STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **13**, May 2025, pp 2119–2135.
Published online in International Academic Press (www.IAPress.org)

Proximal Alternating Linearized Minimization Algorithm for Sparse Tensor Train Decomposition

Zhenlong Hu, Zhongming Chen *

Department of Mathematics, School of Sciences, Hangzhou Dianzi University, Hangzhou 310018, China

Abstract We address the sparse tensor train (TT) decomposition problem by incorporating an L1-norm regularization term. To improve numerical stability, orthogonality constraints are imposed on the problem. The tensor is expressed in the TT format, and the proximal alternating linearized minimization (PALM) algorithm is employed to solve the problem. Furthermore, we verify that the objective function qualifies as a Kurdyka-Łojasiewicz (KL) function and provide a convergence analysis. Numerical experiments on synthetic data and real data also demonstrate the efficiency of the proposed algorithm.

Keywords TT decomposition, L1-norm, PALM, KL function

AMS 2010 subject classifications 15A69, 68W01, 49M27

DOI: 10.19139/soic-2310-5070-2440

1. Introduction

Tensors, which are multidimensional arrays, are extensively utilized in fields such as computer vision [23], signal processing [25], and data mining [9]. There exist several tensor decomposition formats, including CP decomposition [17], Tucker decomposition [22], and TT decomposition [15]. Among these, TT decomposition offers the advantages of linear growth in storage complexity with respect to dimension and enhanced reliability, making it particularly suitable for high-order problems. In 2011, Oseledets proposed the TT decomposition, which represents a tensor as a sequence of third-order tensors. Subsequently, Zhang et al. [30] and Bigoni et al. [7] further developed the TT decomposition. There are different algorithms for computing TT decomposition including TT-SVD, TT-ALS, and so on [28].

It is noteworthy that in many problems, both sparsity constraints and orthogonality constraints play significant roles. Sparsity constraints facilitate feature extraction. For example, in signal transmission where we aim to reconstruct received signals into original signals, since the original signal contains a large number of zero elements, we can impose sparsity constraints on variables to derive useful solutions. This constitutes a compressed sensing problem [8]. Similarly, in video processing where one seeks to extract static components from video footage, by considering dynamic parts as noise, we can decompose the video information matrix into the sum of a low-rank matrix and a noise matrix, requiring the noise matrix to be sparse. This represents a Robust Principal Component Analysis (RPCA) problem [10][13, Section3.8]. Orthogonal constraints contribute to enhancing feature independence and improving numerical stability. For instance, in Principal Component Analysis (PCA), where we aim to find the projection of high-dimensional data points onto a low-dimensional subspace, requiring the bases of this low-dimensional subspace to be orthogonal helps reduce redundant information in the data and

*Correspondence to: Zhongming Chen (Email: zmchen@hdu.edu.cn). Department of Mathematics, School of Sciences, Hangzhou Dianzi University, Hangzhou 310018, China.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

enhances feature independence [14]. Compared with Non-negative Matrix Factorization (NMF), Orthogonal Non-negative Matrix Factorization (ONMF) demonstrates improved clustering performance through such orthogonal constraints [26]. In addition, significant constraints such as non-negative constraints are also crucial. Naturally, these constraints have been incorporated into fundamental tensor decomposition models. For instance, in terms of non-negativity constraints, Max Welling et al. (2001) introduced positivity constraints into tensor factorization [24]. Subsequently, Y. D. Kim et al. (2007) incorporated non-negativity constraints into the Tucker decomposition model, significantly improving the interpretability of the resulting factors [27]. Regarding sparsity constraints, C. Pan and C. Ling et al. (2020) imposed sparse constraints on a Tucker decomposition-based tensor completion model to promote sparse core tensor generation [6]. In 2022, H. Kuang enhanced image recognition accuracy by integrating sparsity constraints into a non-negative TT decomposition framework [12]. Additionally, to address the non-uniqueness inherent in certain tensor decompositions, L. De Lathauwer et al. (2004) introduced orthogonal constraints into tensor decomposition models to improve numerical stability [21]. However, the existing TT decomposition models do not account for the sparsity and orthogonality of TT tensors.

In this paper, we consider the tensor decomposition problem in the TT format. Similar with NMF for the task of image recognition, we can consider the first $d-1$ TT-cores as a set of bases for the recognition space, and treat the last TT-core as the representation vector of objects under these bases. To achieve better feature selection (e.g., highlighting eye and nose information in images), we desire this representation vector to be sparse. Therefore, we choose to impose sparsity constraints on the last TT-core. Furthermore, due to the non-uniqueness of the TT decomposition [3], we impose orthogonality constraints on the other TT-cores to enhance numerical stability, referring to the case of Tucker decomposition [28, Section 2.3]. Additionally, there is another reason for applying orthogonal constraints to the other TT-cores: when performing object classification, we want the bases of the recognition space we select to be orthogonal to reduce redundant information in the data. This can be referenced to the case in PCA [14]. Specifically, the model considered in this paper is as follows:

$$\begin{aligned} \min \quad & \frac{1}{2}\|\Phi(\mathcal{X}_1,...,\mathcal{X}_d)-\Gamma\|_F^2+\mu\|\mathcal{X}_d\|_1 \\ \text{s.t.} \quad & L^T(\mathcal{X}_j)L(\mathcal{X}_j)=I_{r_j}, \quad j\in[d-1], \\ & \mathcal{X}_1\in\mathbb{R}^{r_0\times n_1\times r_1},...,\mathcal{X}_d\in\mathbb{R}^{r_{d-1}\times n_d\times r_d}, \end{aligned} \tag{1.1}$$

where $\Gamma\in\mathbb{R}^{n_1\times n_2\times\cdots\times n_d}$ is the target tensor, $\mu\in\mathbb{R}$ is the regularization parameter. The operator $\Phi(\cdot)$ maps the TT-cores to a tensor and $L(\cdot)$ denotes the left unfolding of a third-order tensor, which will be introduced in the following subsections.

For this type of optimization problem with a block-separable structure, the PALM algorithm proposed by Jérôme Bolte et al. is an effective algorithm [16]. The PALM algorithm selects one block in a cyclic manner and solves the corresponding subproblem by approximating the differentiable term f with a first-order estimate. In this paper, we apply the PALM algorithm to solve the problem (1.1). The numerical experiments are also presented to show the efficiency of the propsoed algorithm.

1.1. Contributions

The main contributions of this paper are as follows:

- A new model for TT decomposition with orthogonal constraints and sparse constraints is proposed, and the PALM algorithm is utilized for solving it.
- Based on the objective function being a KL function, we present a convergence analysis.

1.2. Notation

Throughout this paper, scalars are denoted by lower case letters e.g. a,b,c; vectors are represented by bold lower case letters e.g. $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$; matrices are indicated by capital letters, e.g. A,B,C; tensors of order 3 or higher are expressed by calligraphic letters e.g. $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$. The Kronecker product is denoted as $\otimes$. For any positive integer n, denote $[n]=\{1,2,\dots,n\}$ and let I_n be the identity matrix of size $n\times n$. We use the multi-index $\overline{i_1i_2\cdots i_d}$ to denote the element in $\left[\prod_{k=1}^{d} n_k\right]$ such that

$$\overline{i_1 i_2 \cdots i_d} = i_1 + (i_2 - 1)n_1 + \cdots + (i_d - 1)n_1 \cdots n_{d-1}$$

for any $i_j \in [n_j]$ and $j \in [d]$.

For any 3rd-order tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the ith slice of $\mathcal{A}$ is denoted by $\mathcal{A}(i) \in \mathbb{R}^{n_1 \times n_3}$ for $i \in [n_2]$, the left unfolding $L(\mathcal{A}) \in \mathbb{R}^{n_1 n_2 \times n_3}$ is defined as $L(\mathcal{A})(\overline{i_1 i_2}, i_3) = \mathcal{A}(i_1, i_2, i_3)$ and the right unfolding $R(\mathcal{A}) \in \mathbb{R}^{n_1 \times n_2 n_3}$ is defined as $R(\mathcal{A})(i_1, \overline{i_2 i_3}) = \mathcal{A}(i_1, i_2, i_3)$. Denoted by $L^{-1}(\cdot)$ the inverse operation of $L(\cdot)$ [18]. For any tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$, the kth unfolding matrix $\mathcal{A}_{\langle k \rangle} \in \mathbb{R}^{n_1 \cdots n_k \times n_{k+1} \cdots n_d}$ is defined as

$$\mathcal{A}_{\langle k \rangle}(\overline{i_1 \cdots i_k}, \overline{i_{k+1} \cdots i_d}) = \mathcal{A}(i_1, i_2, \ldots, i_d)$$

where $k \in [d]$. For any two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$, the inner product is defined as

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_d=1}^{n_d} \mathcal{A}(i_1, i_2, \ldots, i_d)\mathcal{B}(i_1, i_2, \ldots, i_d).$$

The L1-norm of $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ is defined as

$$\|\mathcal{A}\|_1 = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_d=1}^{n_d} |\mathcal{A}(i_1, i_2, \ldots, i_d)|,$$

and the Fobenius norm of $\mathcal{A}$ is defined as $\|\mathcal{A}\|_F = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$.

For any matrix $A \in \mathbb{R}^{m \times n}$, the matrix $[A]_+ \in \mathbb{R}^{m \times n}$ is the nonnegative part of A defined as

$$[A]_+(i,j) = \max(A(i,j), 0),$$

the matrix $|A| \in \mathbb{R}^{m \times n}$ is the component-wise absolute values of A defined as

$$|A|(i,j) = |A(i,j)|,$$

and the matrix $sgn(A) \in \mathbb{R}^{m \times n}$ is the component-wise sign function of A defined as

$$sgn(A)(i,j) = \begin{cases} 1, & A(i,j) > 0, \\ 0, & A(i,j) = 0, \\ -1, & A(i,j) < 0. \end{cases}$$

For two matrices $A, B \in \mathbb{R}^{m \times n}$, the *Hadamard product* $A \odot B \in \mathbb{R}^{m \times n}$ is defined as

$$(A \odot B)(i,j) = A(i,j)B(i,j).$$

The notations above are summarized in Table 1.

1.3. Organization

The organization of this paper is as follows. In Section 2, we introduce some preliminaries. The PALM algorithm for solving problem (3.1) is proposed in Section 3, and the convergence analysis is presented in Section 4. In Section 5, we evaluate the algorithm's performance on both synthetic data and real facial dataset. Finally, the conclusions are drawn in Section 6.

Table 1. Description of notations.

Notation	**Meaning**
a	Scalar
$\boldsymbol{a}$	Vector
A	Matrix
$\mathcal{A}$	dth-order tensor $(d \geq 3)$
$[n]$	The set $\{1, 2, \ldots, n\}$
$\otimes$	Kronecker product
I_n	Identity matrix of size $n \times n$
$\mathcal{A}(i)$	The ith slice of 3rd-order tensor $\mathcal{A}$
$L(\mathcal{A})$	The left unfolding of 3rd-order tensor $\mathcal{A}$
$R(\mathcal{A})$	The right unfolding of 3rd-order tensor $\mathcal{A}$
$L^{-1}(\cdot)$	The inverse operation of $L(\cdot)$
$\mathcal{A}_{\langle k\rangle}$	The kth unfolding matrix of tensor $\mathcal{A}$
$\\|\cdot\\|_1$	L1-norm
$\\|\cdot\\|_F$	Frobenius norm
$[A]_+$	The nonnegative part of matrix A
$\|A\|$	The component-wise absolute values of matrix A
$sgn(A)$	The component-wise sign function of matrix A
$A \odot B$	The *Hadamard product* of matrices A and B

2. Preliminaries

2.1. TT Decomposition

For any $\mathcal{X} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_d}$, the TT decomposition factorizes $\mathcal{X}$ into a series of third-order tensors $\{\mathcal{X}_1, ..., \mathcal{X}_d\}$ such that

$$\mathcal{X}(i_1, i_2, \ldots, i_d) = \mathcal{X}_1(i_1)\mathcal{X}_2(i_2)\cdots\mathcal{X}_d(i_d), \tag{2.1}$$

where $\mathcal{X}_k \in \mathbb{R}^{r_{k-1}\times n_k\times r_k}$, $k \in [d]$ are called TT-cores. To make the matrix-by-matrix product a scalar, we assume $r_0 = r_d = 1$. The tensor $\mathcal{X}$ is also denoted by

$$\mathcal{X} = \Phi(\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_d), \tag{2.2}$$

where the notation Φ maps a series of third-order tensors $(\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_d)$ to $\mathcal{X}$ through the product defined in (2.1). The TT-rank of $\mathcal{X}$ is defined as the smallest $\boldsymbol{r} = (r_0, r_1, \ldots, r_d)$ that allows a TT decomposition with core tensors $\mathcal{X}_k \in \mathbb{R}^{r_{k-1}\times n_k\times r_k}$ for $k \in [d]$. It has been shown that the TT-rank of $\mathcal{X}$ is equal to $\boldsymbol{r}$ if and only if $rank(L(\mathcal{X}_k)) = r_k$ and $rank(R(\mathcal{X}_k)) = r_{k-1}$ for $k \in [d]$ [18].

The interface matrices $\mathcal{X}_{\leq k} \in \mathbb{R}^{n_1\cdots n_k\times r_k}$ and $\mathcal{X}_{\geq k} \in \mathbb{R}^{n_k\cdots n_d\times r_{k-1}}$ are defined as

$$\mathcal{X}_{\leq k} = reshape(\Phi(\mathcal{X}_1, ..., \mathcal{X}_k), \prod_{i=1}^{k} n_i, r_k)$$

and

$$\mathcal{X}_{\geq k} = \left(reshape(\Phi(\mathcal{X}_k, ..., \mathcal{X}_d), r_{k-1}, \prod_{i=k}^{d} n_i)\right)^T$$

for $k \in [d]$, where the $reshape$ function rearranges the elements in the input tensor into a tensor with specified shape. Denote $\mathcal{X}_{\leq 0} = \mathcal{X}_{\geq d+1} = 1$.

According to the definitions of $\mathcal{X}_{\leq k}$ and $\mathcal{X}_{\geq k}$, we have the following recursive formulas:

$$\begin{aligned}\mathcal{X}_{\leq k} &= (I_{n_k} \otimes \mathcal{X}_{\leq k-1})L(\mathcal{X}_k),\\ \mathcal{X}_{\geq k} &= (\mathcal{X}_{\geq k+1} \otimes I_{n_k})R(\mathcal{X}_k)^T.\end{aligned} \tag{2.3}$$

By the definition of unfolding matrices, we have

$$\mathcal{X}_{\langle k\rangle} = \mathcal{X}_{\leq k}\mathcal{X}_{\geq k+1}^T, \quad \forall\, k \in [d]. \tag{2.4}$$

2.2. *The KL Property*

In this subsection, we give a brief introduction of the KL property which plays a crucial role in the convergence analysis of the PALM algorithm.

A proper and lower semicontinuous function $h : \mathbb{R}^d \to (0, +\infty]$ is said to have the KL property at $x \in dom\, \partial h := \{y \in \mathbb{R}^d : \partial h \neq \emptyset\}$ if there exists $\eta \in (0, +\infty]$, a neighborhood U of x and a continuous concave function $\varphi : [0, \eta) \to [0, \infty)$ such that

(i) $\varphi(0) = 0$;

(ii) φ is C^1 on $(0, \eta)$ and continuous at 0;

(iii) for all $s \in (0, \eta)$, $\varphi^{'} > 0$;

(iv) for all $y \in U \cap \{y \in \mathbb{R}^d : h(x) < h(y) < h(x) + \eta\}$, there exists $\varphi^{'}(h(y) - h(x))dist(0, \partial h(y)) \leq 1$, where $dist(0, \partial h(y)) = \inf\{\|z\| : z \in \partial h(y)\}$ and ∂ denotes the subdifferential.

The function h is defined as a KL function if h satisfy the KL property at each point of $dom\, \partial h$ [16, 20].

3. Model and Algorithm

In this paper, we investigate the problem of sparse TT decomposition, which can be formally expressed as problem (1.1). This problem aims to decompose a high-order sparse tensor into a sequence of third-order core tensors while preserving its sparsity structure. Specifically, the sparse TT decomposition problem could be equivalently formulated as:

$$\min_{\mathcal{X}_1 \in \mathbb{R}^{r_0 \times n_1 \times r_1}, \ldots, \mathcal{X}_d \in \mathbb{R}^{r_{d-1} \times n_d \times r_d}} F(\mathcal{X}_1, ..., \mathcal{X}_d) = f(\mathcal{X}_1, ..., \mathcal{X}_d) + \sum_{j=1}^{d} g_j(\mathcal{X}_j), \tag{3.1}$$

where

$$\begin{aligned}f(\mathcal{X}_1, ..., \mathcal{X}_d) &= \frac{1}{2}\|\Phi(\mathcal{X}_1, ..., \mathcal{X}_d) - \Gamma\|_F^2,\\ g_j(\mathcal{X}_j) &= \delta_{\mathcal{M}_j}(\mathcal{X}_j), \quad j \in [d-1],\\ g_d(\mathcal{X}_d) &= \mu\|\mathcal{X}_d\|_1.\end{aligned} \tag{3.2}$$

The function $\delta_{\mathcal{M}_j}$: $\mathbb{R}^{r_{j-1} \times n_j \times r_j} \to (\infty, +\infty]$ is the indicator function defined by

$$\delta_{\mathcal{M}_j}(\mathcal{X}) = \begin{cases} 0, & \text{if } \mathcal{X} \in \mathcal{M}_j,\\ +\infty, & \text{otherwise,}\end{cases}$$

and the set $\mathcal{M}_j$ is defined as $\mathcal{M}_j := \{\mathcal{X}_j \in \mathbb{R}^{r_{j-1} \times n_j \times r_j} : L^T(\mathcal{X}_j)L(\mathcal{X}_j) = I_{r_j}\}$ for $j \in [d-1]$.

For problem (3.1), we apply the PALM algorithm which alternates between updating blocks of variables using proximal operators and linearized approximations to efficiently solve non-convex and non-smooth problems [16]. For simplicity, denote $\mathcal{X}^k = (\mathcal{X}_1^k, \mathcal{X}_2^k, \ldots, \mathcal{X}_d^k)$ and

$$\begin{aligned}\mathcal{X}^{k,0} &= (\mathcal{X}_1^k, \mathcal{X}_2^k, \ldots, \mathcal{X}_d^k),\\ \mathcal{X}^{k,1} &= (\mathcal{X}_1^{k+1}, \mathcal{X}_2^k, \ldots, \mathcal{X}_d^k),\\ &\cdots\\ \mathcal{X}^{k,d} &= (\mathcal{X}_1^{k+1}, \mathcal{X}_2^{k+1}, \ldots, \mathcal{X}_d^{k+1}).\end{aligned}$$

Clearly, $\mathcal{X}^{k,0}=\mathcal{X}^k$ and $\mathcal{X}^{k,d}=\mathcal{X}^{k+1}$. Denote by $\nabla_j f(\mathcal{X}_1,\ldots,\mathcal{X}_d)$ the gradient of f with respect to the jth component $\mathcal{X}_j$ when all $\mathcal{X}_i$, $i\neq j$, are fixed. Starting with any $\mathcal{X}^0\in\mathbb{R}^{r_0\times n_1\times r_1}\times\cdots\times\mathbb{R}^{r_{d-1}\times n_d\times r_d}$, the PALM algorithm generates a sequence $\{\mathcal{X}^k\}_{k\in\mathbb{N}}$ via the following successively scheme:

$$\mathcal{X}_j^{k+1}=\underset{\mathcal{X}_j\in\mathbb{R}^{r_{j-1}\times n_j\times r_j}}{\arg\min}\ \langle\nabla_j f(\mathcal{X}^{k,j-1}),\mathcal{X}_j-\mathcal{X}_j^k\rangle+\frac{\overline{\tau}_j^k}{2}\|\mathcal{X}_j-\mathcal{X}_j^k\|^2+g_j(\mathcal{X}_j) \tag{3.3}$$

for $j\in[d]$, where $\overline{\tau}_j^k>\tau_j^k$ and τ_j^k is the Lipschitz modulus of $\nabla_j f(\mathcal{X}^{k,j-1})$.

Before solving these subproblems, we need to compute the gradient of f and estimate the Lipschitz moduli of ∇f with respect to the jth component $\mathcal{X}_j$, which will be shown in the following lemmas.

Lemma 3.1
Let the function f be defined in (3.2). The gradient of f with respect to the jth component at point $(\mathcal{X}_1,\ldots,\mathcal{X}_d)$ is

$$\nabla_j f(\mathcal{X}_1,\ldots,\mathcal{X}_d)=L^{-1}((I_{n_j}\otimes\mathcal{X}_{\le j-1}^T)(\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_d)-\Gamma)_{\langle j\rangle}\mathcal{X}_{\ge j+1}).$$

Proof
For any $\xi_j\in\mathbb{R}^{r_{j-1}\times n_j\times r_j}$, it is not difficult to see that

$$\begin{aligned}&\lim_{t\to0}\frac{f(\mathcal{X}_1,\ldots,\mathcal{X}_j+t\xi_j,\ldots,\mathcal{X}_d)-f(\mathcal{X}_1,\ldots,\mathcal{X}_d)}{t}\\&=\langle\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_d)-\Gamma,\Phi(\mathcal{X}_1,\ldots,\xi_j,\ldots,\mathcal{X}_d)\rangle\\&=\langle(\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_d)-\Gamma)_{\langle j\rangle},(I_{n_j}\otimes\mathcal{X}_{\le j-1})L(\xi_j)\mathcal{X}_{\ge j+1}^T\rangle\\&=\langle(I_{n_j}\otimes\mathcal{X}_{\le j-1}^T)(\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_d)-\Gamma)_{\langle j\rangle}\mathcal{X}_{\ge j+1},L(\xi_j)\rangle\\&=\langle L^{-1}((I_{n_j}\otimes\mathcal{X}_{\le j-1}^T)(\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_d)-\Gamma)_{\langle j\rangle}\mathcal{X}_{\ge j+1}),\xi_j\rangle,\end{aligned}$$

where the second equation follows from (2.3) and (2.4). So the conclusion holds. □

Lemma 3.2
Let the function f be defined in (3.2). For any fixed $\mathcal{X}_i\in\mathbb{R}^{r_{i-1}\times n_i\times r_i}$, $i\neq j$, the partial gradient $\nabla_j f(\mathcal{X}_1,\ldots,\mathcal{X}_d)$ with $j\in[d]$ is globally Lipschitz continuous with moduli $\|I_{n_j}\otimes\mathcal{X}_{\le j-1}^T\|_2^2\|\mathcal{X}_{\ge j+1}\|_F^2$, that is

$$\|\nabla_j f(\mathcal{X}_1,\ldots,\mathcal{X}_j,\ldots,\mathcal{X}_d)-\nabla_j f(\mathcal{X}_1,\ldots,\mathcal{X}_j',\ldots,\mathcal{X}_d)\|_F\le\|I_{n_j}\otimes\mathcal{X}_{\le j-1}^T\|_2^2\|\mathcal{X}_{\ge j+1}\|_F^2\|\mathcal{X}_j-\mathcal{X}_j'\|_F,$$

for any $\mathcal{X}_j,\mathcal{X}_j'\in\mathbb{R}^{r_{j-1}\times n_j\times r_j}$.

Proof
For any $\mathcal{X}_j,\mathcal{X}_j'\in\mathbb{R}^{r_{j-1}\times n_j\times r_j}$, we have

$$\begin{aligned}&\|\nabla_j f(\mathcal{X}_1,\ldots,\mathcal{X}_j,\ldots,\mathcal{X}_d)-\nabla_j f(\mathcal{X}_1,\ldots,\mathcal{X}_j',\ldots,\mathcal{X}_d)\|_F\\&=\|(I_{n_j}\otimes\mathcal{X}_{\le j-1}^T)(\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_j,\ldots,\mathcal{X}_d)-\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_j',\ldots,\mathcal{X}_d))_{\langle j\rangle}\mathcal{X}_{\ge j+1}\|_F\\&\le\|I_{n_j}\otimes\mathcal{X}_{\le j-1}^T\|_2\|(\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_j,\ldots,\mathcal{X}_d)-\Phi(\mathcal{X}_1,\ldots,\mathcal{X}_j',\ldots,\mathcal{X}_d))_{\langle j\rangle}\|_F\|\mathcal{X}_{\ge j+1}\|_F\\&=\|I_{n_j}\otimes\mathcal{X}_{\le j-1}^T\|_2\|(I_{n_j}\otimes\mathcal{X}_{\le j-1})L(\mathcal{X}_j-\mathcal{X}_j')\mathcal{X}_{\ge j+1}^T\|_F\|\mathcal{X}_{\ge j+1}\|_F\\&\le\|I_{n_j}\otimes\mathcal{X}_{\le j-1}^T\|_2^2\|\mathcal{X}_{\ge j+1}\|_F^2\|\mathcal{X}_j-\mathcal{X}_j'\|_F,\end{aligned}$$

which completes the proof. □

Corollary 3.1
If $\mathcal{X}_1,\ldots,\mathcal{X}_{d-1}$ are left-orthogonal, i.e., $\mathcal{X}_i\in\mathcal{M}_i$ for $i\in[d-1]$, the partial gradient $\nabla_j f(\mathcal{X}_1,\ldots,\mathcal{X}_d)$ is $\|\mathcal{X}_d\|_F^2$-Lipschitz for $j\in[d-1]$ and 1-Lipschitz for $j=d$.

Proof
According to [29], the matrix $\mathcal{X}_{\leq j}$ satisfies $\mathcal{X}_{\leq j}^T \mathcal{X}_{\leq j} = I_{n_j}$ for $j \in [d-1]$ since $\mathcal{X}_1, \ldots, \mathcal{X}_{d-1}$ are left-orthogonal. It follows that $\|I_{n_j} \otimes \mathcal{X}_{\leq j-1}^T\|_2 = 1$ for $j \in [d]$. On the other hand, we have $\|\mathcal{X}_{\geq j+1}\|_F = \|\mathcal{X}_d\|_F$ for $j \in [d-1]$ according to [15]. Combining Lemma 3.2, the conclusion holds immediately. □

Remark 3.1
The subsequent convergence analysis requires the Lipschitz modulus of $\nabla_j f(\mathcal{X}^{k,j-1})$ is larger than 0. In practice, we avoid this assumption by introducing a safeguard $v > 0$ and choosing the Lipschitz modulus τ_j^k as

$$\tau_j^k = \begin{cases} \max\{v, \|\mathcal{X}_d^k\|_F^2\}, & j \in [d-1], \\ \max\{v, 1\}, & j = d. \end{cases} \tag{3.4}$$

The parameter $\overline{\tau}_j^k$ in (3.3) is given by $\overline{\tau}_j^k = \gamma_j \tau_j^k$ where $\gamma_j > 1$ for $j \in [d]$.

In each iteration, we need to solve two types of subproblems. For $j \in [d-1]$, subproblem (3.3) becomes

$$\mathcal{X}_j^{k+1} = \mathop{\arg\min}_{\mathcal{X}_j \in \mathbb{R}^{r_{j-1} \times n_j \times r_j}} \langle \nabla_j f(\mathcal{X}^{k,j-1}), \mathcal{X}_j - \mathcal{X}_j^k \rangle + \frac{\overline{\tau}_j^k}{2} \|\mathcal{X}_j - \mathcal{X}_j^k\|^2 + \delta_{\mathcal{M}_j}(\mathcal{X}_j),$$

which is equivalent to solving

$$\mathcal{X}_j^{k+1} = \mathop{\arg\min}_{\mathcal{X}_j \in \mathcal{M}_j} \langle \nabla_j f(\mathcal{X}^{k,j-1}), \mathcal{X}_j - \mathcal{X}_j^k \rangle + \frac{\overline{\tau}_j^k}{2} \|\mathcal{X}_j - \mathcal{X}_j^k\|^2. \tag{3.5}$$

For $j = d$, subproblem (3.3) becomes

$$\mathcal{X}_d^{k+1} = \mathop{\arg\min}_{\mathcal{X}_d \in \mathbb{R}^{r_{d-1} \times n_d \times r_d}} \langle \nabla_d f(\mathcal{X}^{k,d-1}), \mathcal{X}_d - \mathcal{X}_d^k \rangle + \frac{\overline{\tau}_d^k}{2} \|\mathcal{X}_d - \mathcal{X}_d^k\|^2 + \mu \|\mathcal{X}_d\|_1. \tag{3.6}$$

For any $j \in [d]$, denote by $\mathcal{U}_j$ the linear transformation which is given by

$$\mathcal{U}_j(\mathcal{X}_j) = (\mathbf{0}, ..., \mathbf{0}, \mathcal{X}_j, \mathbf{0}., ..., \mathbf{0}),$$

where $\mathcal{X}_j$ is on the jth block. The PALM algorithm for sparse TT decomposition is summarized in Algorithm 1. In the following subsections, we will show how to solve subproblems (3.5) and (3.6), respectively.

3.1. Solving subproblem (3.5)

Subroblem (3.5) can be transformed into the following problem:

$$\begin{aligned} L(\mathcal{X}_j^{k+1}) &= \mathop{\arg\min}_{L(\mathcal{X}_j)^T L(\mathcal{X}_j) = I_{r_j}} \langle L(\nabla_j f(\mathcal{X}^{k,j-1})), L(\mathcal{X}_j) - L(\mathcal{X}_j^k) \rangle + \frac{\overline{\tau}_j^k}{2} \|L(\mathcal{X}_j) - L(\mathcal{X}_j^k)\|_F^2 \\ &= \mathop{\arg\min}_{L(\mathcal{X}_j)^T L(\mathcal{X}_j) = I_{r_j}} \frac{1}{2} \|L(\mathcal{X}_j) - L(\mathcal{X}_j^k - \frac{1}{\overline{\tau}_j^k} \nabla_j f(\mathcal{X}^{k,j-1}))\|_F^2. \end{aligned}$$

This is a classic Procrustes rotation problem, which could be solved in closed-form through singular value decomposition (SVD) [11, Algorithm 6.4.1]. Specifically, let $p = \min\{r_{j-1} n_j, r_j\}$ and suppose that $L(\mathcal{X}_j^k - \frac{1}{\overline{\tau}_j^k} \nabla_j f(\mathcal{X}^{k,j-1})) = U \Sigma V^T$ is the compact SVD of $L(\mathcal{X}_j^k - \frac{1}{\overline{\tau}_j^k} \nabla_j f(\mathcal{X}^{k,j-1}))$, where $U \in \mathbb{R}^{r_{j-1} n_j \times p}$, $\Sigma \in \mathbb{R}^{p \times p}$ and $V \in \mathbb{R}^{r_j \times p}$. It follows that

$$UV^T = \mathop{\arg\min}_{L(\mathcal{X}_j)^T L(\mathcal{X}_j) = I_{r_j}} \frac{1}{2} \|L(\mathcal{X}_j) - L(\mathcal{X}_j^k - \frac{1}{\overline{\tau}_j^k} \nabla_j f(\mathcal{X}^{k,j-1}))\|_F^2.$$

In other words, $L(\mathcal{X}_j^{k+1}) = UV^T$. The method for solving subproblem (3.5) is presented in Algorithm 2.

Algorithm 1: PALM algorithm for sparse TT decomposition (STT-PALM)

```
Input: Initial iterate $\mathcal{X}^0 = (\mathcal{X}_1^0, ..., \mathcal{X}_d^0) \in dom(F)$, maximum number of the iteration steps $k_{max}$, the
       safeguard $v > 0$,$\gamma_j > 1$ for $j \in [d]$
for $k = 0, 1, ..., k_{max}$ do
    Set $\mathcal{X}^{k,0} = \mathcal{X}^k$
    for $j = 1, ..., d-1$ do
        1. Compute $\tau_j^k$ by (3.4)
        2. $\overline{\tau}_j^k = \gamma_j \tau_j^k$, solve the subproblem (3.5)
        3. $\mathcal{X}^{k,j} = \mathcal{X}^{k,j-1} + \mathcal{U}_j(\mathcal{X}_j^{k+1} - \mathcal{X}_j^k)$
    end
    Compute $\tau_d^k$ by (3.4)
    $\overline{\tau}_d^k = \gamma_d \tau_d^k$, solve the subproblem (3.6)
    $\mathcal{X}^{k+1} = \mathcal{X}^{k,d-1} + \mathcal{U}_d(\mathcal{X}_d^{k+1} - \mathcal{X}_d^k)$
end
Output: The sequence $\{\mathcal{X}^k\}$
```

Algorithm 2: Solving subproblem (3.5)

```
Input: $\mathcal{X}_j^k$, $\nabla_j f(\mathcal{X}^{k,j-1})$, $\overline{\tau}_j^k$
1. Compute the compact SVD of $L(\mathcal{X}_j^k - \frac{1}{\overline{\tau}_j^k}\nabla_j f(\mathcal{X}^{k,j-1}))$ and save $U$ and $V$
2. $L(\mathcal{X}_j^{k+1}) = UV^T$
3. $\mathcal{X}_j^{k+1} = L^{-1}(L(\mathcal{X}_j^{k+1}))$
Output: $\mathcal{X}_j^{k+1}$
```

3.2. *Solving subproblems (3.6)*

Subroblem (3.6) can be transformed into the following problem:

$$\begin{aligned} L(\mathcal{X}_d^{k+1}) &= \underset{L(\mathcal{X}_d)}{\arg\min} \langle L(\nabla_d f(\mathcal{X}^{k,d-1})), L(\mathcal{X}_d) - L(\mathcal{X}_d^k)\rangle + \frac{\overline{\tau}_d^k}{2}\|L(\mathcal{X}_d) - L(\mathcal{X}_d^k)\|_F^2 + \mu\|L(\mathcal{X}_d)\|_1 \\ &= \underset{L(\mathcal{X}_d)}{\arg\min} \frac{\mu}{\overline{\tau}_d^k}\|L(\mathcal{X}_d)\|_1 + \frac{1}{2}\|L(\mathcal{X}_d) - L(\mathcal{X}_d^k - \frac{1}{\overline{\tau}_d^k}\nabla_d f(\mathcal{X}^{k,d-1}))\|_F^2. \end{aligned}$$

This problem has a closed-form solution [1, Example 6.8]

$$L(\mathcal{X}_d^{k+1}) = [|L(\mathcal{X}_d^k - \frac{1}{\overline{\tau}_d^k}\nabla_d f(\mathcal{X}^{k,d-1}))| - \frac{\mu}{\overline{\tau}_d^k}\mathbf{e}]_+ \odot sgn(L(\mathcal{X}_d^k - \frac{1}{\overline{\tau}_d^k}\nabla_d f(\mathcal{X}^{k,d-1}))). \tag{3.7}$$

where $\mathbf{e}$ is the all-one vector of proper size. The method for solving subproblem (3.6) is presented in Algorithm 3.

Algorithm 3: Solving subproblem (3.6)

```
Input: $\mathcal{X}_d^k$, $\nabla_d f(\mathcal{X}^{k,d-1})$, $\overline{\tau}_d^k$
1. Compute $L(\mathcal{X}_d^{k+1})$ by (3.7)
2. $\mathcal{X}_d^{k+1} = L^{-1}(L(\mathcal{X}_d^{k+1}))$
Output: $\mathcal{X}_d^{k+1}$
```

4. Convergence Analysis

In this section, we analyze the convergence of the PALM algorithm for the sparse TT decomposition under the framework of [16].

Lemma 4.1
The objective function $F(\mathcal{X}_1, ..., \mathcal{X}_d) = f(\mathcal{X}_1, ..., \mathcal{X}_d) + \sum_{j=1}^{d} g_j(\mathcal{X}_j)$ is a KL function, where $f, g_1, ..., g_d$ are defined in (3.2).

Proof
Since f is a real polynomial function, f is a semi-algebraic function. Since $\mathcal{M}_j$ is a semi-algebraic set, it follows that g_j is the indicator function of a semi-algebraic set for $j \in [d-1]$. Since g_d is the 1-norm of $\mathcal{X}_d$, g_d is a semi-algebraic function. Consequently, $f, g_1, g_2, \ldots, g_d$ are semi-algebraic functions, which implies that F is a semi-algebraic function. According to [16], F is also a KL function. □

Theorem 4.1
Let $\{\mathcal{X}^k\}_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 1. The sequence $\{F(\mathcal{X}^k)\}_{k\in\mathbb{N}}$ is convergent.

Proof
According to the well-known descent lemma [16, 4, 19], since $\nabla_j f(\mathcal{X}_1, ..., \mathcal{X}_d)$ is Lipschitz continuous, we have

$$f(\mathcal{X}^{k,j}) \leq f(\mathcal{X}^{k,j-1}) + \langle \mathcal{X}_j^{k+1} - \mathcal{X}_j^k, \nabla_j f(\mathcal{X}^{k,j-1})\rangle + \frac{\tau_j^k}{2}\|\mathcal{X}_j^{k+1} - \mathcal{X}_j^k\|_F^2 \tag{4.1}$$

for all $j \in [d]$ and $k \geq 0$.

For $j \in [d]$, by setting $\mathcal{X}_j = \mathcal{X}_j^k$ in (3.3), we obtain

$$\langle \mathcal{X}_j^{k+1} - \mathcal{X}_j^k, \nabla_j f(\mathcal{X}^{k,j-1})\rangle + \frac{\overline{\tau}_j^k}{2}\|\mathcal{X}_j^{k+1} - \mathcal{X}_j^k\|_F^2 + g_j(\mathcal{X}_j^{k+1}) \leq g_j(\mathcal{X}_j^k). \tag{4.2}$$

By adding (4.1) and (4.2) together, we obtain

$$\begin{aligned} F(\mathcal{X}^{k,j-1}) - F(\mathcal{X}^{k,j}) &\geq \frac{\overline{\tau}_j^k - \tau_j^k}{2}\|\mathcal{X}_j^{k+1} - \mathcal{X}_j^k\|_F^2 \\ &\geq \frac{(\gamma_j - 1)v}{2}\|\mathcal{X}_j^{k+1} - \mathcal{X}_j^k\|_F^2. \end{aligned} \tag{4.3}$$

Then we have

$$F(\mathcal{X}^k) - F(\mathcal{X}^{k+1}) \geq \frac{(\overline{\gamma} - 1)v}{2}\|\mathcal{X}^{k+1} - \mathcal{X}^k\|_F^2$$

by adding (4.3) from $j = 1$ to d, where $\overline{\gamma} = \min\{\gamma_1, \gamma_2, \ldots, \gamma_d\}$. Therefore, $\{F(\mathcal{X}^k)\}_{k\in\mathbb{N}}$ is a monotonically decreasing sequence. Since $F(\mathcal{X}^k) \geq 0$, it can be concluded that the sequence $\{F(\mathcal{X}^k)\}_{k\in\mathbb{N}}$ is convergent. □

It is noteworthy that the inclusion of orthogonal constraints and sparsity constraints in the model allows us to avoid the assumption that the sequence $\{\mathcal{X}^k\}$ is bounded, which is required in the convergence analysis of the PALM algorithm [16].

Lemma 4.2
The sequence $\{\mathcal{X}^k\}_{k\in\mathbb{N}}$ generated by Algorithm 1 is bounded.

Proof
Since the function F is coercive with respect to the last TT-core $\mathcal{X}_d$ and $\{F(\mathcal{X}^k)\}_{k\in\mathbb{N}}$ is a monotonically decreasing sequence, $\mathcal{X}_d^k$ is bounded. Considering that the TT-cores $\mathcal{X}_1^k, \mathcal{X}_2^k, \ldots, \mathcal{X}_{d-1}^k$ are left-orthogonal, we can derive that the sequence $\{\mathcal{X}^k\}_{k\in\mathbb{N}}$ is bounded. □

Lemma 4.3
For $j \in [d]$, there exists $\lambda_j^-, \lambda_j^+ > 0$ such that

$$\inf\{\tau_j^k : k \in \mathbb{N}\} \geq \lambda_j^-, \quad \sup\{\tau_j^k : k \in \mathbb{N}\} \leq \lambda_j^+. \tag{4.4}$$

Proof
Since we introduce a safeguard $v > 0$, it is evident that $\inf\{\tau_j^k : k \in N\} \geq v$. On the other hand, $\sup\{\tau_j^k : k \in N\} \leq \lambda_j^+$ can be obtained since f is smooth and the generated sequence $\{\mathcal{X}^k\}$ is bounded [16, Remark3(iv)]. □

Theorem 4.2
The sequence $\{\mathcal{X}^k\}_{k\in\mathbb{N}}$ generated by Algorithm 1 converges to a critical point of (3.1).

Proof
Since the function g_j is proper and lower semicontinuous for $j \in [d]$, f is differentiable and by Lemmas 4.1-4.3, the assumptions of [16] are satisfied. According to Theorem 1 of [16], the sequence $\{\mathcal{X}^k\}_{k\in\mathbb{N}}$ generated by Algorithm 1 converges to a critical point of problem. □

5. Experimental results

In this section, we test and verify the convergence of the proposed algorithm. We assign the same value to γ_j for $j \in [d]$, and represent it as γ, then we set $\gamma > 1$. In Section 5.1, we evaluate the impact of varying parameter settings on the performance of Algorithm 1 by using the synthetic data. In Section 5.2, we present the numerical results on face recognition using the ORL face dataset.

All experiments are performed in Matlab 2016b on a 2.7 GHz Intel Core i7 machine with 16 GB RAM. We utilize the MATLAB Tensor Toolbox [5] to conduct our experiments. The code from this paper is available at `https://github.com/zl-hu/Sparse-Tensor-Train-Decomposition`.

5.1. Synthetic Data

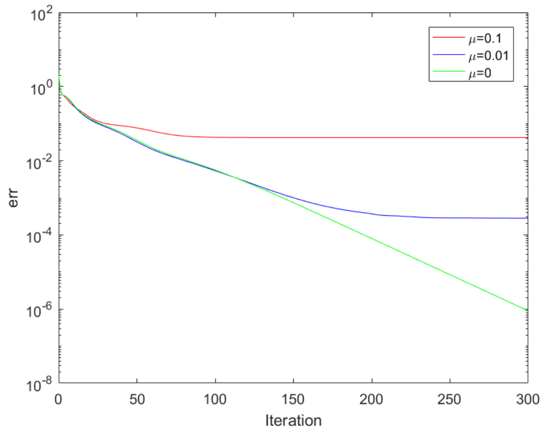
In this subsection, we test how various parameter selections influence the convergence performance of the algorithm using synthetic data. We employ the relative error, denoted as err, as a criterion to assess the algorithm's performance, where

$$err = \frac{\|\mathcal{X} - \mathcal{T}\|_F^2}{\|\mathcal{T}\|_F^2},$$

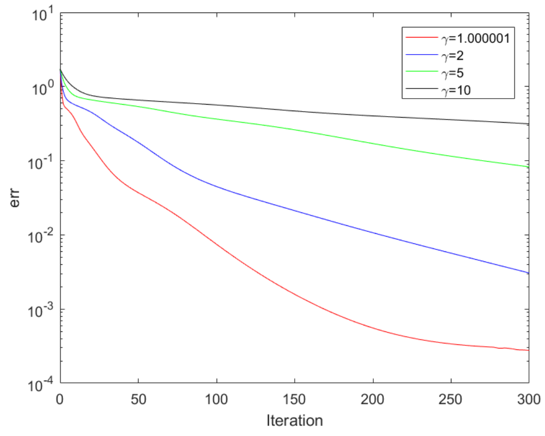
and set the maximum number of iteration $kmax = 300$.

The random target tensor of a given TT-rank used in this subsection is generated as follows: first we randomly generate the TT-core tensors $\{\mathcal{X}_1, ..., \mathcal{X}_d\}$ of proper size. Second, we use the MATLAB command $qr(\cdot)$, which performs the QR decomposition of a given matrix and returns the corresponding orthogonal matrix, on $L(\mathcal{X}_j)$ for $j \in [d-1]$ to make sure that the first $d-1$ TT-cores are left-orthogonal. The initial points are randomly generated in the same way.

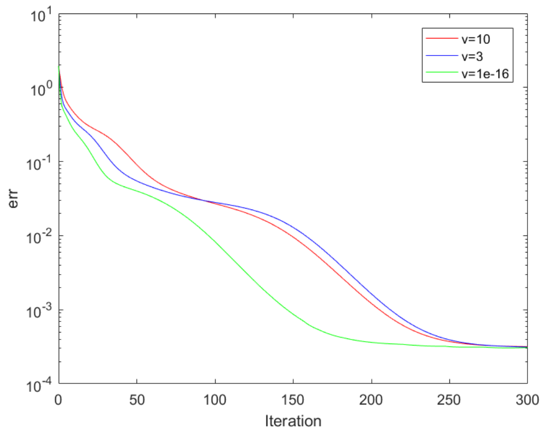
We randomly generate the tensor with size of (10 10 10 10) and TT-rank of (1 4 4 4 1) as the target tensor. Our initial analysis investigates the impact of different regularization parameters μ on the algorithm's convergence performance. We set $\gamma = 1.000001$, $v = 10^{-16}$ and vary the regularization parameter μ with values of 0.1, 0.01, and 0. The relative error as a function of the iteration count for each μ is plotted in Figure 1(a). Then we examine the effects of different γ values on convergence. With the regularization parameter μ fixed at 0.01 and v fixed at 10^{-16}, we assign γ values of 1.000001, 2, 5, and 10. The relative error as a function of iteration number for varying γ is presented in Figure 1(b). Finally, we investigate the impact of parameter v on algorithmic convergence characteristics. Under the configuration with fixed parameters $\mu = 0.01$, $\gamma = 1.000001$, three distinct values $v \in \{10, 3, 10^{-16}\}$ are systematically evaluated, revealing parameter sensitivity patterns through comparative convergence diagnostics. The relative error as a function of iteration number for varying v is presented in Figure 1(c).



(a) Different μ



(b) Different γ



(c) Different v

Figure 1. The err for the algorithm of different parameters

Figure 1(a) illustrates that the relative error diminishes with a decrease in the regularization parameter. This reveals a fundamental issue regarding structural discrepancy (hereafter termed 'gap'). The specific target tensor we select possesses a low-rank structure where the first $d-1$ TT-cores have orthogonalized unfoldings, with the last TT-core being randomly generated. When $\mu = 0$, the algorithm naturally produces tensors exhibiting these structural properties, thereby achieving optimal convergence performance. However, increasing μ imposes additional constraints on the generated tensors, which induces a structural gap between the algorithmic output and the target tensor. Therefore, we consider another category of target tensors generated using the `randn` function in MATLAB with dimensions [10 10 10 10] to investigate the impact of parameter μ on the algorithm. Since the err curves corresponding to different μ values exhibit minimal discrepancies during the initial phase, Figure 2 provides a detailed presentation of the algorithm's err trajectories specifically during the final 100 iterations. For this category of synthetic data and real-world facial data in Section 5.2, $\mu = 0$ does not necessarily yield the best modeling performance. Figure 1(b) demonstrates that the algorithm's convergence speed increases as γ decreases, achieving the most rapid convergence at $\gamma = 1.000001$. As demonstrated in [16, Secrtion 3.1], the parameter $\frac{1}{\tau_j^k}$ functions analogously to the step size in gradient descent algorithms. To accelerate the algorithm's convergence, it is advisable to select the largest possible step size within the algorithm's allowable range, which corresponds to making γ approach 1 as closely as possible. Figure 1(c) demonstrates that the algorithm's convergence speed increases as v decreases, achieving the most rapid convergence at $v = 10^{-16}$. This phenomenon arises because, according to equation (3.4), setting v to values exceeding both $\|\mathcal{X}_d^k\|_F^2\}$ and 1 will result in an enlargement of τ_j^k, thereby leading to an increase in $\overline{\tau}$. Subsequent behavioral patterns can be analyzed by referring to the discussion of Figure 1(b).

In summary, for parameter μ, we recommend first determining the maximum value that ensures stable numerical performance of the algorithm through empirical testing, then selecting a midrange value between this upper bound and zero as the operational μ value. Regarding parameter γ, it should ideally be configured as close to unity as possible. The setting $\gamma = 1.000001$ implemented in our study already satisfies this proximity requirement, as further reduction would yield only negligible improvement in convergence performance. Thus, $\gamma = 1.000001$ also constitutes a reasonable configuration. For parameter v, since minimizing its magnitude does not increase computational overhead, we advocate setting it to the smallest practical value, such as $v = 10^{-16}$.

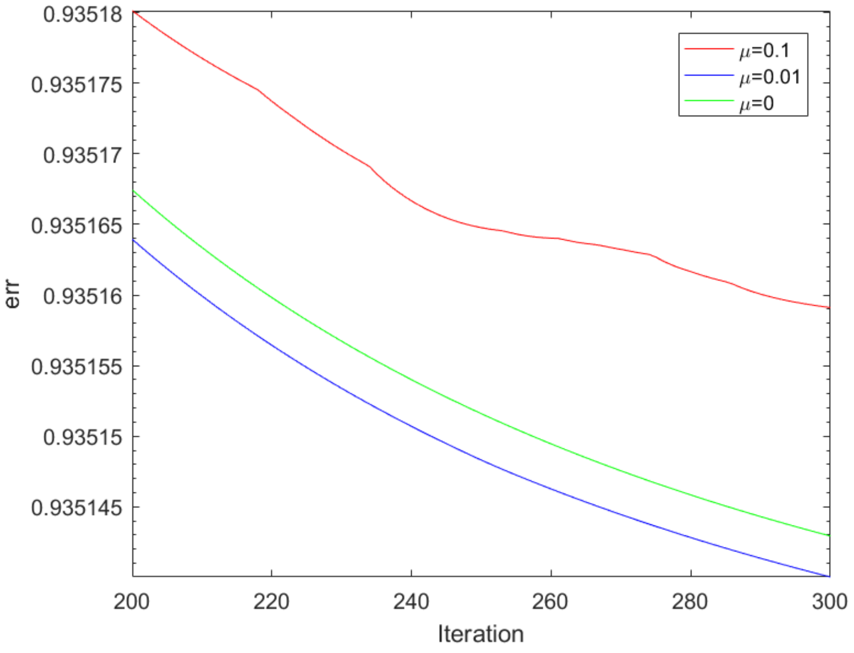


Figure 2. The err for the algorithm of different μ for the new category of synthetic data over the final 100 iterations

5.2. *Face Recognition*

In this subsection, we conduct facial recognition experiments on the ORL face dataset [2]. The ORL dataset contains facial images of 40 subjects, each represented by 10 distinct images. These images are originally 112×92 pixels in resolution. To facilitate processing, we employ MATLAB's $imresize$ function to compress the images to 28×23 pixels. Subsequently, we normalize the pixel values by dividing each by 255, resulting in a preprocessed dataset represented as a tensor with dimensions (28 23 400).

We divide the data into training and test sets according to a certain ratio. The training set tensor is decomposed into the TT format using the testing algorithm. The first two TT-cores are used as the basis in image space, while the last TT-core serves as the representation vector under the basis. We set $\gamma = 1.000001$, $v = 10^{-16}$, $kmax = 300$ and compute the representation vectors of the test set under the learned basis and determine the classification results by calculating the distance between the representation vectors of the test set and those of the training set.

Table 2. Comparison of all tested algorithms

Algorithm	STT-PALM	SNTT-MUR	NOSTT-PALM	PTF
Accuracy	0.8850	0.8550	0.8250	0.8600

The comparative algorithms in this study include: the multiplicative update rules algorithm based on sparse nonnegative TT decomposition (SNTT-MUR) from [12], and the positive tensor factorization (PTF) algorithm from [24]. To demonstrate the performance improvement brought by orthogonality in our method, we removed the orthogonality constraints while keeping all other conditions unchanged, thereby designing a sparse TT decomposition PALM algorithm without orthogonality constraints (NOSTT-PALM) as one of the baselines. Section 5.2.1 presents the recognition accuracy of these algorithms in facial recognition experiments, while Section 5.2.2-5.2.4 provide detailed comparisons with the SNTT-MUR algorithm under varying parameter conditions.

5.2.1. Comparison of tested algorithms in terms of accuracy In this section, we present the performance of all candidate algorithms in facial recognition experiments. For each subject, 5 facial images are randomly selected to constitute the training set, and the other 5 images are for the test set. We set $\mu = 0.01$, TT-rank=(1 20 40 1) in STT-PALM, SNTT-MUR and NOSTT-PALM algorithms. The rank in PTF algorithm is set to 40. We present the recognition accuracy as a function of iteration count for various values of μ in Figure 3 and the recognition accuracy of each algorithm at the final iteration in Table 2.

Figure 3 and Table 2 demonstrate that among all compared algorithms, STT-PALM algorithm achieves higher face recognition accuracy. Notably, only STT-PALM algorithm exhibits rapid accuracy improvement during the initial few iterations, while other algorithms show gradual accuracy increases with iteration steps. The comparison with the NOSTT-PALM algorithm confirms the performance enhancement brought by incorporating orthogonality constraints.

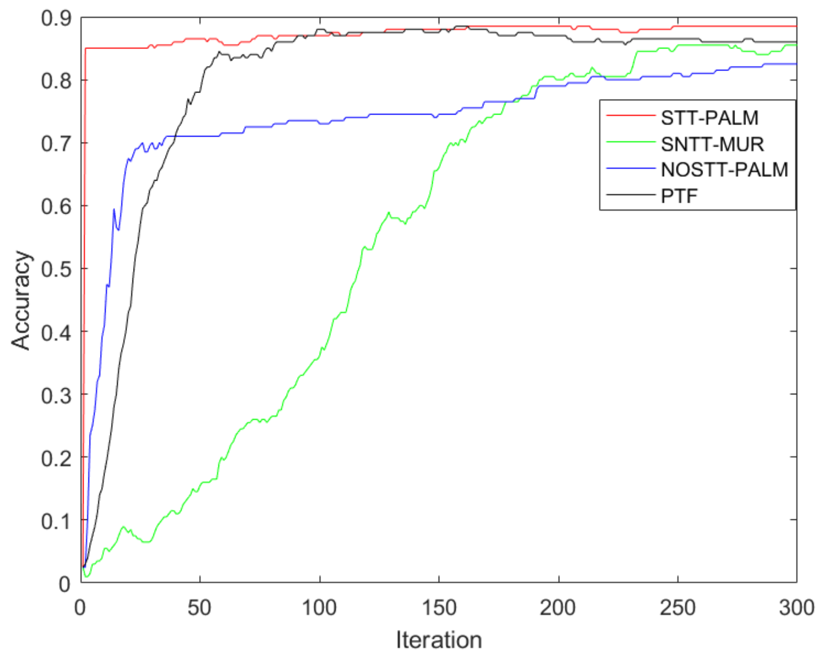


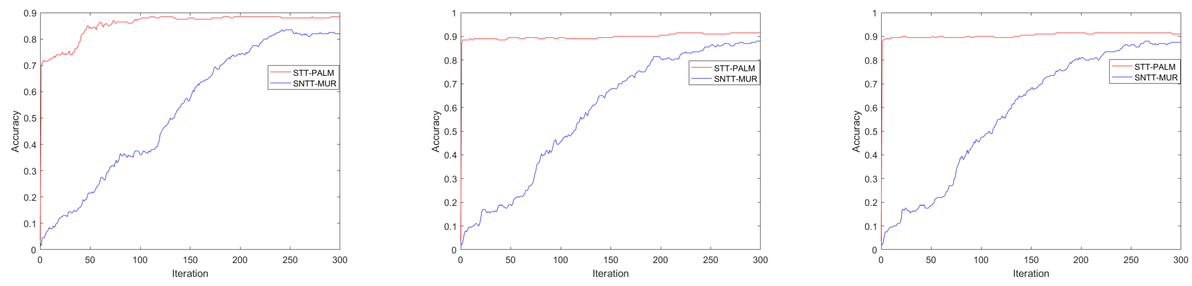
Figure 3. The recognition accuracy of all tested algorithms.

Table 3. Comparison of tested algorithms with different μ

Algorithm	$\mu = 0.1$	$\mu = 0.01$	$\mu = 0.001$
SNTT-MUR	0.8200	0.8800	0.8750
STT-PALM	0.8850	0.9150	0.9100

5.2.2. Different values of the regularization parameter We examine the effects of varying regularization parameters on the tested algorithms' recognition accuracy. For each subject, 5 facial images are randomly selected to constitute the training set, and the other 5 images are for the test set. Following preprocessing, the dimensions of both the training and test sets are (28 23 200). Regularization parameters are set at 0.1, 0.01, and 0.001. Iterations are performed for both algorithms, employing a range of randomly generated initial points that conformed to the algorithmic criteria.

We present the recognition accuracy as a function of iteration count for various values of μ in Figure 4. Additionally, the recognition accuracy of each algorithm at the final iteration is summarized in Table 3.



(a) $\mu = 0.1$ (b) $\mu = 0.01$ (c) $\mu = 0.001$

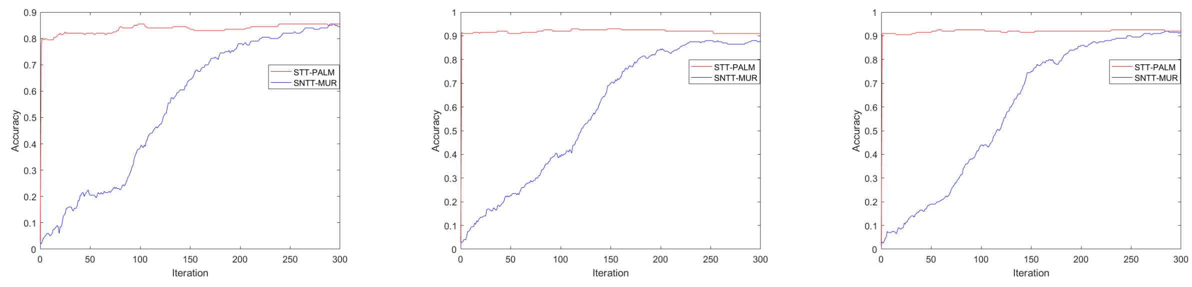
Figure 4. The recognition accuracy for tested algorithms with different μ

The data presented in Figure 4 and Table 3 indicate that the recognition accuracy of the evaluated algorithms improves with a decrease in the regularization parameter μ from 0.1, peaking at a μ value of 0.01, beyond which it stabilizes. This phenomenon arises because a relatively large μ value may lead the model to focus excessively on data sparsity patterns, thereby impeding the model's ability to capture essential facial characteristics and compromising detection accuracy. Conversely, a relatively small μ value could result in insufficient attention allocation to critical facial regions (notably eyes and nose) during feature encoding, whereas adversely affecting recognition performance. We recommend running multiple tests in practice to determine a mid-range μ value between empirically observed extremes.

5.2.3. Different vaules of TT-rank We examine the effect of varying TT-ranks on tested algorithms' recognition accuracy. For each subject, 5 facial images are randomly selected for the training set, and the other 5 images are for the test set. Following preprocessing, the dimensions of both the training and test sets are (28 23 200). TT-ranks are configured to (1 20 20 1), (1 20 40 1) and (1 20 60 1) respectively. The regularization parameter μ is established at 0.01, and iterations for both algorithms proceed from various randomly generated initial points that satisfy the algorithmic criteria.

We present the recognition accuracy as a function of the number of iterations for various TT-ranks in Figure 5, and Table 4 displays the final iteration recognition accuracy for each algorithm.

As shown in Figure 5 and Table 4, the recognition accuracy of the evaluated algorithms improves as the third TT-rank index increases. This is because, according to the TT-rank calculation formula in [18, Section 1.1], the TT-rank of the preprocessed training set is (1 28 200 1). Our experiments indicate that the closer the third index of the TT-rank is to 200, the higher recognition accuracy the algorithm achieves on the training set. However, it should



(a) TT-rank=(1 20 20 1) (b) TT-rank=(1 20 40 1) (c) TT-rank=(1 20 60 1)

Figure 5. The recognition accuracy for tested algorithms with different TT-ranks

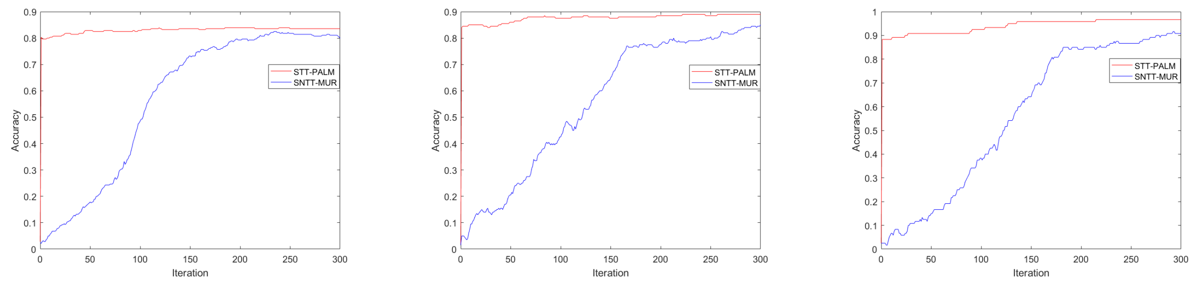
Table 4. Comparison of tested algorithms with different TT-ranks

Algorithm	rank=(1 20 20 1)	rank=(1 20 40 1)	rank=(1 20 60 1)
SNTT-MUR	0.8450	0.8800	0.9150
STT-PALM	0.8550	0.9100	0.9200

be noted that as the third TT-rank index increases, the corresponding matrix dimensions in the algorithm expand proportionally, which may lead to prohibitively long computational time. Therefore, in practical applications we can adopt an intermediate value that achieves an equilibrium between computational cost and recognition accuracy.

5.2.4. Different values of train set size We examine the effects of varying training set sizes on tested algorithms' recognition accuracy. For each subject, we randomly select 3, 5, and 7 facial images to create the training sets. Correspondingly, the remaining 7, 5, and 3 images are allocated as the test sets. Following preprocessing, the dimensions of the training sets are (28 23 120), (28 23 200) and (28 23 280), respectively. The TT-ranks are configured to (1 20 40 1). We set the regularization parameter to 0.01 and performed iterations for both algorithms using a variety of randomly generated initial points that conformed to the algorithmic criteria.

Figure 6 depicts the relationship between recognition accuracy and the number of iterations for various training set sizes, while Table 5 presents the final iteration recognition accuracy of each algorithm.



(a) size=(28 23 120) (b) size=(28 23 200) (c) size=(28 23 280)

Figure 6. The recognition accuracy for tested algorithms with different sizes of training set

Figure 6 and Table 5 illustrate that the recognition accuracy of the tested algorithms improves as the size of the training set expands. This phenomenon can be attributed to two principal factors. Primarily, the expansion of training set size enhances the comprehensiveness of known facial representations, enabling the basis vectors of the

Table 5. Comparison of tested algorithms with different sizes of training set

Algorithm	size=(28 23 120)	size=(28 23 200)	size=(28 23 280)
SNTT-MUR	0.8000	0.8500	0.9083
STT-PALM	0.8357	0.8900	0.9667

learned image space to better approximate those of the true image space. Secondarily, the relative proportion of test set samples diminishes correspondingly as training data increases. However, this scale enlargement inevitably prolongs algorithm iteration time. Therefore, in practical implementation, we can select a moderate value to strike a balance between time cost and recognition accuracy.

It is important to note that across the three facial recognition experiments, the STT-PALM algorithm consistently surpass the SNTT-MUR algorithm in recognition accuracy. Moreover, the STT-PALM algorithm attains high accuracy rapidly within the initial iterations, in contrast to the SNTT-MUR algorithm, which exhibits a gradual and protracted improvement in accuracy with an increasing number of iterations. Therefore, in practice, we can set a smaller maximum number of iterations for the STT-PALM algorithm to save time.

6. Conclusion

In this paper, we propose a novel sparse TT decomposition model, address the problem using the PALM algorithm, and provide a convergence analysis. Numerical experimental results demonstrate that our algorithm exhibits excellent performance. Future work will consider incorporating additional constraints into the model (such as non-negativity constraint).

Acknowledgement

This work was partially supported by National Natural Science Foundation of China (No. 11701132) and Natural Science Foundation of Zhejiang Province (No. LY22A010012).

REFERENCES

1. A. Beck, *First-order methods in optimization*, Society for Industrial and Applied Mathematics, 2017.
2. AT&T Laboratories Cambridge, *The ORL database of faces*, https://cam-orl.co.uk/facedatabase.html.
3. A. Uschmajew, and B. Vandereycken, *The geometry of algorithms using hierarchical tensors*, Linear Algebra and Its Applications, vol. 439, no. 1, pp. 133-166, 2013.
4. B. Dimitri, and J. Tsitsiklis, *Parallel and distributed computation: numerical methods*, Athena Scientific, 2015.
5. B. W. Bader, and T. G. Kolda, *Tensor toolbox for MATLAB, version 2.6*, www.tensortoolbox.org, 2015.
6. C. Pan, C. Ling, H. He, L. Qi, and Y. Xu, *A low-rank and sparse enhanced Tucker decomposition approach for tensor completion*, Applied Mathematics and Computation, vol. 465, pp. 128432, 2024.
7. D. Bigoni, A. P. Engsig-Karup, and Y. M. Marzouk, *Spectral tensor-train decomposition*, SIAM Journal on Scientific Computing, vol. 38, no. 4, pp. A2405-A2439, 2016.
8. D. Donoho, *Compressed sensing*, IEEE Transactions on Information Theory, vol. 52, no. 4, pp. 1289–1306, 2006.
9. E. Acar, S. A. Camtepe, M. S. Krishnamoorthy, and B. Yener, *Modeling and multiway analysis of chatroom tensors*, in Intelligence and Security Informatics: IEEE International Conference on Intelligence and Security Informatics, ISI 2005, Atlanta, GA, USA, May 19-20, 2005.
10. E. J. Candés, X. Li, Y. Ma, et al., *Robust principal component analysis?*, Journal of the ACM (JACM), vol. 58, no. 3, pp. 1–37, 2011.
11. G. Golub, and C. F. Van Loan, *Matrix computations*, The Johns Hopkins University Press, 2013.
12. H. Kuang, *Algorithm research and application of nonnegative tensor decomposition*, M.S. thesis, Hangzhou Dianzi University, Zhejiang, China, 2022.
13. H. Liu, J. Hu, Y. Li, and Z. Wen, *Optimization: Modeling, Algorithm and Theory (in Chinese)*, Higher Education Press, Beijing, 2020.
14. I.T. Jolliffe, *Principal Component Analysis, Second Edition*, Springer-Verlag, 2002.

15. I. V. Oseledets, *Tensor-train decomposition*, SIAM Journal on Scientific Computing, vol. 33, no.5, pp. 2295-2317, 2011.
16. J. Bolte, S. Sabach, and M. Teboulle, *Proximal alternating linearized minimization for nonconvex and nonsmooth problems*, Mathematical Programming, vol. 146, no. 1-2, pp. 459-494, 2014.
17. J. D. Carroll, and J. J. Chang, *Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition*, Psychometrika, vol. 35, no. 3, pp. 283-319, 1970.
18. J. F. Cai, W. Huang, H. Wang, and K. Wei, *Tensor completion via tensor train based low-rank quotient geometry under a preconditioned metric*, arXiv preprint arXiv:2209.04786, 2022.
19. J. M. Ortega, and W. C. Rheinboldt, *Iterative solution of nonlinear equations in several Variables*, Society for Industrial and Applied Mathematics, 2000.
20. K. Kurdyka, *On gradients of functions definable in o-minimal structures*, Annales de l'institut Fourier, vol. 48, no. 3, pp. 769-783, 1998.
21. L. De Lathauwer, and J. Vandewalle, *Dimensionality reduction in higher-order signal processing and rank-(R1, R2, ..., Rn) reduction in multilinear algebra*, Linear Algebra and its Applications, vol. 391, no. 1, pp. 31-55, 2004.
22. L. R. Tucker, *Implications of factor analysis of three-way matrices for measurement of change*, Problems in Measuring Change, vol. 15, pp. 122-137, 1963.
23. M. A. O. Vasilescu, and D. Terzopoulos, *Multilinear analysis of image ensembles: Tensorfaces*, in Computer Vision—ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28-31, 2002.
24. M. Welling, and M. Weber, *Positive tensor factorization*, Pattern Recognition Letters, vol. 22, no. 12, pp. 1255-1261, 2001.
25. P. Comon, *Tensor decompositions: state of the art and applications*, in Institute of Mathematics and its Applications conference series, Oxford, Clarendon, vol. 71, pp. 1-24, 1999.
26. S. Choi, *Algorithms for orthogonal nonnegative matrix factorization*, in Proc. of the Int. Joint Conf. on Neural Networks, pp. 1828–1832, 2008.
27. Y. D. Kim, and S. Choi, *Nonnegative Tucker decomposition*, in IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, 2007.
28. Y. Liu, J. Liu, Z. Long, and C. Zhu, *Tensor computation for data analysis*, Springer, Berlin, 2022.
29. Y. Wu, R. Chen, and Z. Chen, *Solving Sylvester tensor equation based on tensor train decomposition (in Chinese)*, Journal of Hangzhou Dianzi University (Natural Sciences), vol. 41, no. 6, pp. 94-99, 2021.
30. Z. Zhang, X. Yang, I. V. Oseledets, G. E. Karniadakis, and L. Daniel, *Enabling high-dimensional hierarchical uncertainty quantification by anova and tensor-train decomposition*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 34, no. 1, pp. 63-76, 2014.