SVDinsTN: A Tensor Network Paradigm for Efficient Structure Search from Regularized Modeling Perspective

Yu-Bang Zheng[1]   Xi-Le Zhao[2,*]   Junhua Zeng[3,4]   Chao Li[4]

Qibin Zhao[4]   Heng-Chao Li[1]   Ting-Zhu Huang[2]

[1]School of Information Science and Technology, Southwest Jiaotong University, China

[2]School of Mathematical Sciences, University of Electronic Science and Technology of China, China

[3]School of Automation, Guangdong University of Technology, China

[4]Tensor Learning Team, RIKEN Center for Advanced Intelligence Project (AIP), Japan

zhengyubang@163.com, xlzhao122003@163.com, jh.zenggdut@gmail.com, chao.li@riken.jp

qibin.zhao@riken.jp, hcli@home.swjtu.edu.cn, tingzhuhuang@126.com

Abstract

Tensor network (TN) representation is a powerful technique for computer vision and machine learning. TN structure search (TN-SS) aims to search for a customized structure to achieve a compact representation, which is a challenging NP-hard problem. Recent "sampling-evaluation"-based methods require sampling an extensive collection of structures and evaluating them one by one, resulting in prohibitively high computational costs. To address this issue, we propose a novel TN paradigm, named SVD-inspired TN decomposition (SVDinsTN), which allows us to efficiently solve the TN-SS problem from a regularized modeling perspective, eliminating the repeated structure evaluations. To be specific, by inserting a diagonal factor for each edge of the fully-connected TN, SVDinsTN allows us to calculate TN cores and diagonal factors simultaneously, with the factor sparsity revealing a compact TN structure. In theory, we prove a convergence guarantee for the proposed method. Experimental results demonstrate that the proposed method achieves approximately $100 \sim 1000$ times acceleration compared to the state-of-the-art TN-SS methods while maintaining a comparable level of representation ability.

1. Introduction

Tensor network (TN) representation, which aims to express higher-order data with small-sized tensors (called TN cores) by a specific operation among them, has gained significant attention in various areas of data analysis [1, 11, 25, 37], machine learning [5, 9, 27], computer vision [21, 30, 35, 36, 39], etc. By regarding TN cores as nodes and oper-

ations as edges, a TN corresponds to a graph (called TN topology). The vector composed of the weights of all edges in the topology is defined as the TN rank. *TN structure (including topology and rank) search (TN-SS) aims to search for a suitable TN structure to achieve a compact representation for a given tensor, which is known as a challenging NP-hard problem [13, 18]. The selection of TN structure dramatically impacts the performance of TN representation in practical applications [12, 15, 17, 18].

Recently, there have been several notable efforts to tackle the TN-SS problem [12, 15, 17, 18]. But most of them adopt the "sampling-evaluation" framework, which requires sampling a large number of structures as candidates and conducting numerous repeated structure evaluations. For instance, for a tensor of size $40 \times 60 \times 3 \times 9 \times 9$ (used in Section 4.2), TNGA in [15] requires *thousands* of evaluations and TNALE in [18] requires *hundreds* of evaluations, where each evaluation entails solving an optimization problem to compute TN cores iteratively. Consequently, the computational cost becomes exceedingly high. A meaningful question is *whether we can optimize the TN structure simultaneously during the computation of TN cores, thus escaping the "sampling-evaluation" framework and fundamentally addressing the computationally consuming issue.*

In this paper, we introduce for the first time a regularized modeling perspective on solving the TN-SS problem. This perspective enables us to optimize the TN structure simultaneously during the computation of TN cores, effectively eliminating the need for repetitive structure evaluations. To be specific, we propose a novel TN paradigm, termed as *SVD-inspired TN decomposition (SVDinsTN)*, by inserting diagonal factors between any two TN cores in the "fully-connected" topology (see Figure 1(b)). *The intuition behind SVDinsTN is to leverage the sparsity of the inserted diago-*
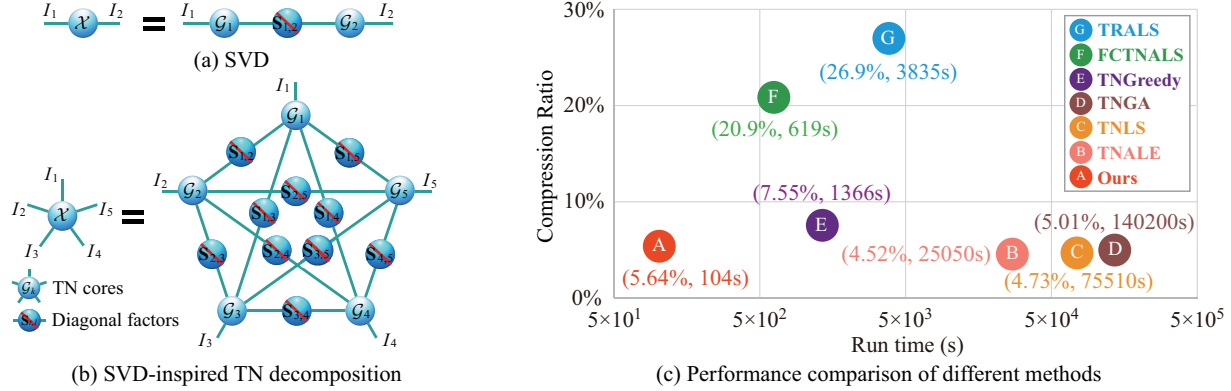
---

*Corresponding author.

Figure 1. (a) A graphical illustration of SVD. (b) A graphical illustration of SVD-inspired TN decomposition on a fifth-order tensor. (c) Comparison of the compression ratio (↓) and run time (↓) of different methods on a fifth-order light field image *Knights*, where the reconstruction error bound is set to 0.05, TRALS [38] and FCTNALS [41] are methods with pre-defined topologies, and TNGreedy [12], TNGA [15], TNLS [17], and TNALE [18] are TN-SS methods (please see more results in Table 2).

nal factors to reveal a compact TN structure and utilize the TN cores (merged with the diagonal factors) to represent a given tensor. Based on SVDinsTN, we establish a regularized model, which updates the TN cores and diagonal factors iteratively and imposes a sparse operator to induce the sparsity of the diagonal factors. In theory, we prove a convergence guarantee for the proposed method and establish an upper bound for the TN rank. In particular, we design a novel initialization scheme for the proposed method based on the upper bound. This initialization scheme enables the proposed method to overcome the high computational cost in the first several iterations, which is caused by the utilization of a "fully-connected" topology as the starting point. As a result, SVDinsTN is capable of capturing a customized TN structure and providing a compact representation for a given tensor in an efficient manner. In summary, we make the following three contributions.

- We propose SVDinsTN, a new TN paradigm, that enables us to optimize the TN structure during the computation of TN cores, greatly reducing the computational cost.
- In theory, we prove a convergence guarantee for the proposed method and establish an upper bound for the TN rank involved in SVDinsTN. The upper bound serves as a guide for designing an efficient initialization scheme.
- Experimental results verify numerically that the proposed method achieves $100 \sim 1000$ times acceleration compared to the state-of-the-art TN-SS methods with a comparable representation ability (see Figure 1(c)).

1.1. Related Works

TN representation[1] aims to find a set of small-sized TN cores to express a large-sized tensor under a given TN struc-

ture (including topology and rank) [4, 5, 31]. In the past decades, many works focused on TN representation with a fixed TN topology, such as tensor train (TT) decomposition with a "chain" topology [24], tensor ring (TR) decomposition with a "ring" topology [38], fully-connected tensor network (FCTN) decomposition with a "fully-connected" topology [41], etc. In addition, these works also presented various methods to optimize the TN cores, such as alternating least square (ALS) [38], gradient descent (GD) [32, 34], proximal alternating minimization (PAM) [41, 42], etc. In contrast, SVDinsTN can reveal a compact TN structure for a given tensor, surpassing methods with pre-defined topologies in terms of representation ability.

TN structure search (TN-SS) aims to search for a suitable or optimal TN structure, including both topology and rank, to achieve a compact representation for a given tensor [8, 12, 15, 17–20, 22, 26]. However, the majority of existing TN-SS methods follow the "sampling-evaluation" framework, which necessitates the use of heuristic search algorithms like the greedy algorithm [12], genetic algorithm [15], and alternating local enumeration algorithm [18] to sample candidate structures and subsequently evaluate them individually. Therefore, these methods inevitably suffer from prohibitively high computational costs due to the numerous repeated evaluations, each involving the iterative calculation of TN cores within an optimization problem. In contrast, SVDinsTN addresses the TN-SS problem from a regularized modeling perspective, thereby avoiding the repeated structure evaluations and significantly reducing computational costs.

2. Notations and Preliminaries

A *tensor* is a multi-dimensional array, and the number of dimensions (also called *modes*) of which is referred to as

---

[1]We focus on TN representation in scientific computing and machine learning, while acknowledging its history of research in physics [7, 23, 31].

the *tensor order*. In the paper, first-order tensors (vectors), second-order tensors (matrices), and Nth-order tensors are denoted by $\mathbf{x} \in \mathbb{R}^{I_1}$, $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$, and $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, respectively. We use $\|\mathcal{X}\|_F$ and $\|\mathcal{X}\|_1$ to denote the Frobenius norm and ℓ_1-norm of $\mathcal{X}$, respectively. For brevity, we let $\mathbb{K}_N$ denote the set $\{1, 2, \cdots, N\}$, which is used for labeling individual indices. Similarly, we let $\mathbb{TL}_N$ denote the set $\{(t, l) | 1 \leq t < l \leq N; t, l \in \mathbb{N}\}$, which is used for labeling paired indices.

We next provide a review of several operations on tensors [41]. The *generalized tensor unfolding* is an operation that converts a tensor into a matrix by merging a group of tensor modes into the rows of the matrix and merging the remaining modes into the columns. For example, an $I_1 \times I_2 \times I_3 \times I_4$ tensor can be unfolded into an $I_1 I_3 \times I_2 I_4$ matrix, denoted by $\mathbf{X}_{[1,3;2,4]}$. We also use $\mathbf{X}_{(2)}$ to simply denote $\mathbf{X}_{[2;1,3,4]} \in \mathbb{R}^{I_2 \times I_1 I_2 I_4}$, which is also called *mode-2 unfolding*, and we use $\mathcal{X} = \text{fold}(\mathbf{X}_{(2)}, 2)$ to denote its corresponding inverse operation [14]. The *generalized tensor transposition* is an operation that rearranges tensor modes. For example, an $I_1 \times I_2 \times I_3 \times I_4$ tensor can be transposed into an $I_3 \times I_2 \times I_1 \times I_4$ tensor, denoted by $\vec{\mathcal{X}}^{\mathbf{n}}$ with $\mathbf{n} = (3, 2, 1, 4)$. The *tensor contraction* is an operation that obtains a new tensor by pairing, multiplying, and summing indices of certain modes of two tensors. For example, if a fourth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times I_4}$ and a third-order tensor $\mathcal{Y} \in \mathbb{R}^{J_1 \times J_2 \times J_3}$ satisfy $I_2 = J_1$ and $I_4 = J_2$, then the tensor contraction between the 2nd and 4th modes of $\mathcal{X}$ and the 1st and 2nd modes of $\mathcal{Y}$ yields a tensor $\mathcal{Z} = \mathcal{X} \times_{2,4}^{1,2} \mathcal{Y} \in \mathbb{R}^{I_1 \times I_3 \times J_3}$. The elements of $\mathcal{Z}$ are calculated as follows:

$$\mathcal{Z}(i_1, i_3, j_3) = \sum_{i_2=1}^{I_2} \sum_{i_4=1}^{I_4} \mathcal{X}(i_1, i_2, i_3, i_4) \mathcal{Y}(i_2, i_4, j_3).$$

The operation "vec()" transforms a tensor to a column vector by lexicographical reordering of its elements. The operation "diag()" extracts the main diagonal elements as a column vector from a diagonal matrix, or constructs a diagonal matrix using a column vector for its main diagonal.

2.1. Tensor Network

In general, a *tensor network (TN)* is defined as a set of small-sized tensors, known as TN cores, in which some or all modes are contracted according to specific operations [5]. The primary purpose of a TN is to represent higher-order data using these TN cores. By considering TN cores as nodes and operations between modes of cores as edges, we define the graph formed by these nodes and edges as the TN topology. Additionally, we assign a non-negative integer weight to each edge to indicate the size of the corresponding mode of TN cores, and call the vector composed of these edge weights the TN rank. Consequently, a TN structure refers to a weighted graph comprising nodes, edges, and weights, encompassing both the TN topology and TN rank.

This paper focuses on only a class of TNs that employs tensor contraction as the operation among TN cores and adopts a simple graph as the TN topology. More particularly, when representing an Nth-order tensor $\mathcal{X}$, this class of TNs comprises precisely N TN cores, each corresponding to one mode of $\mathcal{X}$. A notable method is the FCTN decomposition, which represents an Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ by N small-sized Nth-order cores denoted by $\mathcal{G}_k \in \mathbb{R}^{R_{1,k} \times R_{2,k} \times \cdots \times R_{k-1,k} \times I_k \times R_{k,k+1} \times \cdots \times R_{k,N}}$ for $k \in \mathbb{K}_N$ [41]. In this decomposition, any two cores $\mathcal{G}_l$ and $\mathcal{G}_t$ for $(t, l) \in \mathbb{TL}_N$ share an equal-sized mode $R_{t,l}$ used for tensor contraction. We define the FCTN rank as the vector $(R_{1,2}, R_{1,3}, \cdots, R_{1,N}, R_{2,3}, \cdots, R_{2,N}, \cdots, R_{N-1,N}) \in \mathbb{R}^{N(N-1)/2}$. According to the concept of tensor contraction, removing rank-one edges in a TN topology does not change the expression of the TN. This means that if any element in the FCTN rank is equal to one, the corresponding edge can be harmlessly eliminated from the "fully-connected" topology. For instance, a "fully-connected" topology with rank $(R_{1,2}, 1, \cdots, 1, R_{2,3}, 1, \cdots, 1, R_{N-2,N-1}, R_{N-1,N})$ can be equivalently converted into a "chain" topology with rank $(R_{1,2}, R_{2,3}, \cdots, R_{N-1,N})$ in this manner. This fact can be formally stated as follows.

Property 1 *[16] There exists a one-to-one correspondence between the TN structure and FCTN rank.*

According to Property 1, we can search for a compact TN structure by optimizing the FCTN rank.

3. An Efficient Method for TN-SS

We propose an efficient method to solve the TN-SS problem from a regularized modeling perspective. The *main idea* of the proposed method is to optimize the TN structure (FCTN rank) simultaneously during the computation of TN cores.

3.1. SVDinsTN

We start with the definition of the following SVDinsTN.

Definition 1 (SVDinsTN) *Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ be an Nth-order tensor such that*

$$\mathcal{X}(i_1, i_2, \cdots, i_N) =$$

$$\sum_{r_{1,2}=1}^{R_{1,2}} \sum_{r_{1,3}=1}^{R_{1,3}} \cdots \sum_{r_{1,N}=1}^{R_{1,N}} \sum_{r_{2,3}=1}^{R_{2,3}} \cdots \sum_{r_{2,N}=1}^{R_{2,N}} \cdots \sum_{r_{N-1,N}=1}^{R_{N-1,N}}$$

$$\mathbf{S}_{1,2}(r_{1,2}, r_{1,2}) \mathbf{S}_{1,3}(r_{1,3}, r_{1,3}) \cdots \mathbf{S}_{1,N}(r_{1,N}, r_{1,N})$$

$$\mathbf{S}_{2,3}(r_{2,3}, r_{2,3}) \cdots \mathbf{S}_{2,N}(r_{2,N}, r_{2,N}) \cdots \quad (1)$$

$$\mathbf{S}_{N-1,N}(r_{N-1,N}, r_{N-1,N})$$

$$\mathcal{G}_1(i_1, r_{1,2}, r_{1,3}, \cdots, r_{1,N})$$

$$\mathcal{G}_2(r_{1,2}, i_2, r_{2,3}, \cdots, r_{2,N}) \cdots$$

$$\mathcal{G}_k(r_{1,k}, r_{2,k}, \cdots, r_{k-1,k}, i_k, r_{k,k+1}, \cdots, r_{k,N}) \cdots$$

$$\mathcal{G}_N(r_{1,N}, r_{2,N}, \cdots, r_{N-1,N}, i_N),$$

where $\mathcal{G}_k \in \mathbb{R}^{R_{1,k} \times R_{2,k} \times \cdots \times R_{k-1,k} \times I_k \times R_{k,k+1} \times \cdots \times R_{k,N}}$ for $\forall k \in \mathbb{K}_N$ are Nth-order tensors and called TN cores, and $\mathbf{S}_{t,l} \in \mathbb{R}^{R_{t,l} \times R_{t,l}}$ for $\forall (t,l) \in \mathbb{TL}_N$ are diagonal matrices. Then we call (1) an SVD-inspired TN decomposition (SVDinsTN) of $\mathcal{X}$, denoted by $\mathcal{X} = STN(\mathcal{G}, \mathbf{S})$, where $\mathcal{G}$ denotes $\{\mathcal{G}_k | k \in \mathbb{K}_N\}$ and $\mathbf{S}$ denotes $\{\mathbf{S}_{t,l} | (t,l) \in \mathbb{TL}_N\}$.

As shown in Figure 1(b), SVDinsTN includes both TN cores and diagonal factors, and can use the sparsity of diagonal factors to reveal a compact TN structure and utilize TN cores (merged with diagonal factors) to represent a tensor.

Remark 1 (SVDinsTN & SVD) *SVDinsTN extends upon the "core&diagonal factor&core" form of SVD to higher-order cases (see Figure 1(a) & (b)), incorporating the idea of determining rank through non-zero elements in the diagonal factor. In particular, SVDinsTN can degrade into SVD in second-order cases when TN cores satisfy orthogonality.*

Remark 2 (SVDinsTN & FCTN) *SVDinsTN builds upon FCTN decomposition [41] but can reveal the FCTN rank. It achieves this by inserting diagonal factors between any two TN cores in FCTN decomposition and leveraging the number of non-zero elements in the diagonal factors to determine the FCTN rank. In particular, SVDinsTN can transform into a TN decomposition by merging the diagonal factors into TN cores through the tensor contraction operation.*

3.2. A Regularized Method for TN-SS

We present an SVDinsTN-based regularized method, which updates TN cores and diagonal factors alternately, and imposes a sparse operator to induce the sparsity of diagonal factors to reveal a compact TN structure.

We consider an ℓ_1-norm-based operator for diagonal factors $\mathbf{S}$ and Tikhonov regularization [10] for TN cores $\mathcal{G}$. The ℓ_1-norm-based operator is used to promote the sparsity of $\mathbf{S}$, and the Tikhonov regularization is used to constrict the feasible range of $\mathcal{G}$. Mathematically, the proposed model can be formulated as follows:

$$\min_{\mathcal{G}, \mathbf{S}} \frac{1}{2} \|\mathcal{X} - STN(\mathcal{G}, \mathbf{S})\|_F^2 + \frac{\mu}{2} \sum_{k \in \mathbb{K}_N} \|\mathcal{G}_k\|_F^2 + \sum_{(t,l) \in \mathbb{TL}_N} \lambda_{t,l} \|\mathbf{S}_{t,l}\|_1, \quad (2)$$

where $\lambda_{t,l} > 0$ and $\mu > 0$ are regularization parameters.

We use the PAM-based algorithm [2] to solve (2), whose solution is obtained by alternately updating

$$\begin{cases} \mathcal{G}_k = \underset{\mathcal{G}_k}{\arg\min} \ \frac{1}{2} \|\mathcal{X} - STN(\mathcal{G}, \mathbf{S})\|_F^2 + \frac{\mu}{2} \|\mathcal{G}_k\|_F^2 \\ \qquad + \frac{\rho}{2} \|\mathcal{G}_k - \hat{\mathcal{G}}_k\|_F^2, \ \forall k \in \mathbb{K}_N, \\ \mathbf{S}_{t,l} = \underset{\mathbf{S}_{t,l}}{\arg\min} \ \frac{1}{2} \|\mathcal{X} - STN(\mathcal{G}, \mathbf{S})\|_F^2 + \lambda_{t,l} \|\mathbf{S}_{t,l}\|_1 \\ \qquad + \frac{\rho}{2} \|\mathbf{S}_{t,l} - \hat{\mathbf{S}}_{t,l}\|_F^2, \ \forall (t,l) \in \mathbb{TL}_N, \end{cases} \quad (3)$$

where $\rho > 0$ is a proximal parameter (we fix $\rho = 0.001$), and $\hat{\mathcal{G}}_k$ and $\hat{\mathbf{S}}_{t,l}$ are the solutions of the $\mathcal{G}_k$-subproblem and $\mathbf{S}_{t,l}$-subproblem at the previous iteration, respectively. The detailed solution processes for $\mathcal{G}_k$- and $\mathbf{S}_{t,l}$-subproblems are provided in the *Supplementary Material*[2]. Here, we only present their core processes and results.

1) Update $\mathcal{G}_k$ for $\forall k \in \mathbb{K}_N$: Solving the $\mathcal{G}_k$-subproblem requires fixing the other TN cores and diagonal factors. To address this, we use $\mathbf{M}_k$ to denote the matrix obtained by performing tensor contraction and unfolding operations on all diagonal factors and TN cores except $\mathcal{G}_k$. Then, the $\mathcal{G}_k$-subproblem can be directly solved as follows:

$$\mathbf{G}_{k(k)} = \left(\mathbf{X}_{(k)} \mathbf{M}_k^{\mathrm{T}} + \rho \hat{\mathbf{G}}_{k(k)}\right) \left(\mathbf{M}_k \mathbf{M}_k^{\mathrm{T}} + (\mu + \rho) \mathbf{I}\right)^{-1}. \quad (4)$$

2) Update $\mathbf{S}_{t,l}$ for $\forall (t,l) \in \mathbb{TL}_N$: Solving the $\mathbf{S}_{t,l}$-subproblem requires fixing the other diagonal factors and TN cores. In a similar fashion, we use $\mathbf{H}_{t,l}$ to denote the matrix obtained by performing tensor contraction and unfolding operations on all TN cores and diagonal factors except $\mathbf{S}_{t,l}$. Then, we use an alternating direction method of multipliers (ADMM) [6] to solve the $\mathbf{S}_{t,l}$-subproblem, which can be rewritten as follows:

$$\min_{\mathbf{s}_{t,l}, \mathbf{q}_{t,l}} \frac{1}{2} \|\mathbf{x} - \mathbf{H}_{t,l} \mathbf{q}_{t,l}\|_F^2 + \lambda_{t,l} \|\mathbf{s}_{t,l}\|_1 + \frac{\rho}{2} \|\mathbf{s}_{t,l} - \hat{\mathbf{s}}_{t,l}\|_F^2 \quad (5)$$
$$\text{s.t. } \mathbf{s}_{t,l} - \mathbf{q}_{t,l} = 0,$$

where $\mathbf{x} = \mathrm{vec}(\mathcal{X})$, $\mathbf{s}_{t,l} = \mathrm{diag}(\mathbf{S}_{t,l})$, and $\mathbf{q}_{t,l}$ is an auxiliary variable. The augmented Lagrangian function of (5) can be expressed as the following concise form:

$$L_{\beta_{t,l}}(\mathbf{s}_{t,l}, \mathbf{q}_{t,l}, \mathbf{p}_{t,l}) = \frac{1}{2} \|\mathbf{x} - \mathbf{H}_{t,l} \mathbf{q}_{t,l}\|_F^2 + \lambda_{t,l} \|\mathbf{s}_{t,l}\|_1$$
$$+ \frac{\rho}{2} \|\mathbf{s}_{t,l} - \hat{\mathbf{s}}_{t,l}\|_F^2 + \frac{\beta_{t,l}}{2} \left\| \mathbf{s}_{t,l} - \mathbf{q}_{t,l} + \frac{\mathbf{p}_{t,l}}{\beta_{t,l}} \right\|_F^2, \quad (6)$$

where $\mathbf{p}_{t,l}$ is the Lagrangian multiplier and $\beta_{t,l} > 0$ is the penalty parameter. Within the ADMM framework, $\mathbf{q}_{t,l}$, $\mathbf{s}_{t,l}$, and $\mathbf{p}_{t,l}$ can be solved by alternately updating

$$\begin{cases} \mathbf{q}_{t,l} = \left[\mathbf{H}_{t,l}^{\mathrm{T}} \mathbf{H}_{t,l} + \beta_{t,l} \mathbf{I}\right]^{-1} \left[\mathbf{H}_{t,l}^{\mathrm{T}} \mathbf{x} + \beta_{t,l} \mathbf{s}_{t,l} + \mathbf{p}_{t,l}\right], \\ \mathbf{s}_{t,l} = \mathrm{shrink}\left(\dfrac{\rho \hat{\mathbf{s}}_{t,l} + \beta_{t,l} \mathbf{q}_{t,l} - \mathbf{p}_{t,l}}{\rho + \beta_{t,l}}, \dfrac{\lambda_{t,l}}{\rho + \beta_{t,l}}\right), \\ \mathbf{p}_{t,l} = \mathbf{p}_{t,l} + \beta_{t,l}(\mathbf{s}_{t,l} - \mathbf{q}_{t,l}), \end{cases} \quad (7)$$

where $\mathrm{shrink}(\mathbf{a}, \mathbf{b}) = \max(\mathbf{a} - \mathbf{b}, \mathbf{0}) + \min(\mathbf{a} + \mathbf{b}, \mathbf{0})$.

We describe the pseudocode to optimize model (2) in Algorithm 1, where the operation "$\mathrm{ones}(I_1, \cdots, I_N)$" returns an $I_1 \times \cdots \times I_N$ tensor whose elements are all equal to 1.

Below, we present a brief analysis of the computational complexity and provide a theoretical convergence guarantee for the developed algorithm.

---

[2] https://yubangzheng.github.io

Algorithm 1 PAM-based algorithm to optimize model (2).

Input: A tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and a parameter γ.

Initialization: Initialize $\mathbf{S}_{t,l}$ and $R_{t,l}$ by the initialization scheme in Section 3.3 and let $\beta_{t,l} = 1$ for $\forall (t,l) \in \mathbb{TL}_N$; let $\mathcal{G}_k = 1/\sqrt{I_k} \, \text{ones}(R_{1,k}, R_{2,k}, \cdots, R_{k-1,k}, I_k, R_{k,k+1}, \cdots, R_{k,N})$ for $\forall k \in \mathbb{K}_N$ and $\mu = 1$.

1: **while** not converged **do**
2:     Let $\hat{\mathcal{X}} = \mathcal{X}$ and $\lambda_{t,l} = \gamma \max(\mathbf{S}_{t,l})(\rho + \beta_{t,l})$.
3:     Update $\mathbf{G}_{k(k)}$ by (4) and let $\mathcal{G}_k = \text{fold}(\mathbf{G}_{k(k)}, k)$.
4:     **for** $i = 1$ to 5 **do**
5:         Update $\mathbf{q}_{t,l}$, $\mathbf{s}_{t,l}$, and $\mathbf{p}_{t,l}$ by (7).
6:     **end for**
7:     Delete zero elements in $\mathbf{s}_{t,l}$, let $\mathbf{S}_{t,l} = \text{diag}(\mathbf{s}_{t,l})$, and define the size of $\mathbf{s}_{t,l}$ as $R_{t,l}$.
8:     Delete the corresponding dimensions of $\mathcal{G}_k$ and let $\mathcal{X} = \text{STN}(\mathcal{G}, \mathbf{S})$.
9:     Check the convergence condition: $\frac{\|\mathcal{X} - \hat{\mathcal{X}}\|_F}{\|\hat{\mathcal{X}}\|_F} < 10^{-5}$.
10: **end while**

Output: $\mathcal{G}_k$ for $\forall k \in \mathbb{K}_N$, and $\mathbf{S}_{t,l}$ and $R_{t,l}$ for $\forall (t,l) \in \mathbb{TL}_N$.

Computational complexity. For simplicity, we let the size of the Nth-order tensor $\mathcal{X}$ be $I \times I \times \cdots \times I$ and the initial rank be $(R, R, \cdots, R)$ satisfied $R \leq I$. The computational cost involves updating $\mathcal{G}$ and $\mathbf{S}$, resulting in costs of $\mathcal{O}\left(N \sum_{k=2}^{N} I^k R^{k(N-k)+k-1} + N I^{N-1} R^{2(N-1)} + N^3 I R^N\right)$ and $\mathcal{O}\left(N^2 \sum_{k=2}^{N} I^k R^{k(N-k)+k-1} + N^4 I R^N + N^2 I^N R^2\right)$, respectively. Hence, the computational cost at each iteration is $\mathcal{O}\left(N^2 \sum_{k=2}^{N} I^k R^{k(N-k)+k-1} + N^4 I R^N + N^2 I^N R^2\right)$.

Theorem 1 (Convergence guarantee) *The sequence generated by Algorithm 1, denoted by $\{\mathcal{G}^{(s)}, \mathbf{S}^{(s)}\}_{s \in \mathbb{N}}$, converges to a critical point of the optimization problem (2).*

3.3. Initialization Scheme

SVDinsTN encounters high computational cost in the first several iterations if the TN rank $R_{t,l}$ for $\forall (t,l) \in \mathbb{TL}_N$ are initialized with large values. This is due to the adoption of a "fully-connected" topology as a starting point. To solve this challenge, we design a novel initialization scheme aimed at effectively reducing the initial values of the TN rank.

We first give an upper bound for the TN rank, by which we then design an initialization scheme for the TN rank $R_{t,l}$ and diagonal factors $\mathbf{S}_{t,l} \in \mathbb{R}^{R_{t,l} \times R_{t,l}}$ for $\forall (t,l) \in \mathbb{TL}_N$.

Theorem 2 *Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ be an Nth-order tensor, then there exists an SVDinsTN (1) with the TN rank $R_{t,l} \leq \min(\text{rank}(\mathbf{X}_{(t)}), \text{rank}(\mathbf{X}_{(l)}))$ for $\forall (t,l) \in \mathbb{TL}_N$.*

Theorem 2 indicates that $\min(\text{rank}(\mathbf{X}_{(t)}), \text{rank}(\mathbf{X}_{(l)}))$ can be the initial value of the TN rank $R_{t,l}$. For real-world data, this value is usually embodied by the rank of mode-(t,l) slices[3] of $\mathcal{X}$. Therefore, we initialize $R_{t,l}$ and $\mathbf{S}_{t,l}$ by

---

[3]Mode-(t,l) slices are obtained by fixing all but the mode-t and the mode-l indexes of a tensor [40].

virtue of truncated SVD of mode-(t,l) slices of $\mathcal{X}$, which consists of the following two steps.

Step 1: We first calculate the mean of all mode-(t,l) slices of $\mathcal{X}$ and denote it by $\mathbf{X}_{t,l}$. Then we perform SVD on $\mathbf{X}_{t,l}$ to obtain $\mathbf{s}_{t,l} \in \mathbb{R}^{\min(I_t, I_l)}$, whose elements are singular values of $\mathbf{X}_{t,l}$.

Step 2: We first let $\mathbf{s}_{t,l} = \text{shrink}\left(\mathbf{s}_{t,l}, \frac{\gamma \max(\mathbf{s}_{t,l})}{|\mathbf{s}_{t,l}| + 10^{-16}}\right)$ and delete zero elements in $\mathbf{s}_{t,l}$. Then we let $\mathbf{S}_{t,l} = \text{diag}(\mathbf{s}_{t,l})$ and define the size of $\mathbf{s}_{t,l}$ as $R_{t,l}$.

In practical applications, the shrink operation in Step 2 effectively reduces the initial value of $R_{t,l}$ by projecting very small singular values in $\mathbf{s}_{t,l}$ to zero. As a result, *the challenge of high computational costs in the first several iterations of SVDinsTN can be effectively addressed* (see Figure 2 for a numerical illustration).

4. Numerical Experiments

In this section, we present numerical experiments on both synthetic and real-world data to evaluate the performance of the proposed SVDinsTN. The primary objective is to validate the following three Claims:

A: SVDinsTN can reveal a customized TN structure that aligns with the unique structure of a given tensor.

B: SVDinsTN can greatly reduce time costs while achieving a comparable representation ability to state-of-the-art TN-SS methods. Moreover, SVDinsTN can also surpass existing tensor decomposition methods with predefined topologies regarding representation ability.

C: SVDinsTN can outperform existing tensor decomposition methods in the tensor completion task, highlighting its effectiveness as a valuable tool in applications.

4.1. Experiments for Validating Claim A

We conduct experiments to validate Claim A. Since real-world data lacks a true TN structure, we consider only synthetic data in this experiment.

Data generation. We first randomly generate $\mathcal{G}_k$ for $\forall k \in \mathbb{K}_N$ and $\mathbf{S}_{t,l}$ for $\forall (t,l) \in \mathbb{TL}_N$, whose elements are taken from a uniform distribution between 0 and 1. Then we obtain the synthetic tensor by $\mathcal{X} = \text{STN}(\mathcal{G}, \mathbf{S})$.

Experiment setting. We test both fourth-order tensors of size $16 \times 18 \times 20 \times 22$ and fifth-order tensors of size $14 \times 16 \times 18 \times 20 \times 22$, and consider different kinds of TN structures. For each structure, we conduct 100 independent tests and regenerate the synthetic data to ensure reliable and unbiased results. The ability of the proposed SVDinsTN to reveal TN structure is measured by the *success rate* of the output structures, defined as $S_T/T \times 100\%$, where $T = 100$ is the total number of tests and S_T is the number of tests that accurately output the true TN structure. In all tests, the parameter γ is set to 0.0015.

Table 1. Performance of SVDinsTN on TN structure revealing under 100 independent tests.

True structure								
Success rate	100%	100%	96%	95%	99%	98%	97%	100%

Table 2. Comparison of CR (↓) and run time (×1000s, ↓) of different methods on light field data.

Method	RE bound: 0.01		RE bound: 0.05		RE bound: 0.1	
	CR	Time	CR	Time	CR	Time
Bunny						
TRALS [38]	60.5%	13.54	17.4%	0.471	5.31%	0.118
FCTNALS [41]	65.1%	13.08	20.9%	0.473	3.93%	0.041
TNGreedy [12]	26.1%	11.02	6.32%	1.021	2.34%	0.362
TNGA [15]	27.9%	1014	5.01%	180.3	2.25%	12.52
TNLS [17]	24.3%	1402	4.26%	63.70	2.16%	24.53
TNALE [18]	26.3%	144.5	4.52%	18.36	2.26%	3.064
SVDinsTN	22.4%	**0.745**	6.92%	**0.029**	2.66%	**0.005**
Knights						
TRALS [38]	74.7%	10.31	26.9%	3.835	9.15%	0.423
FCTNALS [41]	73.5%	12.35	20.9%	0.619	3.93%	**0.014**
TNGreedy [12]	32.1%	12.53	7.55%	1.366	3.50%	0.481
TNGA [15]	38.7%	912.9	5.01%	140.2	2.44%	12.52
TNLS [17]	27.3%	1286	4.73%	75.51	2.15%	5.320
TNALE [18]	27.6%	266.4	4.52%	25.05	2.10%	3.386
SVDinsTN	32.0%	**1.548**	5.64%	**0.104**	2.76%	0.019
Truck						
TRALS [38]	62.8%	17.62	22.6%	1.738	6.00%	0.090
FCTNALS [41]	69.3%	7.735	20.9%	2.953	3.93%	0.159
TNGreedy [12]	26.9%	6.676	7.26%	1.259	3.35%	0.488
TNGA [15]	27.9%	1029	5.01%	170.3	2.85%	14.83
TNLS [17]	26.4%	992.6	4.99%	119.8	2.57%	19.35
TNALE [18]	24.7%	239.3	5.77%	19.54	2.90%	5.160
SVDinsTN	23.5%	**1.051**	6.42%	**0.152**	2.83%	**0.023**

Result analysis. Table 1 presents the *success rate* of the output TN structures obtained by the proposed SVDinsTN in 100 independent tests on fourth-order and fifth-order tensors. It can be observed that the proposed SVDinsTN consistently yields high *success rates* of over 95% in all test cases. Notably, in approximately half of the test cases, the *success rates* reach a perfect score of 100%.

4.2. Experiments for Validating Claim B

We conduct experiments to validate Claim B. We consider both real-world data and synthetic data, and use different methods to represent it in this experiment.

Experiment setting. We test three light field data[4], named *Bunny*, *Knights*, and *Truck*, which are fifth-order tensors of size $40 \times 60 \times 3 \times 9 \times 9$ (spatial height × spatial width × color channel × vertical grid × horizontal grid). We employ six representative methods as the compared baselines, including two methods with pre-defined topology: TRALS [38] and FCTNALS [41], and four TN-SS methods: TNGreedy [12], TNGA [15], TNLS [17], and TNALE [18]. We represent the test light field data by different methods and calculate the corresponding *compression ratio (CR)* to achieve a certain *reconstruction error (RE) bound*. The *CR* is defined as $F_{\mathcal{G}}/F_{\mathcal{X}} \times 100\%$, where $F_{\mathcal{G}}$ is the number of elements of TN cores used to represent a tensor and $F_{\mathcal{X}}$ is the number of total elements of the original tensor. The *RE* is defined as $\|\mathcal{X} - \tilde{\mathcal{X}}\|_F/\|\mathcal{X}\|_F$, where $\mathcal{X}$ is the original data and $\tilde{\mathcal{X}}$ is the reconstructed data. In all tests, we select the parameter γ from the interval $[10^{-7}, 10^{-3}]$.

Result analysis. Table 2 reports *CR* and *run time* of different methods on fifth-order light field data. The results show that the proposed SVDinsTN achieves significantly lower *CRs* than TRALS and FCTNALS, which are methods with pre-defined topology. This indicates that SVDinsTN can obtain a more compact structure than the pre-defined one. Furthermore, while SVDinsTN requires the determination of diagonal factors alongside TN cores, its iterative process generates progressively simpler structures, enhancing the computational efficiency. Consequently, SVDinsTN demonstrates *faster* performance compared to TRALS and FCTNALS. Compared to the TN-SS methods, the proposed SVDinsTN achieves *a substantial speed improvement* while maintaining a comparable level of *CR*. Remarkably, SVDinsTN achieves an acceleration of approximately $100 \sim 1000$ times over TNGA, TNLS, and TNALE. This is because TNGreedy, TNGA, TNLS, and TNALE adopt the "sampling-evaluation" framework, necessitating a significant number of repeated structure evaluations. In contrast, SVDinsTN introduces a regularized modeling framework, requiring only a single evaluation.

Impact of the initialization scheme. We analyze the impact of the initialization scheme. In Table 3, we report *CR* and *run time* of SVDinsTN with different initializations on light field data *Truck*. As observed, our initialization

---

[4]http://lightfield.stanford.edu/lfs.html

Table 3. Comparison of CR (↓) and run time (×1000s, ↓) of SVDinsTN with different initializations on light field data *Truck*.

Initialization	RE bound: 0.01		RE bound: 0.05		RE bound: 0.1	
	CR	Time	CR	Time	CR	Time
Random	30.7%	1.203	8.17%	0.474	4.14%	0.208
Ours	23.5%	1.051	6.42%	0.152	2.83%	0.023
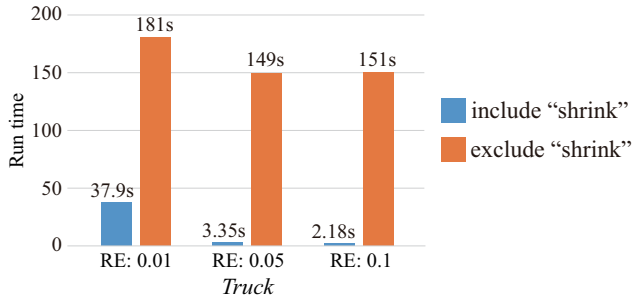
Figure 2. Comparison of the runtime in the first five iterations of SVDinsTN on light field data *Truck* when including and excluding the shrink operation in our initialization scheme.

scheme achieves lower *CRs* compared to random initialization, while maintaining higher efficiency. This corroborates that our initialization scheme can provide a favorable starting point and enhance computational efficiency. In particular, even with random initialization, our method achieves significant acceleration compared to other TN-SS methods. We further analyze the impact of the shrink operation in our initialization scheme. In Figure 2, we present the *run time* comparison of the first five iterations of our method when including and excluding the shrink operation in our initialization scheme. As observed, the shrink operation in our initialization scheme enables our method to greatly reduce the computational costs in the first several iterations.

Higher-order cases. We analyze whether the proposed SVDinsTN still performs well on higher-order tensors. We randomly generate 6th-, 8th-, and 10th-order tensors by using the same procedure in Section 4.1. The size of each tensor mode is randomly selected from $\{5, 6, 7, 8\}$, the edge number of each TN is randomly selected from $\{6, 8, 10\}$, and the rank of each edge is randomly selected from $\{2, 3\}$. For each tensor order, we randomly generate 5 tensors. We compare SVDinsTN and baseline methods in terms of *CR* and *run time* when reaching the RE bound of 0.01, and show the results in Table 4. As observed, SVDinsTN is applicable to higher orders beyond 5, and even up to 10. The behind rational is the truncated SVD used in initialization restricts the initial values of the rank for each edge to a relatively small range, thus improving computational and storage efficiency (see Figure 2). As the iterations progress, the sparsity regularization in the model leads to progressively simpler learned structures, further boosting efficiency.

Table 4. Comparison of the CR (↓) and run time (×1000s, ↓) of different methods when reaching the RE bound of 0.01. The result is the average value of 5 independent experiments and "–" indicates "out of memory".

Method	6th-order		8th-order		10th-order	
	CR	Time	CR	Time	CR	Time
TRALS [38]	1.35%	0.006	0.064%	0.034	–	–
FCTNALS [41]	2.13%	**0.002**	–	–	–	–
TNGreedy [12]	<u>0.88%</u>	0.167	<u>0.016%</u>	2.625	0.0008%	45.39
TNGA [15]	0.94%	3.825	0.024%	51.40	–	–
TNLS [17]	1.11%	0.673	0.038%	59.83	–	–
TNALE [18]	1.65%	0.201	0.047%	19.96	–	–
SVDinsTN	1.13%	**0.002**	<u>0.016%</u>	**0.017**	<u>0.0007%</u>	**0.608**

4.3. Experiments for Validating Claim C

We conduct experiments to validate Claim C. We employ the proposed SVDinsTN to a fundamental application, i.e., tensor completion (TC), and compare it with the state-of-the-art tensor decomposition-based TC methods. Given an incomplete observation tensor $\mathcal{F} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ of $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the proposed TC method first updates $\mathcal{G}$ and $\mathbf{S}$ by Algorithm 1, and then updates the target tensor $\mathcal{X}$ as follows: $\mathcal{X} = \mathcal{P}_{\Omega^c}((\text{STN}(\mathcal{G}, \mathbf{S}) + \rho\hat{\mathcal{X}})/(1 + \rho)) + \mathcal{P}_{\Omega}(\mathcal{F})$, where Ω is the index set of the known elements, $\mathcal{P}_{\Omega}(\mathcal{X})$ is a projection operator that projects the elements in Ω to themselves and all others to zeros, $\hat{\mathcal{X}}$ is the result at the previous iteration, and the initial $\mathcal{X}$ is $\mathcal{F}$.

Experiment setting. We test four color videos[5], named *Bunny*, *News*, *Salesman*, and *Silent*, which are fourth-order tensors of size $144 \times 176 \times 3 \times 50$ (spatial height × spatial width × color channel × frame). We employ six methods for comparison, named FBCP [37], TMac [29], TMacTT [3], TRLRF [33], TW [28], and TNLS[6] [17], respectively. We set the missing ratio (MR) to 90%, which is defined as the ratio of the number of missing elements to the total number of elements. We evaluate the reconstructed quality by *the mean peak signal-to-noise ratio (MPSNR)* computed across all frames. In all tests, the parameter γ is set to 0.0003.

Result analysis. Table 5 reports *MPSNR* and *run time* obtained by different TC methods. As observed, the proposed SVDinsTN consistently achieves the highest *MPSNR* values among all utilized TC methods across all test color videos. In Figure 3, we present the reconstructed images and their corresponding residual images at the 25th frame of *News*. We observe that the proposed SVDinsTN outperforms the baseline methods in terms of visual quality, particularly with respect to background cleanliness and local details (*e.g.* "dancer") recovery.

---

[5] http://trace.eas.asu.edu/yuv/
[6] TNLS excels in the compression task; therefore, we use it as a representative TN-SS method for comparison.

Table 5. Comparison of MPSNR (↑) and run time (in seconds, ↓) of different TC methods on color videos.

Video	FBCP [37]		TMac [29]		TMacTT [3]		TRLRF [33]		TW [28]		TNLS [17]		SVDinsTN	
	MPSNR	Time	MPSNR	Time	MPSNR	Time	MPSNR	Time	MPSNR	Time	MPSNR	Time	MPSNR	Time
Bunny	28.402	1731.2	28.211	1203.5	29.523	453.76	29.163	486.76	30.729	1497.4	28.787	99438	**32.401**	691.33
News	28.234	1720.4	27.882	340.46	28.714	535.97	28.857	978.12	30.027	1426.3	29.761	37675	**31.643**	932.42
Salesman	29.077	1783.2	28.469	353.63	29.534	656.45	28.288	689.35	30.621	1148.7	30.685	76053	**31.684**	769.54
Silent	30.126	1453.9	30.599	316.21	30.647	1305.6	31.081	453.24	31.731	1232.0	28.830	98502	**32.706**	532.31
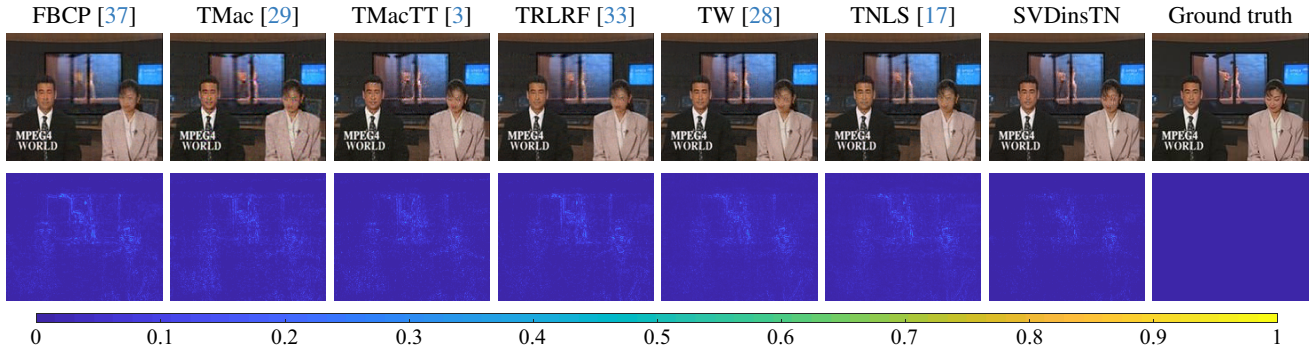
Figure 3. Reconstructed images and residual images obtained by different methods on the 25th frame of *News*. Here the residual image is the average absolute difference between the reconstructed image and the ground truth over R, G, and B channels.
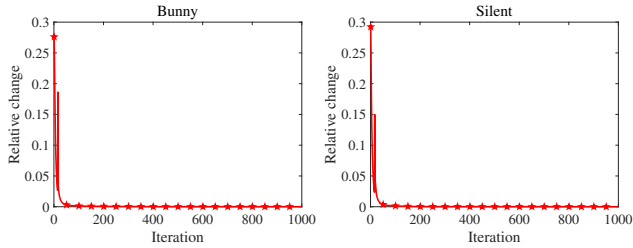

Figure 4. Relative change curves with respect to the iteration number on test color videos *Bunny* and *Silent*. Here the relative change is defined as $\|\mathcal{X} - \hat{\mathcal{X}}\|_F / \|\hat{\mathcal{X}}\|_F$, and $\mathcal{X}$ and $\hat{\mathcal{X}}$ are the results of the current iteration and its previous iteration.

Numerical convergence. In Theorem 1, we provide a *theoretical* convergence guarantee for the proposed method. Here, we select color videos *Bunny* and *Silent* as examples to *numerically* verify the convergence. Figure 4 presents the relative change in the reconstructed color videos at each iteration compared to their respective previous iterations. We observe that the values of the relative change achieved by the proposed method decrease and gradually tend to zero as the number of iterations increases. This justifies the numerical convergence of the proposed method.

5. Conclusion

We propose a novel TN paradigm, called SVDinsTN, which enables us to solve the challenging TN-SS problem from a regularized modeling perspective. This perspective renders our model highly amenable to easy solutions, allowing us to leverage well-established optimization algorithms to solve the regularized model. As a result, the proposed method achieves about $100 \sim 1000$ times acceleration compared to the state-of-the-art TN-SS methods with a comparable representation ability. Besides, SVDinsTN demonstrates its effectiveness as a valuable tool in practical applications.

Limitations. In existing research on TN-SS, two challenging issues remain open. One is the computationally consuming issue, and the other is the theoretical guarantee of the optimal TN structure. SVDinsTN addresses the computationally consuming issue. But the theoretical guarantee of the optimal TN structure is still an open problem. Solving this issue will be the direction of our future work.

Acknowledgements

References

[1] Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M. Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1): 2773–2832, 2014. 1

[2] Hédy Attouch, Jérôme Bolte, Patrick Redont, and Antoine Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, 2010. 4

[3] Johann A Bengua, Ho N Phien, Hoang Duong Tuan, and Minh N Do. Efficient tensor completion for color image and video recovery: Low-rank tensor train. *IEEE Transactions on Image Processing*, 26(5):2466–2479, 2017. 7, 8

[4] Andrzej Cichocki, Danilo P. Mandic, Anh Huy Phan, Cesar F. Caiafa, Guoxu Zhou, Qibin Zhao, and Lieven De Lathauwer. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32(2):145–163, 2015. 2

[5] Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh Huy Phan, Qibin Zhao, and Danilo P. Mandic. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions. *Foundations and Trends® in Machine Learning*, 9(4-5):249–429, 2016. 1, 2, 3

[6] Daniel Gabay and Bertrand Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers and Mathematics with Applications*, 2(1):17–40, 1976. 4

[7] Silvano Garnerone, Thiago R. de Oliveira, and Paolo Zanardi. Typicality in random matrix product states. *Physical Review A*, 81:032336, 2010. 2

[8] Mehrdad Ghadiri, Matthew Fahrbach, Gang Fu, and Vahab Mirrokni. Approximately optimal core shapes for tensor decompositions. *arXiv preprint arXiv:2302.03886*, 2023. 2

[9] Ivan Glasser, Ryan Sweke, Nicola Pancotti, Jens Eisert, and J. Ignacio Cirac. Expressive power of tensor-network factorizations for probabilistic modeling. In *Advances in Neural Information Processing Systems*, 2019. 1

[10] Gene H Golub, Per Christian Hansen, and Dianne P O'Leary. Tikhonov regularization and total least squares. *SIAM journal on matrix analysis and applications*, 21(1):185–194, 1999. 4

[11] Kang Han and Wei Xiang. Multiscale tensor decomposition and rendering equation encoding for view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4232–4241, 2023. 1

[12] Meraj Hashemizadeh, Michelle Liu, Jacob Miller, and Guillaume Rabusseau. Adaptive learning of tensor network structures. *arXiv preprint arXiv:2008.05437*, 2020. 1, 2, 6, 7

[13] Christopher J. Hillar and Lek-Heng Lim. Most tensor problems are NP-hard. *Journal of the ACM*, 60(6): 1–39, 2013. 1

[14] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009. 3

[15] Chao Li and Zhun Sun. Evolutionary topology search for tensor network decomposition. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5947–5957, 2020. 1, 2, 6, 7

[16] Chao Li and Qibin Zhao. Is rank minimization of the essence to learn tensor network structure? In *Second Workshop on Quantum Tensor Networks in Machine Learning (QTNML), Neurips*, 2021. 3

[17] Chao Li, Junhua Zeng, Zerui Tao, and Qibin Zhao. Permutation search of tensor network structures via local sampling. In *Proceedings of the 39th International Conference on Machine Learning*, pages 13106–13124, 2022. 1, 2, 6, 7, 8

[18] Chao Li, Junhua Zeng, Chunmei Li, Cesar Caiafa, and Qibin Zhao. Alternating local enumeration (tnale): Solving tensor network structure search with fewer evaluations. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. 1, 2, 6, 7

[19] Nannan Li, Yu Pan, Yaran Chen, Zixiang Ding, Dongbin Zhao, and Zenglin Xu. Heuristic rank selection with progressively searching tensor ring network. *Complex & Intelligent Systems*, 8(2):771–785, 2022.

[20] Yipeng Liu, Yingcong Lu, Weiting Ou, Zhen Long, and Ce Zhu. Adaptively topological tensor network for multi-view subspace clustering. *arXiv preprint arXiv:2305.00716*, 2023. 2

[21] Yisi Luo, Xi-Le Zhao, Deyu Meng, and Tai-Xiang Jiang. Hlrtf: Hierarchical low-rank tensor factorization for inverse problems in multi-dimensional imaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19281–19290, 2022. 1

[22] Chang Nie, Huan Wang, and Le Tian. Adaptive tensor networks decomposition. In *British Machine Vision Conference*, 2021. 2

[23] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, 2014. 2

[24] Ivan Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011. 2

[25] Piyush Rai, Yingjian Wang, Shengbo Guo, Gary Chen, David Dunson, and Lawrence Carin. Scalable Bayesian low-rank decomposition of incomplete multiway tensors. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, page II–1800–II–1808, 2014. 1

[26] Farnaz Sedighin, Andrzej Cichocki, and Anh Huy Phan. Adaptive rank selection for tensor ring decomposition. *IEEE Journal of Selected Topics in Signal Processing*, 15(3):454–463, 2021. 2

[27] Moein Shakeri and Hong Zhang. Moving object detection under discontinuous change in illumination using tensor low-rank and invariant sparse decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7221–7230, 2019. 1

[28] Zhong-Cheng Wu, Ting-Zhu Huang, Liang-Jian Deng, Hong-Xia Dou, and Deyu Meng. Tensor wheel decomposition and its tensor completion application. In *Advances in Neural Information Processing Systems*, pages 27008–27020, 2022. 7, 8

[29] Yangyang Xu, Ruru Hao, Wotao Yin, and Zhixun Su. Parallel matrix factorization for low-rank tensor completion. *Inverse Problems and Imaging*, 9(2):601–624, 2015. 7, 8

[30] Ryuki Yamamoto, Hidekata Hontani, Akira Imakura, and Tatsuya Yokota. Fast algorithm for low-rank tensor completion in delay-embedded space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2048–2056, 2022. 1

[31] Ke Ye and Lek-Heng Lim. Tensor network ranks. *arXiv preprint arXiv:1801.02662*, 2018. 2

[32] Longhao Yuan, Jianting Cao, Xuyang Zhao, Qiang Wu, and Qibin Zhao. Higher-dimension tensor completion via low-rank tensor ring decomposition. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1071–1076, 2018. 2

[33] Longhao Yuan, Chao Li, Danilo Mandic, Jianting Cao, and Qibin Zhao. Tensor ring decomposition with rank minimization on latent space: An efficient approach for tensor completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9151–9158, 2019. 7, 8

[34] Longhao Yuan, Qibin Zhao, Lihua Gui, and Jianting Cao. High-order tensor completion via gradient-based optimization under tensor train format. *Signal Processing: Image Communication*, 73:53–61, 2019. 2

[35] Shipeng Zhang, Lizhi Wang, Lei Zhang, and Hua Huang. Learning tensor low-rank prior for hyperspectral image reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12001–12010, 2021. 1

[36] Xinyuan Zhang, Xin Yuan, and Lawrence Carin. Nonlocal low-rank tensor factor analysis for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8232–8241, 2018. 1

[37] Qibin Zhao, Liqing Zhang, and Andrzej Cichocki. Bayesian CP factorization of incomplete tensors with automatic rank determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1751–1763, 2015. 1, 7, 8

[38] Qibin Zhao, Guoxu Zhou, Shengli Xie, Liqing Zhang, and Andrzej Cichocki. Tensor ring decomposition. *arXiv preprint arXiv:1606.05535*, 2016. 2, 6, 7

[39] Wen-Jie Zheng, Xi-Le Zhao, Yu-Bang Zheng, Jie Lin, Lina Zhuang, and Ting-Zhu Huang. Spatial-spectral-temporal connective tensor network decomposition for thick cloud removal. *ISPRS Journal of Photogrammetry and Remote Sensing*, 199:182–194, 2023. 1

[40] Yu-Bang Zheng, Ting-Zhu Huang, Xi-Le Zhao, Tai-Xiang Jiang, Teng-Yu Ji, and Tian-Hui Ma. Tensor N-tubal rank and its convex relaxation for low-rank tensor recovery. *Information Sciences*, 532:170–189, 2020. 5

[41] Yu-Bang Zheng, Ting-Zhu Huang, Xi-Le Zhao, Qibin Zhao, and Tai-Xiang Jiang. Fully-connected tensor network decomposition and its application to higher-order tensor completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11071–11078, 2021. 2, 3, 4, 6, 7

[42] Yu-Bang Zheng, Ting-Zhu Huang, Xi-Le Zhao, and Qibin Zhao. Tensor completion via fully-connected tensor network decomposition with regularized factors. *Journal of Scientific Computing*, 92(8):1–35, 2022. 2