



OPEN ACCESS

EDITED BY

Abiy Tasissa,
Tufts University, United States

REVIEWED BY

Lei Wang,
Beihang University, China
Tomohiro Hashizume,
University of Hamburg, Germany

*CORRESPONDENCE

Tatsuya Yokota
✉ t.yokota@nitech.ac.jp

RECEIVED 17 March 2025

ACCEPTED 18 August 2025

PUBLISHED 17 September 2025

CITATION

Mukai M, Hontani H and Yokota T (2025)
Plug-and-play low-rank tensor completion
and reconstruction algorithms with improved
applicability of tensor decompositions.
Front. Appl. Math. Stat. 11:1594873.
doi: 10.3389/fams.2025.1594873

COPYRIGHT

© 2025 Mukai, Hontani and Yokota. This is an
open-access article distributed under the
terms of the [Creative Commons Attribution
License \(CC BY\)](#). The use, distribution or
reproduction in other forums is permitted,
provided the original author(s) and the
copyright owner(s) are credited and that the
original publication in this journal is cited, in
accordance with accepted academic practice.
No use, distribution or reproduction is
permitted which does not comply with these
terms.

Plug-and-play low-rank tensor completion and reconstruction algorithms with improved applicability of tensor decompositions

Manabu Mukai¹, Hidekata Hontani¹ and Tatsuya Yokota^{1,2*}

¹Department of Computer Science, Nagoya Institute of Technology, Aichi, Japan, ²RIKEN Center for Advanced Intelligence Project, Tokyo, Japan

In this paper, we propose a new unified optimization algorithm for general tensor completion and reconstruction problems, which is formulated as an inverse problem for low-rank tensors in general linear observation models. The proposed algorithm supports at least three basic loss functions (ℓ_2 loss, ℓ_1 loss, and generalized KL divergence) and various TD models (CP, Tucker, TT, TR decompositions, non-negative matrix/tensor factorizations, and other constrained TD models). We derive the optimization algorithm based on a hierarchical combination of the alternating direction method of multipliers (ADMM) and majorization-minimization (MM). We show that the proposed algorithm can solve a wide range of applications and can be easily extended to any established TD model in a plug-and-play manner.

KEYWORDS

tensor decompositions, tensor completion, tensor reconstruction, majorization-minimization (MM), alternating direction method of multipliers (ADMM), plug-and-play (PnP), generalized KL divergence

1 Introduction

Tensor decompositions (TDs) are beginning to be used in various fields of applications such as image recovery [1], blind source separation [2], traffic data analysis [3], wireless communications [4], and quantum state tomography [5–7]. Tensor decompositions are mathematical models that are used directly to exploit the low-rank structure of tensors [8–10]. In addition, other various optional structures such as non-negativity [11–13], sparsity [14–17], smoothness [18–21], and manifold constraints [22–24] can be incorporated into factor matrices or core tensors in tensor decompositions. Such flexible modeling in tensor decompositions makes it popular for wide applications. In applications of tensor decompositions, various data analysis tasks can be addressed by designing optimization problems based on the assumptions of low-rank and other additional data structures and optimizing parameters to maximize the consistency (likelihood) with the observed data.

The challenges of applying tensor decomposition to each data analysis task are 2-fold. First, it is necessary to appropriately design TD models and optimization problems with constraints according to the objective of each data analysis task and the observation model of measurement data, which requires deep domain-specific insight and experience. The second challenge is to derive and implement an efficient optimization algorithm for the designed specific tensor decomposition problem, which requires high-level computer science skills such as applied linear algebra, numerical computation, mathematical optimization, and programming languages. Overcoming these two challenges is not easy, as it is rare to find someone who is both an expert in a specific domain and well-versed in

computer science. Constructing a project team that includes experts in the specific domain and computer science for this purpose can be useful, but costs remain high. The first challenge is closely related to research in the target domain and will be difficult to solve fundamentally. However, the second challenge can be solved by technological advances. Here, we will move on to the discussion of solving the second challenge.

In conventional research, a new tensor modeling method is proposed to solve a specific task, the corresponding optimization algorithm is derived, implemented, and experimented on, and the whole is published as a single paper. This research approach is common and robust, but it is very costly because a custom optimization algorithm must be derived and implemented for each subdivided problem. For example, the problem of recovering tensor data that includes missing values is known as the tensor completion problem [25–29]. Now, if we consider a model in which the true tensor is low Tucker/CP rank, the core tensor or factor matrices are non-negative, and the observations contain Gaussian noise, the optimization problem can be formulated as a non-negative Tucker/CP decomposition with missing values based on ℓ_2 loss minimization. It is possible to derive and implement specific optimization algorithms with some expertise [30, 31]. However, in some applications, one often wants to change the assumed noise distribution, try different TD models, or add constraints to the factor matrices [32–37]. It would be possible to derive and implement a specific optimization algorithm each time such a model change is made, but this would be very costly.

One approach to solving the above problems is to develop a universal optimization algorithm that can solve a variety of data analysis problems based on tensor decomposition. Having a single universal solver frees most users from the requirements of derivation and implementation of individual algorithms, allowing them to focus on designing the models. This environment will encourage users to iterate through trial-and-error modeling and accelerate applications of tensor decomposition. The technical challenge in developing a universal optimization algorithm is how to formulate the problem and how to design the structure of the algorithm. A universal optimization algorithm needs to efficiently connect various TD models to various reconstruction problems, but finding the method is not trivial.

In this paper, as a major step toward this goal, we propose a unified algorithm for obtaining any tensor decomposition under various noisy linear observation models. The proposed algorithm supports three basic loss functions (ℓ_2 loss, ℓ_1 loss, and generalized KL divergence) and various low-rank TD models. Since our formulation is considered as a general linear observation model, the proposed algorithm can address a variety of problems such as noise removal, tensor completion, deconvolution, super-resolution, compressed sensing, and medical imaging [1].

We derive the optimization algorithm based on the hierarchical combination of the alternating direction method of multipliers (ADMM) [38], majorization-minimization (MM) [39, 40] algorithm and least-squares (LS) based tensor decomposition. The most distinctive feature of our approach is that it uses LS-based tensor decompositions as plug-and-play modules (denoisers). LS-based algorithms have been well established for various types of TD such as canonical polyadic decomposition (CPD) [41–43], Tucker

decomposition (TKD) [44–48], tensor-train decomposition (TTD) [49, 50], tensor-ring decomposition (TRD) [51], and non-negative matrix/tensor factorizations (NMF/NTF) [11, 52, 53]. Next, we derive an MM algorithm for solving the tensor decomposition problem in a linear observation model based on ℓ_2 loss (Gaussian noise). Since various TD algorithms can be adopted as modules, we can support various TD models at this point. Finally, we derive an ADMM to minimize the ℓ_1 loss and the generalized KL divergence for observations under Laplace and Poisson noises. As a result, an overall ADMM algorithm calls MM in the loop, and the MM calls TD algorithms for the update rule.

It should be noted that the structure of the proposed algorithm can incorporate different TD models in a plug-and-play (PnP) manner. This is inspired by work that applies arbitrary denoisers in a plug-and-play manner in image reconstruction such as PnP-ADMM [54]. LS-based TD can be viewed as a denoising that the low-rank tensor reconstruction from noisy tensor. In our study, we only assume the existence of LS-based TD algorithms and therefore also include *undiscovered TD models* that will be proposed in the future. In addition, the proposed framework can be easily extended for penalized matrix/tensor decompositions. Many sophisticated TD models can be applied as prior to linear inverse problems. For example, our method allows some sophisticated decomposition models proposed for tensor completion to be easily extended to other tasks such as robust tensor decomposition, deconvolution, compressive sensing, and computed tomography.

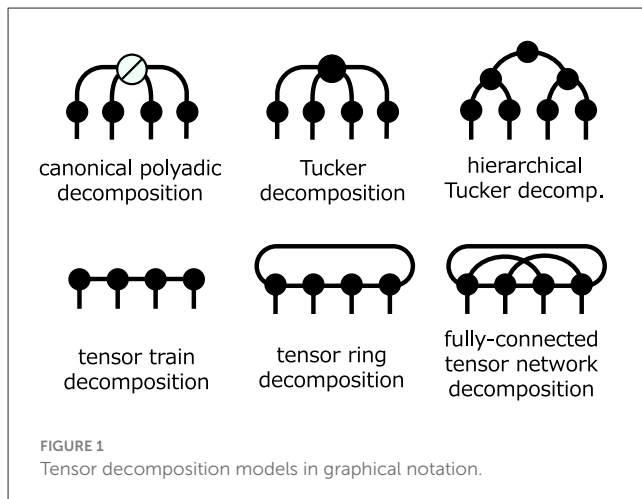
1.1 Notations

A vector, a matrix, and a tensor are indicated by a bold lowercase letter, $\mathbf{a} \in \mathbb{R}^I$, a bold uppercase letter, $\mathbf{B} \in \mathbb{R}^{I \times J}$, and a bold calligraphic letter, $\mathcal{C} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$, respectively. An N th-order tensor, $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$, can be transformed into a vector, which is denoted by using the same character, $\mathbf{x} = \text{vec}(\mathcal{X}) \in \mathbb{R}^{\prod_{n=1}^N I_n}$. An (i_1, i_2, \dots, i_N) -element of \mathcal{X} is denoted by x_{i_1, i_2, \dots, i_N} or $[\mathcal{X}]_{i_1, i_2, \dots, i_N}$. The operators \boxtimes represent the Hadamard product. \dagger represents the Moore-Penrose pseudoinverse. $\text{sign}(\mathcal{X}) \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ and $\text{abs}(\mathcal{X}) \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ are, respectively, operations that return the sign and absolute value for each entry of $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$.

2 Review of existing tensor completion and reconstruction methods from a perspective of plug-and-play algorithms

2.1 LS-based tensor decomposition

Tensor decomposition (TD) is a mathematical model to represent a tensor as a product of tensors/matrices. There are many TD models such as canonical polyadic decomposition (CPD) [41–43, 55], Tucker decomposition (TKD) [44–46], tensor-train decomposition (TTD) [49, 50], tensor-ring decomposition (TRD) [51], block-term decomposition [56–58], coupled tensor



decomposition [59–61], hierarchical Tucker decomposition [62], tensor wheel decomposition [63], fully connected tensor network [64], t-SVD [65], Hankel tensor decomposition [66–69], and convolutional tensor decomposition [21].

Figure 1 shows typical models of tensor decompositions in graphical notation [10]. In graphical notation, each node represents a core tensor and the edges connecting the nodes represent tensor products. As can be seen in the figure, all TD models are expressed by tensor products of core tensors.

In this paper, we will write any TD model as

$$\mathcal{X} = \langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}. \quad (1)$$

Note that Equation 1 is a somewhat abstract expression which only represents that the entire tensor \mathcal{X} is reconstructed by multiplying the L core tensors $\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_{L-1}$ and \mathcal{G}_L . The least squares (LS) based tensor decomposition problem of a given tensor $\mathcal{V} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ is given by

$$(\mathcal{G}_1^*, \mathcal{G}_2^*, \dots, \mathcal{G}_L^*) = \underset{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L}{\operatorname{argmin}} \|\mathcal{V} - \langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle\|_F^2. \quad (2)$$

This squared error is a non-convex function with respect to overall optimization parameters $(\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L)$, but it is a convex quadratic function with respect to only one core tensor \mathcal{G}_l for any $l \in \{1, 2, \dots, L\}$. If we focus on optimizing a certain core tensor \mathcal{G}_l and temporarily consider all other core tensors as constant variables and can be summarized as a single tensor, denoted as \mathcal{G}_{-l} , then the sub-optimization problem is reduced to an LS-based matrix optimization problem:

$$\begin{aligned} & \underset{\mathcal{G}_l}{\operatorname{argmin}} \|\mathcal{V} - \langle \mathcal{G}_l, \mathcal{G}_{-l} \rangle\|_F^2 \\ &= \underset{\mathcal{G}_l}{\operatorname{argmin}} \|\operatorname{mat}_l(\mathcal{V}) - \operatorname{mat}_l(\mathcal{G}_l) \operatorname{mat}_l(\mathcal{G}_{-l})\|_F^2, \end{aligned} \quad (3)$$

where we used abstract notations $\langle \mathcal{G}_l, \mathcal{G}_{-l} \rangle = \langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle$, and mat_m , $m \in \{1, 2, 3\}$, are the corresponding appropriate

matricization operators for the certain l -th core tensor. Solving sub-optimization shown in Equation 3 has a closed-form solution and updating \mathcal{G}_l by

$$\mathcal{G}_l \leftarrow \operatorname{mat}_l^{-1} \left(\operatorname{mat}_l(\mathcal{V}) \operatorname{mat}_l(\mathcal{G}_{-l})^\dagger \right) \quad (4)$$

always reduces (non-increases) the squared error. In other words, putting $\theta = (\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L)$ and $f(\theta) = \|\mathcal{V} - \langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle\|_F^2$, the following algorithm

$$\theta_k \leftarrow (\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L); \quad (5)$$

$$\mathcal{G}_l \leftarrow \operatorname{mat}_l^{-1} \left(\operatorname{mat}_l(\mathcal{V}) \operatorname{mat}_l(\mathcal{G}_{-l})^\dagger \right); \quad (6)$$

$$\theta_{k+1} \leftarrow (\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L); \quad (7)$$

provides the result $f(\theta_k) \geq f(\theta_{k+1})$ for any $l \in \{1, 2, \dots, L\}$. Therefore, the LS-based tensor decomposition problem in Equation 2 can be solved by repeating the steps shown in Equation 6 for every $l \in \{1, 2, \dots, L\}$, and this is called the alternating least squares (ALS) algorithm [42, 43, 47, 50, 51]. ALS is a workhorse algorithm in TDs because it has no hyper-parameter (e.g., step-size of gradient descent), and the objective function can be stably reduced (monotonically non-increasing).

2.1.1 Perspective on optimizing \mathcal{X} by ALS

Here, let us interpret that the ALS algorithm optimizes \mathcal{X} rather than $(\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L)$. This does not make any difference in the process. The LS-based low-rank tensor optimization problem can be formulated by

$$\mathcal{X}^* = \underset{\mathcal{X}}{\operatorname{argmin}} \|\mathcal{V} - \mathcal{X}\|_F^2 + i_{\mathbb{S}_t}(\mathcal{X}), \quad (8)$$

where $\mathbb{S}_t \subset \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ is a set of low-rank tensors which have tensor decomposition $\langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle$,

$$\mathbb{S}_t := \{ \langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle \mid \mathcal{G}_1 \in \mathcal{D}_1, \mathcal{G}_2 \in \mathcal{D}_2, \dots, \mathcal{G}_L \in \mathcal{D}_L \}, \quad (9)$$

\mathcal{D}_l is a domain of \mathcal{G}_l and $i_{\mathbb{S}_t}(\mathcal{X})$ is an indicator function

$$i_{\mathbb{S}_t}(\mathcal{X}) = \begin{cases} 0 & \mathcal{X} \in \mathbb{S}_t \\ \infty & \text{otherwise} \end{cases}. \quad (10)$$

Since the objective functions in Equations 2, 8 are the same when $\mathcal{X} = \langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle$, their minimizers also satisfy $\mathcal{X}^* = \langle \mathcal{G}_1^*, \mathcal{G}_2^*, \dots, \mathcal{G}_L^* \rangle$ if \mathcal{X}^* is unique. Note that the core tensors are generally non-unique (e.g., we have $\langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle = \langle c\mathcal{G}_1, c^{-1}\mathcal{G}_2, \dots, \mathcal{G}_L \rangle$ for any $c \neq 0$).

Remark 1. If \mathcal{X}^* is unique, then the solutions to the problems shown in Equations 2, 8 satisfy $\mathcal{X}^* = \langle \mathcal{G}_1^*, \mathcal{G}_2^*, \dots, \mathcal{G}_L^* \rangle$.

In practice, \mathcal{X}^* may not be unique, but even then there will always be a pair of solutions such that $\mathcal{X}^* = \langle \mathcal{G}_1^*, \mathcal{G}_2^*, \dots, \mathcal{G}_L^* \rangle$. Hence, the problem in Equation 8 can be solved by ALS. Updating

the core tensors θ by ALS simultaneously implies updating \mathcal{X} . Since the series $\theta_0 \rightarrow \theta_1 \rightarrow \dots \rightarrow \theta_\infty$ results in $\mathcal{X}_0 \rightarrow \mathcal{X}_1 \rightarrow \dots \rightarrow \mathcal{X}_\infty$, we pay attention to the fact that the ALS produces a series of $\{\mathcal{X}_k\}$ here. In this paper, we denote the operation of updating \mathcal{X} by ALS as

$$\mathcal{X}_{k+1} = \mathcal{U}_{\mathcal{V}, \mathbb{S}_t}^{\text{ALS}}(\mathcal{X}_k), \quad (11)$$

where $\mathcal{U}_{\mathcal{V}, \mathbb{S}_t}^{\text{ALS}}: \mathbb{S}_t \rightarrow \mathbb{S}_t$ can be considered as an interior point algorithm to a point closer to \mathcal{V} :

$$\|\mathcal{V} - \mathcal{U}_{\mathcal{V}, \mathbb{S}_t}^{\text{ALS}}(\mathcal{X})\|_F \leq \|\mathcal{V} - \mathcal{X}\|_F \quad (12)$$

for any $\mathcal{X} \in \mathbb{S}_t$. Although it is omitted in Equation 11, it is essential for implementation that the input and output of $\mathcal{U}_{\mathcal{V}, \mathbb{S}_t}^{\text{ALS}}$ include not only \mathcal{X} but also θ . We suppose that an operation $\mathcal{U}_{\mathcal{V}, \mathbb{S}_t}^{\text{ALS}}$ outputs a reconstructed tensor resulting from updating at least once for all core tensors. The approximate solution to the problem shown in Equation 8 can be found by repeating the ALS update sufficiently as

$$\mathcal{X}^* \approx \text{Proj}_{\mathbb{S}_t}^{\text{ALS}}(\mathcal{V}, \mathcal{X}_0) := \mathcal{U}_{\mathcal{V}, \mathbb{S}_t}^{\text{ALS}} \circ \dots \circ \mathcal{U}_{\mathcal{V}, \mathbb{S}_t}^{\text{ALS}}(\mathcal{X}_0), \quad (13)$$

where $\mathcal{X}_0 \in \mathbb{S}_t$ is some initialization of \mathcal{X} . Here, the entire ALS is represented as $\text{Proj}_{\mathbb{S}_t}^{\text{ALS}}(\mathcal{V}, \mathcal{X}_0)$. This is because Equation 8 is a problem in finding the closest point in \mathbb{S}_t from a point \mathcal{V} , and it is just a projection of \mathcal{V} onto \mathbb{S}_t . Note that \mathcal{X}_0 cannot be ignored since almost all TD algorithms depend on its initialization.

2.2 LS-based tensor completion and EM-ALS

Tensor completion is a task to estimate missing values in an observed incomplete tensor by using the low-rank structure of a tensor. In case of low-rank matrix completion, theories, algorithms, and applications are well studied [30, 35, 70–73]. Unlike matrices, which have a unique rank definition, tensors have various ranks (e.g., CP rank, Tucker rank, and TT rank), and the meaning of low rank is different for decomposition models [26, 27, 64, 74–77]. Since the appropriate TD model depends on the application, it is desirable to have an environment in which various TD models can be freely selected and tested.

Here, we introduce a basic formulation of LS-based tensor completion as follows:

$$\underset{\mathcal{X}}{\text{minimize}} \quad \|\mathcal{W} \boxtimes (\mathcal{T} - \mathcal{X})\|_F^2 + i_{\mathbb{S}_t}(\mathcal{X}), \quad (14)$$

where $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ is an observed incomplete tensor and \mathcal{W} is a mask tensor of the same size as \mathcal{T} . The entries of \mathcal{W} are given by

$$\mathcal{W}_{i_1 i_2 \dots i_N} = \begin{cases} 1 & \mathcal{T}_{i_1 i_2 \dots i_N} \text{ is observed} \\ 0 & \mathcal{T}_{i_1 i_2 \dots i_N} \text{ is missing} \end{cases}. \quad (15)$$

To solve the problem in Equation 14, many algorithms have been studied [21, 26, 28, 30, 78–80]. The gradient descent-based optimization algorithm [26] is proposed for the CP model. For the Tucker model, which imposes orthogonality on factor matrices, algorithms based on optimization on a manifold are proposed [28, 81]. However, these algorithms are designed specifically for specific TD models and it is difficult to generalize them to various TD models.

Expectation-maximization alternating least squares (EM-ALS) [82] is a versatile tensor completion algorithm that is less dependent on differences in TD models. In fact, EM-ALS is incorporated into various tensor completion algorithms such as TMac [29], TMac-TT [76], MTRD [77], MDT-Tucker [66, 67], and SPC [34]. The algorithm of EM-ALS can be derived from majorization-minimization (MM) [40, 83] which iteratively minimizes an auxiliary function $g(\mathcal{X}, \mathcal{X}')$ that serves as an upper bound on the objective function $f(\mathcal{X})$. In more detail, an auxiliary function holds the following conditions

$$g(\mathcal{X}, \mathcal{X}') \geq f(\mathcal{X}) \text{ and } g(\mathcal{X}, \mathcal{X}) = f(\mathcal{X}), \quad (16)$$

and the update rule of MM is given by

$$\mathcal{X}_{k+1} = \underset{\mathcal{X}}{\text{argmin}} \quad g(\mathcal{X}, \mathcal{X}_k). \quad (17)$$

This ensures that the objective function is monotonically decreasing as follows:

$$f(\mathcal{X}_k) = g(\mathcal{X}_k, \mathcal{X}_k) \geq g(\mathcal{X}_{k+1}, \mathcal{X}_k) \geq f(\mathcal{X}_{k+1}). \quad (18)$$

Specifically, the objective function and its auxiliary functions are given by

$$f(\mathcal{X}) := \|\mathcal{W} \boxtimes (\mathcal{T} - \mathcal{X})\|_F^2 + i_{\mathbb{S}_t}(\mathcal{X}), \quad (19)$$

$$g(\mathcal{X}, \mathcal{X}') := f(\mathcal{X}) + \|\overline{\mathcal{W}} \boxtimes (\mathcal{X}' - \mathcal{X})\|_F^2, \quad (20)$$

where $\overline{\mathcal{W}} = \mathbf{1} - \mathcal{W}$ is a tensor that flips the 0 and 1 of \mathcal{W} . Looking at the additional terms in Equation 20, we can see that they clearly satisfy the condition shown in Equation 16. Furthermore, update rule can be transformed as

$$\mathcal{X}_{k+1} = \underset{\mathcal{X}}{\text{argmin}} \quad \|\mathcal{V}_k - \mathcal{X}\|_F^2 + i_{\mathbb{S}_t}(\mathcal{X}), \quad (21)$$

where $\mathcal{V}_k = \mathcal{W} \boxtimes \mathcal{T} + \overline{\mathcal{W}} \boxtimes \mathcal{X}_k$. Since the structure of Equation 21 is the same as Equation 8, it can be solved by ALS. Finally, EM-ALS can be given by the following algorithm:

$$\mathcal{V}_k = \mathcal{W} \boxtimes \mathcal{T} + \overline{\mathcal{W}} \boxtimes \mathcal{X}_k; \quad (22)$$

$$\mathcal{X}_{k+1} = \text{Proj}_{\mathbb{S}_t}^{\text{ALS}}(\mathcal{V}_k, \mathcal{X}_k); \quad (23)$$

where $\mathcal{X}_0 \in \mathbb{S}_t$ is required for the initialization. The steps shown in Equations 22, 23 are called the E-step and the M-step, respectively.

Since M-step is an iterative algorithm, EM-ALS becomes a double iterative algorithm and is inefficient. Since the auxiliary function can also be decreased by one step of ALS $\mathcal{X}_{k+1} = \mathcal{U}_{\mathcal{V}_k, \mathcal{S}_t}^{\text{ALS}}(\mathcal{X}_k)$, the M-step can often be replaced with this. In EM-ALS, the operation that depends on the TD model is only the part in Equation 23. This means that any update rule of various LS-based TDs that decreases the objective function can be used as is.

Remark 2. Plug-and-play of TD algorithms is possible for tensor completion in EM-ALS.

2.3 Robust tensor decomposition and ADMM

Robust tensor decomposition (RTD) is a task to reconstruct a low-rank tensor from an observed tensor with outliers. Typically, the outlier components are assumed to be sparse (or follow a Laplace distribution) and the problem is formulated as a tensor decomposition based on ℓ_1 loss as follows:

$$\mathcal{X}^* = \underset{\mathcal{X}}{\operatorname{argmin}} \|\mathcal{B} - \mathcal{X}\|_1 + i_{\mathcal{S}_t}(\mathcal{X}), \quad (24)$$

where $\mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ is an observed tensor that includes outlier components. Various TD models and algorithms for RTD have been proposed, such as CPD [84], TKD [85], TRD [86], t-SVD [87], and more sophisticated models for specific domains [37].

Zhang and Ding [85] have proposed Tucker-based RTD using the alternating direction method of multipliers (ADMM). Zhang and Ding's ADMM formulation is also versatile, like EM-ALS, and we review it briefly here. First, we introduce a new variable $\mathcal{E} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ and add a constraint $\mathcal{E} = \mathcal{B} - \mathcal{X}$ to the RTD problem shown in Equation 24. For the constrained optimization problem, the augmented Lagrangian is given by

$$\mathcal{L}_{\text{RTD}}(\mathcal{X}, \mathcal{E}, \Lambda) = \|\mathcal{E}\|_1 + i_{\mathcal{S}_t}(\mathcal{X}) + \langle \Lambda, \mathcal{B} - \mathcal{X} - \mathcal{E} \rangle + \frac{\beta}{2} \|\mathcal{B} - \mathcal{X} - \mathcal{E}\|_F^2, \quad (25)$$

where $\Lambda \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ is a Lagrange multiplier, and $\beta > 0$ is a hyperparameter. In each step of ADMM, \mathcal{X} is updated as a minimizer of $\mathcal{L}_{\text{RTD}}(\mathcal{X}, \mathcal{E}, \Lambda)$ with respect to \mathcal{X} , \mathcal{E} is updated as a minimizer of $\mathcal{L}_{\text{RTD}}(\mathcal{X}, \mathcal{E}, \Lambda)$ with respect to \mathcal{E} , and Λ is updated by the method of multipliers. ADMM algorithm is given by

$$\mathcal{X}_{k+1} = \underset{\mathcal{X}}{\operatorname{argmin}} i_{\mathcal{S}_t}(\mathcal{X}) + \frac{\beta}{2} \left\| \mathcal{B} - \mathcal{X} - \mathcal{E}_k + \frac{\Lambda_k}{\beta} \right\|_F^2; \quad (26)$$

$$\mathcal{E}_{k+1} = \underset{\mathcal{E}}{\operatorname{argmin}} \|\mathcal{E}\|_1 + \frac{\beta}{2} \left\| \mathcal{B} - \mathcal{X}_{k+1} - \mathcal{E} + \frac{\Lambda_k}{\beta} \right\|_F^2; \quad (27)$$

$$\Lambda_{k+1} = \Lambda_k + \beta(\mathcal{B} - \mathcal{X}_{k+1} - \mathcal{E}_{k+1}); \quad (28)$$

where initializations \mathcal{X}_0 , \mathcal{E}_0 , and Λ_0 are required. The sub-optimization problems shown in Equations 26, 27 can be solved

by LS-based TD algorithm and soft-thresholding, respectively. In practice, Equations 26, 27 can be replaced as

$$\mathcal{V}_k = \mathcal{B} - \mathcal{E}_k + \frac{\Lambda_k}{\beta}; \quad (29)$$

$$\mathcal{X}_{k+1} = \operatorname{Proj}_{\mathcal{S}_t}^{\text{ALS}}(\mathcal{V}_k, \mathcal{X}_k); \quad (30)$$

$$\mathcal{E}_{k+1} = \operatorname{soft}_{\frac{1}{\beta}} \left[\mathcal{B} - \mathcal{X}_{k+1} + \frac{\Lambda_k}{\beta} \right]; \quad (31)$$

where $\operatorname{soft}_{\rho}(\cdot)$ with $\rho > 0$ is a soft-thresholding operator:

$$\operatorname{soft}_{\rho}[\mathcal{A}] := \operatorname{sign}(\mathcal{A}) \square \max(\operatorname{abs}(\mathcal{A}) - \rho, 0). \quad (32)$$

In a similar way to EM-ALS, $\operatorname{Proj}_{\mathcal{S}_t}^{\text{ALS}}(\mathcal{V}_k, \mathcal{X}_k)$ often be replaced with $\mathcal{U}_{\mathcal{V}_k, \mathcal{S}_t}^{\text{ALS}}(\mathcal{X}_k)$ in Equation 30, and the any update rule of various LS-based TD can be used as is.

Remark 3. Plug-and-play of TD algorithms is possible for robust tensor decomposition in ADMM.

2.4 Missing elements for versatile tensor reconstruction

In this paper, we consider solving a more versatile tensor reconstruction problem than tensor completion and RTD. First, we assume the following linear observation model

$$b_j = \langle \mathcal{A}_j, \mathcal{X} \rangle + n_j \quad (33)$$

for $j \in \{1, 2, \dots, J\}$, where an observation b_j is obtained from innerproduct between a given tensor $\mathcal{A}_j \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ and an unknown low-rank tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ with noise component n_j . This observation model can be transformed into vector-matrix form as

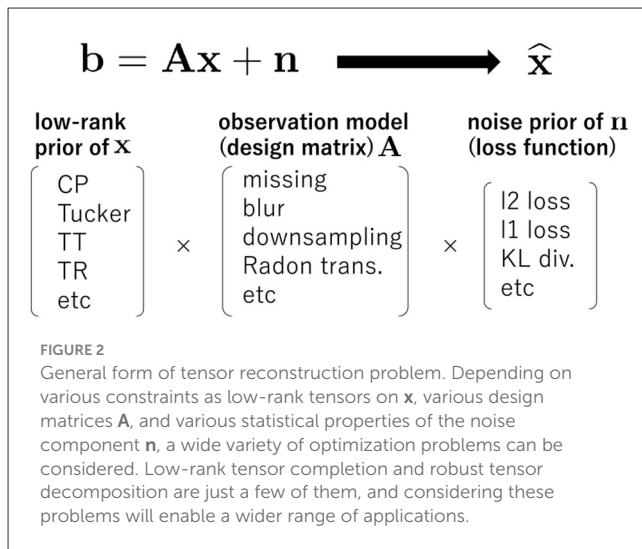
$$\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{n}, \quad (34)$$

where $\mathbf{b} = [b_1, b_2, \dots, b_J]^T \in \mathbb{R}^J$ is an observed signal, $\mathbf{n} = [n_1, n_2, \dots, n_J]^T \in \mathbb{R}^J$ is noise component, $\mathbf{x} = \operatorname{vec}(\mathcal{X})$ is a vector form of low-rank tensor, and

$$\mathbf{A} = \begin{bmatrix} \operatorname{vec}(\mathcal{A}_1)^T \\ \operatorname{vec}(\mathcal{A}_2)^T \\ \vdots \\ \operatorname{vec}(\mathcal{A}_J)^T \end{bmatrix} \in \mathbb{R}^{J \times (I_1 I_2 \dots I_N)} \quad (35)$$

is a design matrix. Introducing a low-rank TD constraint $\mathbf{x} \in \mathbb{S}$ instead of $\mathcal{X} \in \mathcal{S}_t$ and loss function based on the noise assumption of \mathbf{n} , the optimization problem can be given as

$$\underset{\mathbf{x}}{\operatorname{minimize}} D(\mathbf{b}, \mathbf{A}\mathbf{x}) + i_{\mathbb{S}}(\mathbf{x}), \quad (36)$$



where $\mathbb{S} := \{\text{vec}(\langle \mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L \rangle) \mid \mathcal{G}_1 \in \mathcal{D}_1, \mathcal{G}_2 \in \mathcal{D}_2, \dots, \mathcal{G}_L \in \mathcal{D}_L\}$ is a space of vectorization of low-rank tensors, $D(\cdot, \cdot)$ stands for loss function such as ℓ_2 loss, ℓ_1 loss and generalized Kullback-Leibler (KL) divergence. This study aims to solve the problem shown in Equation 36. Note that $\mathbf{x} \in \mathbb{R}^I$ is a I -dimensional vector in the form, and we consider that \mathbf{x} represents a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ by $\mathbf{x} = \text{vec}(\mathcal{X})$, where $I = \prod_{i=1}^N I_i$.

The problem in Equation 36 includes low-rank tensor completion and RTD. When the loss function is ℓ_2 , $\mathbf{b} = \text{vec}(\mathcal{W} \boxtimes \mathcal{T})$, and $\mathbf{A} = \text{diag}(\text{vec}(\mathcal{W}))$, then the problem in Equation 36 reduces to the tensor completion problem in Equation 14. When the loss function is ℓ_1 , $\mathbf{b} = \text{vec}(\mathcal{B})$, and $\mathbf{A} = \mathbf{I}$, then the problem in Equation 36 reduces to the RTD problem in Equation 24.

Figure 2 shows the concept of the tensor reconstruction problem considered in this study. In our problem formulation in Equation 36 includes various other patterns of tensor reconstruction. Tensor completion and RTD are just a few of them, and there are still many missing patterns. We aim to solve the problem in Equation 36 for any design matrix $\mathbf{A} \in \mathbb{R}^{I \times I}$, and other loss functions such as the generalized Kullback-Leibler (KL) divergence. This generalization is important for various applications. For the design matrix \mathbf{A} , the Toeplitz matrix is used in image deblurring, the downsampling matrix is used in the super-resolution task, the Radon transform matrix is used in computed tomography and the random projection matrix is used in compressed sensing [1]. For loss functions, ℓ_2 loss is used in Gaussian noise setting, ℓ_1 loss is used in Laplace noise (or sparse noise) setting, and generalized KL divergence is used in Poisson noise setting.

Furthermore, we consider a penalized version as follow:

$$\underset{\mathbf{x}}{\text{minimize}} D(\mathbf{b}, \mathbf{Ax}) + i_{\mathbb{S}}(\mathbf{x}) + \alpha p(\theta), \quad (37)$$

where $\alpha \geq 0$ and $p(\theta)$ is a penalty function for core tensors $\theta = (\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L)$. This is a generalization of the problem in Equation 36, since Equation 37 with $\alpha = 0$ is

equivalent to Equation 36. This plays an important role for the introduction of Tikhonov regularization, sparse regularization, low-rank regularization (e.g., nuclear norm), and smoothing. In particular, Tikhonov regularization of the core tensors reduces the problem of non-uniqueness of scale and improves the convergence of the TD algorithm [88].

3 Proposed method

3.1 Sketch of optimization framework

The key idea to solve a diverse set of problems formulated in Equation 36 is to employ the *plug-and-play* (PnP) approach of TD algorithms such as EM-ALS and ADMM. In this study, we first solve the case of ℓ_2 loss using LS-based TD with the MM framework, which is a generalization of EM-ALS. Furthermore, we use it to solve the cases of ℓ_1 loss and KL divergence with the ADMM framework. Thus, we call the proposed algorithm *ADMM-MM*. By replacing the LS-based TD module in a PnP manner, various types of TD can be easily generalized and applied to various applications.

3.2 Optimization

In this section, we explain one-by-one how to optimize the problem in Equation 37 for three loss functions: ℓ_2 loss, ℓ_1 loss, and generalized KL divergence. Then, we explain how to combine these three cases as the ADMM-MM algorithm.

3.2.1 Preliminary of LS-based TD

Key module in ADMM-MM is LS-based TD. In vector formulation \mathbf{x} instead of \mathcal{X} , the ALS algorithm of LS-based TD for $\mathbf{v} = \text{vec}(\mathcal{V})$ is denoted as

$$\begin{aligned} & \underset{\mathbf{x}}{\text{argmin}} \|\mathbf{v} - \mathbf{x}\|_2^2 + i_{\mathbb{S}}(\mathbf{x}) \\ & \approx \text{Proj}_{\mathbb{S}}^{\text{ALS}}(\mathbf{v}, \mathbf{x}_0) := \mathcal{U}_{\mathbf{v}, \mathbb{S}}^{\text{ALS}} \circ \dots \circ \mathcal{U}_{\mathbf{v}, \mathbb{S}}^{\text{ALS}}(\mathbf{x}_0), \end{aligned} \quad (38)$$

where $\text{Proj}_{\mathbb{S}}^{\text{ALS}}$, $\mathcal{U}_{\mathbf{v}, \mathbb{S}}^{\text{ALS}}$ and \mathbf{x}_0 correspond to $\text{Proj}_{\mathbb{S}_t}^{\text{ALS}}$, $\mathcal{U}_{\mathbf{v}, \mathbb{S}_t}^{\text{ALS}}$ and \mathcal{X}_0 , respectively.

Note that $\text{Proj}_{\mathbb{S}}^{\text{ALS}}$ and $\mathcal{U}_{\mathbf{v}, \mathbb{S}}^{\text{ALS}}$ do not necessarily have to be strictly based on ALS, and may be replaced by hierarchical ALS (HALS) for CPD [89] or multiplicative update rule for non-negative matrix factorization (NMF) [11, 53]. Furthermore, we aim to solve the penalized version of LS-based TD. In this study, we assume *existence of iterative penalized LS-based TD algorithm* that minimizes the squared error with penalty, and we denote more general LS-based TD as

$$\begin{aligned} & \underset{\mathbf{x}}{\text{argmin}} \|\mathbf{v} - \mathbf{x}\|_2^2 + i_{\mathbb{S}}(\mathbf{x}) + \alpha p(\theta) \\ & \approx \text{Proj}_{\mathbb{S}_{\alpha}}(\mathbf{v}, \mathbf{x}_0) := \mathcal{U}_{\mathbf{v}, \mathbb{S}_{\alpha}} \circ \dots \circ \mathcal{U}_{\mathbf{v}, \mathbb{S}_{\alpha}}(\mathbf{x}_0). \end{aligned} \quad (39)$$

It should be noted that the above LS-based TD algorithm is expressed from the perspective of updating $\mathbf{x}(\theta)$, however, the

core tensors $\theta = (\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_L)$ are also actually updated in practical implementation. In other words, in this algorithm, \mathbf{x} and θ are always considered a pair. For simplicity, we will not write it explicitly, but the input to $\mathcal{U}_{\mathbf{v}, \mathbb{S}_\alpha}$ requires not only \mathbf{x}_0 but also θ_0 .

3.2.2 MM for ℓ_2 loss

Here, we consider a case to minimize ℓ_2 loss in Equation 37 as

$$f(\mathbf{x}) = \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 + i_{\mathbb{S}}(\mathbf{x}) + \alpha p(\theta). \quad (40)$$

To minimize $f(\mathbf{x})$, we propose to employ MM approach:

$$\mathbf{x}_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} g(\mathbf{x}|\mathbf{x}_k), \quad (41)$$

where auxiliary function $g(\mathbf{x}|\mathbf{x}_k)$ is given by

$$g(\mathbf{x}|\mathbf{x}_k) := f(\mathbf{x}) + (\mathbf{x} - \mathbf{x}_k)^\top (\lambda \mathbf{I} - \mathbf{A}^\top \mathbf{A})(\mathbf{x} - \mathbf{x}_k) \quad (42)$$

$$= \lambda \left\| \mathbf{x} - \left(\mathbf{x}_k - \frac{1}{\lambda} (\mathbf{A}^\top \mathbf{A} \mathbf{x}_k - \mathbf{A}^\top \mathbf{b}) \right) \right\|_2^2 + i_{\mathbb{S}}(\mathbf{x}) + \alpha p(\theta) + \text{const}. \quad (43)$$

From Equation 42, the auxiliary function satisfies the conditions of Equation 16 when λ is greater than the maximum eigenvalue of $\mathbf{A}^\top \mathbf{A}$. Then, the MM step in Equation 41 can be reduced to

$$\mathbf{v}_k = \mathbf{x}_k - \frac{1}{\lambda} (\mathbf{A}^\top \mathbf{A} \mathbf{x}_k - \mathbf{A}^\top \mathbf{b}); \quad (44)$$

$$\mathbf{x}_{k+1} = \operatorname{Proj}_{\mathbb{S}_\alpha}(\mathbf{v}_k, \mathbf{x}_k); \quad (45)$$

Note that the penalty parameter for LS-based TD becomes $\frac{\alpha}{\lambda}$, and we set λ to be the maximum eigenvalue of $\mathbf{A}^\top \mathbf{A}$ in practice.

3.2.3 ADMM for other loss functions

Next, we consider the problem in Equation 37 for other loss functions and propose to employ ADMM as Zhang and Ding's formulation [85]. First, we introduce a new variable $\mathbf{y} \in \mathbb{R}^J$ and a linear constraint $\mathbf{y} = \mathbf{A}\mathbf{x}$, and its augmented Lagrangian is given by

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{z}) = D(\mathbf{b}, \mathbf{y}) + i_{\mathbb{S}}(\mathbf{x}) + \alpha p(\theta) + \langle \mathbf{z}, \mathbf{y} - \mathbf{A}\mathbf{x} \rangle + \frac{\beta}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2, \quad (46)$$

where $\mathbf{z} \in \mathbb{R}^J$ is a Lagrange multiplier and $\beta > 0$ is a hyperparameter. In each step of ADMM, \mathbf{x} is updated as a minimizer of $\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{z})$ with respect to \mathbf{x} , \mathbf{y} is updated as a minimizer

of $\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{z})$ with respect to \mathbf{y} , and \mathbf{z} is updated by the method of multipliers. ADMM algorithm is given by

$$\mathbf{x}_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} i_{\mathbb{S}}(\mathbf{x}) + \alpha p(\theta) + \frac{\beta}{2} \left\| \mathbf{y}_k + \frac{\mathbf{z}_k}{\beta} - \mathbf{A}\mathbf{x} \right\|_2^2; \quad (47)$$

$$\mathbf{y}_{k+1} = \underset{\mathbf{y}}{\operatorname{argmin}} D(\mathbf{b}, \mathbf{y}) + \frac{\beta}{2} \left\| \mathbf{y} + \frac{\mathbf{z}_k}{\beta} - \mathbf{A}\mathbf{x}_{k+1} \right\|_2^2; \quad (48)$$

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \beta(\mathbf{y}_{k+1} - \mathbf{A}\mathbf{x}_{k+1}); \quad (49)$$

where initializations \mathbf{x}_0 , \mathbf{y}_0 and \mathbf{z}_0 are required.

3.2.3.1 Update rule for \mathbf{x}

Here, we consider practical process in Equation 47. Since $\mathbf{y}_k + \frac{\mathbf{z}_k}{\beta}$ is constant and $i_{\mathbb{S}}(\mathbf{x})$ is invariant to scalar, the structure of Equation 47 is the same as in Equation 40. By using MM approach, update rule in Equation 47 can be replaced with

$$\mathbf{v}_k = \mathbf{x}_k - \frac{1}{\lambda} \left(\mathbf{A}^\top \mathbf{A} \mathbf{x}_k - \mathbf{A}^\top \left(\mathbf{y}_k + \frac{\mathbf{z}_k}{\beta} \right) \right); \quad (50)$$

$$\mathbf{x}_{k+1} = \operatorname{Proj}_{\mathbb{S}_{\frac{2\alpha}{\lambda\beta}}}(\mathbf{v}_k, \mathbf{x}_k); \quad (51)$$

Note that the penalty parameter for LS-based TD becomes $\frac{2\alpha}{\lambda\beta}$.

3.2.3.2 Update rule for \mathbf{y}

Update rule in Equation 48 is derived depending on the loss function $D(\mathbf{b}, \mathbf{y})$. Many loss functions take the form of a sum of entry-wise losses $D(\mathbf{b}, \mathbf{y}) = \sum_{j=1}^J d(b_j, y_j)$, in which case the subproblem for updating \mathbf{y} is separable for each entry y_j as

$$y_j^* = \underset{y_j}{\operatorname{argmin}} d(b_j, y_j) + \frac{\beta}{2} (d_j - y_j)^2, \quad (52)$$

where we put $\mathbf{d} = [d_1, d_2, \dots, d_J]^\top = \mathbf{A}\mathbf{x}_{k+1} - \frac{\mathbf{z}_k}{\beta}$. The solution in Equation 52 is unique if $d(b_j, y_j)$ is convex. Our algorithm is possible to support loss functions that the problems formulated in Equation 52 have closed-form solutions. Combettes and Pesquet [90] and Parikh and Boyd [91] can be useful for obtaining solutions to several distance functions $d(b_j, y_j)$. For example, when we consider ℓ_1 loss $d(b_j, y_j) = |b_j - y_j|$, the solution can be given by soft-thresholding as

$$y_j^* = b_j + \operatorname{soft}_{\frac{1}{\beta}}[d_j - b_j]. \quad (53)$$

When we consider generalized KL divergence for positive $y_j > 0$ as $d(b_j, y_j) = b_j \log \frac{b_j}{y_j} + y_j - b_j$, the solution can be given by

$$y_j^* = \frac{\beta d_j - 1 + \sqrt{(\beta d_j - 1)^2 + 4\beta b_j}}{2\beta}. \quad (54)$$

```

1: input:  $\mathbf{b}$ ,  $\mathbf{A}$ ,  $\mathcal{S}$ , type of loss,  $\alpha$ ,  $\beta > 0$ 
2: Initialize  $\mathbf{x} \in \mathcal{S}$  and  $\mathbf{z} = \mathbf{0}$ ;
3:  $\lambda \leftarrow$  maximum eigenvalue of  $\mathbf{A}^\top \mathbf{A}$ ;
4:  $\gamma \leftarrow \begin{cases} \frac{\alpha}{\lambda} & (\ell_2) \\ \frac{2\alpha}{\beta\lambda} & (\ell_1 \text{ or KL}) \end{cases}$ 
5: repeat
6:    $\mathbf{d} \leftarrow \mathbf{Ax} - \frac{1}{\beta}\mathbf{z}$ ;
7:    $y_j \leftarrow \begin{cases} b_j & (\ell_2) \\ b_j + \text{soft}_{\frac{1}{\beta}}(d_j - b_j) & (\ell_1) \\ \frac{\beta d_j - 1 + \sqrt{(\beta d_j - 1)^2 + 4\beta b_j}}{2\beta} & (\text{KL}) \end{cases}$ 
8:    $\mathbf{z} \leftarrow \begin{cases} \mathbf{0} & (\ell_2) \\ \mathbf{z} + \beta(\mathbf{y} - \mathbf{Ax}) & (\ell_1 \text{ or KL}) \end{cases}$ 
9:    $\mathbf{v} \leftarrow \mathbf{x} - \frac{1}{\lambda}(\mathbf{A}^\top \mathbf{Ax} - \mathbf{A}^\top(\mathbf{y} + \frac{1}{\beta}\mathbf{z}))$ ;
10:   $\mathbf{x} \leftarrow \text{Proj}_{\mathcal{S}_\gamma}(\mathbf{v}, \mathbf{x})$  or  $\mathcal{U}_{\mathbf{v}, \mathcal{S}_\gamma}(\mathbf{x})$ ;
11: until convergence
12: output:  $\mathbf{x}$ 

```

Algorithm 1. ADMM-MM algorithm.

3.3 Proposed algorithm

Finally, the proposed ADMM-MM algorithm that supports three typical loss functions can be summarized in Algorithm 1. Our ADMM-MM allows for the immediate application of sophisticated TD models to other tasks such as completion, robust reconstruction, and compressive sensing. By simply selecting the update rule in the 7th line, we can accommodate the three loss functions. The majorization step on the 9th line is important to allow for accommodating a variety of design matrices \mathbf{A} . Many LS-based TD with penalty can be applied into the ADMM-MM algorithm in a plug-and-play manner at the 10th line. The basic expectation of plug-and-play module, $\text{Proj}_{\mathcal{S}_\gamma}(\mathbf{v}, \mathbf{x})$ or $\mathcal{U}_{\mathbf{v}, \mathcal{S}_\gamma}(\mathbf{x})$, is that it has a monotonically decreasing property with respect to the penalized squared error shown in Equation 39. Note that our algorithm can also involve tensor nuclear norm regularization [65] although not a direct tensor decomposition. That is, a proximal mapping given in closed form of nuclear norm regularization can be replaced directly with $\text{Proj}_{\mathcal{S}_\gamma}(\mathbf{v}, \mathbf{x})$.

The proposed algorithm can be regarded as a unified and extended algorithm of EM-ALS and Zhang and Ding's ADMM. When \mathbf{A} is the diagonal matrix of vectorization of the mask tensor \mathcal{W} (e.g., $\mathbf{A} = \text{diag}(\text{vec}(\mathcal{W}))$) and ℓ_2 loss is assumed, the ADMM-MM algorithm results in EM-ALS. When $\mathbf{A} = \mathbf{I}$ and ℓ_1 loss is assumed, the ADMM-MM algorithm results in Zhang and Ding's ADMM.

From the perspective of an optimization algorithm that uses ADMM with MM, the proposed algorithm can be regarded as a special case of linearized ADMM [92, 93]. However, the linearized ADMM work [92] considers only convex optimizations, while the other work [93] considers only non-convex matrix norms which have effective proximal mappings. The major difference in the proposed method from the above methods is that the update rule of the iterative algorithm is applied to penalized TD instead of proximal mapping. Our proposal to plug-and-play various TDs

is a new and meaningful attempt, which will greatly improve the applicability of TDs.

The computational complexity of a single iteration in the ADMM-MM algorithm is often dominated by the LS-based TD at the 10th line. The complexity of lines 6 through 9 is $\mathcal{O}(\Omega)$, where Ω is the number of nonzero elements in the design matrix $\mathbf{A} \in \mathbb{R}^{I \times I}$. We often assume that the design matrix is sparse and $J \leq \Omega \ll IJ$. On the other hand, the complexity of LS-based TD for N -th order tensor $\mathcal{V} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ depends on the model and algorithm. CPD/NNCPD can be solved by ALS and HALS [2, 8]. The complexity of CP-ALS is $\mathcal{O}(NIR_{\text{CP}}) + \mathcal{O}(NR_{\text{CP}}^3)$, while the complexity of HALS is $\mathcal{O}(NIR_{\text{CP}})$, assuming that the CP rank is denoted as R_{CP} and $I = \prod_{i=1}^N I_i$. ALS is computationally more expensive than HALS because it requires matrix inversion. TKD is usually solved by the higher-order orthogonal iteration (HOOI) algorithm [48], which solves N eigenvalue problems alternately. Since the symmetric matrix of size (I_n, I_n) by tensor-matrix multiplication is usually more expensive than its eigenvalue problem, then the complexity of HOOI is $\mathcal{O}(NIR_{\text{TK}})$, assuming that the Tucker rank is $(R_{\text{TK}}, R_{\text{TK}}, \dots, R_{\text{TK}})$. TTD is usually solved by TT-ALS with QR orthogonalization [50]. The advantage of this algorithm is that QR orthogonalization allows each least-squares solution to be found without matrix inversion. The complexity of TT-ALS is $\mathcal{O}(NIR_{\text{TT}})$, assuming that the TT-rank is $(R_{\text{TT}}, R_{\text{TT}}, \dots, R_{\text{TT}})$. TRD is usually solved by TR-ALS [51]. The complexity of TR-ALS is $\mathcal{O}(NIR_{\text{TR}}^2) + \mathcal{O}(NR_{\text{TR}}^6)$, assuming that the TR-rank is $(R_{\text{TR}}, R_{\text{TR}}, \dots, R_{\text{TR}})$. From a complexity perspective, TKD and TTD are highly efficient. TRD has a higher complexity with respect to TR rank. CPD often requires a larger CP rank, making it more expensive than other models.

3.4 Discussions on the convergence

Unfortunately, this study does not include new theoretical results on the convergence of the ADMM-MM algorithm using LS-based TD algorithms for sub-optimization. Instead, this section introduces existing theoretical results on convergence related to TD, EM/MM, and ADMM, and discusses their relevance to this study.

3.4.1 Case of ℓ_2 loss with MM

The theory of monotonicity and convergence of the EM and MM algorithms has been discussed [83, 94], and global convergence has been shown when the minimizer of the auxiliary function is unique [95]. In this study, ALS and MM are combined for ℓ_2 loss minimization. From a point of view of only MM, the results of MM [95] cannot be applied to our case since our auxiliary function is non-convex and its minimizer is not unique in general. On the other hand, the convergence of tensor completion via non-negative tensor factorization [96] has been studied, which corresponds to a special case of our algorithm.

In Section 3.2.2, we explain the proposed MM algorithm, which first derives the auxiliary function and then minimizes it using ALS (or BCD). However, the same auxiliary function and algorithm can be derived by considering the opposite way which first considers

to minimize the original function using ALS (or BCD) and then derives the auxiliary function for each sub-optimization problem. From this perspective, the proposed algorithm for ℓ_2 loss is included in a framework named block majorization minimization or block successive upper bound minimization [97, 98]. In short, from the results of convergence [97, 98], the proposed algorithm has global convergence if the auxiliary function of each sub-optimization has a unique minimizer. Although the solution to the sub-optimization in ALS is generally not unique, this can sometimes be resolved by adding regularization. For example, in the case of the use of Tikhonov regularized ALS for sub-optimization, global convergence is guaranteed in the proposed MM algorithm.

3.4.2 Case of other loss with ADMM

When $\mathbf{A} = \mathbf{I}$, the proposed algorithm is reduced to the standard ADMM. Usually, the convergence of ADMM is based on the non-expansive property of projection onto a convex set or proximal mapping of a convex function [99], and it can not be applicable to our case since low-rank tensor approximation is characterized as a projection onto a non-convex set, $\text{Proj}_{\mathbb{S}}$, or non-convex sub-optimization, $\text{Proj}_{\mathbb{S}_y}$. As a related work [100], the ADMM algorithm and its convergence for the completion of the matrix with NMF have been discussed, however, in their formulation, each factor matrix of NMF is treated separately as an optimization variable in ADMM, which differs from our formulation. On the other hand, the non-convex ADMM [101] includes many non-convex functions and indicator functions of non-convex sets such as the Stiefel/Grassmannian manifold, and it is close to our problem. According to this theory, the compactness of the set \mathbb{S} , the coercivity and smoothness of the objective function, and the continuity of the sub-optimization paths are required for convergence. It is not trivial whether these conditions apply in our case because $\text{Proj}_{\mathbb{S}_y}$ is a composite problem of tensor decomposition and penalization of the core tensors.

When $\mathbf{A} \neq \mathbf{I}$, the proposed algorithm is characterized as ADMM with MM for its sub-optimization. This combination of ADMM and MM is often referred to as linearized ADMM [92]. Non-convex linearized ADMM [93] and its convergence have been discussed, and it is applied to non-convex low-rank matrix reconstruction problem. However, it is not trivial whether the results of convergence [93] can be applied in our case. From the perspective of PnP-ADMM, the results of the convergence analysis based on the Lipschitz condition of the denoiser have been reported [102]. Thus, although there are various relevant results, whether they apply specifically to our case is an open problem. This open problem may be solved by analyzing the properties of the penalized TD algorithm.

4 Related works

There are several studies on algorithms for TD using ADMM. AO-ADMM [73] is an algorithm for constrained matrix/tensor factorization which solves the sub-problems for updating factor matrices using ADMM. Although alternating optimization (AO) is the main-routine and ADMM is subroutine in AO-ADMM, in

TABLE 1 Comparison of objective functions and computational time [sec] at convergence for PG, BCD, and the proposed algorithms.

		PG		BCD		ADMM-MM (proposed)	
		Obj.	Time	Obj.	Time	Obj.	Time
ℓ_2	Noise	-	-	1,623	410.3	1614	8.924
	Missing	-	-	8,091	1,866	119.0	7.792
	Blur	-	-	4,657	13,684	26.10	298.5
	Down	-	-	80.68	1,165	2.365	11.36
ℓ_1	Noise	42.92	13.02	43.18	41.66	42.87	2.16
	Missing	39.74	48.92	38.65	89.79	35.09	84.9
	Blur	39.14	103.9	39.57	171.7	38.93	133.8
	Down	1.040	108.9	1.759	42.18	1.004	4.85
KL	Noise	5.361	20.50	5.394	146	5.358	4.65
	Missing	0.526	54.95	0.363	220.8	0.247	8.532
	Blur	5.109	158.2	5.155	576.4	4.905	669.3
	Down	0.189	98.59	0.304	32.16	0.135	7.65

The best objective values and computational times are highlighted by bold.

contrast, the proposed ADMM-MM algorithm is ADMM is used as main-routine. AO-ADMM supports several loss functions, but it does not support various design matrices. In addition, AO-PDS [103] has been proposed using primal-dual splitting (PDS) instead of ADMM.

Robust Tucker decomposition (RTKD) with ℓ_1 loss has been proposed [85] and its algorithm has been developed based on ADMM. RTKD employs ADMM as the main routine, ALS is used as the subroutine, and it can be regarded as a special case of our proposed algorithm. RTKD does not support any other loss function, various design matrices, and other constraints.

The penalized TD using ADMM has been actively studied and many algorithms have been proposed [100, 104–107]. Each of these algorithms is a detailed formulation of ADMM tailored to the problem, and the optimization variables are separated into many blocks and updated alternately. The algorithms are not structured, making them difficult to generalize and extend.

In the context of generalized TD, generalized CPD [108] has been proposed. The purpose of the study is to make the CP decomposition compatible with various loss functions. Basically, a BCD-based gradient algorithm has been proposed. However, other TD models and perspectives on design matrices are not discussed.

PnP-ADMM [54] is a framework for using some black-box models (e.g., trained deep denoiser) instead of proximal mapping in ADMM. It is highly extensible, in that any model can be applied to various design matrices. The structure of using LS-based TD in a plug-and-play manner in the proposed algorithm is basically the same as that of PnP-ADMM. If we consider LS-based TD as a denoiser, the proposed algorithm may be considered a type of PnP-ADMM. In this sense, the proposed algorithm and PnP-ADMM are very similar, but they are significantly different in that our objective function is not a black box.

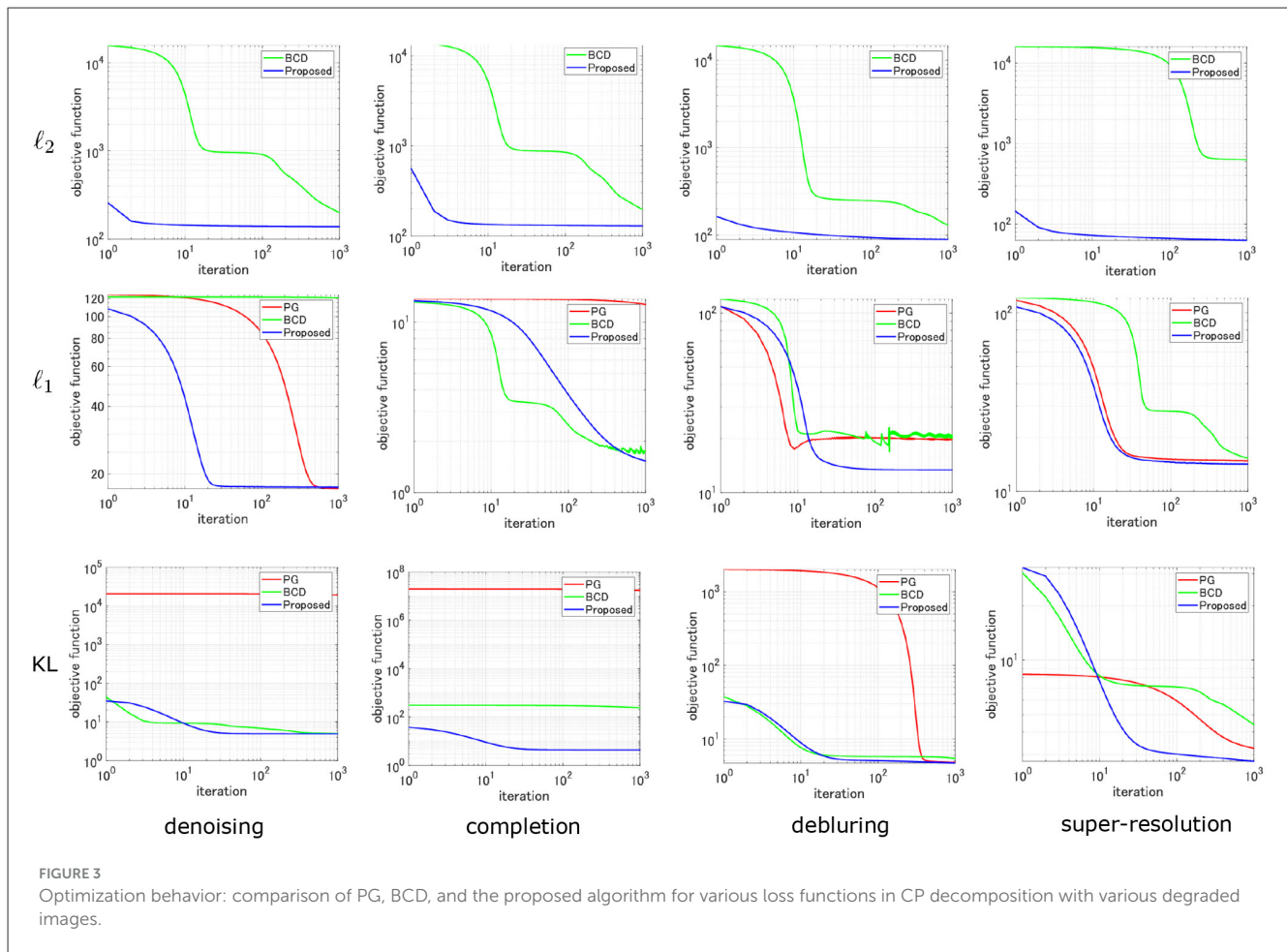


FIGURE 3

Optimization behavior: comparison of PG, BCD, and the proposed algorithm for various loss functions in CP decomposition with various degraded images.

5 Experiments

The purpose of the experiments is to evaluate the performance of the proposed algorithm in terms of optimization and to investigate its usefulness for versatile tensor reconstruction tasks.

5.1 Optimization behaviors for various tensor reconstruction tasks

Tensor reconstruction tasks in this experiment include tensor denoising, completion, de-blurring, and super-resolution. We used an RGB image named “facade” represented as a third-order tensor of size $256 \times 256 \times 3$ for ground truth in denoising, completion, de-blurring, and super-resolution tasks.

For tensor denoising task, we set $\mathbf{A} = \mathbf{I}$. Gaussian noise, salt-and-pepper noise, and Poisson noise are added for individual tasks. For tensor completion tasks, we set $\mathbf{A} = \text{diag}(\text{vec}(\mathbf{W}))$ with a randomly generated mask tensor $\mathbf{W} \in \{0, 1\}^{256 \times 256 \times 3}$, where 90% of the entries are missing. For de-blurring tasks, we used a motion blur window of size 21×21 , and the constructed block Toeplitz matrix is used as \mathbf{A} . For super-resolution tasks, we used a

Lanczos2 kernel for downsampling to 1/4 size, and the constructed downsampling matrix is used as \mathbf{A} .

5.1.1 Comparison with gradient methods

In this experiment, we compare the proposed algorithm with two gradient-based algorithms: projected gradient (PG) and block coordinate descent (BCD) with the low-rank CP decomposition model:

$$\mathbf{x} = \text{vec}([\mathbf{G}^{(1)}, \mathbf{G}^{(2)}, \mathbf{G}^{(3)}]) \in \mathbb{S} \subset \mathbb{R}^{256 \cdot 256 \cdot 3}, \quad (55)$$

$$\mathbb{S} := \{\text{vec}([\mathbf{G}^{(1)}, \mathbf{G}^{(2)}, \mathbf{G}^{(3)}]) \mid \mathbf{G}^{(1)} \in \mathbb{R}^{256 \times R}, \mathbf{G}^{(2)} \in \mathbb{R}^{256 \times R}, \mathbf{G}^{(3)} \in \mathbb{R}^{3 \times R}\}. \quad (56)$$

In PG, \mathbf{x} is moved along the gradient descent direction and then projected onto the set \mathbb{S} .

$$\mathbf{v}_k = \mathbf{x}_k - \mu \nabla_{\mathbf{x}} D(\mathbf{b}, \mathbf{A}\mathbf{x}_k); \quad (57)$$

$$\mathbf{x}_{k+1} = \text{Proj}_{\mathbb{S}}^{\text{ALS}}(\mathbf{v}_k, \mathbf{x}_k); \quad (58)$$

where $\mu > 0$ is a step-size. Note that it is the same as the proposed method when $D(\mathbf{b}, \mathbf{A}\mathbf{x}) = \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2$ and $\mu = \frac{1}{\lambda}$. In BCD based gradient algorithm is given by

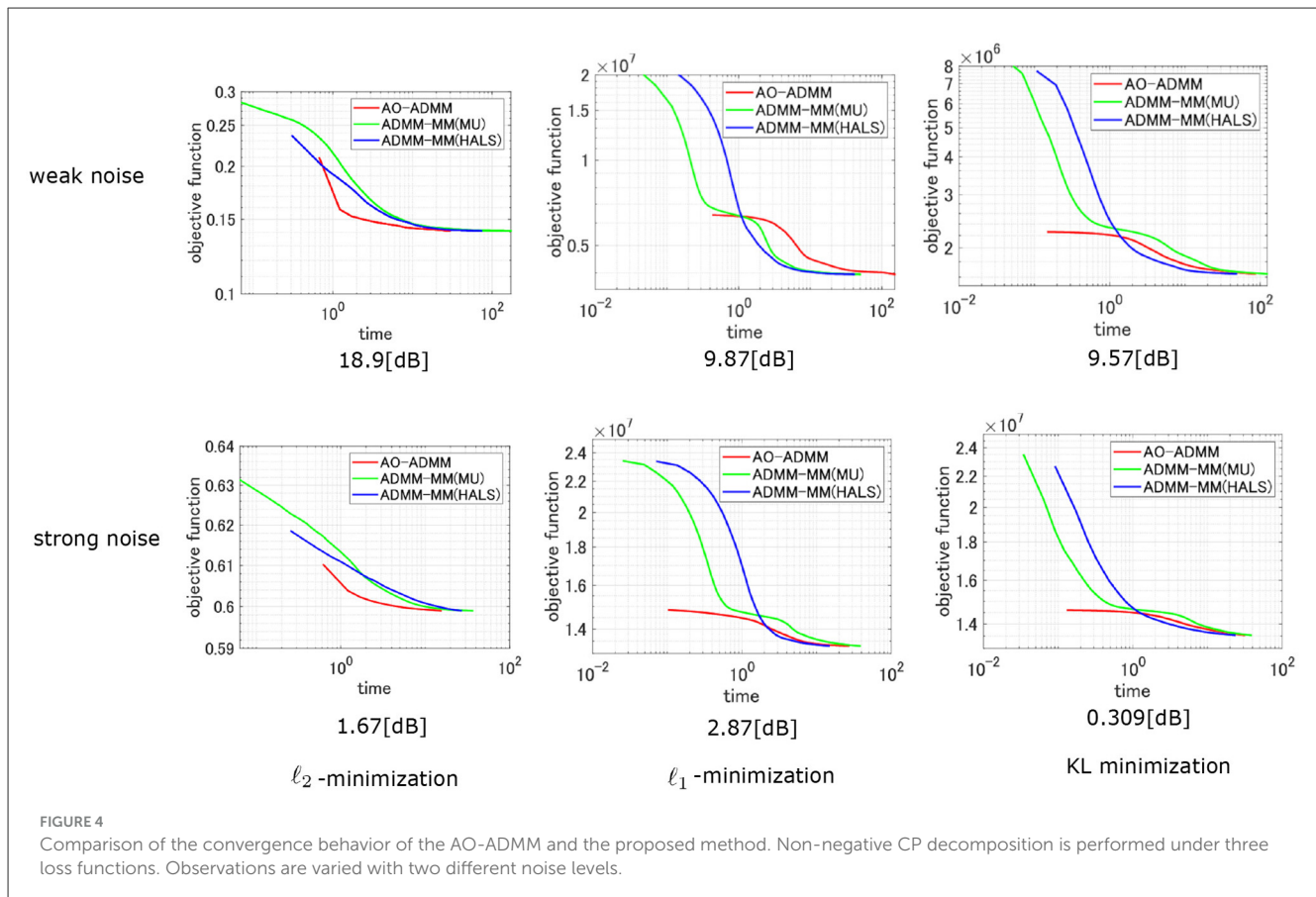


FIGURE 4

Comparison of the convergence behavior of the AO-ADMM and the proposed method. Non-negative CP decomposition is performed under three loss functions. Observations are varied with two different noise levels.

TABLE 2 Comparison of the time required for the objective function to converge in noise removal using AO-ADMM and ADMM-MM.

		AO-ADMM	ADMM-MM (MU)	ADMM-MM (HALS)
ℓ_2 loss	Weak noise	30.15	172.8	74.70
	Strong noise	15.62	36.75	27.40
ℓ_1 loss	Weak noise	153.1	50.20	41.46
	Strong noise	27.31	38.35	14.98
KL divergence	Weak noise	86.36	123.7	48.27
	Strong noise	31.67	38.45	23.76

The best objective values and computational times are highlighted by bold.

$$\mathbf{G}_{k+1}^{(1)} = \mathbf{G}_k^{(1)} - \mu \nabla_{\mathbf{G}^{(1)}} D(\mathbf{b}, \text{Avec}(\llbracket \mathbf{G}_k^{(1)}, \mathbf{G}_k^{(2)}, \mathbf{G}_k^{(3)} \rrbracket)); \quad (59)$$

$$\mathbf{G}_{k+1}^{(2)} = \mathbf{G}_k^{(2)} - \mu \nabla_{\mathbf{G}^{(2)}} D(\mathbf{b}, \text{Avec}(\llbracket \mathbf{G}_{k+1}^{(1)}, \mathbf{G}_k^{(2)}, \mathbf{G}_k^{(3)} \rrbracket)); \quad (60)$$

$$\mathbf{G}_{k+1}^{(3)} = \mathbf{G}_k^{(3)} - \mu \nabla_{\mathbf{G}^{(3)}} D(\mathbf{b}, \text{Avec}(\llbracket \mathbf{G}_{k+1}^{(1)}, \mathbf{G}_{k+1}^{(2)}, \mathbf{G}_k^{(3)} \rrbracket)); \quad (61)$$

In both PG and BCD, step-size μ was manually adjusted for the best performance. The proposed algorithm has an optimization parameter β and was also manually adjusted. We experimented with four tasks of denoising, completion, deblurring and super-resolution under Gaussian, salt-and-pepper, and Poisson noise measurement. ℓ_2 loss, ℓ_1 loss, and generalized KL divergence are used for Gaussian, salt-and-pepper, and Poisson noise measurements, respectively.

Table 1 shows the achieved values of the objective function and its computational time [sec] for the three optimization methods in various settings. The best values are highlighted in bold. Figure 3 shows its optimization behaviors in the various tasks based on three loss functions. The proposed method stably and efficiently reduces the objective function in various settings in comparison to PG and BCD. Note that since the proposed method for ℓ_2 loss and PG are equivalent, they are not compared.

5.1.2 Comparison with AO-ADMM

In this experiment, we compare the proposed method with AO-ADMM in standard non-negative CP decomposition (NNCPD) under three loss functions. The optimization problem is given by

$$\begin{aligned} \text{minimize } D(\mathbf{b}, \text{vec}(\llbracket \mathbf{G}^{(1)}, \mathbf{G}^{(2)}, \mathbf{G}^{(3)} \rrbracket)), \text{ s.t. } \mathbf{G}^{(1)} \in \mathbb{R}_{\geq 0}^{256 \times R}, \\ \mathbf{G}^{(2)} \in \mathbb{R}_{\geq 0}^{256 \times R}, \mathbf{G}^{(3)} \in \mathbb{R}_{\geq 0}^{3 \times R}, \end{aligned} \quad (62)$$

where $\mathbb{R}_{\geq 0}$ is a set of non-negative real numbers. In AO-ADMM, each $\mathbf{G}^{(i)}$ is updated by sub-optimization using ADMM. We used the original implementation of AO-ADMM by Huang¹ with some modifications to adopt it for the TD problem. In the proposed method, we plug-and-played the multiplicative update (MU) [11] and the hierarchical alternating least squares (HALS) [89] for LS-based NNCPD.

¹ <https://www.cise.ufl.edu/~kejun/code.html>

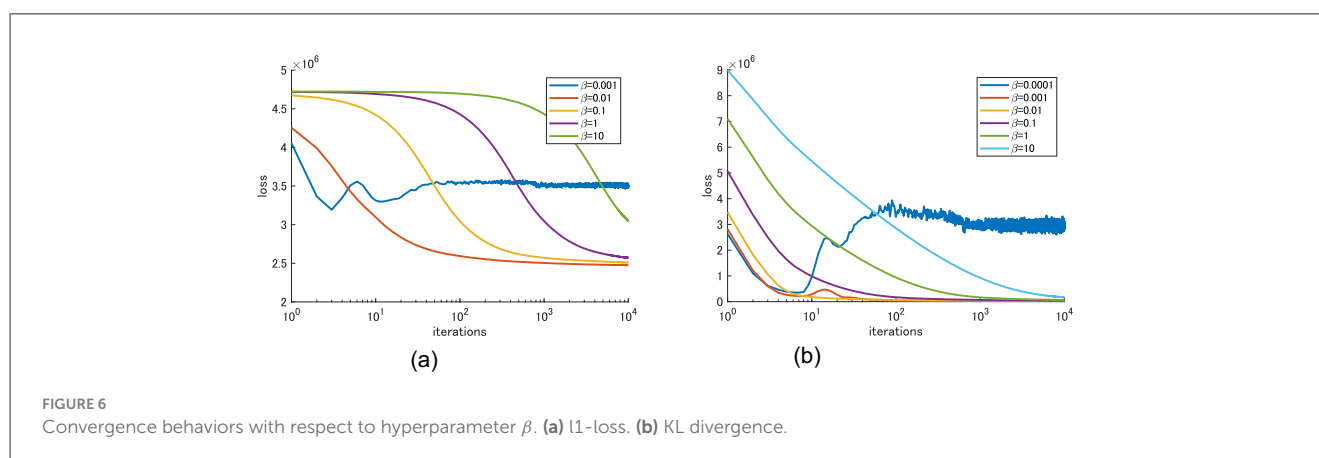
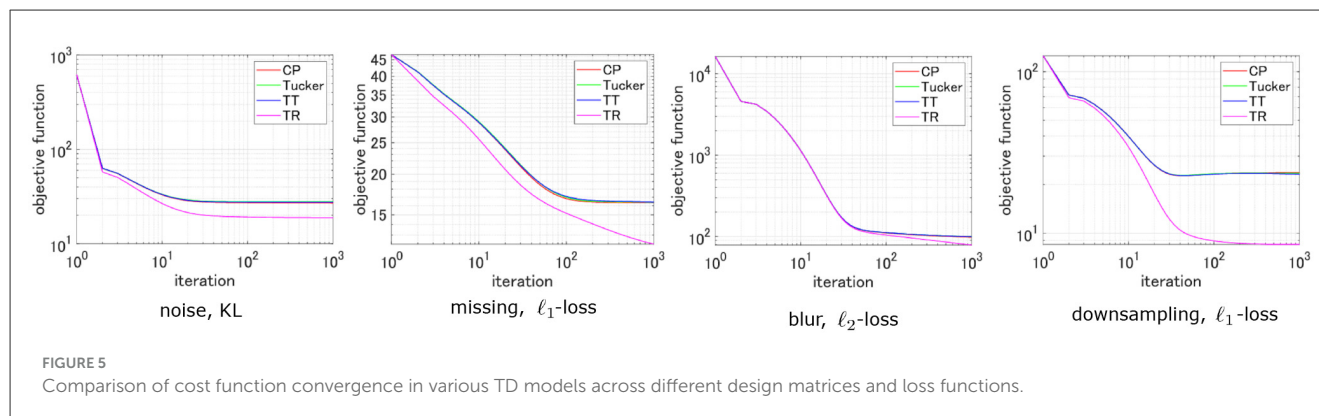


Figure 4 and Table 2 show the comparison of the convergence behaviors between the proposed method and the AO-ADMM. We experimented with weak and strong noise setting under three types of noise. The value displayed below each plot represents the signal-to-noise ratio (SNR) of noisy measurements. Although the MU and HALS used in plug-and-play are not pure least squares minimizers, we can see that they successively converged. In the early stages of optimization, the objective function decreased faster with AO-ADMM, but there were no significant differences in the convergence speed. When comparing the time to convergence, AO-ADMM was slightly faster to minimize ℓ_2 loss, while the proposed method was slightly faster to minimize ℓ_1 loss and generalized KL divergence.

It should be noted that it is difficult to make a fair comparison of the convergence of the AO-ADMM and ADMM-MM algorithms. AO-ADMM has an AO main-iteration and an ADMM sub-iteration for sub-problems, making it a double-loop algorithm. In this study, 10 sub-iterations were performed, but changing this may change the convergence results. Also the convergence behavior changes significantly depending on the parameter β in ADMM. The appropriate beta value differs depending on the task, loss function, and model.

5.1.3 Various tensor decomposition models

Here, we apply the proposed ADMM-MM algorithm to four types of TD models: CP, Tucker, TT, and

TR decompositions. The standard ALS algorithm is used for CP and TR decompositions [42, 43, 51], and orthogonalized ALS is used for Tucker and TT decompositions [48, 50].

Figure 5 shows the selected results, and various tensor reconstruction settings for all TD models can be succinctly optimized by the proposed ADMM-MM algorithm.

5.1.4 Sensitivity of a hyperparameter β

Here, we show the differences in convergence behaviors of ADMM-MM algorithm with respect to the values of hyperparameter β . In this experiment, the tensor completion task was solved using the ALS algorithm of CP decomposition with Tikhonov regularization. β is a hyperparameter related to ADMM, so there are two settings: minimizing ℓ_1 loss and KL divergence. Figure 6 shows the convergence behaviors of the loss function obtained with various values of β . Similar trends were obtained for both loss functions. When β is large, it is stable but convergence is slow. Making β smaller will speed up the convergence, but if it is too small, the convergence becomes unstable.

5.2 Image processing applications

In this experiment, we demonstrate that the proposed algorithm connects various TD models with various image processing tasks.

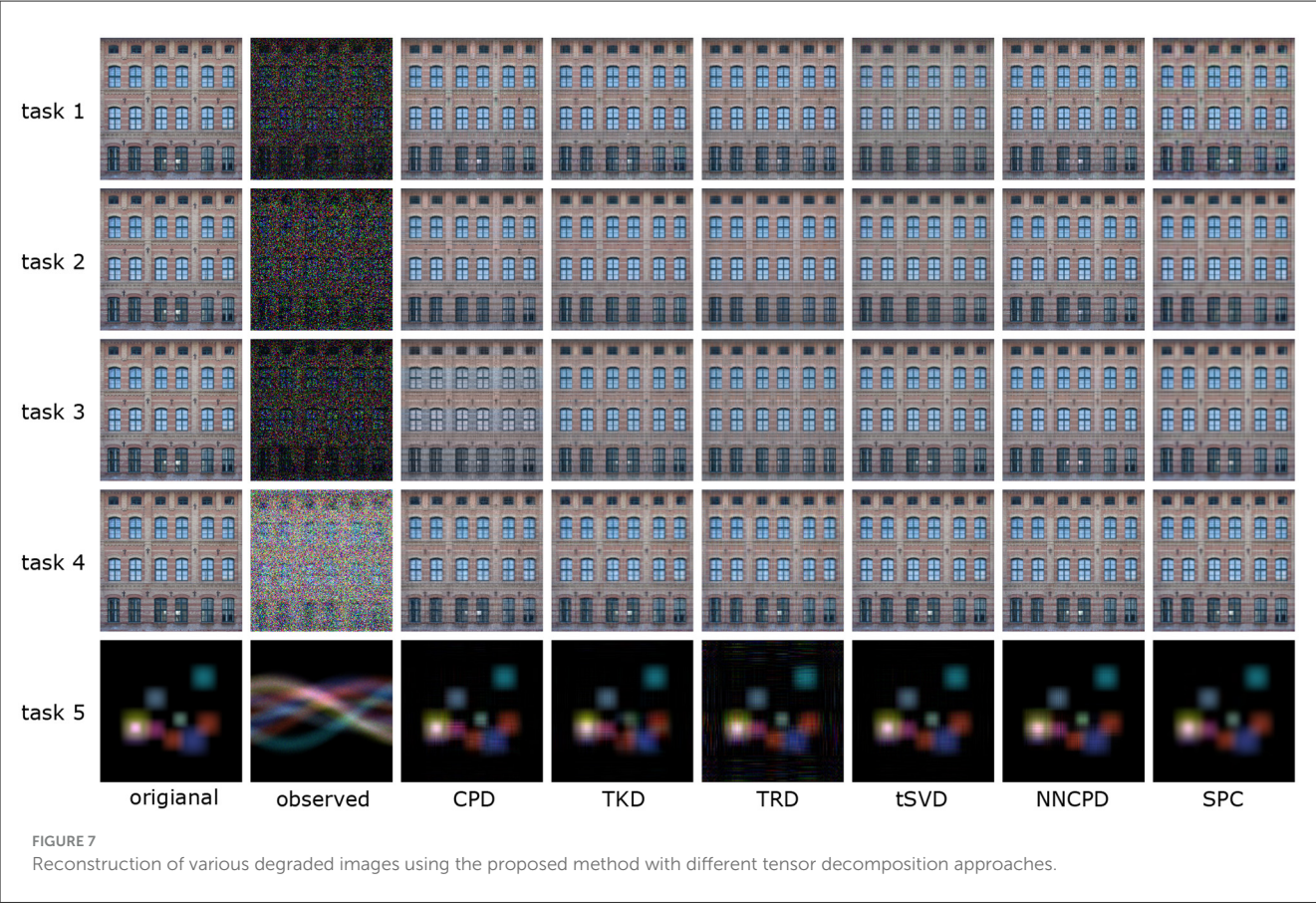


TABLE 3 Comparison of TD models with PSNR, RSE, and SSIM metrics in light field image recovery.

Method	PSNR [dB]	RSE	SSIM
FTTNN [109]	42.05	0.0247	0.9923
TKD (by ADMM-MM)	42.74	0.0229	0.9946
TTD (by ADMM-MM)	39.99	0.0314	0.9925
TRD (by ADMM-MM)	45.66	0.0163	0.9968
CPD (by ADMM-MM)	45.76	0.0161	0.9948
NNCPD (by ADMM-MM)	47.14	0.0138	0.9978
SPC (by ADMM-MM)	48.98	0.0111	0.9973

5.2.1 Color image reconstruction and computed tomography

Here, we show the results of color image reconstruction and computed tomography using the proposed ADMM-MM algorithm. An RGB image named “facade” is used for three image-inpainting tasks under three different noises (tasks 1, 2, and 3) and for an image deblurring task under sparse noise (task 4). An artificial low-rank tensor of size $128 \times 128 \times 3$ is used for computed tomography under Poisson noise (task 5).

We apply various TD models for various image reconstruction tasks to demonstrate the potential of the proposed method. We used six TD models: CP decomposition (CPD), Tucker decomposition (TKD), TR decomposition (TRD) [51], tensor nuclear norm regularization (tSVD) [65], NNCP decomposition (NNCPD) [12], and smooth PARAFAC (SPC) [34]. tSVD stands

for tensor nuclear norm regularization using singular value shresholding. NNCPD stands for CP decomposition with non-negative constraints of factor matrices. Each factor matrices are updated by multiplicative update rules. SPC stands for CP decomposition with smoothness constraints of factor matrices. Although the SPC has been originally proposed for LS-based tensor completion, our framework can extend it to ℓ_1 loss and KL-divergence with arbitrary design matrix \mathbf{A} .

Figure 7 shows the results obtained by six TD models in image processing tasks: tensor completion under Gaussian noise (task 1), tensor completion under sparse noise (task 2), tensor completion under Poisson noise (task 3), de-blurring under sparse noise (task 4), and computed tomography under Poisson noise (task 5). We can see that the proposed method allows to plug-and-play many TD models for application to a variety of image processing tasks.

5.2.2 Application to light field image recovery

Here, we apply the proposed algorithm to the problem of light field image restoration as one of the case studies. The light field image used is a fourth-order tensor named “vinyl” with size $(128,128,3,81)$.² An algorithm for light field image restoration under the hybrid model of Tucker and TT decompositions, named fast tensor train nuclear norm (FTTNN),³ has been studied [109]. The task is to restore the original image from an image in which

2 Data is available online: <https://lightfield-analysis.uni-konstanz.de/>.
3 Code is available online: <https://github.com/ynqiu/fast-TTRPCA>.

20% of the pixels are randomly selected and overwritten with random values.

Our framework allows us to apply different kinds of TD model to this task and compare them with the existing algorithm. Table 3 shows the results of the comparison with PSNR, RSE, and SSIM metrics. We applied TKD, TTD, TRD, CPD, NNCPD, and SPC to this task by using the proposed ADMM-MM algorithm. In particular, we were able to demonstrate a high level of performance with the constrained CPD models (i.e., NNCPD and SPC) in light field image recovery.

6 Conclusion

In this study, we proposed a versatile tensor reconstruction framework to plug-and-play various LS-based TD algorithms and apply them to various applications. This framework is very practical because many TD models are often initially studied on the basis of least squares. The newly proposed TD algorithm can be plug-and-played and operated based on any design matrix and at least three loss functions. In addition, any loss function having a proximal mapping can be easily introduced.

In experiments, we demonstrated the effectiveness of the proposed method compared to existing gradient-based optimization algorithms and AO-ADMM. Although the convergence of the proposed algorithm is theoretically ambiguous, experimentally we confirmed that it has been successfully optimized for various problems, models, and hyperparameter settings. It was also demonstrated that it is useful for a wide range of image processing applications.

In this study, we plug-and-played various TD algorithms and confirmed their effectiveness, but it would be meaningful to investigate a class of tensor decomposition algorithms that can be plug-and-played by theoretical analysis. Furthermore, there are some properties, such as convergence, that remain unclear, so continued investigation is necessary. In addition, it would also be worthwhile to conduct a study on ways to accelerate the convergence of the algorithm.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found here: <https://github.com/yokotatsuya/PnP-Tensor-Decomposition>.

References

1. Yokota T, Caiafa CF, Zhao Q. Tensor methods for low-level vision. In: Liu Y, editor. *Tensors for Data Processing: Theory, Methods, and Applications*. Academic Press Inc Elsevier Science (2021). p. 371–425. doi: 10.1016/B978-0-12-824447-0.00017-0
2. Cichocki A, Zdunek R, Phan AH, Amari SI. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*. New York: John Wiley & Sons, Ltd. (2009). doi: 10.1002/9780470747278
3. Li L, Li Y, Li Z. Efficient missing data imputing for traffic flow by considering temporal and spatial dependence. *Transport Res Part C*. (2013) 34:108–20. doi: 10.1016/j.trc.2013.05.008
4. Chen H, Ahmad F, Vorobyov S, Porikli F. Tensor decompositions in wireless communications and MIMO radar. *IEEE J Sel Top Signal Process*. (2021) 15:438–53. doi: 10.1109/JSTSP.2021.3061937
5. Kalev A, Kosut RL, Deutsch IH. Quantum tomography protocols with positivity are compressed sensing protocols. *NPJ Quant Inf*. (2015) 1:1–6. doi: 10.1038/npjqi.2015.18
6. Kyrillidis A, Kalev A, Park D, Bhojanapalli S, Caramanis C, Sanghavi S. Provable compressed sensing quantum state tomography via non-convex methods. *NPJ Quant Inf*. (2018) 4:36. doi: 10.1038/s41534-018-0080-4

Author contributions

MM: Writing – review & editing, Writing – original draft. HH: Writing – review & editing. TY: Writing – original draft, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research and/or publication of this article. This work was partially supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 23K28109.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that Gen AI was used in the creation of this manuscript. We used generative AI to proofread the English in the manuscript.

Any alternative text (alt text) provided alongside figures in this article has been generated by Frontiers with the support of artificial intelligence and reasonable efforts have been made to ensure accuracy, including review by the authors wherever possible. If you identify any issues, please contact us.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

7. Qin Z, Jameson C, Gong Z, Wakin MB, Zhu Z. Quantum state tomography for matrix product density operators. *IEEE Trans Inf Theory*. (2024) 70:5030–56. doi: 10.1109/TIT.2024.3360951
8. Kolda TG, Bader BW. Tensor decompositions and applications. *SIAM Rev*. (2009) 51:455–500. doi: 10.1137/07070111X
9. Cichocki A, Mandic D, De Lathauwer L, Zhou G, Zhao Q, Caiafa C, et al. Tensor decompositions for signal processing applications: from two-way to multiway component analysis. *IEEE Signal Process Mag*. (2015) 32:145–63. doi: 10.1109/MSP.2013.2297439
10. Yokota T. Very basics of tensors with graphical notations: unfolding, calculations, and decompositions. *arXiv preprint arXiv:241116094* (2024).
11. Lee D, Seung HS. Algorithms for non-negative matrix factorization. In: *Advances in Neural Information Processing Systems* (2000). p. 13.
12. Cichocki A, Zdunek R, Amari Si. Nonnegative matrix and tensor factorization [lecture notes]. *IEEE Signal Process Mag*. (2007) 25:142–5. doi: 10.1109/MSP.2008.4408452
13. Gillis N. *Nonnegative Matrix Factorization*. London: SIAM (2020). doi: 10.1137/1.9781611976410
14. Papalexakis EE, Faloutsos C, Sidiropoulos ND. Parcube: Sparse parallelizable tensor decompositions. In: *Proceedings of European Conference on Machine Learning and Intelligent Systems (SCIS) and International Symposium on Advanced Intelligent Systems (ISIS)*. IEEE (2014). p. 478–483. doi: 10.1109/SCIS-ISIS.2014.7044685
15. Yokota T, Cichocki A. Multilinear tensor rank estimation via sparse Tucker decomposition. In: *Proceedings of International Conference on Soft Computing and Intelligent Systems (SCIS) and International Symposium on Advanced Intelligent Systems (ISIS)*. IEEE (2014). p. 478–483. doi: 10.1109/SCIS-ISIS.2014.7044685
16. Caiafa CF, Cichocki A. Block sparse representations of tensors using Kronecker bases. In: *Proceedings of ICASSP*. IEEE (2012). p. 2709–2712. doi: 10.1109/ICASSP.2012.6288476
17. Caiafa CF, Cichocki A. Computing sparse representations of multidimensional signals using Kronecker bases. *Neural Comput*. (2013) 25:186–220. doi: 10.1162/NECO_a_00385
18. Essid S, Févotte C. Smooth nonnegative matrix factorization for unsupervised audiovisual document structuring. *IEEE Trans Multimed*. (2012) 15:415–25. doi: 10.1109/TMM.2012.2228474
19. Yokota T, Zdunek R, Cichocki A, Yamashita Y. Smooth nonnegative matrix and tensor factorizations for robust multi-way data analysis. *Signal Proc*. (2015) 113:234–49. doi: 10.1016/j.sigpro.2015.02.003
20. Yokota T, Kawai K, Sakata M, Kimura Y, Hontani H. Dynamic PET image reconstruction using nonnegative matrix factorization incorporated with deep image prior. In: *Proceedings of ICCV* (2019). p. 3126–3135. doi: 10.1109/ICCV.2019.00322
21. Takayama H, Yokota T. A new model for tensor completion: smooth convolutional tensor factorization. *IEEE Access*. (2023) 11:67526–39. doi: 10.1109/ACCESS.2023.3291744
22. Cai D, He X, Han J, Huang TS. Graph regularized nonnegative matrix factorization for data representation. *IEEE Trans Pattern Anal Mach Intell*. (2010) 33:1548–60. doi: 10.1109/TPAMI.2010.231
23. Yin M, Gao J, Lin Z. Laplacian regularized low-rank representation and its applications. *IEEE Trans Pattern Anal Mach Intell*. (2015) 38:504–17. doi: 10.1109/TPAMI.2015.2462360
24. Li X, Ng MK, Cong G, Ye Y, Wu Q. MR-NTD manifold regularization nonnegative Tucker decomposition for tensor data dimension reduction and representation. *IEEE Trans Neural Netw Learn Syst*. (2016) 28:1787–800. doi: 10.1109/TNNLS.2016.2545400
25. Cai JF, Candes EJ, Shen Z. A singular value thresholding algorithm for matrix completion. *SIAM J Optimiz*. (2010) 20:1956–82. doi: 10.1137/080738970
26. Acar E, Dunlavy DM, Kolda TG, Mørup M. Scalable tensor factorizations for incomplete data. *Chemometr Intell Lab Syst*. (2011) 106:41–56. doi: 10.1016/j.chemolab.2010.08.004
27. Liu J, Musialski P, Wonka P, Ye J. Tensor completion for estimating missing values in visual data. *IEEE Trans Pattern Anal Mach Intell*. (2013) 35:208–20. doi: 10.1109/TPAMI.2012.39
28. Kressner D, Steinlechner M, Vandereycken B. Low-rank tensor completion by Riemannian optimization. *BIT Numer Mathem*. (2014) 54:447–68. doi: 10.1007/s10543-013-0455-z
29. Xu Y, Hao R, Yin W, Su Z. Parallel matrix factorization for low-rank tensor completion. *Inverse Problems Imag*. (2015) 9:601–24. doi: 10.3934/ipi.2015.9.601
30. Kim YD, Choi S. Weighted nonnegative matrix factorization. In: *Proceedings of ICASSP*. IEEE (2009). p. 1541–1544. doi: 10.1109/ICASSP.2009.4959890
31. Nielsen SFV, Mørup M. Non-negative tensor factorization with missing data for the modeling of gene expressions in the human brain. In: *Proceedings of IEEE International Workshop on Machine Learning for Signal Processing*. IEEE (2014). p. 1–6. doi: 10.1109/MLSP.2014.6958919
32. Mørup M, Hansen LK, Arnfred SM. Algorithms for sparse nonnegative Tucker decompositions. *Neural Comput*. (2008) 20:2112–2131. doi: 10.1162/neco.2008.11-06-407
33. Zhou G, Cichocki A, Zhao Q, Xie S. Efficient nonnegative Tucker decompositions: Algorithms and uniqueness. *IEEE Trans Image Proc*. (2015) 24:4990–5003. doi: 10.1109/TIP.2015.2478396
34. Yokota T, Zhao Q, Cichocki A. Smooth PARAFAC decomposition for tensor completion. *IEEE Trans Signal Proc*. (2016) 64:5423–36. doi: 10.1109/TSP.2016.2586759
35. Ghalamkari K, Sugiyama M. Fast rank-1 NMF for missing data with KL divergence. In: *Proceedings of International Conference on Artificial Intelligence and Statistics*. PMLR (2022). p. 2927–2940.
36. Durand A, Roueff F, Jicquel JM, Paul N. New penalized criteria for smooth non-negative tensor factorization with missing entries. *IEEE Trans Signal Proc*. (2024). doi: 10.1109/TSP.2024.3392357
37. Lyu C, Lu QL, Wu X, Antoniou C. Tucker factorization-based tensor completion for robust traffic data imputation. *Trans Res Part C*. (2024) 160:104502. doi: 10.1016/j.trc.2024.104502
38. Boyd S, Parikh N, Chu E, Peleato B, Eckstein J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found Trends Mach Learn*. (2011) 3:1–122. doi: 10.1561/22000000016
39. Hunter DR, Lange K. A tutorial on MM algorithms. *Am Stat*. (2004) 58:30–7. doi: 10.1198/0003130042836
40. Sun Y, Babu P, Palomar DP. Majorization-minimization algorithms in signal processing, communications, and machine learning. *IEEE Trans Signal Proc*. (2016) 65:794–816. doi: 10.1109/TSP.2016.2601299
41. Hitchcock FL. The expression of a tensor or a polyadic as a sum of products. *J Mathem Phys*. (1927) 6:164–89. doi: 10.1002/sapm192761164
42. Carroll JD, Chang JJ. Analysis of individual differences in multidimensional scaling via an N-way generalization of “Eckart-Young” decomposition. *Psychometrika*. (1970) 35:283–319. doi: 10.1007/BF02310791
43. Harshman RA. Foundations of the PARAFAC procedure: models and conditions for an “explanatory” multimodal factor analysis. *UCLA Work Paper Phonet*. (1970) 16:1–84.
44. Tucker LR. Implications of factor analysis of three-way matrices for measurement of change. *Probl Measur Change*. (1963) 12:122–137.
45. Tucker LR. The extension of factor analysis to three-dimensional matrices. *Contr Mathem Psychol*. (1964) 51:109–127.
46. Tucker LR. Some mathematical notes on three-mode factor analysis. *Psychometrika*. (1966) 31:279–311. doi: 10.1007/BF02289464
47. Kroonenberg PM, De Leeuw J. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*. (1980) 45:69–97. doi: 10.1007/BF02293599
48. De Lathauwer L, De Moor B, Vandewalle J. On the best rank-1 and rank-(r_1, r_2, r_3) approximation of higher-order tensors. *SIAM J Matrix Anal Appl*. (2000) 21:1324–42. doi: 10.1137/S0895479898346995
49. Oseledets IV. Tensor-train decomposition. *SIAM J Sci Comput*. (2011) 33:2295–317. doi: 10.1137/090752286
50. Holtz S, Rohwedder T, Schneider R. The alternating linear scheme for tensor optimization in the tensor train format. *SIAM J Sci Comput*. (2012) 34:A683–713. doi: 10.1137/100818893
51. Zhao Q, Zhou G, Xie S, Zhang L, Cichocki A. Tensor ring decomposition. *arXiv preprint arXiv:1606.5535* (2016).
52. Kim YD, Choi S. Nonnegative Tucker decomposition. In: *Proceedings of CVPR*. IEEE (2007). p. 1–8. doi: 10.1109/CVPR.2007.383405
53. Gillis N, Glineur F. Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization. *Neural Comput*. (2012) 24:1085–105. doi: 10.1162/NECO_a_00256
54. Venkatakrisnan SV, Bouman CA, Wohlberg B. Plug-and-play priors for model based reconstruction. In: *Proceedings of GlobalSIP*. IEEE (2013). p. 945–948. doi: 10.1109/GlobalSIP.2013.6737048
55. Kiers HA. Towards a standardized notation and terminology in multiway analysis. *J Chemometrics*. (2000) 14:105–22. doi: 10.1002/1099-128X(200005/06)14:3<105::AID-CEM582>3.0.CO;2-I
56. De Lathauwer L. Decompositions of a higher-order tensor in block terms—Part I: Lemmas for partitioned matrices. *SIAM J Matrix Anal Appl*. (2008) 30:1022–32. doi: 10.1137/060661685
57. De Lathauwer L. Decompositions of a higher-order tensor in block terms—Part II: Definitions and uniqueness. *SIAM J Matrix Anal Appl*. (2008) 30:1033–66. doi: 10.1137/070690729
58. De Lathauwer L, Nion D. Decompositions of a higher-order tensor in block terms—Part III: Alternating least squares algorithms. *SIAM J Matrix Anal Appl*. (2008) 30:1067–83. doi: 10.1137/070690730

59. Van Mechelen I, Smilde AK. A generic linked-mode decomposition model for data fusion. *Chemometr Intell Lab Syst.* (2010) 104:83–94. doi: 10.1016/j.chemolab.2010.04.012
60. Yokoya N, Yairi T, Iwasaki A. Coupled nonnegative matrix factorization unmixing for hyperspectral and multispectral data fusion. *IEEE Trans Geosci Rem Sens.* (2011) 50:528–37. doi: 10.1109/TGRS.2011.2161320
61. Lahat D, Adali T, Jutten C. Multimodal data fusion: an overview of methods, challenges, and prospects. *Proc IEEE.* (2015) 103:1449–77. doi: 10.1109/JPROC.2015.2460697
62. Grasedyck L. Hierarchical singular value decomposition of tensors. *SIAM J Matrix Anal Appl.* (2010) 31:2029–54. doi: 10.1137/090764189
63. Wu ZC, Huang TZ, Deng LJ, Dou HX, Meng D. Tensor wheel decomposition and its tensor completion application. In: *Advances in Neural Information Processing Systems* (2022). p. 27008–27020.
64. Zheng YB, Huang TZ, Zhao XL, Zhao Q, Jiang TX. Fully-connected tensor network decomposition and its application to higher-order tensor completion. In: *Proceedings of the AAAI Conference on Artificial Intelligence* (2021). p. 11071–11078. doi: 10.1609/aaai.v35i12.17321
65. Zhang Z, Ely G, Aeron S, Hao N, Kilmer M. Novel methods for multilinear data completion and de-noising based on tensor-SVD. In: *Proceedings of CVPR* (2014). p. 3842–3849. doi: 10.1109/CVPR.2014.485
66. Yokota T, Erem B, Guler S, Warfield SK, Hontani H. Missing slice recovery for tensors using a low-rank model in embedded space. In: *Proceedings of CVPR* (2018). p. 8251–8259. doi: 10.1109/CVPR.2018.00861
67. Yamamoto R, Hontani H, Imakura A, Yokota T. Fast algorithm for low-rank tensor completion in delay-embedded space. In: *Proceedings of CVPR* (2022). p. 2048–2056. doi: 10.1109/CVPR52688.2022.00210
68. Sedighin F, Cichocki A, Yokota T, Shi Q. Matrix and tensor completion in multiway delay embedded space using tensor train, with application to signal reconstruction. *IEEE Signal Process Lett.* (2020) 27:810–4. doi: 10.1109/LSP.2020.2990313
69. Sedighin F, Cichocki A. Image completion in embedded space using multistage tensor ring decomposition. *Front Artif Intell.* (2021) 4:687176. doi: 10.3389/frai.2021.687176
70. Candes EJ, Recht B. Exact matrix completion via convex optimization. *Found Comput Mathem.* (2009) 9:717–72. doi: 10.1007/s10208-009-9045-5
71. Gillis N, Glineur F. Low-rank matrix approximation with weights or missing data is NP-hard. *SIAM J Matrix Anal Appl.* (2011) 32:1149–65. doi: 10.1137/110820361
72. Hamon R, Emiya V, Févotte C. Convex nonnegative matrix factorization with missing data. In: *Proceedings of IEEE International Workshop on Machine Learning for Signal Processing.* IEEE (2016). p. 1–6. doi: 10.1109/MLSP.2016.7738910
73. Huang K, Sidiropoulos ND, Liavas AP. A flexible and efficient algorithmic framework for constrained matrix and tensor factorization. *IEEE Trans Signal Proc.* (2016) 64:5052–65. doi: 10.1109/TSP.2016.2576427
74. Gandy S, Recht B, Yamada I. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Probl.* (2011) 27:25010. doi: 10.1088/0266-5611/27/2/025010
75. Chen YL, Hsu CT, Liao HYM. Simultaneous tensor decomposition and completion using factor priors. *IEEE Trans Pattern Anal Mach Intell.* (2013) 36:577–91. doi: 10.1109/TPAMI.2013.164
76. Bengua JA, Phien HN, Tuan HD, Do MN. Efficient tensor completion for color image and video recovery: low-rank tensor train. *IEEE Trans Image Proc.* (2017) 26:2466–79. doi: 10.1109/TIP.2017.2672439
77. Yu J, Zhou G, Zhao Q, Xie K. An effective tensor completion method based on multi-linear tensor ring decomposition. In: *Proceedings of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC).* IEEE (2018). p. 1344–1349. doi: 10.23919/APSIPA.2018.8659492
78. Srebro N, Jaakkola T. Weighted low-rank approximations. In: *Proceedings of ICML* (2003). p. 720–727.
79. Tomasi G, Bro R. PARAFAC and missing values. *Chemometr Intell Lab Syst.* (2005) 75:163–80. doi: 10.1016/j.chemolab.2004.07.003
80. Filipovic M, Jukic A. Tucker factorization with missing data with application to low-n-rank tensor completion. *Multidimens Syst Signal Process.* (2015) 26:677–92. doi: 10.1007/s11045-013-0269-9
81. Kasai H, Mishra B. Low-rank tensor completion: a Riemannian manifold preconditioning approach. In: *Proceedings of ICML* (2016). p. 1012–1021.
82. Dempster AP, Laird NM, Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. *J R Stat Soc.* (1977) 39:1–22. doi: 10.1111/j.2517-6161.1977.tb01600.x
83. Lange K, Hunter DR, Yang I. Optimization transfer using surrogate objective functions. *J Comput Graph Stat.* (2000) 9:1–20. doi: 10.1080/10618600.2000.10474858
84. Zhao Q, Zhou G, Zhang L, Cichocki A, Amari SI. Bayesian robust tensor factorization for incomplete multiway data. *IEEE Trans Neural Netw Learn Syst.* (2015) 27:736–48. doi: 10.1109/TNNLS.2015.2423694
85. Zhang M, Ding C. Robust Tucker tensor decomposition for effective image representation. In: *Proceedings of ICCV* (2013). p. 2448–2455. doi: 10.1109/ICCV.2013.304
86. Huang H, Liu Y, Long Z, Zhu C. Robust low-rank tensor ring completion. *IEEE Trans Comput Imag.* (2020) 6:1117–26. doi: 10.1109/TCL.2020.3006718
87. Lu C, Feng J, Chen Y, Liu W, Lin Z, Yan S. Tensor robust principal component analysis with a new tensor nuclear norm. *IEEE Trans Pattern Anal Mach Intell.* (2019) 42:925–38. doi: 10.1109/TPAMI.2019.2891760
88. Uschmajew A. Local convergence of the alternating least squares algorithm for canonical tensor approximation. *SIAM J Matrix Anal Appl.* (2012) 33:639–52. doi: 10.1137/110843587
89. Cichocki A, Zdunek R, Amari SI. Hierarchical ALS algorithms for nonnegative matrix and 3D tensor factorization. In: *Proceedings of International Conference on Independent Component Analysis and Signal Separation.* Springer (2007). p. 169–176. doi: 10.1007/978-3-540-74494-8_22
90. Combettes PL, Pesquet JC. Proximal splitting methods in signal processing. In: *Fixed-point Algorithms for Inverse Problems in Science and Engineering.* (2011). p. 185–212. doi: 10.1007/978-1-4419-9569-8_10
91. Parikh N, Boyd S. Proximal algorithms. *Found Trends Optim.* (2014) 1:127–239. doi: 10.1561/2400000003
92. Lin Z, Liu R, Su Z. Linearized alternating direction method with adaptive penalty for low-rank representation. In: *Advances in Neural Information Processing Systems* (2011).
93. Hien LTK, Phan DN, Gillis N. Inertial alternating direction method of multipliers for non-convex non-smooth optimization. *Comput Optim Appl.* (2022) 83:247–85. doi: 10.1007/s10589-022-00394-8
94. Wu CJ. On the convergence properties of the EM algorithm. *Ann Stat.* (1983) 2:95–103.
95. Vaida F. Parameter convergence for EM and MM algorithms. *Statistica Sinica.* (2005) 15:831–840. Available online at: <https://www3.stat.sinica.edu.tw/statistica/J15N3/j15n316/j15n316.html>
96. Xu Y, Yin W. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM J Imaging Sci.* (2013) 6:1758–89. doi: 10.1137/120887795
97. Razaviyayn M, Hong M, Luo ZQ. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM J Optim.* (2013) 23:1126–53. doi: 10.1137/120891009
98. Hong M, Razaviyayn M, Luo ZQ, Pang JS. A unified algorithmic framework for block-structured optimization involving big data: with applications in machine learning and signal processing. *IEEE Signal Process Mag.* (2015) 33:57–77. doi: 10.1109/MSP.2015.2481563
99. Eckstein J, Bertsekas DP. On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathem Progr.* (1992) 55:293–318. doi: 10.1007/BF01581204
100. Xu Y, Yin W, Wen Z, Zhang Y. An alternating direction algorithm for matrix completion with nonnegative factors. *Front Mathem China.* (2012) 7:365–84. doi: 10.1007/s11464-012-0194-5
101. Wang Y, Yin W, Zeng J. Global convergence of ADMM in nonconvex nonsmooth optimization. *J Sci Comput.* (2019) 78:29–63. doi: 10.1007/s10915-018-0757-z
102. Ryu E, Liu J, Wang S, Chen X, Wang Z, Yin W. Plug-and-play methods provably converge with properly trained denoisers. In: *Proceedings of ICML.* PMLR (2019). p. 5546–5557.
103. Ono S, Kasai T. Efficient constrained tensor factorization by alternating optimization with primal-dual splitting. In: *Proceedings of ICASSP.* IEEE (2018). p. 3379–3383. doi: 10.1109/ICASSP.2018.8461790
104. Sun DL, Fevotte C. Alternating direction method of multipliers for non-negative matrix factorization with the beta-divergence. In: *Proceedings of ICASSP.* IEEE (2014). p. 6201–6205. doi: 10.1109/ICASSP.2014.6854796
105. Hajinezhad D, Chang TH, Wang X, Shi Q, Hong M. Nonnegative matrix factorization using ADMM: Algorithm and convergence analysis. In: *Proceedings of ICASSP.* IEEE (2016). p. 4742–4746. doi: 10.1109/ICASSP.2016.7472577
106. Xue J, Zhao Y, Liao W, Chan JCW, Kong SG. Enhanced sparsity prior model for low-rank tensor completion. *IEEE Trans Neural Netw Learn Syst.* (2019) 31:4567–81. doi: 10.1109/TNNLS.2019.2956153
107. Xue J, Zhao Y, Huang S, Liao W, Chan JCW, Kong SG. Multilayer sparsity-based tensor decomposition for low-rank tensor completion. *IEEE Trans Neural Netw Learn Syst.* (2021) 33:6916–30. doi: 10.1109/TNNLS.2021.3083931
108. Hong D, Kolda TG, Diersch JA. Generalized canonical polyadic tensor decomposition. *SIAM Rev.* (2020) 62:133–63. doi: 10.1137/18M1203626
109. Qiu Y, Zhou G, Huang Z, Zhao Q, Xie S. Efficient tensor robust PCA under hybrid model of Tucker and tensor train. *IEEE Signal Process Lett.* (2022) 29:627–31. doi: 10.1109/LSP.2022.3143721