked ability to characterize multiway (high order) data and capture intrinsic correlations across its different dimensions. This fact explains their wide usage and efficacy in numerous applications of computer vision [1], [2], pattern recognition [3], [4], [5], [6], [7], and signal processing [8], [9], [10].

Manuscript received 26 October 2020; revised 22 November 2021 and 16 June 2022; accepted 15 May 2023. This work was supported in part by the NSF CAREER Award under Grant CCF-1552497 and in part by the NSF Award under Grant CCF-2106339. *(Corresponding author: Yicong He.)*

Yicong He is with the Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32816 USA (e-mail: yicong.he@ucf.edu).

George K. Atia is with the Department of Electrical and Computer Engineering and the Department of Computer Science, University of Central Florida, Orlando, FL 32816 USA (e-mail: george.atia@ucf.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2023.3280086.

Digital Object Identifier 10.1109/TNNLS.2023.3280086

Similar to matrices, the data represented by tensors may contain redundant information, which is referred to as the low-rank property of tensors. To exploit the underlying low-rank structure of high-order tensors, several low-rank tensor models have been proposed based on different tensor decompositions, including CANDECOMP/PARAFAC (CP) decomposition [11], Tucker decomposition [12], tensor ring decomposition [13], and tensor singular value decomposition (t-SVD) [14].

Tensor completion, a generalization of the popular matrix completion problem [15], [16], is the task of filling in the missing entries of a partially observed tensor, typically by exploiting the low-rank property of the tensor. There exist several tensor completion algorithms tailored to different low-rank tensor models, such as the CP decomposition-based alternating minimization algorithm [11], [17], Tucker decomposition-based tensor completion using the Riemannian manifold approach [12], [18] and alternating minimization [19], the t-SVD-based completion algorithm using convex relaxation [20], alternating minimization [21], [22], and Grassmannian optimization [23].

### A. Robust Tensor Completion

In the real world, the data observed could be perturbed by different kinds of noise originating from human errors and/or signal interference. Existing algorithms largely utilize the second-order statistics as their error measure, which works well in certain noisy settings, such as with noise from a Gaussian distribution. However, when the data are contaminated with large outliers, the performance of traditional algorithms is unsatisfactory in general. This motivated the development of robust algorithms for low-rank tensor recovery that are not unduly affected by the outliers [24], [25], [26]. While many such algorithms presume that all the entries of the tensor data are observed, several algorithms were designed to deal with incomplete or grossly corrupted data, which is the main focus of this work.

The vast majority of existing robust tensor completion algorithms are based on tensor rank models that are different from the tubal rank model considered here. In [27], an $\ell_1$-norm regularized sum of nuclear norm (SNN-L1) completion algorithm is proposed. Rather than directly applying complex Tucker decomposition, which decomposes the tensor into a set of matrices and one small core tensor, SNN-L1

TABLE I
OBJECTIVE FUNCTIONS OF ROBUST TENSOR COMPLETION ALGORITHMS

| Algorithm | Rank model | Objective function |
|---|---|---|
| SNN-L1 [24] | Tucker | $\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{S}}} \sum_{i=1}^{N} \left\|\boldsymbol{X}_{i,(i)}\right\|_* + \lambda\|\boldsymbol{\mathcal{S}}\|_1$ s.t. $\boldsymbol{\mathcal{P}} \circ \left(\sum_{i=1}^{N} \boldsymbol{\mathcal{X}}_i + \boldsymbol{\mathcal{S}}\right) = \boldsymbol{\mathcal{P}} \circ \boldsymbol{\mathcal{M}}$ |
| SNN-ST [28] | Tucker | $\min_{\boldsymbol{\mathcal{X}}} \sum_{i_1 \ldots i_N \in \Omega} \rho_\sigma((\mathcal{M}_{i_1 \ldots i_N} - (\sum_{m=1}^{N} \boldsymbol{\mathcal{X}}_m)_{i_1 \ldots i_N})) + \lambda \sum_{j=1}^{N} \left\|\boldsymbol{X}_{j,(j)}\right\|_*$ |
| SNN-HT [28] | Tucker | $\min_{\boldsymbol{\mathcal{X}}} \sum_{i_1 \ldots i_N \in \Omega} \rho_\sigma((\mathcal{M}_{i_1 \ldots i_N} - (\sum_{m=1}^{N} \boldsymbol{\mathcal{X}}_m)_{i_1 \ldots i_N})) + \sum_{j=1}^{N} \delta_{M_{r_j}}(\boldsymbol{X}_{j,(j)})$ |
| TRNN-L1 [29] | Tensor ring | $\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{S}}} \sum_{d=1}^{N} w_d \left\|\boldsymbol{X}_{\{d,L\}}\right\|_* + \lambda_d\|\boldsymbol{\mathcal{S}}\|_1$ s.t. $\boldsymbol{\mathcal{P}} \circ (\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{S}}) = \boldsymbol{\mathcal{P}} \circ \boldsymbol{\mathcal{M}}$ |
| TNN-L1 [31], [32] | Tubal | $\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{S}}} \frac{1}{n_3} \sum_{i=1}^{n_3} \left\|\bar{\boldsymbol{X}}^{(i)}\right\|_* + \lambda\|\boldsymbol{\mathcal{S}}\|_1$ s.t., $\boldsymbol{\mathcal{P}} \circ (\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{S}}) = \boldsymbol{\mathcal{P}} \circ \boldsymbol{\mathcal{M}}$ |
| HQ-TCTF /HQ-TCASD (This paper) | Tubal | $\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}}} \sum_{i,j,k} \mathcal{P}_{ijk}\sigma^2\left(1 - G_\sigma\left(\mathcal{M}_{ijk} - (\boldsymbol{\mathcal{X}} * \boldsymbol{\mathcal{Y}})_{ijk}\right)\right)$ |

relaxes the low-tucker-rank constraint using a (weighted) sum of nuclear norms of tensor unfolding matrices. Using a similar convex relaxation of Tucker decomposition, a robust low-tucker-rank tensor completion algorithm that uses soft/hard thresholding (SNN-ST/HT) was developed in [28]. It introduces two M-estimators, the Welsch loss and the Cauchy loss, as error measures, which improves on SNN-L1. In the same vein, Huang et al. [29] developed a robust $\ell_1$-norm regularized tensor ring nuclear norm (TRNN-L1) algorithm based on the tensor ring model. Similar to SNN, TRNN-L1 solves the complex tensor ring decomposition problem by minimizing the nuclear norm of circular unfolding matrices.

Recently, the tubal rank tensor model has been introduced in the context of tensor completion [30]. The tubal rank characterization is based on t-SVD [14], which rests on the tensor-to-tensor product (t-product) as an extension of matrix algebra. Compared with other tensor models, the tubal rank model has provable theoretical guarantees [30]. It also maintains the intrinsic multidimensional tensor structures as it relates to tensor factorization as a product of tensors. Its superior performance over alternate rank models in tensor completion has been established in various works [20], [21], [22]. We will further discuss the advantages of low-tubal-rank-based tensor completion in Section II-B. Several low-tubal-rank-based tensor completion algorithms have been developed in the literature, including tensor nuclear norm (TNN) [20] and its robust $\ell_1$-norm regularized version TNN-L1 [31], [32].

Table I summarizes the abovementioned robust tensor completion algorithms along with the tensor rank model they adopt and the corresponding objective functions. As shown, existing robust algorithms utilize the matrix nuclear norm for regularization, which requires performing an SVD in every iteration. For large matrices, SVD incurs a high computational cost. Furthermore, because of the complex computation of an SVD on a large matrix, algorithms, such as TNN-L1, are not readily amenable to parallel implementation on GPU.

### B. Contribution

In sharp contrast to the foregoing work, we propose a novel SVD-free and parallelizable robust tensor completion method based on tensor factorization and the maximum correntropy criterion (MCC) [33], [34] under the tubal rank model. Tensor factorization is theoretically grounded on the fact that a best tubal rank-$r$ approximation can be obtained from truncation of the t-SVD. Furthermore, algorithms based on tensor factorization (as opposed to minimizing norms of tensor unfoldings) were shown to yield accurate performance [21], [22]. Correntropy is an information-theoretic nonlinear similarity measure that can provably handle the negative effect of large outliers [35], [36], [37]. Compared with the commonly used $\ell_1$-norm, correntropy is everywhere differentiable and is at the heart of several robust algorithms in different fields [38], [39]. By introducing correntopy as our error measure, we propose a novel correntropy-based objective function for robust low-tubal-rank tensor completion. To efficiently solve the formulated completion problem, we first leverage a half-quadratic (HQ) optimization technique [40] to transform the nonconvex problem into a weighted tensor factorization problem. Then, two efficient and simple algorithms based on alternating minimization and alternating steepest descent (ASD) are developed, and we analytically establish the convergence of both algorithms. Also, we propose an adaptive kernel width selection strategy to further improve the convergence rate and accuracy. The main contributions of the work are summarized as follows.

1) We propose a novel objective function for robust low-tubal-rank tensor completion, which uses tensor factorization to capture the low-rank structure and correntropy as the error measure to give robustness against outliers. As shown in Table I, our approach imposes the low-rank structure through factorization. Compared with other existing nuclear-norm-based robust tensor completion algorithms, our factorization-based method does not need to perform multiple SVD computations.
2) We reformulate the complex correntropy-based optimization problem as a weighted tensor factorization by leveraging the HQ minimization technique (Section III-B). We develop two efficient algorithms [half-quadratic-based tensor completion by tensor factorization (HQ-TCTF) and half-quadratic based tensor completion by alternating steepest descent (HQ-TCASD)] for robust tensor completion (see Sections III-C and III-D). The algorithms utilize alternating minimization and ASD, which avoid the costly computation of the SVD operations and are readily parallelizable on GPU. We also analyze the convergence and computational complexity of the proposed algorithms.
3) We demonstrate the robust and efficient performance of the proposed algorithms through extensive numerical

experiments performed with both synthetic and real data. The proposed methods can outperform nuclear-norm-based methods in many noisy settings in terms of peak signal-to-noise ratio (PSNR). With the use of parallel computation, the proposed methods can also run significantly faster than other algorithms.

This article is organized as follows. In Section II, we introduce our notation and provide some preliminary background on the tensor properties, tensor completion, and the MCC. In Section III, we propose the new correntropy-based tensor completion cost and propose two HQ-based algorithms. In Section IV, we present experimental results to demonstrate the reconstruction performance. Finally, the conclusion is given in Section V.

## II. Preliminaries

### A. Definitions and Notation

In this section, we review some important definitions and introduce the notation used throughout this article. Boldface uppercase script letters are used to denote tensors (e.g., $\boldsymbol{\mathcal{X}}$), and boldface letters are used to denote matrices (e.g., $\boldsymbol{X}$). Unless stated otherwise, we focus on third-order tensors, i.e., $\boldsymbol{\mathcal{X}} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ where $n_1, n_2$, and $n_3$ are the dimensions of each way of the tensor. The notations $\boldsymbol{\mathcal{X}}(i,:,:)$, $\boldsymbol{\mathcal{X}}(:,i,:)$, and $\boldsymbol{\mathcal{X}}(:,:,i)$ denote the horizontal, lateral and frontal slices of $\boldsymbol{\mathcal{X}}$, respectively, and $\boldsymbol{\mathcal{X}}(:,j,k)$,$\boldsymbol{\mathcal{X}}(i,:,k)$, and $\boldsymbol{\mathcal{X}}(i,j,:)$ denote the mode-1, mode-2, and mode-3 tubes, respectively, while $\mathcal{X}_{ijk}$ denotes the $(i,j,k)$th entry of tensor $\boldsymbol{\mathcal{X}}$. The Frobenius norm of tensor is defined as $\|\boldsymbol{\mathcal{X}}\|_F = (\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}|\mathcal{X}_{ijk}|^2)^{1/2}$.

In the frequency domain, $\bar{\boldsymbol{\mathcal{X}}}$ denotes the Fourier transform along the third mode of $\boldsymbol{\mathcal{X}}$. We use the convention, $\bar{\boldsymbol{\mathcal{X}}} = \text{fft}(\boldsymbol{\mathcal{X}}, [\,], 3)$ to denote the Fourier transform along the third dimension. Similarly, we use $\boldsymbol{\mathcal{X}} = \text{ifft}(\bar{\boldsymbol{\mathcal{X}}}, [\,], 3)$ for the inverse transform. We also define the matrix $\bar{\boldsymbol{X}} \in \mathbb{R}^{n_1 n_3 \times n_2 n_3}$

$$\bar{\boldsymbol{X}} = \text{bdiag}(\bar{\boldsymbol{\mathcal{X}}}) = \begin{bmatrix} \bar{\boldsymbol{X}}^{(1)} & & & \\ & \bar{\boldsymbol{X}}^{(2)} & & \\ & & \ddots & \\ & & & \bar{\boldsymbol{X}}^{(n_3)} \end{bmatrix}$$

where $\boldsymbol{X}^{(i)} := \boldsymbol{\mathcal{X}}(:,:,i)$ and bdiag$(\cdot)$ denotes the operator that maps the tensor $\bar{\boldsymbol{\mathcal{X}}}$ to the block diagonal matrix $\bar{\boldsymbol{X}}$. The block-circulant operator bcirc$(\cdot)$ is defined as

$$\text{bcirc}(\boldsymbol{\mathcal{X}}) = \begin{bmatrix} \boldsymbol{X}^{(1)} & \boldsymbol{X}^{(n_3)} & \cdots & \boldsymbol{X}^{(2)} \\ \boldsymbol{X}^{(2)} & \boldsymbol{X}^{(1)} & \cdots & \boldsymbol{X}^{(3)} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{X}^{(n_3)} & \boldsymbol{X}^{(n_3-1)} & \cdots & \boldsymbol{X}^{(1)} \end{bmatrix}.$$

Therefore, the following relation holds:

$$\left(\boldsymbol{F}_{n_3} \otimes \boldsymbol{I}_{n_1}\right) \text{bcirc}(\boldsymbol{\mathcal{X}}) \left(\boldsymbol{F}_{n_3}^{-1} \otimes \boldsymbol{I}_{n_2}\right) = \bar{\boldsymbol{X}} \tag{1}$$

where $\boldsymbol{F}_{n_3} \in \mathbb{C}^{n_3 \times n_3}$ is the discrete Fourier transform (DFT) matrix, $\otimes$ is the Kronecker product, and $\boldsymbol{I}_{n_1} \in \mathbb{R}^{n_1 \times n_1}$ is the identity matrix. Furthermore, $\boldsymbol{F}_{n_3}^{-1}$ can be computed as $\boldsymbol{F}_{n_3}^{-1} = \boldsymbol{F}_{n_3}^{*}/n_3$, where $\boldsymbol{X}^{*}$ denotes the Hermitian transpose of $\boldsymbol{X}$.

To define the t-product, we first define the unfold operator unfold$(\cdot)$, which maps the tensor $\boldsymbol{\mathcal{X}}$ to a matrix $\tilde{\boldsymbol{X}} \in \mathbb{C}^{n_1 n_3 \times n_2}$

$$\tilde{\boldsymbol{X}} = \text{unfold}(\boldsymbol{\mathcal{X}}) = \begin{bmatrix} \boldsymbol{X}^{(1)} \\ \boldsymbol{X}^{(2)} \\ \vdots \\ \boldsymbol{X}^{(n_3)} \end{bmatrix}$$

and its inverse operator fold$(\cdot)$ is defined as

$$\text{fold}(\tilde{\boldsymbol{X}}) = \boldsymbol{\mathcal{X}}\,.$$

We can readily state the definition of the t-product.

*Definition 1 (t-Product [14]):* The t-product $\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}$ of $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{n_2 \times n_4 \times n_3}$ is the tensor of size $n_1 \times n_4 \times n_3$ given by

$$\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}} = \text{fold}(\text{bcirc}(\boldsymbol{\mathcal{A}}) \cdot \text{unfold}(\boldsymbol{\mathcal{B}})).$$

Furthermore, we will need the following lemma from [14].

*Lemma 1 [14]:* Suppose that $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{n_2 \times n_4 \times n_3}$ are two arbitrary tensors. Let $\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}$. Then, the following properties hold.

1) $\|\boldsymbol{\mathcal{A}}\|_F^2 = \frac{1}{n_3}\|\bar{\boldsymbol{A}}\|_F^2$
2) $\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}$ and $\bar{\boldsymbol{F}} = \bar{\boldsymbol{A}}\bar{\boldsymbol{B}}$ are equivalent.

According to the second property in Lemma 1, the t-product is equivalent to matrix multiplication in the frequency domain.

Next, we state the definitions of the t-SVD and the tubal rank.

*Theorem 1 (t-SVD [14], [41]):* The tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ can be factorized as $\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^*$, where $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{n_1 \times n_1 \times n_3}$ and $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{n_2 \times n_2 \times n_3}$ are orthogonal and $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is an $f$-diagonal tensor, i.e., each of the frontal slices of $\boldsymbol{\mathcal{S}}$ is a diagonal matrix. The diagonal entries in $\boldsymbol{\mathcal{S}}(:,:,1)$ are called the singular values of $\boldsymbol{\mathcal{A}}$.

*Definition 2 (Tensor Tubal Rank [30]):* For any $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the tensor tubal rank, $\text{rank}_t(\boldsymbol{\mathcal{A}})$, is the number of nonzero singular tubes of $\boldsymbol{\mathcal{S}}$ from the t-SVD, i.e.,

$$\text{rank}_t(\boldsymbol{\mathcal{A}}) = \#\{i : \boldsymbol{\mathcal{S}}(i,i,:) \neq 0\}\,.$$

We will also need the following definition of tensor multirank.

*Lemma 2 (Best Tubal Rank-r Approximation [30]):* Let the t-SVD of $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ be $\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^*$. Given a tubal rank $r$, define $\boldsymbol{\mathcal{A}}_r := \sum_{s=1}^{r} \boldsymbol{\mathcal{U}}(:,s,:) * \boldsymbol{\mathcal{S}}(s,s,:) * \boldsymbol{\mathcal{V}}^*(:,s,:)$. Then,

$$\boldsymbol{\mathcal{A}}_r = \underset{\check{\boldsymbol{\mathcal{A}}} \in \mathbb{A}}{\arg\min} \|\boldsymbol{\mathcal{A}} - \check{\boldsymbol{\mathcal{A}}}\|_F$$

where $\mathbb{A} := \{\boldsymbol{\mathcal{X}} * \boldsymbol{\mathcal{Y}} \mid \boldsymbol{\mathcal{X}} \in \mathbb{R}^{n_1 \times r \times n_3}, \boldsymbol{\mathcal{Y}} \in \mathbb{R}^{r \times n_2 \times n_3}\}$.

*Definition 3 (Tensor Multirank [30]):* For any tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, its multirank $\text{rank}_m(\boldsymbol{\mathcal{A}})$ is a vector defined as $\boldsymbol{r} = (\text{rank}(\bar{\boldsymbol{A}}^{(1)}); \ldots; \bar{\boldsymbol{A}}^{(n_3)})$. Specifically, the relation between tubal rank and tensor multirank is

$$\text{rank}_t(\boldsymbol{\mathcal{A}}) = \max\left(r_1, \ldots, r_{n_3}\right)$$

where $r_i$ is the $i$th element of $\boldsymbol{r}$.

### B. Low-Tubal-Rank Tensor Completion

Tensor completion is the task of recovering a tensor $\mathcal{M} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ from a subset of its entries by leveraging the low-rank property of the tensor. When using tubal rank for the definition of the rank, the low-tubal-rank property typically amounts to $\text{rank}_t(\mathcal{M}) \ll \max\{n_1, n_2\}$. Specifically, by defining the observed subset of entries $\mathbf{\Omega} \subseteq [n_1] \times [n_2] \times [n_3]$ and its indicator tensor $\mathcal{P}$

$$\mathcal{P}_{ijk} = \begin{cases} 1, & \text{if } (i, j, k) \in \Omega \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

the low-tubal-rank tensor completion problem can be formulated through the minimization

$$\min_{\mathcal{Z} \in \mathbb{R}^{n_1 \times n_2 \times n_3}} \text{rank}_t(\mathcal{Z}), \quad \text{s.t.} \mathcal{P} \circ (\mathcal{Z} - \mathcal{M}) = \mathbf{0} \tag{3}$$

where $\circ$ denotes the Hadamard (elementwise) product of the two same-size tensors. It is known that (3) is NP-hard. To address this problem, several methods were proposed, which can be categorized into two main categories.

1) *Convex Relaxation [20], [42]:* In this approach, (3) is relaxed to obtain a convex optimization problem. Specifically, by defining the TNN

$$\|\mathcal{A}\|_{\text{TNN}} = \frac{1}{n_3} \sum_{i=1}^{n_3} \left\| \bar{A}^{(i)} \right\|_*$$

where $\|\cdot\|_*$ denotes the matrix nuclear norm, (3) can be relaxed to

$$\min_{\mathcal{Z} \in \mathbb{R}^{n_1 \times n_2 \times n_3}} \sum_{i=1}^{n_3} \left\| \bar{Z}^{(i)} \right\|_*, \text{ s.t. } \mathcal{P} \circ (\mathcal{Z} - \mathcal{M}) = \mathbf{0}\,. \tag{4}$$

The iterative solver to the nuclear norm-based relaxation has to compute an SVD at each iteration, which incurs high computational complexity for large-scale high-dimensional data.

2) *Tensor Factorization:* Similar to the powerfactorization method proposed for matrix completion [43], a low-tubal-rank tensor can be represented as the t-product of two smaller tensors [14]. Specifically, the recovered tensor $\mathcal{M} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ can be factorized into the t-product of two tensors $\mathcal{X} \in \mathbb{R}^{n_1 \times r \times n_3}$ and $\mathcal{Y} \in \mathbb{R}^{r \times n_2 \times n_3}$, where $r$ is the tubal rank of $\mathcal{M}$ [21]. The tensor factorization then solves tensor completion by utilizing the objective function

$$\min_{\mathcal{X}, \mathcal{Y}} J(\mathcal{X}, \mathcal{Y}) := \|\mathcal{P} \circ (\mathcal{X} * \mathcal{Y} - \mathcal{M})\|_F^2\,. \tag{5}$$

Tensor factorization can avoid the high complexity associated with performing the SVD, and the complexity is reduced due to the inherent low-rank property. Two algorithms based on tensor factorization were proposed, namely, Tubal-Altmin (TAM) [22] and TCTF [21].

Low-tubal-rank-based tensor completion offers several advantages over tensor completion using other tensor rank models (e.g., CP rank, Tucker rank, and tensor ring rank). First, other methods usually impose low-rank constraints through the nuclear norm minimization on unfolding matrices of the tensor, which may destroy the original multidimensional structure of the tensor data. In contrast, based on tensor algebra, the tubal rank-based methods directly impose a low-tubal-rank constraint on a tensor and can well capture the inherent low-rank structure of a tensor [21], [22]. Second, unlike other rank models for which it is hard or infeasible to obtain an optimal approximation with truncated decomposition, in the tubal rank model, such an approximation is given in Lemma 2, which gives a theoretical footing for our proposed method. Third, if the tensor has a large $n_3$, the dimensions of the unfolding matrices will be very large, which degrades the computational efficiency of said algorithms. On the other hand, for the tubal-rank-based method, the SVD or factorization is applied to matrices of size $n_1 \times n_2$, which is smaller than the unfolding matrices.

### C. Maximum Correntropy Criterion

Correntropy is a local and nonlinear similarity measure between two random variables within a "window" in the joint space determined by the kernel width. Given two random variables $X$ and $Y$, the correntropy is defined as [33]

$$V(X, Y) = \mathbb{E}[\kappa_\sigma(X, Y)] = \int \kappa_\sigma(x, y) dF_{XY}(x, y) \tag{6}$$

where $\kappa_\sigma$ is a shift-invariant Mercer kernel with kernel width $\sigma$, $F_{XY}(x, y)$ denotes the joint probability distribution function of $X$ and $Y$, and $\mathbb{E}[.]$ is the expectation operator. Given a finite number of samples $\{x_i, y_i\}_{i=1}^N$ and using the Gaussian kernel $G_\sigma(x) = \exp(-(x^2/2\sigma^2))$ as the kernel function, the correntropy can be approximated by

$$\hat{V}(X, Y) = \frac{1}{N} \sum_{i=1}^{N} \exp\left(-\frac{e_i^2}{2\sigma^2}\right) \tag{7}$$

where $e_i = x_i - y_i$.

Compared with the $\ell_2$-norm-based second-order statistic of the error, the correntropy involves all the even moments of the difference between $X$ and $Y$ and is insensitive to outliers. Replacing the second-order measure with the correntropy measure leads to the MCC [44]. The MCC solution is obtained by maximizing the following utility function:

$$J_{\text{mcc}} = \mathbb{E}[G_\sigma(e(i))]\,. \tag{8}$$

Moreover, in practice, the MCC can also be formulated as minimizing the following correntropy-induced loss (C-loss) function [45]:

$$J_{C-loss} = \frac{1}{M} \sum_{i=1}^{M} \sigma^2 (1 - G_\sigma(e(i)))\,. \tag{9}$$

The cost function above is closely related to Welsch's cost function, originally introduced in [46].

Fig. 1 shows the different error measures. As can be seen, the correntropy-based error measure can efficiently reduce the effect of a large error $e$ caused by large outliers. Compared with the $\ell_1$-norm-based error, it is also differentiable at 0, which is convenient for optimization and allows us to leverage an HQ technique to efficiently solve the problem. The superior performance of correntropy over $\ell_1$ and $\ell_2$ norm has been verified in many fields [36], [37] and is also verified in this work.

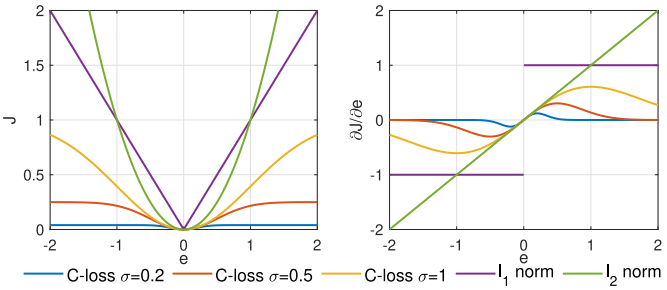


Fig. 1. Curves of different error measures with error $e$. Left: cost function $J$ with $e$. Right: derivative $\partial J/\partial e$ with respect to $e$.

## III. Proposed Methods

### A. Correntropy-Based Tensor Completion

Before we state our objective function for tensor completion, we first rewrite (5) as

$$\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}}} J(\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}}) := \sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\mathcal{P}_{ijk}\big(\mathcal{M}_{ijk}-(\boldsymbol{\mathcal{X}}*\boldsymbol{\mathcal{Y}})_{ijk}\big)^2. \tag{10}$$

When the observed entries $\mathcal{M}_{ijk}$ are corrupted or contain large outliers, the $\ell_2$ error measure can bias the optimization, which degrades the performance of tensor completion. To enhance robustness, in this work, we utilize the correntropy as the error measure. By replacing the $\ell_2$ error measure with correntropy, we obtain the new optimization problem

$$\begin{aligned}&\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}}} J_{G_\sigma}(\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}})\\ &:=\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\mathcal{P}_{ijk}\sigma^2\big(1-G_\sigma\big(\mathcal{M}_{ijk}-(\boldsymbol{\mathcal{X}}*\boldsymbol{\mathcal{Y}})_{ijk}\big)\big).\end{aligned} \tag{11}$$

The formulation in (11) generalizes the correntropy-based formulation in [36] for matrix completion. In particular, for the special case where $n_3 = 1$, the optimization in (11) reduces to the correntropy-based matrix completion. Surely, since tensor algebra is substantially different from the algebra of matrices (even the definition of tensor rank is not unique), the solution in [36] is no longer suitable for tensor completion, a fact that will also be verified in Section IV. Thus, here, we seek new approaches to solve (11).

### B. Optimization via HQ Minimization

In general, (11) is nonconvex and is difficult to be directly optimized. To tackle this difficulty, we utilize the HQ optimization technique to optimize the correntropy-based cost function. According to the HQ optimization theory [40], there exists a convex conjugated function $\varphi$ such that

$$G_\sigma(e) = \max_t\left(\frac{e^2t}{\sigma^2}-\varphi(t)\right) \tag{12}$$

where $t\in\mathbb{R}$ and the maximum is reached at $t=-G_\sigma(e)$. Equation (12) can be rewritten as

$$\sigma^2(1-G_\sigma(e)) = \min_t\big(-e^2t+\sigma^2\varphi(t)\big)\,. \tag{13}$$

By defining $s=-t$ and $\phi(s)=\sigma^2\varphi(-s)$, (13) can be written as

$$\min_e \sigma^2(1-G_\sigma(e)) = \min_{e,s}\big(e^2s+\phi(s)\big)\,. \tag{14}$$

Thus, minimizing the nonconvex C-loss function in terms of $e$ is equivalent to minimizing an augmented cost function in an enlarged parameter space $\{e, s\}$. Therefore, by substituting (14) into (11), the correntropy-based objective function $J_{G_\sigma}(\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}})$ can be expressed as

$$\begin{aligned}J_{G_\sigma}(\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}}) = \min_{\boldsymbol{\mathcal{W}}}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\Big(&\mathcal{W}_{ijk}\mathcal{P}_{ijk}\big(\mathcal{M}_{ijk}-(\boldsymbol{\mathcal{X}}*\boldsymbol{\mathcal{Y}})_{ijk}\big)^2\\ &+\mathcal{P}_{ijk}\phi\big(\mathcal{W}_{ijk}\big)\Big).\end{aligned} \tag{15}$$

Furthermore, by defining the augmented cost function

$$J_{\mathrm{HQ}}(\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}},\boldsymbol{\mathcal{W}}) = \|\sqrt{\boldsymbol{\mathcal{W}}}\circ\boldsymbol{\mathcal{P}}\circ(\boldsymbol{\mathcal{M}}-\boldsymbol{\mathcal{X}}*\boldsymbol{\mathcal{Y}})\|_F^2+\psi_{\boldsymbol{\Omega}}(\boldsymbol{\mathcal{W}}) \tag{16}$$

where $\psi_{\boldsymbol{\Omega}}(\boldsymbol{\mathcal{W}}) = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\mathcal{P}_{ijk}\phi(\mathcal{W}_{ijk})$, we have the following relation:

$$\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}}} J_{G_\sigma}(\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}}) = \min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}},\boldsymbol{\mathcal{W}}} J_{\mathrm{HQ}}(\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}},\boldsymbol{\mathcal{W}})\,. \tag{17}$$

Therefore, the correntropy-based optimization problem is formulated as an HQ-based optimization.

We propose the following alternating minimization procedure to solve the optimization problem (16).

*1) Optimizing $\boldsymbol{\mathcal{W}}$:* According to (12) and (14), given a certain $e$, the minimum is reached at $s=G_\sigma(e)$. Therefore, given the fixed $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$, the optimal solutions of $\mathcal{W}_{ijk}$ for $(i,j,k)\in\boldsymbol{\Omega}$ can be obtained as

$$\mathcal{W}_{ijk} = G_\sigma\big(\mathcal{M}_{ijk}-(\boldsymbol{\mathcal{X}}*\boldsymbol{\mathcal{Y}})_{ijk}\big),\ (i,j,k)\in\boldsymbol{\Omega}\,. \tag{18}$$

Since computing $\mathcal{W}_{ijk}$ for $(i,j,k)\notin\boldsymbol{\Omega}$ does not affect the solution of (11) due to the multiplication with $\boldsymbol{\mathcal{P}}$, henceforth we use $\mathcal{W}_{ijk}$ for all the entries to simplify the expressions.

*2) Optimizing $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$:* Given a fixed $\boldsymbol{\mathcal{W}}$, (16) becomes a weighted tensor completion problem

$$\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}}}\|\sqrt{\boldsymbol{\mathcal{W}}}\circ\boldsymbol{\mathcal{P}}\circ(\boldsymbol{\mathcal{M}}-\boldsymbol{\mathcal{X}}*\boldsymbol{\mathcal{Y}})\|_F^2. \tag{19}$$

The weighting tensor $\boldsymbol{\mathcal{W}}$ assigns different weights to each observed entry based on error residuals. Given the nature of the Gaussian function, a large error will lead to a small weight such that the negative impact of large outliers for error statistics can be greatly reduced. In the following, we propose and develop two algorithms to solve (19).

### C. Alternating Minimization-Based Algorithm

Inspired by TCTF [21], we first propose an alternating minimization-based approach to solve (19). By introducing an auxiliary tensor variable $\boldsymbol{\mathcal{Z}}$, (19) can be rewritten as

$$\begin{aligned}\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}},\boldsymbol{\mathcal{Z}}} J(\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{Y}},\boldsymbol{\mathcal{Z}}) := &\|\sqrt{\boldsymbol{\mathcal{W}}}\circ\boldsymbol{\mathcal{P}}\circ(\boldsymbol{\mathcal{M}}-\boldsymbol{\mathcal{Z}})\|_F^2\\ &+\beta\|\boldsymbol{\mathcal{X}}*\boldsymbol{\mathcal{Y}}-\boldsymbol{\mathcal{Z}}\|_F^2\end{aligned} \tag{20}$$

where $\beta$ is the regularization parameter. To solve (20), one can again utilize alternating minimization and update $\mathcal{Z}$, $\mathcal{X}$, and $\mathcal{Y}$ in turn. Specifically, by fixing $\mathcal{X}$ and $\mathcal{Y}$, we can update $\mathcal{Z}$ as

$$\mathcal{Z} = \arg\min_{\mathcal{Z}} \|\sqrt{\mathcal{W}} \circ \mathcal{P} \circ (\mathcal{M} - \mathcal{Z})\|_F^2 + \beta \|\mathcal{X} * \mathcal{Y} - \mathcal{Z}\|_F^2. \tag{21}$$

To solve (21), we set the first derivative of $J(\mathcal{X}, \mathcal{Y}, \mathcal{Z})$ with respect to $\mathcal{Z}$ to zero, i.e.,

$$\frac{\partial J}{\partial \mathcal{Z}} = 2(\mathcal{W} \circ \mathcal{P} \circ (\mathcal{Z} - \mathcal{M}) + \beta \mathcal{Z} - \beta \mathcal{X} * \mathcal{Y}) = \mathbf{0}. \tag{22}$$

Equation (22) is equivalent to the requirement that

$$\begin{aligned} \mathcal{P} \circ (\mathcal{W} \circ \mathcal{Z} - \mathcal{W} \circ \mathcal{M} + \beta \mathcal{Z} - \beta \mathcal{X} * \mathcal{Y}) &= \mathbf{0} \\ (\mathbf{1} - \mathcal{P}) \circ (\mathcal{Z} - \mathcal{X} * \mathcal{Y}) &= \mathbf{0}. \end{aligned} \tag{23}$$

Thus, $\mathcal{Z}$ can be obtained in a closed form as

$$\begin{aligned} \mathcal{Z} &= \mathcal{P} \circ \mathcal{Z} + (\mathbf{1} - \mathcal{P}) \circ \mathcal{Z} \\ &= \mathcal{X} * \mathcal{Y} + \frac{\mathcal{W}}{\beta \mathbf{1} + \mathcal{W}} \circ \mathcal{P} \circ (\mathcal{M} - \mathcal{X} * \mathcal{Y}) \end{aligned} \tag{24}$$

where $\mathbf{1}$ denotes the tensor of all ones, and the division is elementwise. Furthermore, by fixing $\mathcal{Z}$, (20) reduces to the following minimization:

$$\min_{\mathcal{X}, \mathcal{Y}} \|\mathcal{X} * \mathcal{Y} - \mathcal{Z}\|_F^2 . \tag{25}$$

According to Lemma 1, we have

$$\|\mathcal{X} * \mathcal{Y} - \mathcal{Z}\|_F^2 = \frac{1}{n_3} \|\bar{\mathbf{X}} \bar{\mathbf{Y}} - \bar{\mathbf{Z}}\|_F^2 . \tag{26}$$

Given the block structure of $\bar{\mathbf{X}}$, $\bar{\mathbf{Y}}$, and $\bar{\mathbf{Z}}$, the above minimization problem is equivalent to solving the $n_3$ subproblems

$$\min_{\bar{\mathbf{X}}^{(k)}, \bar{\mathbf{Y}}^{(k)}} \|\bar{\mathbf{X}}^{(k)} \bar{\mathbf{Y}}^{(k)} - \bar{\mathbf{Z}}^{(k)}\|_F^2, \quad k = 1, \ldots, n_3 . \tag{27}$$

For each $k$, we can alternate between least-squares solutions to $\bar{\mathbf{X}}^{(k)}$ and $\bar{\mathbf{Y}}^{(k)}$, i.e.,

$$\begin{aligned} \bar{\mathbf{X}}^{(k)} &= \bar{\mathbf{Z}}^{(k)} \left(\bar{\mathbf{Y}}^{(k)}\right)^* \left(\bar{\mathbf{Y}}^{(k)} \left(\bar{\mathbf{Y}}^{(k)}\right)^*\right)^\dagger \\ \bar{\mathbf{Y}}^{(k)} &= \left(\bar{\mathbf{Y}}^{(k)} \left(\bar{\mathbf{Y}}^{(k)}\right)^*\right)^\dagger \left(\bar{\mathbf{Y}}^{(k)}\right)^* \bar{\mathbf{Z}}^{(k)} \end{aligned} \tag{28}$$

where $\boldsymbol{A}^\dagger$ denotes the Moore–Penrose pseudoinverse of matrix $\boldsymbol{A}$. Therefore, to solve (16), we alternate between the updates in (18), (24), and (28) until convergence. We name this algorithm "HQ-TCTF." The pseudocode of HQ-TCTF is summarized in Algorithm 1. Note that in step 3 of the algorithm, we use an adaptive kernel width to enhance the rate of convergence. More details about this strategy are discussed in Section III-E.

Note that the $n_3$ subproblems in each alternating minimization step are independent of each other. Thus, the solution to these subproblems can be parallelized to further speed up computation.

*Remark 1:* One can observe that as $\sigma \to \infty$, $G_\sigma(e)$ approaches 1, and thus, all the entries of $\mathcal{W}$ become 1. In this special case, one does not need to optimize $\mathcal{W}$ in (18), and (16) reduces to

$$\min_{\mathcal{X}, \mathcal{Y}} \|\mathcal{P} \circ (\mathcal{M} - \mathcal{X} * \mathcal{Y})\|_F^2 \tag{29}$$

which is the tensor completion problem in (5). Furthermore, by setting $\beta = 0$ in (24), the updates of $\mathcal{Z}$, $\mathcal{X}$, and $\mathcal{Y}$ will be the same as in TCTF.

**Algorithm 1** HQ-TCTF for Robust Tensor Completion

**Input:** $\mathcal{P}$, $\mathcal{P} \circ \mathcal{M}$, $\beta$ and $r$
1: initial tensors $\mathcal{X}^0$ and $\mathcal{Y}^0$, $t = 0$
2: **repeat**
3: compute $\sigma^{t+1}$ and $\mathcal{W}^{t+1}$ using (18).
4: compute $\mathcal{Z}^{t+1}$ using (21).
5: **for** $k = 1, \ldots, n_3$ **do**
6: compute $\mathcal{X}^{(k),t+1}$ and $\mathcal{Y}^{(k),t+1}$ using (28)
7: **end for**
8: $t = t + 1$
9: **until** stopping criterion is satisfied
**Output:** $\mathcal{X}^t * \mathcal{Y}^t$

*Remark 2:* The adaptive tubal rank estimation method developed for TCTF [21] can be naturally applied to HQ-TCTF. Specifically, the scalar rank parameter $r$ in Algorithm 1 can be replaced with a multirank vector $\boldsymbol{r} = [r_1, \ldots, r_{n_3}]$ and the adaptive approach in [19] and [21] iteratively estimates the rank of the tensor.

The following proposition establishes the convergence guarantees for HQ-TCTF.

*Proposition 1:* Define the cost function

$$\begin{aligned} J(\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \mathcal{W}) = \|\sqrt{\mathcal{W}} \circ \mathcal{P} \circ (\mathcal{M} - \mathcal{X} * \mathcal{Y})\|_F^2 \\ + \|\mathcal{X} * \mathcal{Y} - \mathcal{Z}\|_F^2 + \psi_{\boldsymbol{\Omega}}(\mathcal{W}). \end{aligned} \tag{30}$$

The sequence $\{J_{\sigma^t}(\mathcal{X}^t, \mathcal{Y}^t, \mathcal{Z}^t, \mathcal{W}^t), t = 1, 2, \ldots\}$ generated by Algorithm 1 converges.

*Proof:* Since $\mathcal{W}$ and $\mathcal{Z}$ are optimal solutions to (18) and (21), respectively, we have

$$J\left(\mathcal{X}^{t+1}, \mathcal{Y}^{t+1}, \mathcal{Z}^{t+1}, \mathcal{W}^{t+1}\right) \le J\left(\mathcal{X}^{t+1}, \mathcal{Y}^{t+1}, \mathcal{Z}^{t}, \mathcal{W}^{t}\right). \tag{31}$$

Then, from Lemma 3 in the Supplementary Material of [21], one can obtain that for each matrix $\bar{\mathbf{X}}^{(k)}, \bar{\mathbf{Y}}^{(k)}, k = 1, \ldots, n_3$ generated from (28), the following inequality holds:

$$\|\bar{\mathbf{X}}^{(k),t+1} \bar{\mathbf{Y}}^{(k),t+1} - \bar{\mathbf{Z}}\|_F^2 \le \|\bar{\mathbf{X}}^{(k),t} \bar{\mathbf{Y}}^{(k),t} - \bar{\mathbf{Z}}\|_F^2. \tag{32}$$

From Lemma 1, we have $\|\mathcal{X}^t * \mathcal{Y}^t - \mathcal{Z}\|_F^2 = (1/n_3) \sum_{k=1}^{n_3} \|\bar{\mathbf{X}}^{(k),t} \bar{\mathbf{Y}}^{(k),t} - \bar{\mathbf{Z}}\|_F^2$. Thus, the following inequality holds:

$$\|\mathcal{X}^{t+1} * \mathcal{Y}^{t+1} - \mathcal{Z}\|_F^2 \le \|\mathcal{X}^t * \mathcal{Y}^t - \mathcal{Z}\|_F^2. \tag{33}$$

Combining (31) and (33), we have

$$J(\mathcal{X}^{t+1}, \mathcal{Y}^{t+1}, \mathcal{Z}^{t+1}, \mathcal{W}^{t+1}) \le J(\mathcal{X}^t, \mathcal{Y}^t, \mathcal{Z}^t, \mathcal{W}^t). \tag{34}$$

It can be also verified that $J(\mathcal{X}^t, \mathcal{Y}^t, \mathcal{Z}^t, \mathcal{W}^t)$ is always bounded below for arbitrary $t$. Thus, $\{J(\mathcal{X}^t, \mathcal{Y}^t, \mathcal{Z}^t, \mathcal{W}^t), t = 1, 2, \ldots\}$ will converge. □

### D. ASD-Based Algorithm

In the context of matrix completion, ASD was introduced to efficiently solve the completion problem [47]. ASD has a lower per-iteration complexity than PowerFactorization and can recover high-rank matrices. In this section, we introduce the ASD method for tensor completion and develop an efficient robust tensor completion algorithm.

As mentioned in Section III-B, we first optimize $\mathcal{W}$ using (18). Then, instead of directly optimizing (19), we gradually update $\mathcal{X}$ and $\mathcal{Y}$ using gradient descent. For convenience, we first add a multiplicative factor of $(1/2)$ to (19) such that the minimization problem becomes

$$\frac{1}{2}\min_{\mathcal{X},\mathcal{Y}} \|\sqrt{\mathcal{W}} \circ \mathcal{P} \circ (\mathcal{M} - \mathcal{X} * \mathcal{Y})\|_F^2. \tag{35}$$

Then, using the relation (1) and Definition 1 in Section II-A, (35) can be rewritten as

$$\frac{1}{2}\min_{\mathcal{X},\mathcal{Y}} \|\sqrt{\tilde{\boldsymbol{W}}} \circ \tilde{\boldsymbol{P}} \circ \left(\tilde{\boldsymbol{M}} - \text{bcirc}(\mathcal{X})\tilde{\boldsymbol{Y}}\right)\|_F^2. \tag{36}$$

Based on the block-circulant diagonalization [30], we have

$$\begin{aligned}\text{bcirc}(\mathcal{X})\tilde{\boldsymbol{Y}} &= \left(\boldsymbol{F}_{n_3}^{-1} \otimes \boldsymbol{I}_{n_1}\right)\bar{\boldsymbol{X}}\hat{\boldsymbol{Y}} \\ &= \boldsymbol{F}^{-1}\bar{\boldsymbol{X}}\hat{\boldsymbol{Y}} \\ &= \boldsymbol{U}\hat{\boldsymbol{Y}}\end{aligned} \tag{37}$$

where $\boldsymbol{F}^{-1} = \boldsymbol{F}_{n_3}^{-1} \otimes \boldsymbol{I}_{n_1}$ (consequently, $\boldsymbol{F} = \boldsymbol{F}^{-1} \times n_3$), $\boldsymbol{U} = \boldsymbol{F}^{-1}\bar{\boldsymbol{X}}$, and $\hat{\boldsymbol{A}} = \text{unfold}(\bar{\mathcal{A}})$. Finally, (36) can be reformulated as

$$\min J(\boldsymbol{U}, \hat{\boldsymbol{Y}}) := \frac{1}{2}\left\|\sqrt{\tilde{\boldsymbol{W}}} \circ \tilde{\boldsymbol{P}} \circ \left(\tilde{\boldsymbol{M}} - \boldsymbol{U}\hat{\boldsymbol{Y}}\right)\right\|_F^2. \tag{38}$$

Using the matrix derivatives, the partial derivative of $J(\boldsymbol{U}, \hat{\boldsymbol{Y}})$ with respect to $\boldsymbol{U}$ can be computed as

$$\boldsymbol{g}_U = \frac{\partial J}{\partial \boldsymbol{U}} = -\tilde{\boldsymbol{W}} \circ \tilde{\boldsymbol{P}} \circ \left(\tilde{\boldsymbol{M}} - \boldsymbol{U}\hat{\boldsymbol{Y}}\right)\hat{\boldsymbol{Y}}^*. \tag{39}$$

Note that $\bar{\boldsymbol{X}} = \boldsymbol{F}\boldsymbol{U}$ is a block diagonal matrix. Following the method in [23], we force the update of $\bar{\boldsymbol{X}}$ at each iteration to be block diagonal. Specifically, by defining the operator bdiagz$(\cdot)$ which sets the nonblock-diagonal entries of a matrix to zero, the updated gradient can be obtained as

$$\boldsymbol{g}'_U = \boldsymbol{F}^{-1}\,\text{bdiagz}(\boldsymbol{F}\boldsymbol{g}_U). \tag{40}$$

The steepest descent step size $\mu'_U$ for $\boldsymbol{g}'_U$ can be obtained in the following minimizer

$$\begin{aligned}\mu'_U &= \arg\min_{\mu} \left\|\sqrt{\tilde{\boldsymbol{W}}} \circ \tilde{\boldsymbol{P}} \circ \left(\tilde{\boldsymbol{M}} - \left(\boldsymbol{U} - \mu\boldsymbol{g}'_U\right)\hat{\boldsymbol{Y}}\right)\right\|_F^2 \\ &= \frac{\|\boldsymbol{g}'_U\|_F^2}{\left\|\sqrt{\tilde{\boldsymbol{W}}} \circ \tilde{\boldsymbol{P}} \circ \left(\boldsymbol{g}'_U\hat{\boldsymbol{Y}}\right)\right\|_F^2}\end{aligned} \tag{41}$$

and the matrix $\boldsymbol{U}$ can be updated as

$$\boldsymbol{U}^{t+1} = \boldsymbol{U}^t - \mu'^t_U \boldsymbol{g}'^t_U. \tag{42}$$

Similarly, by fixing $\boldsymbol{U}$, the partial derivative of $\boldsymbol{J}$ with respect to $\hat{\boldsymbol{Y}}$ can be obtained as

$$\boldsymbol{g}_{\hat{Y}} = \frac{\partial J}{\partial \hat{\boldsymbol{Y}}} = -\boldsymbol{U}^*\left(\tilde{\boldsymbol{W}} \circ \tilde{\boldsymbol{P}} \circ \left(\tilde{\boldsymbol{M}} - \boldsymbol{U}\hat{\boldsymbol{Y}}\right)\right). \tag{43}$$

The corresponding step size $\mu_{\hat{Y}}$ will be

$$\mu_{\hat{Y}} = \frac{\|\boldsymbol{g}_{\hat{Y}}\|_F^2}{\left\|\sqrt{\tilde{\boldsymbol{W}}} \circ \tilde{\boldsymbol{P}} \circ \left(\boldsymbol{U}\boldsymbol{g}_{\hat{Y}}\right)\right\|_F^2}. \tag{44}$$

Similar to ASD, the foregoing update process suffers from a slow rate of convergence when directly applied to image and video completion tasks. To tackle this problem, following a Newton-like method for scaled ASD [47], we scale the gradient descent direction for $\hat{\boldsymbol{Y}}$ in (43) by $(\boldsymbol{U}^*\boldsymbol{U})^{-1}$, i.e.,

$$\boldsymbol{g}'_{\hat{Y}} = (\boldsymbol{U}^*\boldsymbol{U})^{-1}\boldsymbol{g}_{\hat{Y}} \tag{45}$$

and the corresponding step size $\mu'_{\hat{Y}}$ with exact line search is

$$\mu'_{\hat{Y}} = \frac{\langle \boldsymbol{g}_{\hat{Y}}, \boldsymbol{g}'_{\hat{Y}}\rangle}{\left\|\sqrt{\tilde{\boldsymbol{W}}} \circ \tilde{\boldsymbol{P}} \circ \left(\boldsymbol{U}\boldsymbol{g}'_{\hat{Y}}\right)\right\|_F^2} \tag{46}$$

where $\langle \boldsymbol{A}, \boldsymbol{B}\rangle := \sum_{1\le i,j\le n} A^*_{ij}B_{ij}$. Therefore, the matrix $\hat{\boldsymbol{Y}}$ at the $t$th iteration can be updated by combining (43) and (45), i.e.,

$$\hat{\boldsymbol{Y}}^{t+1} = \hat{\boldsymbol{Y}}^t - (1-\lambda)\mu^t_{\hat{Y}}\boldsymbol{g}^t_{\hat{Y}} - \lambda\mu'^t_{\hat{Y}}\boldsymbol{g}'^t_{\hat{Y}} \tag{47}$$

where $0 \le \lambda \le 1$ is a free parameter to be chosen.

Therefore, the matrices $\boldsymbol{U}$ and $\hat{\boldsymbol{Y}}$ can be alternately updated using (42) and (47) until convergence. We term the above algorithm "HQ-TCASD."

Similar to HQ-TCTF, adaptive selection of the kernel width $\sigma$ is used to improve the rate of convergence and the performance of HQ-TCASD. HQ-TCASD is summarized in Algorithm 2. We remark that the matrices $\boldsymbol{U}(\bar{\boldsymbol{X}})$ and $\hat{\boldsymbol{Y}}$ have a block structure, so the matrix computation can be processed block-by-block. Also, since we have $\boldsymbol{F}\tilde{\boldsymbol{A}} = \text{unfold}(\text{fft}(\mathcal{A}, [\,], 3))$ for a tensor $\mathcal{A}$, the conventional fast Fourier transform (FFT) operation can be used in (40) instead of matrix multiplication to further accelerate the computation.

**Algorithm 2** HQ-TCASD for Robust Tensor Completion

**Input:** $\mathcal{P}$, $\mathcal{P} \circ \mathcal{M}$, $r$ and $\lambda$

1: initial matrices $\boldsymbol{U}^0$ and $\hat{\boldsymbol{Y}}^0$, $t = 0$
2: **repeat**
3: compute $\sigma^{t+1}$ and $\mathcal{W}^{t+1}$ using (18).
4: compute $\boldsymbol{U}^{t+1}$ using (42).
5: compute $\hat{\boldsymbol{Y}}^{t+1}$ using (47).
6: $t = t + 1$
7: **until** stopping criterion is satisfied

**Output:** $\boldsymbol{U}^t * \hat{\boldsymbol{Y}}^t$

The following proposition verifies the convergence of the proposed HQ-TCSAD.

*Proposition 2:* Define the cost function

$$J(\mathcal{X}, \mathcal{Y}, \mathcal{W}) = \frac{1}{2}\|\sqrt{\mathcal{W}} \circ \mathcal{P} \circ (\mathcal{M} - \mathcal{X} * \mathcal{Y})\|_F^2 + \frac{1}{2}\psi_{\boldsymbol{\Omega}}(\mathcal{W}). \tag{48}$$

The sequence $\{J(\mathcal{X}^t, \mathcal{Y}^t, \mathcal{W}^t), t = 1, 2, \ldots\}$ generated by Algorithm 2 will converge.

*Proof:* See the Appendix. □

*Remark 3:* As $\sigma \to \infty$ [i.e., for the standard tensor completion cost function in (5)], one can set $\tilde{\boldsymbol{W}}$ to be all one's

matrix and alternately update (42) and (47). This is itself a new algorithm, which we term TCASD. It can be used for tensor completion in noise-free settings or with Gaussian noise.

### E. Stopping Criterion and Adaptive Kernel Width Selection

The relative error between iterations can be used to measure the speed of convergence and develop a stopping criterion. Specifically, the residual error tensor at the $t$th iteration $\boldsymbol{\mathcal{E}}^t$ is defined as

$$\boldsymbol{\mathcal{E}}^t = \sqrt{\boldsymbol{\mathcal{W}}^t} \circ \boldsymbol{\mathcal{P}} \circ (\boldsymbol{\mathcal{M}} - \boldsymbol{\mathcal{X}}^t * \boldsymbol{\mathcal{Y}}^t) \,. \tag{49}$$

If $\|\boldsymbol{\mathcal{E}}^t\|_F - \|\boldsymbol{\mathcal{E}}^{t-1}\|_F$ falls below a sufficiently small value $\varepsilon$, the algorithm is considered to have converged to a local minimum, and the iterative procedure terminates.

To further improve the performance and achieve a faster rate of convergence, we use an adaptive kernel width selection strategy. Specifically, the kernel width at the $(t+1)$th iteration is determined by

$$\sigma^{t+1} = \max\left(\eta\left(\max\left(\boldsymbol{e}^t_{\Omega(0.25)}, \boldsymbol{e}^t_{\Omega(0.75)}\right)\right), \sigma_{\min}\right) \tag{50}$$

where $\boldsymbol{e}^t_\Omega \in \mathbb{R}^{|\Omega|\times 1}$ denotes the vector composed of all nonzero entries of $\boldsymbol{\mathcal{P}} \circ (\boldsymbol{\mathcal{M}} - \boldsymbol{\mathcal{X}}^t * \boldsymbol{\mathcal{Y}}^t)$ and $\boldsymbol{y}_{(q)}$ denotes the $q$th quantile of $\boldsymbol{y}$. The parameter $\eta$ controls the kernel width, and $\sigma_{\min}$ is a lower bound on $\sigma$.

### F. Complexity Analysis

We first present a complexity analysis of HQ-TCTF. Computing $\sigma$ involves computing $\boldsymbol{\mathcal{X}} * \boldsymbol{\mathcal{Y}}$ and finding the quantile of $e_\Omega$, whose time complexities are $\mathcal{O}(r(n_1+n_2)n_3 \log n_3 + rn_1n_2n_3)$ and $\mathcal{O}(n_1n_2n_3)$, respectively. The complexity of computing $\boldsymbol{\mathcal{W}}$ and $\boldsymbol{\mathcal{Z}}$ is both $\mathcal{O}(n_1n_2n_3)$ since $\boldsymbol{\mathcal{X}} * \boldsymbol{\mathcal{Y}}$ was already computed. Then, the cost of updating $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ is $\mathcal{O}(r(n_1+n_2)n_3 \log n_3 + rn_1n_2n_3)$. Therefore, the overall complexity of HQ-TCTF is $\mathcal{O}(r(n_1+n_2)n_3 \log n_3 + rn_1n_2n_3)$.

For HQ-TCASD, similar to HQ-TCTF, the complexity of computing $\sigma$ is $\mathcal{O}(r(n_1+n_2)n_3 \log n_3 + rn_1n_2n_3)$. Computing $\boldsymbol{g}'_U$ using FFT has complexity $\mathcal{O}(r(n_1+n_2)n_3 \log n_3 + rn_1n_3 \max(n_2, n_3))$, and calculation of $\mu'_U$, $\boldsymbol{g}_{\hat{Y}}$ and $\mu_{\hat{Y}}$ is of complexity $\mathcal{O}(r(n_1+n_2)n_3 \log n_3 + rn_1n_2n_3)$. Therefore, the overall complexity of HQ-TCASD is $\mathcal{O}(r(n_1+n_2)n_3 \log n_3 + rn_1n_3 \max(n_2, n_3))$.

One can observe that if $n_2 > n_3$, both HQ-TCTF and HQ-TCASD have the same order complexity. Furthermore, as both HQ-TCTF and HQ-TCASD are SVD-free algorithms and are readily parallelizable, the computation can be easily accelerated through parallel computation, which is verified in the experiments.

## IV. Experiments

In this section, we thoroughly evaluate the performance of the proposed algorithms HQ-TCTF, HQ-TCASD, and TCASD using both synthetic and real data. We compare to existing tensor completion algorithms, including TCTF [21], TAM [22], and TNN [20], and robust tensor completion algorithms, including SNN-L1 [24], SNN-HT/ST with Welsch loss (SNN-WHT/WST) [28], TRNN-L1 [29], and TNN-L1 [31]. For a fair comparison, the adaptive kernel width selection method is also applied to SNN-WHT and SNN-WST in the experiments. Furthermore, the correntropy-based robust matrix completion algorithm [36] is also included in the comparisons, where the tensor is treated as $n_3$ matrices of dimension $n_1 \times n_2$. In the experiments, we refer to this matrix-completion-based method as HQ-MCASD. We also report the run time of the proposed methods on a GPU (designated with suffix "parallel") by simply using the "gpuArray" data structure in MATLAB. All algorithms are implemented using MATLAB r2019b on a standard 16-GB memory PC with a 2.6-GHz CPU and an NVIDIA RTX3070 GPU.

In all simulations, the maximum number of iterations of all algorithms is set to 500 unless explicitly mentioned. The parameter $\eta$ in (50) for adaptive kernel width selection is set to 6 and 2 for synthetic data and real data, respectively. The lower bound $\sigma_{\min}$ for kernel width selection is experimentally set to 0.3 for synthetic data and 0.15 for real data. The threshold $\varepsilon$ for the stopping criterion is set to $10^{-9}$ for synthetic data and $10^{-5}$ for real data. The regularization parameter $\beta$ for HQ-TCTF is set to 1. For real data, $\lambda$ for HQ-TCASD is fixed to 0.2. Other parameters for each algorithm are tuned to achieve the best performance in each task. Note that the parameters of the different algorithms are not adapted across different noise settings in each simulation. Fixing the parameters is important since the noise properties could be changing and may not be measurable in practice.

### A. Synthetic Data

In this section, we verify the performance of the proposed algorithms using synthetic data. The dimensions of the tensor are set to $n_1 = n_2 = 200$ and $n_3 = 20$. The low-tubal-rank tensor $\boldsymbol{\mathcal{M}}$ with tubal rank $\bar{r}$ is obtained by the t-product of two tensors whose entries are generated from a zero mean Gaussian distribution with unit variance. The indicator tensor $\boldsymbol{\mathcal{P}}$ with observation fraction $p$ is generated by randomly and uniformly assigning $p \times 100\%$ of the entries of $\boldsymbol{\mathcal{P}}$ the value 1. The performance of an instance of tensor completion is evaluated using the relative error

$$\text{rel.err} = \frac{\|\hat{\boldsymbol{\mathcal{M}}} - \boldsymbol{\mathcal{M}}\|_F}{\|\boldsymbol{\mathcal{M}}\|_F} \tag{51}$$

where $\hat{\boldsymbol{\mathcal{M}}}$ is the recovered tensor. The performance is evaluated by taking the ensemble average of the relative error over $T$ independent Monte Carlo runs of different instances of $\boldsymbol{\mathcal{P}}$ and the noise. In this section, we only compare the performance of the proposed algorithms to TNN, TNN-L1, TAM, and TCTF since the other algorithms are using different definitions for the tensor rank.

In the experiments, the observed entries of the tensor are perturbed by additive noise generated from the standard two-component Gaussian mixture model (GMM). The probability density function is given by $(1-c)N(0, \sigma_A^2) + cN(0, \sigma_B^2)$, where $N(0, \sigma_A^2)$ represents the general Gaussian noise disturbance with variance $\sigma_A^2$ and $N(0, \sigma_B^2)$ with a large variance $\sigma_B^2$ captures the outliers. The variable $c$ controls the occurrence probability of outliers.

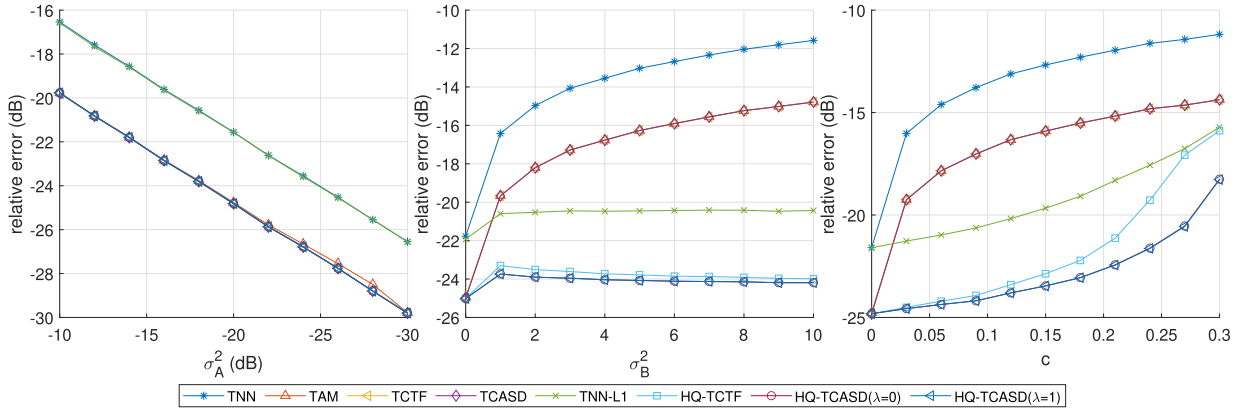


Fig. 2. Curves of average relative error under different noise environments. Left: $c = 0$. Middle: $\sigma_A^2 = 0.01, c = 0.1$. Right: $\sigma_A^2 = 0.01$ and $\sigma_B^2 = 1$.

We first investigate the performance of the algorithms under different settings for the noise. The observation fraction $p$ is set to 0.5 and the tubal rank $\bar{r}$ of $\boldsymbol{\mathcal{M}}$ is set to 10. The rank parameter for all the algorithms is set to the true value, i.e., $r = 10$. For each noise distribution, we average over 20 Monte Carlo runs. The average relative error under different noise distributions is shown in Fig. 2. One can observe that for Gaussian noise (i.e., $c = 0$), all algorithms expect TNN and TNN-L1 achieve the same favorable performance, and however, for GMM noise with $c \neq 0$, the proposed robust algorithms HQ-TCTF and HQ-TCASD outperform all the other algorithms. Also, HQ-TCASD is shown to slightly outperform HQ-TCTF.

In many practical situations, the actual rank $\bar{r}$ may not be known. Therefore, we study the performance under different settings of $r$. Again, the observation fraction $p$ is set to 0.5 and the actual tubal rank $\bar{r} = 10$. We use a Gaussian noise distribution with $\sigma_A^2 = 0.01$. For all factorization-based algorithms, we gradually change the rank parameter $r$ between 5 and 50. Note that TNN and TNN-L1 do not require setting the rank since they use convex relaxation as described in Section II-B. The other parameters are set as in the previous simulation. For HQ-TCTF, an additional algorithm with adaptive rank estimation (namely, HQ-TCTF-RE) is also included for comparison. The average relative error under different rank parameters $r$ is shown in Fig. 3 for the different algorithms. As shown, HQ-TCASD is still able to successfully complete the tensor $\boldsymbol{\mathcal{M}}$ with low relative error even when $r$ is set larger than actual $\bar{r}$.

Finally, we compare the performance of the proposed algorithms and TNN-L1 with different tensor sizes under the GMM noise model. Here, we only compare to TNN-L1 since it is the only algorithm other than the proposed methods that can yield successful recovery under the GMM noise, as shown in Fig. 2. The tensor size is set to $n_1 = n_2$ and $n_3 = 20$. The parameters of the GMM noise are set to $c = 0.1, \sigma_A^2 = 0.01$, and $\sigma_B^2 = 10$. The rank $\bar{r}$ is set to $n_1 \times 0.05$. The rank of HQ-TCASD with $\lambda = 1$ is set to $\bar{r} + 5$ for fast completion speed. We gradually increase $n_1$ from 100 to 1000 and average the relative error over 20 Monte Carlo runs. The average relative error and the average running time are shown in Fig. 4.

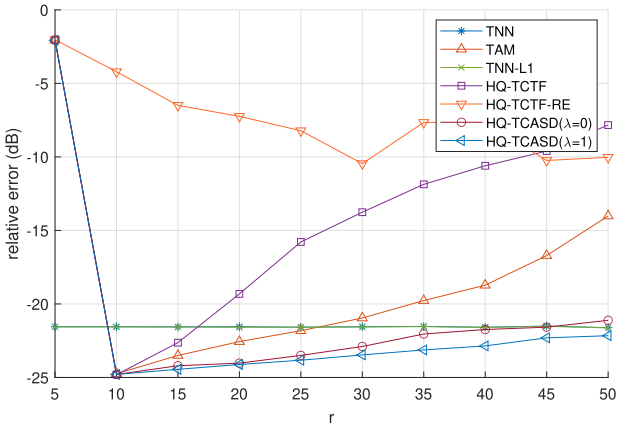


Fig. 3. Average relative error as a function of the rank parameter $r$ with Gaussian noise.

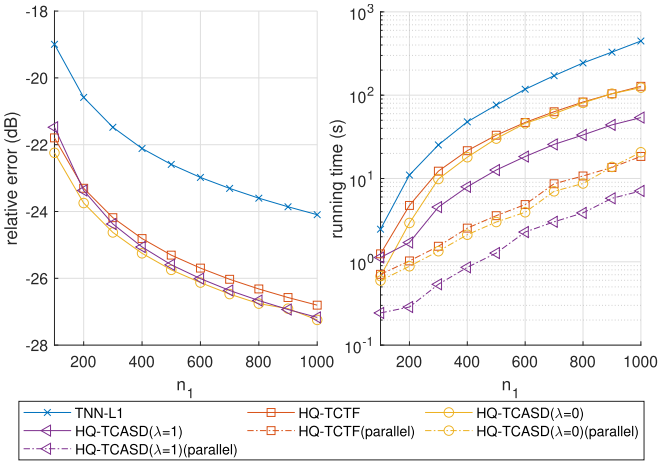


Fig. 4. Average relative error (left) and average run time (right) as a function of $n_1$ under GMM noise.

One can observe that the proposed algorithms always yield a significantly lower relative error and smaller computation time than TNN-L1. Specifically, the parallel computation can further speed up the computation of the proposed methods by an order of magnitude.

TABLE II
COMPLETION PERFORMANCE (PSNR) COMPARISON ON FOUR IMAGES FROM THE DAVIS AND SIDD DATASETS

| Image | Missing Pattern | Noise | c | Image SNR | SNN-L1 | SNN-WST | SNN-WHT | TRNN-L1 | TNN-L1 | HQ-MCASD | HQ-TCTF | HQ-TCASD |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bus | random(50%) | Stripe GMM | 0 | 23.62 | 25.36 | 26.27 | 25.61 | 24.58 | 27.25 | 26.26 | 28.76 | **29.12** |
| | | | 0.1 | 7.69 | 25.01 | 25.69 | 25.19 | 23.95 | 26.25 | 25.75 | 28.16 | **28.45** |
| | | | 0.2 | 3.03 | 24.13 | 24.85 | 23.84 | 23.21 | 24.98 | 20.64 | 26.47 | **27.26** |
| | | | 0.3 | -1.34 | 23.10 | 23.66 | 19.97 | 22.46 | 23.22 | 11.95 | 20.45 | **24.50** |
| dance | watermark | Stripe PSP | 0 | 22.44 | 29.06 | 29.96 | 29.47 | 29.85 | 29.25 | 29.59 | 27.47 | **31.24** |
| | | | 0.1 | 6.01 | 27.07 | 21.63 | 28.91 | 28.36 | 27.12 | 28.94 | 27.22 | **30.88** |
| | | | 0.2 | 3.21 | 24.05 | 13.24 | 27.99 | 25.94 | 23.27 | 25.41 | 26.20 | **29.49** |
| | | | 0.3 | 1.38 | 17.69 | 11.28 | **24.53** | 23.10 | 15.36 | 11.94 | 18.47 | 24.38 |
| board | random(50%) | Unknown | \ | 24.49 | 37.91 | 38.59 | 34.44 | 37.94 | 38.23 | 37.23 | 39.43 | **39.58** |
| | | | | 18.88 | 34.12 | 37.68 | 33.89 | 37.23 | 37.40 | 36.77 | 38.55 | **38.66** |
| | | | | 13.90 | **30.89** | 30.77 | 28.32 | 29.73 | 29.74 | 28.41 | 30.50 | 30.46 |
| | | | | 9.20 | **24.98** | 23.42 | 21.43 | 23.35 | 23.32 | 21.79 | 23.94 | 23.75 |
| alphabet | watermark | Unknown | \ | 22.63 | 37.50 | **38.37** | 36.91 | 37.23 | 37.88 | 35.65 | 36.76 | 38.01 |
| | | | | 19.21 | 31.65 | 33.02 | 33.70 | 30.59 | 31.78 | 33.20 | 34.28 | **34.70** |
| | | | | 16.39 | 28.40 | 29.23 | 30.72 | 27.29 | 28.48 | 30.73 | 31.84 | **31.92** |
| | | | | 12.98 | 25.04 | 25.46 | 27.00 | 24.06 | 25.09 | 27.42 | 28.54 | **28.43** |

### B. Image Inpainting

It aims to recover the missing pixels of an image from the observed pixels of the image. Because many images can be well approximated by a low-rank representation, image inpainting can be seen as a matrix or tensor completion task [21] and has been widely used for evaluating performance of matrix and tensor completion algorithms. When the observed pixels are corrupted with impulsive noise or outliers, the image inpainting task is more challenging. In this section, we evaluate the performance of the proposed HQ-TCTF and HQ-TCASD algorithms, along with other state-of-the-art robust completion algorithms, on the robust color image inpainting task with multiple noise distributions and missing pixel patterns. The performance evaluation metric is the PSNR defined as

$$\text{PSNR} = 10\log_{10}\frac{I_{\max}^3 n_1 n_2 n_3}{\left\|\hat{\mathcal{M}} - \mathcal{M}\right\|_F^2}$$

where $I_{\max}$ denotes the largest value of the pixels of the image data. A higher PSNR signifies better recovery performance.

We evaluate the completion performance of the different algorithms using four images. The first two images "bus" and "paragliding" are chosen from the Densely Annotated Video Segmentation (DAVIS) 2016 dataset.[1] [48] Different kinds of synthetic noise are added to these two images to obtain the noisy images. The last two images "board" and "alphabet" are selected from the Smartphone Image Denoising Dataset (SIDD).[2] For each image, four (noisy) photographs captured using a Samsung Galaxy S6 Edge are provided with different lighting conditions along with the ground-truth (noiseless) image. The noise comes from the camera itself and no synthetic noise is added. All images are scaled to $1920 \times 1080$, so each color image can be regarded as a $1920 \times 1080 \times 3$ tensor.

[1]https://davischallenge.org/davis2016/code.html
[2]https://www.eecs.yorku.ca/~kamel/sidd/index.php

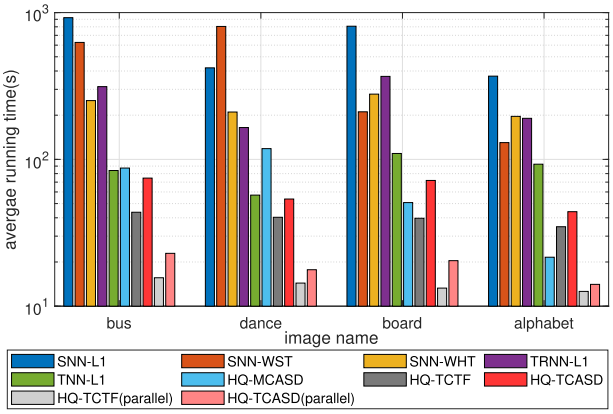


Fig. 5. Average running time for each image.

The completion performance is tested on two types of missing pixel patterns. In the first pattern, we independently and randomly select 50% pixels from each channel as the missing pixels. In the second pattern, a watermark is added to all channels of the image, and the missing pixels correspond to pixels covered by the watermark.

We evaluate the performance using four types of noises.

1) *GMM Noise*: All observed pixels are perturbed by GMM noise described in the previous experiment with $\sigma_A^2 = 0.001$ and $\sigma_B^2 = 1$ and parameter $c$.
2) *Possion+Salt-and-Pepper (PSP) Noise*: Nearly, $c \times 100\%$ of the observed pixels are randomly selected and perturbed with salt-and-pepper noise, and the remaining observed pixels are perturbed with Poisson noise.
3) *Stripe GMM noise*: For each channel, 50% of the columns are randomly selected, of which $2c \times 100\%$ of the observed pixels are perturbed with Gaussian noise $N(0, 1)$. The remaining observed pixels are perturbed by Gaussian noise $N(0, 0.01)$.

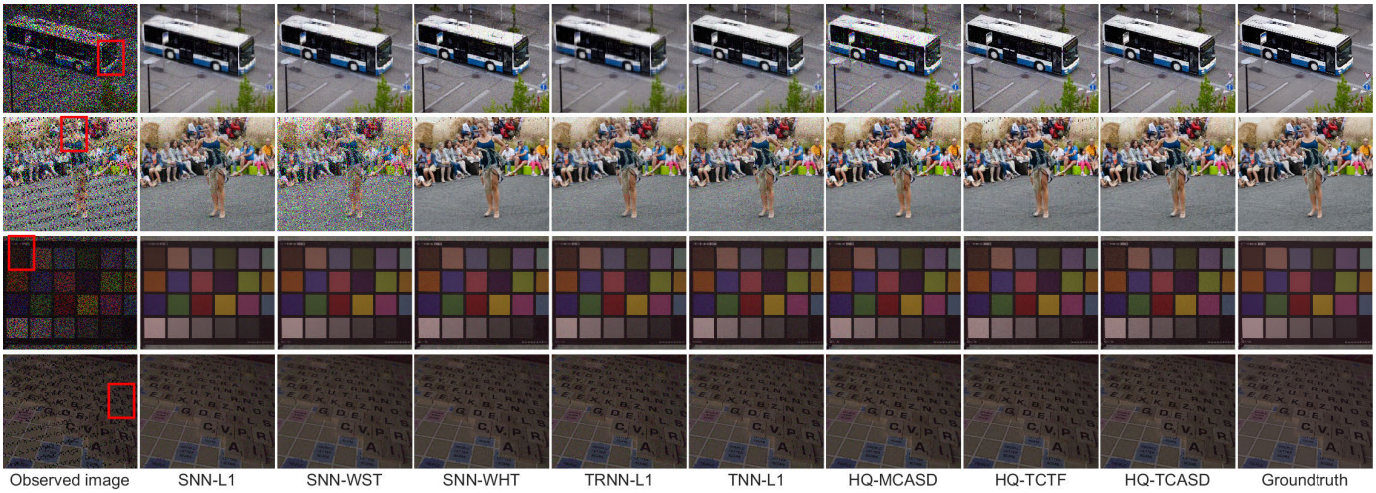


Fig. 6. Images recovered by different algorithms under different noise distributions with $c = 0.2$.

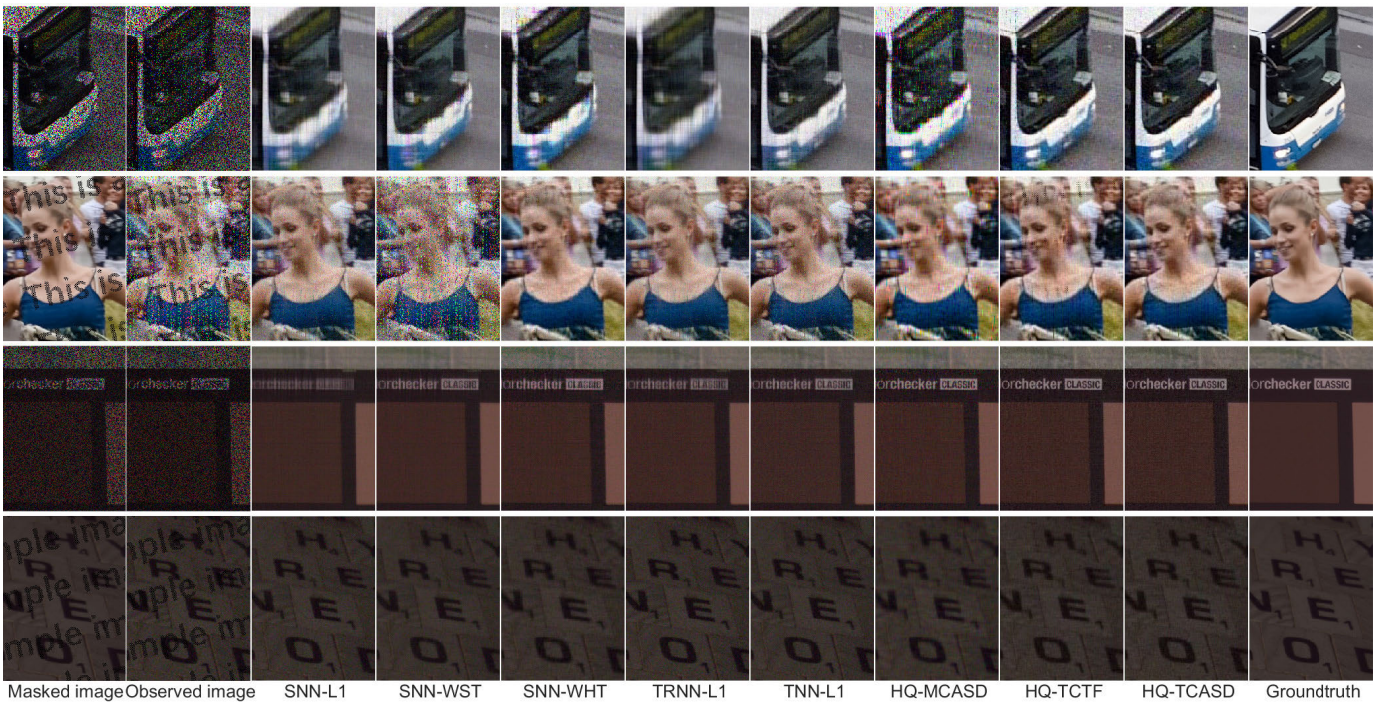


Fig. 7. Enlarged regions (red rectangles of Fig. 6) from the images recovered by different algorithms.

4) *Stripe PSP Noise*: This is similar to stripe GMM noise, but we replace the Gaussian noise $N(0, 1)$ and $N(0, 0.01)$ in stripe GMM noise with salt-and-pepper noise and Poisson noise, respectively.

The multirank vectors for HQ-TCASD and HQ-TCTF are set to [150, 20, 20] and [120, 20, 20], respectively. The average PSNR for the four images is reported in Table II for different values of the noise parameter $c$, and the average run time for each image is shown in Fig. 5. One can observe that HQ-TCASD achieves the highest PSNR for most of the images, and HQ-TCTF is the second best. Furthermore, parallel computation significantly reduces the computational cost of the proposed HQ-TCTF and HQ-TCASD. Examples of the recovered full and partially enlarged images are shown in Figs. 6 and 7, respectively. As shown, the methods proposed yield visually clearer texture and more accurate colors than the other methods.

### C. Video Data Completion

In this section, we evaluate the performance of the algorithms using video data. Four grayscale video sequences from the DAVIS 2016 dataset are used for testing completion performance. Due to the limitation of the computer memory, the resolution of the video is scaled down to $1280 \times 720$ from the original $1920 \times 1080$ resolution, and the first 30 frames of each sequence are selected, such that each video sequence

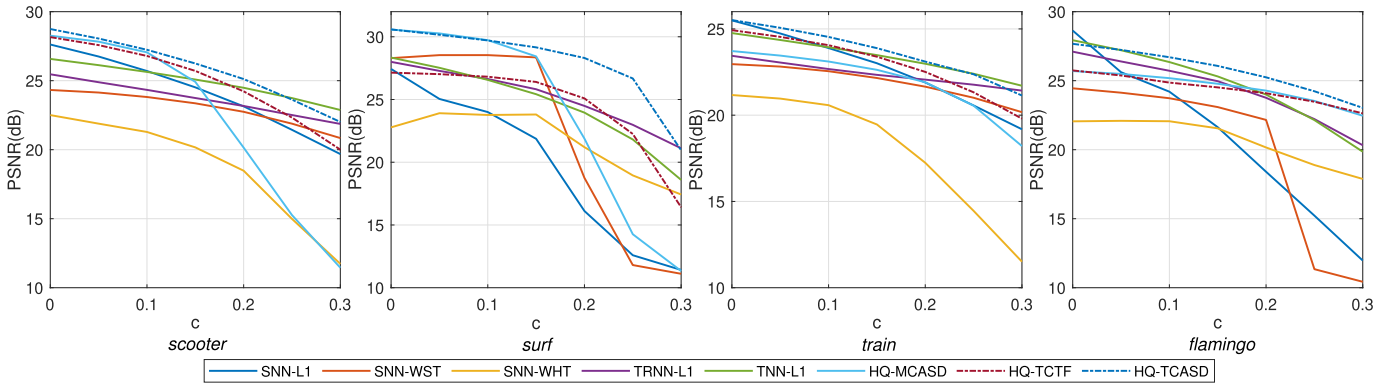


Fig. 8. Average PSNR on four videos from the DAVIS dataset versus parameter $c$. Missing patterns (from left to right): random (50%), watermark, and random (50%), watermark. Noise distributions (from left to right): stripe GMM, stripe PSP, GMM, and PSP. The dashed lines are for the proposed algorithms.

TABLE III
COMPLETION PERFORMANCE (RELATIVE ERROR) COMPARISON ON TRAFFIC DATA

| Parameter | Metric | SNN-L1 | SNN-WST | SNN-WHT | TRNN-L1 | TNN-L1 | HQ-MCASD | HQ-TCTF | HQ-TCASD |
|---|---|---|---|---|---|---|---|---|---|
| $c = 0.1$ | *rel.err* | 0.0673 | 0.0705 | 0.0753 | 0.0489 | 0.0436 | **0.0377** | 0.0385 | 0.0397 |
| | time(s) | 14437.7 | 8145.53 | 6891.92 | 12208.6 | 568.65 | 189.78 | 316.56 (223.52) | 959.25 (370.15) |
| $c = 0.2$ | *rel.err* | 0.0692 | 0.0728 | 0.0809 | 0.0511 | 0.0466 | 0.0575 | 0.0451 | **0.0423** |
| | time(s) | 10440.9 | 9423.65 | 6745.36 | 13850.3 | 710.64 | 311.89 | 351.35 (243.23) | 817.58 (327.38) |
| $c = 0.3$ | *rel.err* | 0.0726 | 0.0755 | 0.0935 | 0.0536 | 0.0527 | 0.1179 | 0.0560 | **0.0476** |
| | time(s) | 13092.0 | 10932.2 | 6328.92 | 14832.2 | 1011.82 | 618.94 | 396.08 (269.25) | 738.43 (291.34) |

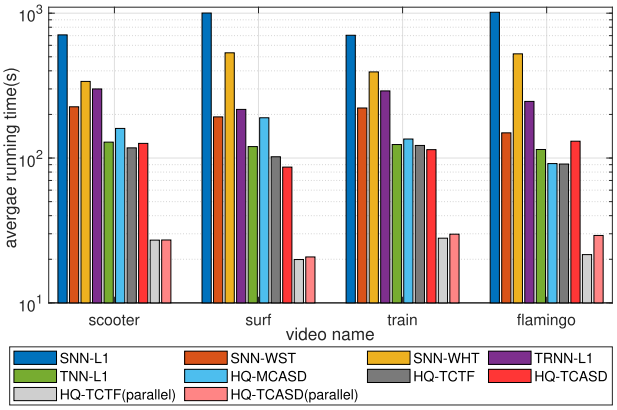


Fig. 9. Average running time for each video.

forms a tensor of size $1280 \times 720 \times 30$. The multirank vectors for HQ-TCASD and HQ-TCTF are set to $[80, 80, \ldots, 80]$ and $[80, 60, \ldots, 60]$, respectively.

Similar to the image inpainting task, we compare performance under different missing pixel patterns and noise distributions. The curves of average PSNR for different values of $c$ are shown in Fig. 8. The corresponding average running times are shown in Fig. 9. The proposed HQ-TCASD algorithm achieves the highest PSNR values in most situations, and HQ-TCASD achieves a threefold speedup over other algorithms using parallel computation. To shed more light on performance, Fig. 10 shows the examples of recovered video frames from four video sequences. In Fig. 11, we zoom in on the regions of Fig. 10 surrounded by the red rectangles. It can be seen that HQ-TCASD yields the frames that are less noisy and with better contrast than the ones recovered by the other methods.

We also investigate the performance with an increasing number of video frames. The "train" video with GMM noise $c = 0.2$ is utilized in this experiment. The video length is increased from 1 to 50 and the corresponding average PSNR curves of different algorithms are shown in Fig. 12 (right). As shown, first, the average PSNR increases rapidly as the number of frames increases. Then, the average PSNR of all algorithms remains unchanged or slightly decreases except SNN-WHT and SNN-L1. To better understand the tubal rank property with an increasing number of frames, we set the tubal multirank to the same value $r$ and compute the PSNR for the best $r$-tubal-rank approximation of the original video. The results are shown in Fig. 12 (left). It can be seen that the performance only degrades slightly as the number of frames increases. Therefore, one can use a fixed setting of the tubal rank for different numbers of frames, which is also verified in Fig. 12 (right).

### D. Traffic Data Prediction

In this section, we further evaluate the performance of the algorithms using traffic data. The traffic data are generated from the large-scale PeMS traffic speed dataset[3] [49]. The data register traffic speed time series from 11 160 sensors over four

[3]https://doi.org/10.5281/zenodo.3939793

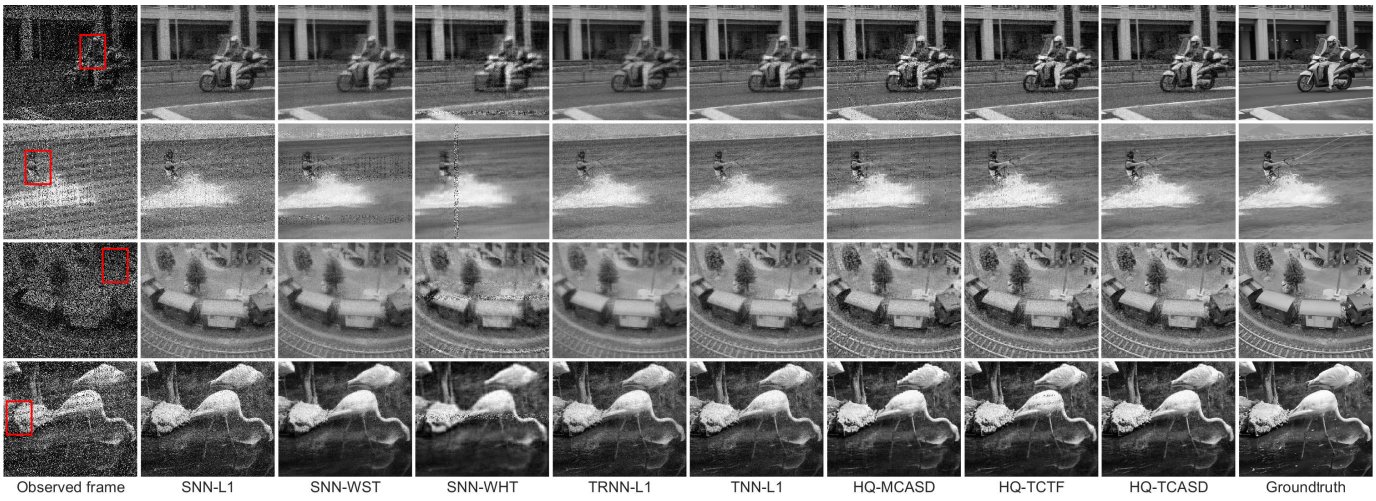


Fig. 10. Frames recovered by different algorithms under different noise distributions with $c = 0.2$.

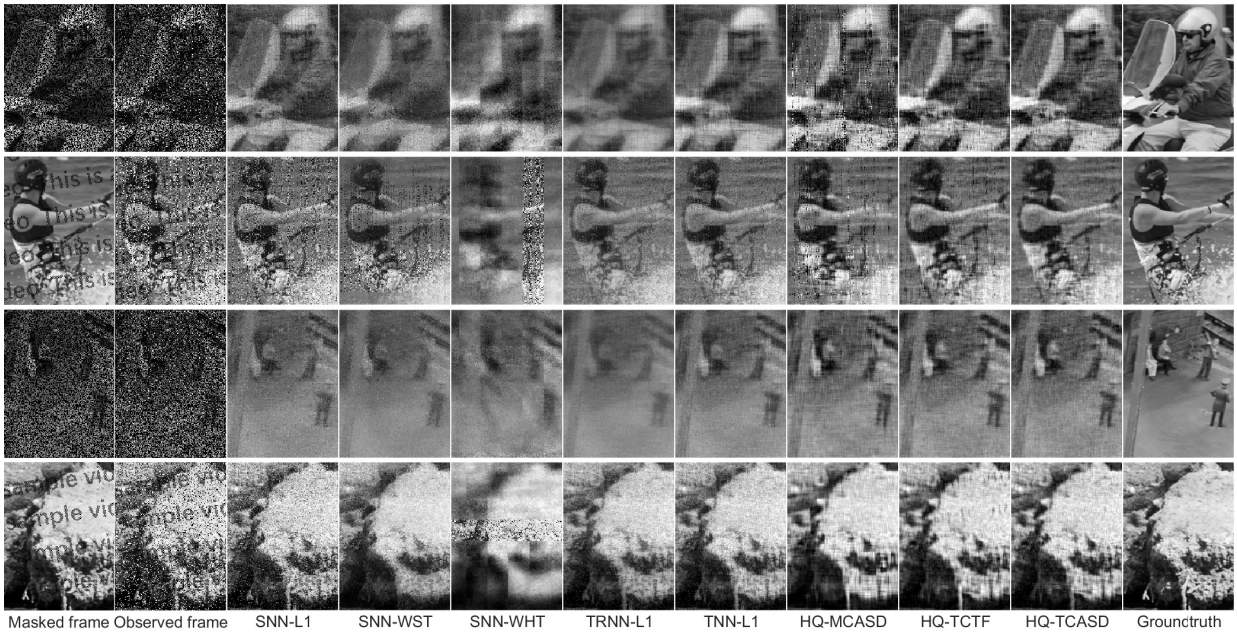


Fig. 11. Enlarged regions (red rectangles of Fig. 10) of recovered frames by different algorithms.

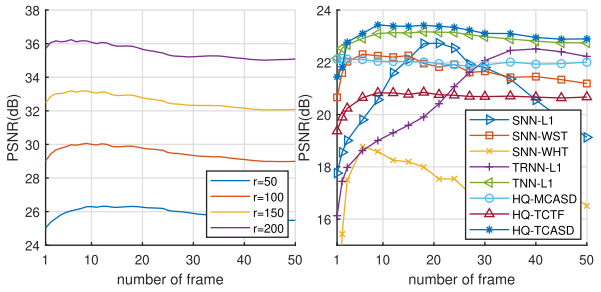


Fig. 12. Left: PSNR curves of best $r$-tubal-rank approximation of the original video with a different number of frames. Right: PSNR curves of different algorithms versus number of frames.

weeks with 288 time points per day (i.e., 5-min frequency) in California, USA. Thus, it forms a 11 160 $\times$ 288 $\times$ 28 tensor. Each value of the data is normalized such that all data are in the range [0, 1]. In this experiment, we randomly and uniformly selected 50% of the data points as the observed data. The noise parameter $\sigma_A^2$ is set to zero and the outliers have $\sigma_B^2 = 1$. For HQ-TCASD and HQ-TCTF, the elements of the multirank vector are all fixed at 20. Twenty Monte Carlo runs are performed for each value of $c$ with different selections of observed data and noise. The values of the average relative error under different simulation settings are reported in Table III. The running time in seconds of HQ-TCASD and HQ-TCTF using parallel computation is shown between brackets. HQ-TCASD achieves the best performance for $c = 0.2$ and 0.3. To better illustrate the recovery performance, an example of the data recovered from sensor No. 9960 on the 26th day under $c = 0.3$ is shown in Fig. 13. It can be seen that the proposed HQ-TCASD outperforms the other algorithms.

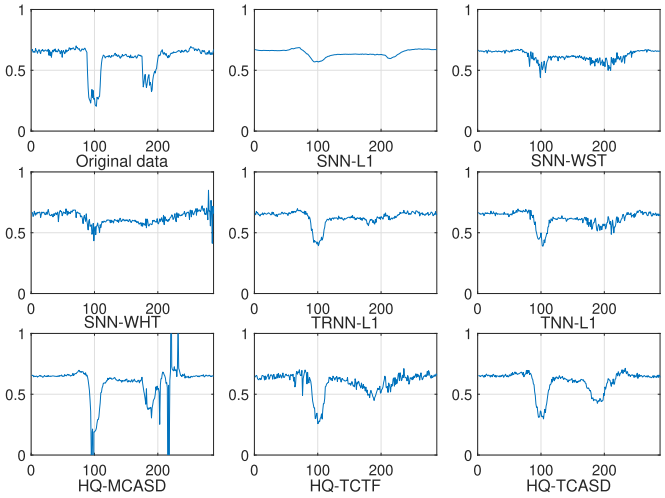


Fig. 13. Examples of the recovered missing signals of traffic data.

## V. Conclusion

In this article, we proposed a novel robust tensor completion method that utilizes tensor factorization to impose a low-tubal-rank structure, which avoids the computation of the SVD. The correntropy measure is introduced to alleviate the impact of large outliers. Based on an HQ minimization technique, two efficient robust tensor completion algorithms, HQ-TCTF and HQ-TCASD, were proposed and their convergence is analyzed. Experiments on both synthetic and real datasets demonstrate the superior performance of the proposed methods compared to existing state-of-the-art algorithms.

## Appendix
## Proof of Proposition 2

Since $\mathcal{W}$ is an optimal solution for (18), we have

$$J\left(\mathcal{X}^{t+1},\mathcal{Y}^{t+1},\mathcal{W}^{t+1}\right)\leq J\left(\mathcal{X}^{t+1},\mathcal{Y}^{t+1},\mathcal{W}^{t}\right). \tag{52}$$

By fixing $\mathcal{W}$ and defining $\boldsymbol{Q}=\sqrt{\tilde{\boldsymbol{W}}}\circ\tilde{\boldsymbol{P}}$, we obtain the following:

$$\begin{aligned}
&2J(\boldsymbol{U}^{t+1},\hat{\boldsymbol{Y}}^{t})-2J(\boldsymbol{U}^{t},\hat{\boldsymbol{Y}}^{t})\\
&=\left\|\boldsymbol{Q}\circ\left(\tilde{\boldsymbol{M}}-\boldsymbol{U}^{t+1}\hat{\boldsymbol{Y}}^{t}\right)\right\|_F^2-\left\|\boldsymbol{Q}\circ\left(\tilde{\boldsymbol{M}}-\boldsymbol{U}^{t}\hat{\boldsymbol{Y}}^{t}\right)\right\|_F^2\\
&=\left\|\boldsymbol{Q}\circ\left(\tilde{\boldsymbol{M}}-\left(\boldsymbol{U}^{t}-\mu_U^{\prime t}\boldsymbol{g}_U^{\prime t}\right)\hat{\boldsymbol{Y}}^{t}\right)\right\|_F^2-\left\|\boldsymbol{Q}\circ\left(\tilde{\boldsymbol{M}}-\boldsymbol{U}^{t}\hat{\boldsymbol{Y}}^{t}\right)\right\|_F^2\\
&=\left(\mu_U^{\prime t}\right)^2\left\|\boldsymbol{Q}\circ\left(\boldsymbol{g}_U^{\prime t}\hat{\boldsymbol{Y}}^{t}\right)\right\|_F^2+2\mu_U^{\prime t}\left\langle\boldsymbol{Q}\circ\left(\tilde{\boldsymbol{M}}-\boldsymbol{U}^{t}\hat{\boldsymbol{Y}}^{t}\right),\boldsymbol{g}_U^{\prime t}\hat{\boldsymbol{Y}}^{t}\right\rangle\\
&=\frac{(\|\boldsymbol{g}_U^{\prime t}\|_F^2)^2}{\left\|\boldsymbol{Q}\circ\left(\boldsymbol{g}_U^{\prime t}\hat{\boldsymbol{Y}}\right)\right\|_F^2}-2\mu_U^{\prime t}\left\langle\boldsymbol{g}_U^{t},\boldsymbol{g}_U^{\prime t}\right\rangle.
\end{aligned}\tag{53}$$

We can further simplify $\langle\boldsymbol{g}_U,\boldsymbol{g}_U'\rangle$ as

$$\begin{aligned}
\left\langle\boldsymbol{g}_U,\boldsymbol{g}_U'\right\rangle&=\operatorname{tr}\left(\boldsymbol{g}_U^{*}\boldsymbol{F}^{-1}\operatorname{bdiagz}(\boldsymbol{F}\boldsymbol{g}_U)\right)\\
&=\frac{1}{n_3}\operatorname{tr}\left((\boldsymbol{F}\boldsymbol{g}_U)^{*}\operatorname{bdiagz}(\boldsymbol{F}\boldsymbol{g}_U)\right)\\
&=\frac{1}{n_3}\left\|\operatorname{bdiagz}(\boldsymbol{F}\boldsymbol{g}_U)\right\|_F^2
\end{aligned}\tag{54}$$

where $\operatorname{tr}(\cdot)$ denotes the trace operator. Furthermore, $\|\boldsymbol{g}_U'\|_F^2$ can be simplified as

$$\|\boldsymbol{g}_U'\|_F^2=\frac{1}{n_3}\|\boldsymbol{F}\operatorname{bdiagz}(\boldsymbol{F}\boldsymbol{g}_U)\|_F^2=\frac{1}{n_3}\|\operatorname{bdiagz}(\boldsymbol{F}\boldsymbol{g}_U)\|_F^2 \tag{55}$$

where we use the fact that $\boldsymbol{F}^{*}\boldsymbol{F}=\boldsymbol{I}$. Therefore, according to (54) and (55), we have

$$\left\|\boldsymbol{g}_U'\right\|_F^2=\left\langle\boldsymbol{g}_U,\boldsymbol{g}_U'\right\rangle \tag{56}$$

and (53) can be written as

$$J(\boldsymbol{U}^{t+1},\hat{\boldsymbol{Y}}^{t})-J(\boldsymbol{U}^{t},\hat{\boldsymbol{Y}}^{t})=-\frac{\left(\|\boldsymbol{g}_U^{\prime t}\|_F^2\right)^2}{2\left\|\boldsymbol{Q}\circ\left(\boldsymbol{g}_U^{\prime t}\hat{\boldsymbol{Y}}\right)\right\|_F^2}\leq 0. \tag{57}$$

Similarly, we can obtain

$$\begin{aligned}
&J(\boldsymbol{U}^{t+1},\hat{\boldsymbol{Y}}^{t+1})-J(\boldsymbol{U}^{t+1},\hat{\boldsymbol{Y}}^{t})\\
&=-(1-\lambda)\frac{\left(\|\boldsymbol{g}_{\hat{\boldsymbol{Y}}}^{t}\|_F^2\right)^2}{2\left\|\boldsymbol{Q}\circ\left(\boldsymbol{U}^{t+1}\boldsymbol{g}_{\hat{\boldsymbol{Y}}}^{t}\right)\right\|_F^2}-\lambda\frac{|\langle\boldsymbol{g}_{\hat{\boldsymbol{Y}}}^{t},\boldsymbol{g}_{\hat{\boldsymbol{Y}}}^{\prime t}\rangle|^2}{2\left\|\boldsymbol{Q}\circ\left(\boldsymbol{U}^{t+1}\boldsymbol{g}_{\hat{\boldsymbol{Y}}}^{\prime t}\right)\right\|_F^2}\\
&\leq 0
\end{aligned}\tag{58}$$

and (57) and (58) imply that

$$J(\boldsymbol{U}^{t+1},\hat{\boldsymbol{Y}}^{t+1})\leq J(\boldsymbol{U}^{t},\hat{\boldsymbol{Y}}^{t}). \tag{59}$$

Thus, according to the relation between $\boldsymbol{U}$ and $\mathcal{X}$, and $\hat{\boldsymbol{Y}}$ and $\mathcal{Y}$, we have that

$$J(\mathcal{X}^{t+1},\mathcal{Y}^{t+1},\mathcal{W}^{t+1})\leq J(\mathcal{X}^{t},\mathcal{Y}^{t},\mathcal{W}^{t}). \tag{60}$$

It can also be verified that $J(\mathcal{X}^{t},\mathcal{Y}^{t},\mathcal{W}^{t})$ is always bounded below for arbitrary $t$. Thus, $\{J(\mathcal{X}^{t},\mathcal{Y}^{t},\mathcal{W}^{t}),t=1,2,\ldots\}$ will converge.

## References

[1] R. Dian, L. Fang, and S. Li, "Hyperspectral image super-resolution via non-local sparse tensor factorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3862–3871.

[2] S. Zhang, L. Wang, Y. Fu, X. Zhong, and H. Huang, "Computational hyperspectral imaging based on dimension-discriminative low-rank tensor recovery," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 10182–10191.

[3] Y. Xie, D. Tao, W. Zhang, Y. Liu, L. Zhang, and Y. Qu, "On unifying multi-view self-representations for clustering by tensor multi-rank minimization," *Int. J. Comput. Vis.*, vol. 126, no. 11, pp. 1157–1179, Nov. 2018.

[4] T. Zhao et al., "Multi-agent tensor fusion for contextual trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12118–12126.

[5] X. He, D. Cai, and P. Niyogi, "Tensor subspace analysis," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 499–506.

[6] Y. Xie et al., "Robust kernelized multiview self-representation for subspace clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 2, pp. 868–881, Feb. 2021.

[7] Y. Tang, Y. Xie, C. Zhang, Z. Zhang, and W. Zhang, "One-step multiview subspace segmentation via joint skinny tensor learning and latent clustering," *IEEE Trans. Cybern.*, vol. 52, no. 9, pp. 9179–9193, Sep. 2022.

[8] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, Jul. 2017.

[9] A. Cichocki et al., "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Process. Mag.*, vol. 32, no. 2, pp. 145–163, Mar. 2015.

[10] Y. Tang, Y. Xie, X. Yang, J. Niu, and W. Zhang, "Tensor multi-elastic kernel self-paced learning for time series clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 3, pp. 1223–1237, Mar. 2021.

[11] H. A. L. Kiers, "Towards a standardized notation and terminology in multiway analysis," *J. Chemometrics*, vol. 14, no. 3, pp. 105–122, 2000.

[12] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, Sep. 1966.

[13] Q. Zhao, G. Zhou, S. Xie, L. Zhang, and A. Cichocki, "Tensor ring decomposition," 2016, *arXiv:1606.05535*.

[14] M. E. Kilmer and C. D. Martin, "Factorization strategies for third-order tensors," *Linear Algebra its Appl.*, vol. 435, no. 3, pp. 641–658, Aug. 2011.

[15] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Found. Comput. Math.*, vol. 9, no. 6, pp. 717–772, Dec. 2009.

[16] E. J. Candes and Y. Plan, "Matrix completion with noise," *Proc. IEEE*, vol. 98, no. 6, pp. 925–936, Jun. 2010.

[17] P. Jain and S. Oh, "Provable tensor factorization with missing data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1431–1439.

[18] H. Kasai and B. Mishra, "Low-rank tensor completion: A Riemannian manifold preconditioning approach," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1012–1021.

[19] Y. Xu, R. Hao, W. Yin, and Z. Su, "Parallel matrix factorization for low-rank tensor completion," *Inverse Problems Imag.*, vol. 9, no. 2, pp. 601–624, 2015.

[20] Z. Zhang and S. Aeron, "Exact tensor completion using t-SVD," *IEEE Trans. Signal Process.*, vol. 65, no. 6, pp. 1511–1526, Mar. 2017.

[21] P. Zhou, C. Lu, Z. Lin, and C. Zhang, "Tensor factorization for low-rank tensor completion," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1152–1163, Mar. 2018.

[22] X. Liu, S. Aeron, V. Aggarwal, and X. Wang, "Low-tubal-rank tensor completion using alternating minimization," *IEEE Trans. Inf. Theory*, vol. 66, no. 3, pp. 1714–1737, Mar. 2020.

[23] K. Gilman, D. Ataee Tarzanagh, and L. Balzano, "Grassmannian optimization for online tensor completion and tracking with the t-SVD," 2020, *arXiv:2001.11419*.

[24] D. Goldfarb and Z. Qin, "Robust low-rank tensor recovery: Models and algorithms," *SIAM J. Matrix Anal. Appl.*, vol. 35, no. 1, pp. 225–253, Jan. 2014.

[25] X. Chen et al., "A generalized model for robust tensor factorization with noise modeling by mixture of Gaussians," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5380–5393, Nov. 2018.

[26] K. Inoue, K. Hara, and K. Urahama, "Robust multilinear principal component analysis," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 591–597.

[27] B. Huang, C. Mu, D. Goldfarb, and J. Wright, "Provable low-rank tensor recovery," *Optim.-Online*, vol. 4252, no. 2, pp. 455–500, 2014.

[28] Y. Yang, Y. Feng, and J. A. K. Suykens, "Robust low-rank tensor recovery with regularized redescending M-estimator," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 9, pp. 1933–1946, Sep. 2016.

[29] H. Huang, Y. Liu, Z. Long, and C. Zhu, "Robust low-rank tensor ring completion," *IEEE Trans. Comput. Imag.*, vol. 6, pp. 1117–1126, 2020.

[30] M. E. Kilmer, K. Braman, N. Hao, and R. C. Hoover, "Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging," *SIAM J. Matrix Anal. Appl.*, vol. 34, no. 1, pp. 148–172, Jan. 2013.

[31] Q. Jiang and M. Ng, "Robust low-tubal-rank tensor completion via convex optimization," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2649–2655.

[32] A. Wang, X. Song, X. Wu, Z. Lai, and Z. Jin, "Robust Low-tubal-rank tensor completion," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 3432–3436.

[33] W. Liu, P. P. Pokharel, and J. C. Principe, "Correntropy: Properties and applications in non-Gaussian signal processing," *IEEE Trans. Signal Process.*, vol. 55, no. 11, pp. 5286–5298, Nov. 2007.

[34] B. Chen, L. Xing, H. Zhao, N. Zheng, and J. C. Principe, "Generalized correntropy for robust adaptive filtering," *IEEE Trans. Signal Process.*, vol. 64, no. 13, pp. 3376–3387, Jul. 2016.

[35] R. He, W.-S. Zheng, and B.-G. Hu, "Maximum correntropy criterion for robust face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1561–1576, Aug. 2011.

[36] Y. He, F. Wang, Y. Li, J. Qin, and B. Chen, "Robust matrix completion via maximum correntropy criterion and half-quadratic optimization," *IEEE Trans. Signal Process.*, vol. 68, pp. 181–195, 2020.

[37] Y. Zheng, B. Chen, S. Wang, and W. Wang, "Broad learning system based on maximum correntropy criterion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 7, pp. 3083–3097, Jul. 2021.

[38] B. Chen, X. Liu, H. Zhao, and J. C. Principe, "Maximum correntropy Kalman filter," *Automatica*, vol. 76, pp. 70–77, Feb. 2017.

[39] S. Zhao, B. Chen, and J. C. Príncipe, "Kernel adaptive filtering with maximum correntropy criterion," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2011, pp. 2012–2017.

[40] M. Nikolova and M. K. Ng, "Analysis of half-quadratic minimization methods for signal and image recovery," *SIAM J. Sci. Comput.*, vol. 27, no. 3, pp. 937–966, Jan. 2005.

[41] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, "Tensor robust principal component analysis with a new tensor nuclear norm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 4, pp. 925–938, Apr. 2020.

[42] W. Hu, D. Tao, W. Zhang, Y. Xie, and Y. Yang, "The twist tensor nuclear norm for video completion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 12, pp. 2961–2973, Dec. 2017.

[43] J. P. Haldar and D. Hernando, "Rank-constrained solutions to linear matrix equations using PowerFactorization," *IEEE Signal Process. Lett.*, vol. 16, no. 7, pp. 584–587, Jul. 2009.

[44] A. Singh and J. C. Principe, "Using correntropy as a cost function in linear adaptive filters," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Feb. 2009, pp. 2950–2955.

[45] A. Singh, R. Pokharel, and J. Principe, "The C-loss function for pattern classification," *Pattern Recognit.*, vol. 47, no. 1, pp. 441–453, Jan. 2014.

[46] J. E. Dennis and R. E. Welsch, "Techniques for nonlinear least squares and robust regression," *Commun. Statist. Simul. Comput.*, vol. 7, no. 4, pp. 345–359, Jan. 1978.

[47] J. Tanner and K. Wei, "Low rank matrix completion by alternating steepest descent methods," *Appl. Comput. Harmon. Anal.*, vol. 40, no. 2, pp. 417–429, Mar. 2016.

[48] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 724–732.

[49] T. Mallick, P. Balaprakash, E. Rask, and J. Macfarlane, "Transfer learning with graph neural networks for short-term highway traffic forecasting," 2020, *arXiv:2004.08038*.