Robust Low-Tubal-Rank Tensor Completion Based on Tensor Factorization and Maximum Correntropy 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].

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

TABLE I

| Algorithm          | Rank model | Objective function |
|--------------------|------------|--------------------|
| SNN-L1 [24]        | Tucker     | \( \min_{\mathcal{X}} \sum_{i=1}^{N} \| X_{i} \|_{r} + \lambda \| S \|_{1} \) \|
|                    |            | \( \text{s.t. } \mathcal{P} \circ \left( \sum_{i=1}^{N} X_{i} + S \right) = \mathcal{P} \circ \mathcal{M} \) |

| SNN-ST [28]        | Tucker     | \( \min_{\mathcal{X}_{11\cdots i_{n}}} \sum_{i_{1}\cdots i_{n} \in \Omega} \rho_{0} \left( \left\langle M_{i_{1}\cdots i_{n}} - \sum_{m=1}^{M} X_{m_{i_{1}\cdots i_{n}}} \right\rangle \right) + \lambda \sum_{j=1}^{J} \| X_{j} \|_{r} \) |
|                    |            | \( \text{s.t. } \mathcal{P} \circ \left( \sum_{i_{1}\cdots i_{n} \in \Omega} X_{i_{1}\cdots i_{n}} + S \right) = \mathcal{P} \circ \mathcal{M} \) |

| SNN-HT [28]        | Tucker     | \( \min_{\mathcal{X}_{11\cdots i_{n}}} \sum_{i_{1}\cdots i_{n} \in \Omega} \rho_{0} \left( \left\langle M_{i_{1}\cdots i_{n}} - \sum_{m=1}^{M} X_{m_{i_{1}\cdots i_{n}}} \right\rangle \right) + \lambda \sum_{j=1}^{J} \| X_{j} \|_{r} \) |
|                    |            | \( \text{s.t. } \mathcal{P} \circ \left( \sum_{i_{1}\cdots i_{n} \in \Omega} X_{i_{1}\cdots i_{n}} + S \right) = \mathcal{P} \circ \mathcal{M} \) |

| TRNN-L1 [29]       | Tensor ring| \( \min_{\mathcal{X}, \mathcal{S}} \sum_{i=1}^{N} \| X_{i} \|_{L} + \lambda \| S \|_{1} \) \|
|                    |            | \( \text{s.t. } \mathcal{P} \circ \left( \mathcal{X} + S \right) = \mathcal{P} \circ \mathcal{M} \) |

| TNN-L1 [31, 32]    | Tubal      | \( \min_{\mathcal{X}, \mathcal{S}} \| \mathcal{R} \|_{1} + \lambda \| S \|_{1} \) \|
|                    |            | \( \text{s.t. } \mathcal{P} \circ \left( \mathcal{X} + S \right) = \mathcal{P} \circ \mathcal{M} \) |

| HQ-TCTF            | Tubal      | \( \min_{\mathcal{X}, j, k, \mathcal{P}_{j,k}} \sum_{i=1}^{N} \mathcal{P}_{j,k} \| X_{i,j,k} \|_{1} \) \|
|                    |            | \( \text{s.t. } \mathcal{P} \circ \left( \mathcal{X} + S \right) = \mathcal{P} \circ \mathcal{M} \) |

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 intro-
duces 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 com-
pletion 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 correntropy 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 prob-
lem. Then, two efficient and simple algorithms based on alter-
nating 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 factor-
ization 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 opti-
mization 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 oper-
ations 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 courrentropy-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., $\mathbf{X}$), and boldface letters are used to denote matrices (e.g., $X$). Unless stated otherwise, we focus on third-order tensors, i.e., $\mathbf{X} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ where $n_1$, $n_2$, and $n_3$ are the dimensions of each way of the tensor. The notations $\mathbf{X}(i, :, :)$, $\mathbf{X}(:, j, :)$, and $\mathbf{X}(::, k, :)$ denote the horizontal, lateral, and frontal slices of $\mathbf{X}$, respectively, and $\mathbf{X}(i, j, k)$, $\mathbf{X}(i, :, k)$, and $\mathbf{X}(i, :, :) -\mathbf{X}(::, k, :) -\mathbf{X}(::, :, k)$ denote the mode-1, mode-2, and mode-3 tubes, respectively, while $\mathbf{X}_{ijk}$ denotes the $(i, j, k)$th entry of tensor $\mathbf{X}$. The Frobenius norm of tensor is defined as $\|\mathbf{X}\|_F = (\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} |\mathbf{X}_{ijk}|^2)^{1/2}$.

In the frequency domain, $\mathbf{X}$ denotes the Fourier transform along the third mode of $\mathbf{X}$. We use the convention, $\tilde{\mathbf{X}} = \text{fft}(\mathbf{X}, [1, 3])$ to denote the Fourier transform along the third dimension. Similarly, we use $\mathbf{X} = \text{ifft}(\tilde{\mathbf{X}}, [1, 3])$ for the inverse transform. We also define the matrix $\tilde{\mathbf{X}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$

$$\tilde{\mathbf{X}} = \text{bdiag}(\tilde{\mathbf{X}}) = \begin{bmatrix} \tilde{X}^{(1)} & & \\ & \tilde{X}^{(2)} & \\ & & \vdots \\ & & & \tilde{X}^{(n_3)} \end{bmatrix}$$

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

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

Therefore, the following relation holds:

$$(F_{n_3} \otimes I_{n_1})bcirc(\tilde{\mathbf{X}})(F_{n_3}^{-1} \otimes I_{n_1}) = \tilde{\mathbf{X}}$$

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

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

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

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

$$\text{fold}(\tilde{\mathbf{X}}) = \mathbf{X}.$$ 

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

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

$$\mathbf{A} \ast \mathbf{B} = \text{fold}(bcirc(\mathbf{A}) \cdot \text{unfold}(\mathbf{B})).$$

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

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

1. $\|\mathbf{A}\|_F^2 = \frac{1}{n_1} \|\mathbf{A}\|_F^2$
2. $\mathbf{F} = \mathbf{A} \ast \mathbf{B}$ and $\tilde{\mathbf{F}} = \tilde{\mathbf{A}} \tilde{\mathbf{B}}$ 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 $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ can be factorized as $\mathbf{A} = \mathbf{U} \ast \mathbf{S} \ast \mathbf{V}^*$, where $\mathbf{U} \in \mathbb{R}^{n_1 \times n_1 \times n_3}$ and $\mathbf{V} \in \mathbb{R}^{n_2 \times n_2 \times n_3}$ are orthogonal and $\mathbf{S} \in \mathbb{R}^{n_2 \times n_2 \times n_3}$ is an $f$-diagonal tensor, i.e., each of the frontal slices of $\mathbf{S}$ is a diagonal matrix. The diagonal entries in $\mathbf{S}(::, 1)$ are called the singular values of $\mathbf{A}$.

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

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

We will also need the following definition of tensor multi-rank.

**Lemma 2 (Best Tubal Rank-r Approximation [30]):** Let the t-SVD of $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, be $\mathbf{A} = \mathbf{U} \ast \mathbf{S} \ast \mathbf{V}^*$, given a tubal rank $r$, define $\mathcal{A}_r := \sum_{i=1}^{n_1} \mathbf{U}(::, s, :) \ast \mathbf{S}(s, s, :) \ast \mathbf{V}^*(::, s, :)$. Then, $\mathcal{A}_r = \arg \min_{\mathbf{A} \in \mathcal{A}} \|\mathbf{A} - \tilde{\mathbf{A}}\|_F$ where $\mathcal{A} := \{\mathbf{X} \ast \mathbf{Y} \mid \mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}, \mathbf{Y} \in \mathbb{R}^{r \times n_2 \times n_3}\}$.

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

$$\text{rank}_m(\mathbf{A}) = \max(r_1, \ldots, r_{n_3})$$

where $r_i$ is the $i$th element of $r$. 

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
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 rank$_t(\mathcal{M}) \ll \max(n_1, n_2)$. Specifically, by defining the observed subset of entries $\Omega \subseteq [n_1] \times [n_2] \times [n_3]$ and its indicator tensor $\mathcal{P}$

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

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

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

(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} \|\mathcal{A}^{(i)}\|_*$

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

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

(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 $\mathcal{X} \in \mathbb{R}^{r_1 \times n_2 \times n_3}$ and $\mathcal{Y} \in \mathbb{R}^{r_2 \times n_2 \times n_3}$, where $r$ is the tubal rank of $\mathcal{M}$ [21]. The tensor factorization then solves tensor completion by utilizing the objective function

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

(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 $n_1$, 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]

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

(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 $\mathbb{E}[\cdot]$ is the expectation operator. Given a finite number of samples $\{x_i, y_i\}_{i=1}^N$ and using the Gaussian kernel $G_\sigma(x) = \exp\left(-\frac{(x^2) \sigma^2}{2}\right)$ as the kernel function, the correntropy can be approximated by

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

(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:

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

(8)

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

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

(9)

The cost function above is closely related to Welch’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

$$
\min_{\mathbf{X}, \mathbf{Y}} J(\mathbf{X}, \mathbf{Y}) := \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} \mathcal{P}_{ijk} \sigma^2 \left( 1 - G_\sigma(\mathcal{M}_{ijk} - (\mathbf{X} \ast \mathbf{Y})_{ijk}) \right). \tag{11}
$$

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

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

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

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

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

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

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 $J_{G_\sigma}(\mathbf{X}, \mathbf{Y})$ can be expressed as

$$
J_{G_\sigma}(\mathbf{X}, \mathbf{Y}) = \min_{\mathbf{W}} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} \left( \mathcal{W}_{ijk} \mathcal{P}_{ijk} \left( \mathcal{M}_{ijk} - (\mathbf{X} \ast \mathbf{Y})_{ijk} \right)^2 + \mathcal{P}_{ijk} \phi(\mathcal{W}_{ijk}) \right). \tag{15}
$$

Furthermore, by defining the augmented cost function

$$
J_{HQ}(\mathbf{X}, \mathbf{Y}, \mathbf{W}) = \| \sqrt{\mathbf{W}} \circ \mathcal{P} \circ (\mathcal{M} - \mathbf{X} \ast \mathbf{Y}) \|^2_F + \psi_\mathbf{G}(\mathbf{W}) \tag{16}
$$

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

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

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

$$
\mathcal{W}_{ijk} = G_\sigma(M_{ijk} - (\mathbf{X} \ast \mathbf{Y})_{ijk}), (i, j, k) \in \Omega. \tag{18}
$$

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

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

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

The weighting tensor $\mathbf{W}$ 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 $\mathbf{Z}$, (19) can be rewritten as

$$
\min_{\mathbf{X}, \mathbf{Y}, \mathbf{Z}} J(\mathbf{X}, \mathbf{Y}, \mathbf{Z}) := \| \sqrt{\mathbf{W}} \circ \mathcal{P} \circ (\mathcal{M} - \mathbf{Z}) \|^2_F + \beta \| \mathbf{X} \ast \mathbf{Y} - \mathbf{Z} \|^2_F \tag{20}
$$

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
where $\beta$ is the regularization parameter. To solve (20), one can again utilize alternating minimization and update $Z$, $X$, and $Y$ in turn. Specifically, by fixing $X$ and $Y$, we can update $Z$ as

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

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

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

Equation (22) is equivalent to the requirement that $P \circ (W \circ Z - W \circ M + \beta Z - \beta X \star Y) = 0$.

$$\frac{(1 - P) \circ (Z - X \star Y) = 0. \tag{23} \rightleftharpoons}$$

Thus, $Z$ can be obtained in a closed form as

$$Z = P \circ Z + (1 - P) \circ Z = X \star Y + \frac{W}{\beta 1 + W} \circ P \circ (M - X \star Y) \tag{24}$$

where $1$ denotes the tensor of all ones, and the division is elementwize. Furthermore, by fixing $Z$, (20) reduces to the following minimization:

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

According to Lemma 1, we have

$$\| X \star Y - Z \|^2_F = \frac{1}{n_3} \| X Y - Z \|^2_F. \tag{26}$$

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

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

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

$$\tilde{X}^{(k)} = \tilde{Z}^{(k)} (\tilde{Y}^{(k)})^* (\tilde{Y}^{(k)})^* \tilde{X}^{(k)} \quad \tilde{Y}^{(k)} = (\tilde{Y}^{(k)} (\tilde{Y}^{(k)})^*)^{1/2} (\tilde{X}^{(k)})^* \tilde{Z}^{(k)} \tag{28}$$

where $A^+$ denotes the Moore–Penrose pseudoinverse of matrix $A$. 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 $W$ become 1.

Algorithm 1 HQ-TCTF for Robust Tensor Completion

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

In this special case, one does not need to optimize $W$ in (18), and (16) reduces to

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

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

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 $r = \{r_1, \ldots, r_{n_3}\}$ 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

$$J(X, Y, Z, W) = \| \sqrt{W} \circ P \circ (M - X \star Y) \|^2_F + \| X \star Y - Z \|^2_F + \psi_\Theta(W). \tag{30}$$

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

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

$$J(X^{t+1}, Y^{t+1}, Z^{t+1}, W^{t+1}) \leq J(X^{t+1}, Y^{t+1}, Z^t, W^t). \tag{31}$$

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

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

From Lemma 3, we have

$$\| X^{t+1} \star Y^{t+1} - Z^{t+1} \|^2_F = (1/n_3) \sum_{k=1}^{n_3} \| X^{(k), t+1} \star Y^{(k), t+1} - Z^{(k), t+1} \|^2_F. \tag{33}$$

Combining (31) and (33), we have

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

It can be also verified that $J(X^t, Y^t, Z^t, W^t)$ is always bounded below for arbitrary $t$. Thus, $\{J(X^t, Y^t, Z^t, W^t), t = 1, 2, \ldots\}$ 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 \( WW \) using (18). Then, instead of directly optimizing (19), we gradually update \( XX \) and \( YY \) using gradient descent. For convenience, we first add a multiplicative factor of \((1/2)\) to (19) such that the minimization problem becomes

\[
\frac{1}{2} \min_{XX,YY} \| \sqrt{WW} \circ PP \circ (MM - XX \ast YY) \|_F^2. \tag{35}
\]

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

\[
\frac{1}{2} \min_{XX,YY} \| \sqrt{WW} \circ PP \circ (\hat{MM} - \text{bcirc}(XX)\hat{YY}) \|_F^2. \tag{36}
\]

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

\[
\text{bcirc}(XX)\hat{YY} = (F_{n-1}^{-1} \otimes I_n)XX\hat{YY} = F^{-1}XX\hat{YY} = U\hat{YY}, \tag{37}
\]

where \( F^{-1} = F_{n-1}^{-1} \otimes I_n \) (consequently, \( F = F^{-1} \times n_3 \)), \( U = F^{-1}XX \), and \( \hat{A} = \text{unfold}(A) \). Finally, (36) can be reformulated as

\[
\min_{U,YY} J(U,\hat{YY}) := \frac{1}{2} \| \sqrt{WW} \circ PP \circ (\hat{MM} - UYY) \|_F^2. \tag{38}
\]

Using the matrix derivatives, the partial derivative of \( J(U,YY) \) with respect to \( U \) can be computed as

\[
g_U = \frac{\partial J}{\partial U} = -\sqrt{WW} \circ PP \circ (\hat{MM} - UYY)^+. \tag{39}
\]

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

\[
g_U = F^{-1} \text{bdiag}(FG_U) . \tag{40}
\]

The steepest descent step size \( \mu_U \) for \( g_U \) can be obtained in the following minimizer

\[
\mu_U = \arg \min_{\mu} \left\{ \frac{\|g_U\|_F^2}{\sqrt{WW} \circ PP \circ (\hat{MM} - U + \mu U)YY} \right\}^2 . \tag{41}
\]

and the matrix \( U \) can be updated as

\[
U^{t+1} = U^t - \mu_U gg_U . \tag{42}
\]

Similarly, by fixing \( U \), the partial derivative of \( J \) with respect to \( YY \) can be obtained as

\[
g_Y = \frac{\partial J}{\partial YY} = -U^* (\sqrt{WW} \circ PP \circ (\hat{MM} - YY))^+ , \tag{43}
\]

The corresponding step size \( \mu_Y \) will be

\[
\mu_Y = \frac{\|g_Y\|_F^2}{\sqrt{WW} \circ PP \circ (U^*YY)} . \tag{44}
\]

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 \( YY \) in (43) by \( (U^*U)^{-1} \), i.e.,

\[
g_Y = (U^*U)^{-1}gg_Y , \tag{45}
\]

and the corresponding step size \( \mu_Y \) with exact line search is

\[
\mu_Y = \frac{\|g_Y\|_F^2}{\sqrt{WW} \circ PP \circ (U^*YY)} , \tag{46}
\]

where \( (A, B) := \sum_{1 \leq i, j \leq r} A_{ij} B_{ij} \). Therefore, the matrix \( YY \) at the \( t \)th iteration can be updated by combining (43) and (45), i.e.,

\[
YY^{t+1} = YY^t - (1 - \lambda)g_Yg_Y^+ - \mu_Yg_Y^+ , \tag{47}
\]

where \( 0 \leq \lambda \leq 1 \) is a free parameter to be chosen.

Therefore, the matrices \( U \) and \( YY \) can be alternately updated using (42) and (47) until convergence. We term the above algorithm “HQ-TCASD.”

Similar to HQ-TCF, 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 \( U(XX) \) and \( YY \) have a block structure, so the matrix computation can be processed block-by-block. Also, since we have \( FF_A = \text{unfold}(ft(\mathcal{A}, [], 3)) \) 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: | \( P, P \circ MM, r, \lambda , \sigma \) |
|-------|-------------------------------------|
| 1:    | initial matrices \( U^0 \) and \( YY^0 \), \( t = 0 \) |
| 2:    | repeat |
| 3:    | compute \( \sigma^{t+1} \) and \( YY^{t+1} \) using (18) |
| 4:    | compute \( U^{t+1} \) using (42) |
| 5:    | compute \( YY^{t+1} \) using (47) |
| 6:    | \( t = t + 1 \) |
| 7:    | until stopping criterion is satisfied |
| Output: | \( U^t \ast YY^t \) |

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

**Proposition 2:** Define the cost function

\[
J(XX, YY, WW) = \frac{1}{2} \| \sqrt{WW} \circ PP \circ (MM - XX \ast YY) \|_F^2 + \frac{1}{2} \psi_W(W). \tag{48}
\]

The sequence \( \{J(XX', YY', WW')\}, t = 1, 2, \ldots \) 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 \( WW \) 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 \( \mathcal{E}' \) is defined as

\[
\mathcal{E}' = \sqrt{\mathcal{W}' \circ \mathcal{P} \circ (\mathcal{M} - \mathcal{X}' \ast \mathcal{Y}')},
\]

(49)

If \( \| \mathcal{E}' \|_F - \| \mathcal{E}'^{-1} \|_F \) 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

\[
\sigma^{t+1} = \max \left( \eta \left( \max \left( e_{21}^{t+1}(0.25), e_{21}^{t+1}(0.75) \right) \right), \sigma_{\min} \right)
\]

(50)

where \( e_{21} \in \mathbb{R}^{[2] \times 1} \) denotes the vector composed of all nonzero entries of \( \mathcal{P} \circ (\mathcal{M} - \mathcal{X}' \ast \mathcal{Y}') \) and \( \sigma_{\min} \) denotes the \( q \)th quantile of \( \mathcal{Y} \). 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 \( \mathcal{X} \ast \mathcal{Y} \) and finding the quantile of \( e_{21} \), whose time complexities are \( \mathcal{O}(r_n + n_2 n_3)^3 \log n_3 + r n_1 n_2 n_3) \) and \( \mathcal{O}(n_1 n_2 n_3) \), respectively. The complexity of computing \( \mathcal{W} \) and \( \mathcal{Z} \) is both \( \mathcal{O}(n_1 n_2 n_3) \) since \( \mathcal{X} \ast \mathcal{Y} \) was already computed. Then, the cost of updating \( \mathcal{X} \) and \( \mathcal{Y} \) is \( \mathcal{O}(r_n + n_2 n_3)^3 \log n_3 + r n_1 n_2 n_3) \). Therefore, the overall complexity of HQ-TCTF is \( \mathcal{O}(r_n + n_2 n_3)^3 \log n_3 + r n_1 n_2 n_3) \).

For HQ-TCASD, similar to HQ-TCTF, the complexity of computing \( \sigma \) is \( \mathcal{O}(r_n + n_2 n_3)^3 \log n_3 + r n_1 n_2 n_3) \). Computing \( g_{t'} \) using FFT has complexity \( \mathcal{O}(r_n + n_2 n_3)^3 \log n_3 + r n_1 n_2 n_3) \), and calculation of \( \mu_t \) and \( \mu_{t'} \) is of complexity \( \mathcal{O}(r_n + n_2 n_3)^3 \log n_3 + r n_1 n_2 n_3) \). Therefore, the overall complexity of HQ-TCASD is \( \mathcal{O}(r_n + n_2 n_3)^3 \log n_3 + r n_1 n_2 n_3) \).

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 a matrix of dimension \( n_1 \times n_2 \).

In the experiments, we refer to this matrix-completion-based method as HQ-MCASD. 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 \( \mathcal{M} \) with tubal rank \( \hat{r} \) 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 \( \mathcal{P} \) with observation fraction \( p \) is generated by randomly and uniformly assigning \( p \times 100\% \) of the entries of \( \mathcal{P} \) the value 1. The performance of an instance of tensor completion is evaluated using the relative error

\[
\text{rel.err} = \frac{\| \hat{\mathcal{M}} - \mathcal{M} \|_F}{\| \mathcal{M} \|_F}
\]

(51)

where \( \hat{\mathcal{M}} \) 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 \( \mathcal{P} \) 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 \((1-c)N(0, \sigma^2_1) + cN(0, \sigma^2_2)\), where \(N(0, \sigma^2_1)\) represents the general Gaussian noise disturbance with variance \(\sigma^2_1\) and \(N(0, \sigma^2_2)\) with a large variance \(\sigma^2_2\) captures the outliers. The variable \(c\) controls the occurrence probability of outliers.
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 $\tilde{r}$ of $\mathcal{M}$ 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 $\tilde{r}$ 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 $\tilde{r} = 10$. We use a Gaussian noise distribution with $\sigma^2_A = 0.01$. 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 $\mathcal{M}$ with low relative error even when $r$ is set larger than actual $\tilde{r}$.

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 $c = 0.1$, $\sigma^2_A = 0.01$, and $\sigma^2_B = 10$. The rank $\tilde{r}$ is set to $n_1 \times 0.05$. The rank of HQ-TCASD with $\lambda = 1$ is set to $\tilde{r} + 5$ 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. 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 | SNNT-L1 | TRNN-L1 | TNN-L1 | HQ-MCASD | HQ-TCTF | HQ-TCASD |
|--------|-----------------|--------|-----|-----------|--------|---------|---------|---------|--------|----------|---------|----------|
| bus    | random(50%)     | Stripe | 0   | 23.62     | 25.36  | 26.27   | 25.51   | 24.58   | 27.25   | 26.26   | 28.76    | 29.12   | 28.45    |
|        |                 | GMM    | 0.1 | 7.69      | 25.01  | 25.69   | 25.19   | 23.95   | 26.26   | 25.75   | 28.16    | 27.28   | 28.45    |
|        |                 |        | 0.2 | 3.03      | 24.13  | 24.87   | 23.64   | 23.12   | 24.98   | 20.64   | 26.47    | 27.26   | 28.45    |
|        |                 |        | 0.3 | -1.34     | 23.10  | 23.66   | 19.97   | 22.46   | 23.22   | 11.95   | 20.45    | 24.50   |          |
| dance  | watermark       | Stripe | 0   | 22.44     | 29.06  | 29.96   | 29.47   | 29.85   | 29.25   | 29.59   | 27.47    | 31.24   |          |
|        |                 | PSP    | 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   | 15.56   | 11.94   | 18.47    | 21.58   |          |
| board  | random(50%)     | Unknown\ | \   | 24.49     | 37.91  | 35.89   | 34.44   | 37.94   | 38.23   | 37.23   | 39.43    | 39.58   |          |
|        |                 | Unknown\ | \   | 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    | 23.73   |          |
| alphabet | watermark  | Unknown\ | \   | 22.63     | 37.50  | 38.37   | 36.91   | 37.23   | 37.88   | 35.65   | 36.76    | 38.01   |          |
|        |                 | Unknown\ | \   | 19.21     | 31.65  | 33.02   | 33.02   | 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

\[
\text{PSNR} = 10 \log_{10} \left( \frac{I_{\text{max}}^3}{\| \mathbf{M} - \hat{\mathbf{M}} \|_F^2} \right)
\]

where \( I_{\text{max}} \) 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.\(^1\) 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. \(\text{GMM Noise}\): 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. \(\text{Poisson+Salt-and-Pepper (PSP) Noise}\): 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 by Poisson noise.
3. \(\text{Stripe GMM noise}\): For each channel, 50% of the columns are randomly selected, of which \(2c \times 100\%\) of the observed pixels are perturbed with Gaussian noise \(\mathcal{N}(0, 1)\). The remaining observed pixels are perturbed by Gaussian noise \(\mathcal{N}(0, 0.01)\).

---

\(^1\)https://davischallenge.org/davis2016/code.html
\(^2\)https://www.eecs.yorku.ca/~kamel/sidd/index.php

Fig. 5. Average running time for each image.
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

| Parameter | Metric | SNN-L1 | SNN-WST | SNN-WHT | TRNN-L1 | TNN-L1 | HQ-MCASD | HQ-TCTF | HQ-TCASD |
|-----------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| $c = 0.1$ | \textit{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) |
| $c = 0.2$ | \textit{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) |
| $c = 0.3$ | \textit{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\footnote{https://doi.org/10.5281/zenodo.3939793} [49]. The data register traffic speed time series from 11 160 sensors over four
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 $11 \times 160 \times 28 \times 28$ 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 $\sigma^2$ is set to zero and the outliers have $\sigma^2_B = 1$. 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.
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 corentropy 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

\[
J(\mathbf{X}^{t+1}, \mathbf{Y}^{t+1}, \mathbf{W}^{t+1}) \leq J(\mathbf{X}^{t+1}, \mathbf{Y}^{t+1}, \mathbf{W}^t). \tag{52}
\]

By fixing \( \mathcal{W} \) and defining \( \mathcal{Q} = \sqrt{\mathcal{W}} \circ \hat{\mathcal{P}} \), we obtain the following:

\[
2J(U^{t+1}, \hat{Y}^t) - 2J(U^t, \hat{Y}^t) = \| \mathcal{Q} \circ (M - U^{t+1} \hat{Y}^t) \|_F^2 - \| \mathcal{Q} \circ (M - U^t \hat{Y}^t) \|_F^2
= \| \mathcal{Q} \circ (M - (U^t - \mu_t \hat{g}^t) \hat{Y}^t) \|_F^2 - \| \mathcal{Q} \circ (M - U^t \hat{Y}^t) \|_F^2
= (\mu_t^2) \| \mathcal{Q} \circ (\hat{g}^t \hat{Y}^t) \|_F^2 + 2\mu_t \mathcal{Q} \circ (M - U^t \hat{Y}^t), \hat{g}^t \hat{Y}^t)
= -\frac{(\| \hat{g}^t \|_F^2)}{\| \mathcal{Q} \circ (\hat{g}^t \hat{Y}^t) \|_F^2} - 2\mu_t \mathcal{Q} \circ (M - U^t \hat{Y}^t), \hat{g}^t \hat{Y}^t). \tag{53}
\]

We can further simplify \( \langle \hat{g}^t, \hat{g}^t \rangle \) as

\[
\langle \hat{g}^t, \hat{g}^t \rangle = \text{tr}(\hat{g}^t F^{-1} \text{bdiag}(F \hat{g}^t)) = \frac{1}{n_3} \text{tr}(F \hat{g}^t) \text{bdiag}(F \hat{g}^t) = \frac{1}{n_3} \| \text{bdiag}(F \hat{g}^t) \|_F^2. \tag{54}
\]

where \( \text{tr}(\cdot) \) denotes the trace operator. Furthermore, \( \| \hat{g}^t \|_F^2 \) can be simplified as

\[
\| \hat{g}^t \|_F^2 = \frac{1}{n_3} \| F \text{bdiag}(F \hat{g}^t) \|_F = \frac{1}{n_3} \| \text{bdiag}(F \hat{g}^t) \|_F^2
\]

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

\[
\| \hat{g}^t \|_F^2 = \langle \hat{g}^t, \hat{g}^t \rangle \tag{56}
\]

and (53) can be written as

\[
J(U^{t+1}, \hat{Y}^t) - J(U^t, \hat{Y}^t) = \frac{\| \hat{g}^t \|_F^2}{2} - \frac{\lambda}{2} \langle \hat{g}^t, \hat{g}^t \rangle \leq 0. \tag{57}
\]

Similarly, we can obtain

\[
J(U^{t+1}, \hat{Y}^{t+1}) - J(U^{t+1}, \hat{Y}^t)
= -(1 - \lambda) \frac{\langle \hat{g}^t, \hat{g}^t \rangle}{2} - \frac{\lambda}{2} \langle \hat{g}^t, \hat{g}^t \rangle^2 \leq 0 \tag{58}
\]

and (57) and (58) imply that

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

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

\[
J(\mathbf{X}^{t+1}, \mathbf{Y}^{t+1}, \mathbf{W}^{t+1}) \leq J(\mathbf{X}^t, \mathbf{Y}^t, \mathbf{W}^t). \tag{60}
\]

It can also be verified that \( J(\mathbf{X}^t, \mathbf{Y}^t, \mathbf{W}^t) \) is always bounded below for arbitrary \( t \). Thus, \( \{J(\mathbf{X}^t, \mathbf{Y}^t, \mathbf{W}^t), t = 1, 2, \ldots \} \) 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 multi-subrepresentations 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. 3551–3562, Jul. 2017.
[9] A. Cichocki et al., “Tensor decompositions for signal processing applications: From two-way to multway component analysis,” IEEE Signal Process. Mag., vol. 32, no. 2, pp. 145–163, Mar. 2015.
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.

H. A. L. Kiers, “Towards a standardized notation and terminology in multiway analysis,” J. Chemometrics, vol. 14, no. 3, pp. 105–122, 2000.

L. R. Tucker, “Some mathematical notes on three-mode factor analysis,” Psychometrika, vol. 31, no. 3, pp. 279–311, Sep. 1966.

Q. Zhao, G. Zhou, S. Xie, L. Zhang, and A. Cichocki, “Tensor ring decomposition,” 2016, arXiv:1606.05375.

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.

E. J. Candès and B. Recht, “Exact matrix completion via convex optimization,” Found. Comput. Math., vol. 9, no. 6, pp. 717–772, Dec. 2009.

E. J. Candès and Y. Plan, “Matrix completion with noise,” Proc. IEEE, vol. 98, no. 6, pp. 925–936, Jun. 2010.

P. Jain and S. Oh, “Provable tensor factorization with missing data,” in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 1431–1439.

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

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.

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

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.

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.

K. Gilman, D. Ataee Tarzanagh, and L. Balzano, “Grassmannian optimization for online tensor completion and tracking with the t-SVD,” 2020, arXiv:2001.11419.

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.

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.

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.

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

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.

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.

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.