Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

TuckerAPP: A novel spatiotemporal Tucker decomposition approach for traffic imputation

Wenwu Gong [b], Zhejun Huang [c], Jiaxin Lu [b], Lili Yang [a,b,*]

[a] Shenzhen Key Laboratory of Safety and Security for Next Generation of Industrial Interne, Guangdong, Shenzhen 518055, PR China
[b] Department of Statistics and Data Science, Southern University of Science and Technology, Guangdong, Shenzhen 518055, PR China
[c] School of Science, UNSW Canberra, 2600, ACT Australia

ARTICLE INFO

Keywords:
Traffic data imputation
Regularized Tucker decomposition
Spatiotemporal constraints
Alternating proximal gradient

ABSTRACT

Missing traffic data caused by sensor failures poses a significant challenge to reliable traffic prediction and control. Existing spatiotemporal imputation methods struggle in extreme cases with over 80% missing data and complex non-random patterns. To address these limitations, we propose the spatiotemporal regularized Tucker APProach (TuckerAPP), which integrates tensor sparsity with spatiotemporal constraints to jointly capture long-term trends and short-term dynamics. TuckerAPP offers two key advantages: (i) adaptive rank determination via full-size Tucker decomposition with core tensor sparsity, and (ii) hierarchical spatiotemporal modeling through graph-regularized spatial factors that encode road network topology and Toeplitz-constrained temporal factors that capture periodic traffic patterns. We further develop a multi-block alternating proximal gradient algorithm with guaranteed convergence for large-scale tensors. Extensive experiments on urban traffic and network flow datasets demonstrate that TuckerAPP consistently outperforms six state-of-the-art baselines under extreme missing scenarios. These results confirm TuckerAPP's robustness in preserving spatiotemporal consistency and highlight its superiority over existing tensor-based approaches.

1. Introduction

With the rapid advancement of sensor technologies and intelligent transportation systems, the collection of multidimensional traffic data has become increasingly widespread. Road loop sensors capture key metrics such as speed, flow, and occupancy, while GPS-equipped vehicles record movement patterns, including origin–destination pairs, travel times, and transportation modes. This rich traffic information is essential for transportation management and route planning. However, missing data issues caused by sensor failures, transmission errors, or adverse weather conditions significantly impact traffic analysis and forecasting accuracy [1]. Developing robust imputation methods is crucial for ensuring data continuity and supporting effective traffic management [2].

Traditional two-dimensional matrix imputation methods fail to fully exploit the rich spatiotemporal structures inherent in traffic data, often leading to suboptimal accuracy [3,4]. In contrast, higher-order traffic tensors preserve natural spatial and temporal correlations, providing a more comprehensive representation. Over the past two decades, tensor-based approaches have demonstrated strong capabilities in capturing short-term dynamics [5] and efficiently processing large-scale traffic datasets [6]. Existing methods typically rely on tensor decomposition or nuclear norm minimization to exploit low-rank priors, while incorporating additional regularizers to maintain local similarity.

Besides, current tensor-based methods face critical challenges under extreme missing conditions, particularly when data loss exceeds 80% or follows complex non-random patterns. Moreover, they often fail to fully leverage long-term periodic traffic behaviors and spatial consistency across road networks, fundamentally limiting their imputation performance. Addressing these limitations requires models that can simultaneously capture long-term periodicity and short-term variability, while maintaining robustness against severe data sparsity.

1.1. *Literature review*

Given the long-term trend and dynamic pattern inherent in traffic data, research has shown that combining the low-rank assumption with spatiotemporal constraints often yields superior performance compared to other methods for traffic data imputation [7,8].

1.1.1. *Low-rank tensor imputation*

The low-rank property, which captures global correlations, is a critical assumption in the traffic data imputation (TDI) problem. Ran

[*] Corresponding author.
E-mail address: yangll@sustech.edu.cn (L. Yang).

https://doi.org/10.1016/j.sigpro.2025.110383
Received 1 April 2025; Received in revised form 8 October 2025; Accepted 2 November 2025
Available online 5 November 2025

et al. [9] were the first to employ the tensor nuclear norm minimization method to recover traffic flow. Chen et al. [1] proposed a multi-rank model based on truncated singular values of tensor unfolding matrices to address the high computational cost associated with singular value decomposition. Other studies have also used the low-rank Hankel structure to capture the long-term trend of sensory traffic data [4]. However, enforcing low-rankness through the singular values of TDI disrupts the inherent tensor structure and incurs high computational costs due to repeated matrix unfolding operations [10].

Low-rank tensor decomposition, which captures traffic patterns by decomposing the tensor into simpler components, encodes the traffic information of a specific mode and has a higher parameterization efficiency in tensor modeling. For example, Tan et al. [6,11] proposed the Tucker decomposition model to exploit long-term traffic trends. Yokota et al. [12] introduced a smoothed CANDECOMP/PARAFAC (CP) decomposition and Chen et al. [13] proposed a Bayesian-augmented CP decomposition model incorporating traffic domain knowledge to improve imputation performance. Zhang et al. [14] proposed a tensor train (TT) model to characterize the global information of the traffic tensor. He et al. [15] introduced a tubal-rank-based spatiotemporal imputation model to exploit the hidden structures of traffic tensors. However, choosing the appropriate tensor rank remains challenging in tensor-based imputation methods.

1.1.2. Imputation with spatiotemporal constraints

Numerous studies have shown that low-rank tensor models are insufficient for scenarios with high missing data ratios [16,17]. Traffic data, with their complex spatiotemporal patterns, have led to the extensive exploration of various constraint-based methods, including total variation [18], graph regularization [19], and temporal regularization [8]. Rose et al. [20] were pioneers in analyzing traffic data using spatiotemporal characteristics within a low-rank learning framework, accounting for similarities in both spatial and temporal dimensions. Chen et al. [8] introduced a temporal correlation term to improve their traffic data imputation model. Additionally, Chen et al. [21] developed a large-scale framework for traffic data imputation that integrates temporal regularization with low-tubal rank tensor completion.

The Tucker model offers a distinct advantage in capturing spatiotemporal correlations in traffic data within a subspace [7,17]. For example, Goulart et al. [22] applied the orthogonal Tucker model with tensor core thresholding to enhance traffic flow imputation performance. Wang et al. [23] proposed a regularized non-negative Tucker decomposition model to identify interpretable traffic patterns in urban traffic flows. Su et al. [24] combined the Tucker model with l_2 norm regularization of factor matrices to improve imputation accuracy. Pan et al. [10] introduced a sparse enhanced Tucker decomposition model to better leverage long- and short-term information for traffic data imputation. However, many of these existing approaches overlook the spatial similarity patterns inherent in traffic data, particularly the correlations between nearby sensors. This oversight results in models that lack robustness when dealing with high missing data rates or structured missing patterns, as they fail to leverage the spatial redundancy that could compensate for missing observations.

1.2. Limitations of existing approaches

In reviewing these related works, we observe that most approaches rely on tensor decomposition or the associated nuclear norm to capture the essential low-rank prior, while incorporating additional regularizers to enforce temporal local consistency. The core motivation of our proposal stems from two critical gaps in current Tucker-based traffic data imputation methods. First, existing methods either require manual specification of tensor ranks [10], which is often impractical without domain expertise, or rely on computationally expensive rank estimation procedures [25]. Our first key insight is that spatiotemporal feature tensors naturally exhibit sparse core structures when decomposed using full-size Tucker decomposition. This sparsity arises from the multi-scale nature of traffic patterns, where only a subset of factor combinations contributes significantly to the traffic dynamics. By enforcing sparsity on the core tensor through l_1 norm while maintaining full-size factor matrices, we perform low-rank representation without requiring prior knowledge of the appropriate tensor rank.

The second motivation addresses the insufficient exploitation of spatiotemporal informations in existing methods. Traffic data exhibits strong spatial correlations due to network topology (nearby sensors tend to have similar traffic patterns) and temporal correlations due to periodic patterns (rush hours, weekly cycles, seasonal variations) [7,8]. Our spatiotemporal regularization addresses this limitation through two complementary mechanisms. For spatial correlations, we employ graph regularization on factor matrices corresponding to spatial modes, where the Laplacian structure L captures the similarity between traffic sensors based on their historical patterns. For temporal correlations, we introduce Toeplitz-constrained regularization on temporal factor matrices, which captures the smooth evolution of traffic patterns over time.

1.3. Contributions

This study proposes spatiotemporal regularized Tucker approach (TuckerAPP), an enhanced Tucker decomposition model for traffic data imputation. To address the challenge of rank selection in Tucker decomposition, TuckerAPP adopts a full-size Tucker decomposition strategy and characterizes the tensor's low-rank structure through core tensor sparsity. In addition, spatiotemporal constraints are incorporated to improve imputation accuracy, especially under high missing data rates.

- Novel Tucker-based imputation model. We propose TuckerAPP, which captures long-term traffic patterns via core tensor sparsity, integrates non-negative factor matrices, and leverages spatiotemporal regularization to model short-term dynamics. TuckerAPP eliminates manual rank selection under full-size Tucker decomposition. Besides, the non-negativity constraints imposed on factor matrices further strengthen spatiotemporal correlations and enhance imputation accuracy.
- Efficient optimization. We design an alternating proximal gradient algorithm with closed-form update rules for all variables. The algorithm is computationally efficient and comes with theoretical convergence guarantees, ensuring stability on large-scale traffic tensors.
- Empirical validation. Experiments on two real-world traffic datasets under both random and structured missing scenarios demonstrate that TuckerAPP consistently outperforms baselines. Notably, its superiority becomes more pronounced under extreme missing conditions, confirming the robustness of the proposed model.

The remainder of this paper is organized as follows. Section 2 introduces notations and problem formulation. Section 3 presents the TuckerAPP method with detailed model analysis. Section 4 details the optimization algorithm and its convergence analysis. Section 5 reports experimental results. Finally, Section 6 concludes the study and discusses future research directions.

2. Preliminaries

2.1. Notations

We denote the mode-n unfolding operator as $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times \prod_{j \neq n} I_j}$. Based on the Kronecker matrix product $\otimes$, we represent the Tucker decomposition $\mathcal{X} = \mathcal{G} \times_{n=1}^{N} \mathbf{U}_n$ by $\mathbf{X}_{(n)} = \mathbf{U}_n \mathbf{G}_{(n)} \mathbf{V}_n^{\mathrm{T}}$, $\mathbf{V}_n = \left(\mathbf{U}_N \otimes \cdots \otimes \mathbf{U}_{n+1} \otimes \mathbf{U}_{n-1} \otimes \cdots \otimes \mathbf{U}_1\right)$ and the superscript 'T' represent the matrix transpose. We can verify that $\mathrm{vec}(\mathcal{X}) = \left(\mathbf{U}_N \otimes \cdots \otimes \mathbf{U}_n \otimes \cdots \otimes \mathbf{U}_1\right) \mathrm{vec}(\mathcal{G}) = \otimes_{n=N}^{1} \mathbf{U}_n \, \mathrm{vec}(\mathcal{G})$. We define $\mathcal{X}_{\Omega}$ as a projector that keeps the nonzero terms and leaves the other values as zero values. Please refer to [26] for more information on the notation (see Table 1).

Table 1
Notations

$\mathcal{X}, \mathbf{U}, \alpha$	A tensor, matrix and real value, respectively.
$\mathbb{R}_{+}^{I_1 \times I_2 \times \cdots \times I_N}$	Set of N-th order non-negative array.
$\mathcal{P}_{+}(\mathbf{U})$	Operator yielding a non-negative matrix of $u_{ij} = \max\left(u_{ij}, 0\right), \forall i, j$.
$\mathcal{S}_{\mu}(x)$	Shrinkage operator with μ in component-wise.
$\Omega, \bar{\Omega}$	Observed index set and its complement.
$\mathcal{X}_{\Omega}$	Observed entries supported on the observed index.
$\mathcal{H}$	Tensorization operator.
$\times_n$	Mode-n product.
$\otimes, \odot$	Kronecker product and Hadamard product.
$\\|*\\|_F$	Frobenius norm.
$\mathbf{X}_{(n)}$	Mode-n unfolding of tensor $\mathcal{X}$.
tr	Matrix trace operator.

2.2. *Traffic data imputation*

Traffic data is typically collected from M sensors over J days with I time points. The missing multivariate time series is indicated as $\mathbf{Y}_{\Omega} \in \mathbb{R}^{M \times IJ}$ with the observed index set Ω, as shown in Fig. 1(a). Prior studies have demonstrated the effectiveness of low-rank tensor completion methods, which model traffic data as low-rank tensors due to obvious dependencies in the spatial dimension and in the temporal dimension. Chen et al. [8] showed that a low-rank tensor can effectively capture long-term trends in traffic data. However, traffic data tends to be similar along nearby sensors and correlates at adjacent time points, reflecting short-term patterns [27]. Existing methods often fail under such conditions because they rely primarily on global low-rank assumptions without adequately leveraging local spatiotemporal structure.

This study introduces the tensorization operator [28] $\mathcal{H}$ to stack one-day traffic sensory data and reshape it into a third-order tensor $\mathcal{X} \in \mathbb{R}^{M \times I \times J}$. Then, an enhanced Tucker decomposition combined with the spatiotemporal constraints model (TuckerAPP) is proposed to capture long- and short-term traffic patterns. The main idea of the TuckerAPP model is to find a low-rank Tucker approximation $\hat{\mathcal{X}}$ for the traffic tensor. In contrast, the inverse operator $\hat{\mathbf{Y}} = \mathcal{H}^{-1}(\hat{\mathcal{X}})$ converts the reconstructed tensor into the original traffic matrix and then estimates the missing values.

Mathematically, the proposed model is illustrated by minimizing the following objective:

$$\underset{\mathcal{X};\mathcal{G};\{\mathbf{U}_n\}}{\text{minimize}} \ \frac{1}{2}\left\|\mathcal{X} - \mathcal{G} \times_{n=1}^{N} \mathbf{U}_n\right\|_F^2 + \mathcal{R}(\mathcal{G}, \{\mathbf{U}_n\}), \quad \text{s.t.,} \ \mathcal{X}_{\Omega} = \mathcal{H}(\mathbf{Y}_{\Omega}), \tag{1}$$

$\mathcal{R}(\cdot)$ is the user-defined regularization constraint to encode structured priors of the underlying traffic data effectively. Under different missing scenarios Ω, we estimate traffic tensor $\mathcal{X}$ by the equation (2)

$$\hat{\mathcal{X}} = \mathcal{X}_{\Omega} + \{\hat{\mathcal{G}} \times_{n=1}^{N} \hat{\mathbf{U}}_n\}_{\bar{\Omega}}. \tag{2}$$

3. Proposed model

Traffic data often shows multidimensional dependencies in the spatial dimension and in the temporal dimension, resulting in low rankness of traffic data. This section describes the formulation of the TuckerAPP model using spatial and temporal constraints in a low-rank Tucker decomposition framework. The proposed model incorporates four key components, including core tensor sparsity, non-negative factor matrices, Laplacian regularization, and Toeplitz constraint.

3.1. *Sparsity measure*

As discussed in the introduction, the key challenge of Tucker decomposition lies in selecting the appropriate rank. To address this issue, we penalize the sparsity of the core tensor in the Tucker decomposition. Drawing inspiration from previous studies [29,30], we set a full-size Tucker decomposition and characterize the tensor's low-rank structure through the core tensor's sparsity. To further explore this sparsity, we introduce constraints on non-negative factor matrices [31]. Specifically, applying non-negativity constraints to the factor matrices leads to better compression performance compared to unconstrained alternatives [32].

3.2. *Spatiotemporal constraints*

Previous studies show that traffic data have short-term patterns along the spatial and temporal modes. The similarity between the rows of the traffic matrix characterizes the spatial pattern, and the correlations between columns capture the temporal correlation [33]. Recent research extracts spatiotemporal features using regularization but does not clearly explain the rationale. Temporal correlations have been studied, but few papers have discussed spatial similarity. This research addresses the spatiotemporal correlations relying on the graph regularization and temporal constraint on factor matrices, which leads to better performance and clearly explains the rationale behind modeling spatiotemporal features for the TDI problem.

3.2.1. *Spatial similarity*

To exploit the spatial correlations among traffic sensors, we leverage the similarity of neighboring nodes to encode the spatial local consistency. Inspired by graph theory [34,35], we employ graph Laplacian regularization to capture the inherent geometric structure of the traffic network. Specifically, the adjacency matrix of the road graph is constructed based on the pairwise distances between sensor locations within the network. For each traffic sensor, we select its p nearest neighbors and construct a similarity matrix using a kernel-based weighting scheme:

$$w_{ij} = \exp\left(-\frac{|y_i - y_j|^2}{\sigma^2}\right), \tag{3}$$

where y_i and y_j denote the traffic value of sensors i and j, respectively. The scaling parameter is fixed to $\sigma^2 = 1$, ensuring a uniform control of the adjacent matrix distribution. This exponential kernel ensures that sensors with similar historical patterns receive higher weights, while dissimilar sensors receive weights approaching zero. The choice of exponential kernel is motivated by its ability to create smooth, localized neighborhoods in the pattern space. Unlike binary adjacent matrices based solely on physical proximity, our similarity matrix captures functional relationships between sensors based on their actual traffic behaviors.

Based on the constructed similarity matrix $\mathbf{W} \in \mathbb{R}^{I_1 \times I_1}$, we apply graph Laplacian regularization to capture local spatial consistency:

$$\sum_{i=1}^{I_1}\sum_{j=1}^{I_1} w_{ij}\left|\mathbf{u}_i - \mathbf{u}_j\right| 2^2 = \operatorname{tr}\left(\mathbf{U}^{\mathrm{T}}\mathbf{L}\mathbf{U}\right), \quad \mathbf{L} = \mathbf{D} - \mathbf{W}, \tag{4}$$

where $\mathbf{u}_i$ denotes the column vector of $\mathbf{U}^{\mathrm{T}}$, $\mathbf{D} \in \mathbb{R}^{I_1 \times I_1}$ is a diagonal degree matrix with entries $d_{ii} = \sum_{j=1}^{I_1} w_{ij}$. This formulation enforces that sensors located closer in the spatial domain produce traffic data with higher similarity, thereby encoding spatial smoothness in the TDI problem.

3.2.2. *Temporal correlation*

The correlations between adjacent time points in the time dimension are modeled using a temporal constraint [8]. Considering the non-stationary in the temporal dimension, the original traffic data is often correlated at adjacent time points. For adjacent $j-1$th and jth time points in the traffic matrix $\mathbf{Y}$, we consider the Toeplitz operator $\mathbf{T}$ defined in the traffic tensor $\mathcal{X}$ to capture the temporal correlation, i.e., $\left\|\mathbf{Y}_{\cdot j} - \mathbf{Y}_{\cdot j-1}\right\|_F^2 = \|\mathcal{X} \times_n \mathbf{T}\|_F^2$.

$$\begin{aligned} \|\mathcal{X} \times_n \mathbf{T}\|_F^2 &= \left\|\mathcal{G} \times_1 \mathbf{U}_1 \cdots \times_n \left(\mathbf{T}\mathbf{U}_n\right) \times_{n+1} \cdots \times_N \mathbf{U}_N\right\|_F^2 = \left\|\left(\mathbf{T}\mathbf{U}_n\right)\left(\mathcal{G} \times_{p=1,p\neq n}^{N} \mathbf{U}_p\right)\right\|_F^2 \\ &\leq \|\mathbf{T}\mathbf{U}_n\|_F^2 \left\|\mathcal{G} \times_{p=1,p\neq n}^{N} \mathbf{U}_p\right\|_F^2 \leq \text{const.}\, \|\mathbf{T}\mathbf{U}_n\|_F^2 . \end{aligned} \tag{5}$$

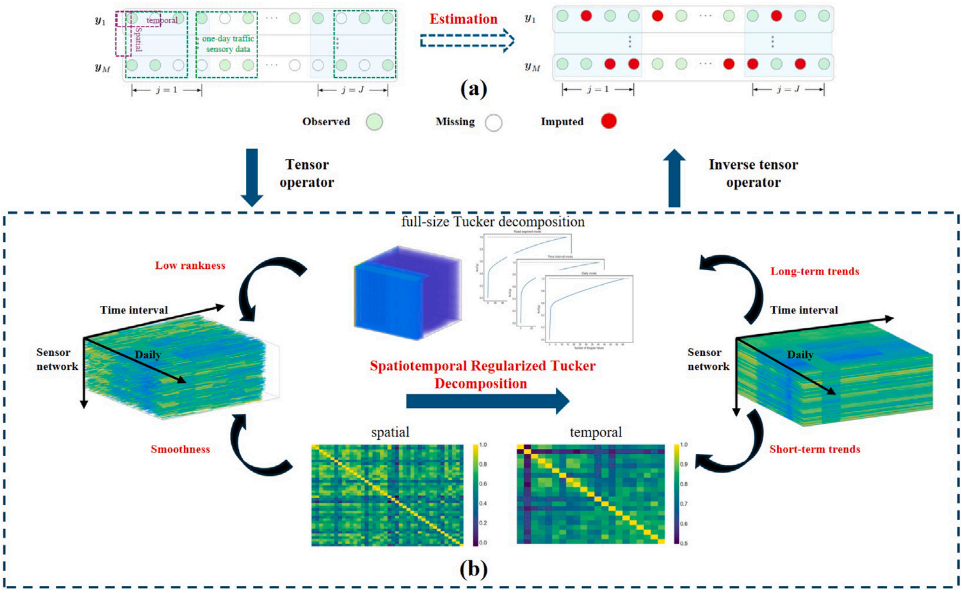

Fig. 1. Framework of the proposed TuckerAPP model. (a) Problem setup, data tensorization, and estimation: the original traffic matrix $\mathbf{Y} \in \mathbb{R}^{M\times IJ}$ with partially observed entries is reshaped into a third-order tensor $\mathcal{X} \in \mathbb{R}^{M\times I\times J}$ via tensorization operator $\mathcal{H}$; the imputed tensor $\hat{\mathcal{X}}$ is then converted back to matrix form $\hat{\mathbf{Y}} = \mathcal{H}^{-1}(\hat{\mathcal{X}})$. (b) TuckerAPP architecture: the model employs a full-size Tucker decomposition with a sparsity-constrained core tensor. Spatial regularization is enforced through a Laplacian structure capturing sensor similarity, while fractional difference regularization via Toeplitz constraint exploits temporal local consistency.

Consequently, we use $\|\mathbf{TU}\|_F^2$ to characterize the temporal correlation of the traffic data.

Remark 1. Previous studies have applied differential operators on unfolding matrices as regularization terms to model traffic dynamics in TDI problems [8,24]. However, the traffic dynamics are more pronounced in the latent space, where Tucker decomposition captures them more effectively [36,37]. The proposed spatiotemporal constraints capture the short-term dynamics by regularizing the factor matrices within the Tucker decomposition, enhancing interpretability and improving performance.

3.3. Spatiotemporal regularized tucker decomposition model

Let $\mathcal{X}^0 \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ be the missing traffic tensor. Based on Eq. (1) and the aforementioned spatiotemporal constraints, we consider the following optimization problem

$$\begin{aligned}\underset{\mathcal{G};\{\mathbf{U}_n\};\mathcal{X}}{\text{minimize}}\ \mathbb{F}(\mathcal{G},\{\mathbf{U}_n\},\mathcal{X}) \triangleq \{&\frac{1}{2}\left\|\mathcal{X}-\mathcal{G}\times_{n=1}^{N}\mathbf{U}_n\right\|_F^2+\alpha\|\mathcal{G}\|_1\\ &+\sum_{n=1}^{K}\frac{\beta_n}{2}\operatorname{tr}\left(\mathbf{U}_n^{\mathrm{T}}\mathbf{L}_n\mathbf{U}_n\right)+\sum_{n=K+1}^{N}\frac{\gamma_n}{2}\|\mathbf{T}_n\mathbf{U}_n\|_F^2\} \qquad (6)\\ &\text{s.t., } \mathbf{U}_n\in\mathbb{R}_+^{I_n\times I_n}, n=1,\dots,N \text{ and } \mathcal{X}_\Omega=\mathcal{X}_\Omega^0,\end{aligned}$$

where $\alpha, \beta_n, \gamma_n$ are positive penalty parameters, K represents the numbers of spatial modes, $\mathbf{L}_n$ captures the spatial similarity, and $\mathbf{T}_n$ encodes the temporal. Notably, imposing non-negativity constraints on the factor matrix enhances dependencies across spatial and temporal dimension [38]. Furthermore, the Tucker core tensor becomes sparser [39] and captures more long-term trends [40].

Remark 2. The Tucker components' constraint ensures that the TuckerAPP model is well-defined. On the one hand, if all penalty parameters and non-negative vanish, there are product combinations $\{\lambda_{n+1}\}$ such that $\{\lambda_1\mathcal{G}, \lambda_2\mathbf{U}_1, \dots, \lambda_{n+1}\mathbf{U}_n\}$ does not change the value of (6). Hence, the low-rank Tucker approximation may not be a unique solution. On the other hand, spatiotemporal constraints imply that the gradients of (6) are Lipschitz continuous and have a bounded Lipschitz constant under proximal linear operators (See Propositions 1 and 2), which guarantees that the solution set is nonempty.

4. Solving TuckerAPP model

The nature of our optimization problem in (6) presents a multi-block non-convex optimization challenge with mixed constraints including non-negativity constraints and spatiotemporal regularization terms on factor matrices, sparsity regularization on the core tensor. To solve the optimization problem (6), we employ the alternating proximal gradient (APG) method to transform the objective function into multiple solvable subproblems and iterate alternately to find the critical point. The proposed algorithm uses the proximal operator to find approximate solutions of (6), reducing each iteration's time cost and improving the imputation performance.

4.1. Optimization for the TuckerAPP model

The TuckerAPP model is the regularized block multi-convex optimization problem, which can be solved using the proximal linear operator and finds an inexact solution. We first transform the (6) in mode-n unfolding, and the subproblems are given as the following three types.

- Basic non-negative matrix factorization

$$\underset{\mathbf{U}_n\geq 0}{\text{minimize}}\ \ell(\mathbf{U}_n)=\frac{1}{2}\left\|\mathbf{X}_{(n)}-\mathbf{U}_n\mathbf{G}_{(n)}\mathbf{V}_n^{\mathrm{T}}\right\|_{\mathrm{F}}^2,\quad \mathbf{V}_n=\otimes_{p=N,p\neq n}^{1}\mathbf{U}_p. \qquad (7)$$

- Graph regularization on factor matrix

$$\underset{\mathbf{U}_n\geq 0}{\text{minimize}}\ \ell(\mathbf{U}_n)=\frac{1}{2}\left\|\mathbf{X}_{(n)}-\mathbf{U}_n\mathbf{G}_{(n)}\mathbf{V}_n^{\mathrm{T}}\right\|_{\mathrm{F}}^2+\frac{\beta_n}{2}\operatorname{tr}\left(\mathbf{U}_n^{\mathrm{T}}\mathbf{L}_n\mathbf{U}_n\right) \qquad (8)$$

where $\mathbf{L}_n = \mathbf{D}_n - \mathbf{W}_n$ represents the Laplacian matrix.

- Temporal constraint on factor matrix

$$\underset{\mathbf{U}_n\geq 0}{\text{minimize}}\ \ell(\mathbf{U}_n)=\frac{1}{2}\left\|\mathbf{X}_{(n)}-\mathbf{U}_n\mathbf{G}_{(n)}\mathbf{V}_n^{\mathrm{T}}\right\|_{\mathrm{F}}^2+\frac{\gamma_n}{2}\left\|\mathbf{T}_n\mathbf{U}_n\right\|_{\mathrm{F}}^2 \qquad (9)$$

where $\mathbf{T}_n$ is a self-defined temporal constraint matrix.

Proposition 1. *Subproblems (7)–(9) are differentiable and convex. Besides, the gradients $\nabla_{\mathbf{U}_n}\ell(\mathbf{U}_n)$ are both Lipschitz continuous with the following Lipschitz constant.*

$$L_{\mathbf{U}_n}=\begin{cases}\left\|\mathbf{G}_{(n)}\mathbf{V}_n^{\mathrm{T}}\mathbf{V}\mathbf{G}_{(n)}^{\mathrm{T}}\right\|_2+\beta_n\left\|\mathbf{L}_n\right\|_2, & \textit{Spatial}\\ \left\|\mathbf{G}_{(n)}\mathbf{V}_n^{\mathrm{T}}\mathbf{V}\mathbf{G}_{(n)}^{\mathrm{T}}\right\|_2+\gamma_n\left\|\mathbf{T}_n^{\mathrm{T}}\mathbf{T}_n\right\|_2, & \textit{Temporal}\\ \left\|\mathbf{G}_{(n)}\mathbf{V}_n^{\mathrm{T}}\mathbf{V}\mathbf{G}_{(n)}^{\mathrm{T}}\right\|_2, & \textit{otherwise.}\end{cases}$$

Then, we can use the prox-linear operator to solve factor matrices subproblems, i.e.,

$$\hat{\mathbf{U}}_n=\underset{\mathbf{U}_n\ge 0}{\operatorname{argmin}}\ \left\langle\nabla_{\mathbf{U}_n}\ell(\tilde{\mathbf{U}}_n),\mathbf{U}_n-\tilde{\mathbf{U}}_n\right\rangle+\frac{L_{\mathbf{U}_n}}{2}\|\mathbf{U}_n-\tilde{\mathbf{U}}_n\|_F^2,$$

where $\tilde{\mathbf{U}}_n$ denotes the extrapolated point. We take the derivative and find out

$$\hat{\mathbf{U}}_n=\mathcal{P}_+\left(\tilde{\mathbf{U}}_n-\frac{1}{L_{\mathbf{U}_n}}\nabla_{\mathbf{U}_n}\ell\left(\tilde{\mathbf{U}}_n\right)\right),\tag{10}$$

where $\mathcal{P}_+(\cdot)$ projects the negative entries of matrix into zeros.

Secondly, we update the subproblem $\mathcal{G}$ using the vectorization optimization problem

$$\underset{\mathcal{G}}{\text{minimize}}\ \frac{1}{2}\left\|\operatorname{vec}(\mathcal{X})-\otimes_{n=N}^{1}\mathbf{U}_n\operatorname{vec}(\mathcal{G})\right\|_{\mathrm{F}}^2+\alpha\|\operatorname{vec}(\mathcal{G})\|_1=f(\mathcal{G})+\alpha\|\operatorname{vec}(\mathcal{G})\|_1.$$

Proposition 2. *$\nabla_{\mathcal{G}}f(\mathcal{G})$ is Lipschitz continuous with the bounded Lipschitz constant*

$$L_{\mathcal{G}}=\left\|\otimes_{n=N}^{1}\mathbf{U}_n^{\mathrm{T}}\mathbf{U}_n\right\|_2=\prod_{n=1}^{N}\left\|\mathbf{U}_n^{\mathrm{T}}\mathbf{U}_n\right\|_2.$$

Given by the result of Proposition 2, the core tensor prox-linear function is denoted as

$$\hat{\mathcal{G}}=\underset{\mathcal{G}}{\operatorname{argmin}}\left\langle\nabla_{\mathcal{G}}f(\tilde{\mathcal{G}}),\mathcal{G}-\tilde{\mathcal{G}}\right\rangle+\frac{L_{\mathcal{G}}}{2}\|\mathcal{G}-\tilde{\mathcal{G}}\|_F^2+\alpha\|\mathcal{G}\|_1,$$

where $\tilde{\mathcal{G}}$ denotes the extrapolated point. Using the soft-thresholding operator [41], the result of composite model is given by

$$\hat{\mathcal{G}}=\mathcal{S}_{\frac{\alpha}{L_{\mathcal{G}}}}\left(\tilde{\mathcal{G}}-\frac{1}{L_{\mathcal{G}}}\nabla_{\mathcal{G}}f\left(\tilde{\mathcal{G}}\right)\right)\tag{11}$$

where $\mathcal{S}_\mu(\cdot)$ is 'shrinkage' operator defining component-wisely as $\mathcal{S}_\mu(x)=\operatorname{sign}(x)\cdot\max(0,|x|-\mu)$.

Third, since spatiotemporal priors are imposed on the factor matrices $\{\mathbf{U}_n\}$, our proposed algorithm follows the sequence $\mathcal{G},\mathbf{U}_1,\mathbf{U}_2,\ldots,\mathbf{U}_N$ in its design. During the kth iteration, the core tensor $\mathcal{G}^k$ is updated by

$$\begin{aligned}&\mathcal{G}^{k+1}=\mathcal{S}_{\frac{\alpha}{L_{\mathcal{G}}^k}}\left(\tilde{\mathcal{G}}^k-\frac{1}{L_{\mathcal{G}}^k}\nabla_{\mathcal{G}}f\left(\tilde{\mathcal{G}}^k\right)\right),\\&\tilde{\mathcal{G}}^k=\mathcal{G}^k+\omega_k\left(\mathcal{G}^k-\mathcal{G}^{k-1}\right),\ \omega_k=\min\{\frac{t^{k-1}-1}{t^k},0.999\sqrt{\frac{L_{\mathcal{G}}^{k-1}}{L_{\mathcal{G}}^k}}\}\end{aligned}\tag{12}$$

and the factor matrices $\{\mathbf{U}_n^k\}$

$$\begin{aligned}&\mathbf{U}_n^{k+1}=\mathcal{P}_+\left(\tilde{\mathbf{U}}_n^k-\frac{1}{L_{\mathbf{U}_n}^k}\nabla_{\mathbf{U}_n}\ell\left(\tilde{\mathbf{U}}_n^k\right)\right),\\&\tilde{\mathbf{U}}_n^k=\mathbf{U}_n^k+\omega_k\left(\mathbf{U}_n^k-\mathbf{U}_n^{k-1}\right),\ \omega_k=\min\{\frac{t^{k-1}-1}{t^k},0.999\sqrt{\frac{L_{\mathbf{U}}^{k-1}}{L_{\mathbf{U}}^k}}\}.\end{aligned}\tag{13}$$

Technically, we speed up Algorithm 1 using a parameterized iterative shrinkage-thresholding (PIST) scheme [42], given by $t^k=\frac{0.8+\sqrt{4(t^{k-1})^2+0.8}}{2}$, $t^0=1$.

Furthermore, we use the control rule [39] to re-update tensor $\mathcal{Z}^k$ at the end of iteration $k+1$

$$\mathcal{Z}^{k+1}{}_{\Omega}=\mathcal{X}^0{}_{\Omega}+\phi(\mathcal{Z}^{k-1}{}_{\Omega}-\{\mathcal{G}^k\times_{n=1}^{N}\mathbf{U}_n^k\}_{\Omega}),\quad\mathcal{Z}^{k+1}{}_{\bar{\Omega}}=\{\mathcal{G}^k\times_{n=1}^{N}\mathbf{U}_n^k\}_{\bar{\Omega}},\tag{14}$$

where tensor $\mathcal{Z}^{k-1},\mathcal{Z}^0=\mathcal{X}^0$ is the last iteration, $0\le\phi\le 1$ is a hyperparameter, and $\bar{\Omega}$ represents the complement of set Ω. We ensure that the value of $\mathbb{F}\left(\mathcal{G}^{k+1},\{\mathbf{U}_n^{k+1}\},\mathcal{Z}^{k+1}\right)$ decreases before re-updating the $\tilde{\mathcal{G}}$, $\{\tilde{\mathbf{U}}_n\}$. Once this is achieved, the complete tensor $\hat{\mathcal{X}}=\mathcal{X}_{\Omega}^0+\mathcal{Z}^{k+1}{}_{\bar{\Omega}}$ is computed as the imputed result when the condition in (15) is satisfied.

$$\left\|\Omega\odot(\mathcal{Z}^{k+1}-\mathcal{X}^0)\right\|_F\left\|\Omega\odot\mathcal{X}^0\right\|_F^{-1}<\text{tol},\ \text{for some }k.\tag{15}$$

Algorithm 1 APG-based solver for the TuckerAPP model

```
Input: Missing traffic tensor $\mathcal{X}^0\in\mathbb{R}_+^{I_1\times I_2\times\cdots\times I_N}$, $\Omega$ containing indices of observed entries, and the parameters $\alpha\ge 0$, $\beta_n\ge 0$, $\gamma_n\ge 0$, tol $=1e^{-4}$.
Output: Reconstructed tensor $\hat{\mathcal{X}}$.
Construct positive semi-definite similarity matrix $\mathbf{W}_n$ and temporal constraint matrix $\mathbf{T}_n$;
Initialize $\mathcal{G}^0,\mathbf{U}_n^0\in\mathbb{R}_+^{I_n\times I_n}$ $(1\le n\le N)$ randomly and then processed by normalization;
for $k=1$ to $K$ do
    Optimize $\mathcal{G}$ according to (12);
    for $n=1$ to $N$ do
        Optimize $\mathbf{U}_n$ using (13);
    end for
    Find out $\mathcal{Z}^{k+1}$ using (14);
    Re-update $\tilde{\mathcal{G}}$, $\{\tilde{\mathbf{U}}_n\}$ when $\mathbb{F}\left(\mathcal{G}^{k+1},\{\mathbf{U}_{k+1}\},\mathcal{Z}^{k+1}\right)<\mathbb{F}\left(\mathcal{G}^k,\{\mathbf{U}_k\},\mathcal{Z}^k\right)$ until (15) satisfied.
end for
return $\hat{\mathcal{X}}_{\Omega}=\mathcal{X}^0{}_{\Omega}$, $\hat{\mathcal{X}}_{\bar{\Omega}}=\mathcal{Z}^K{}_{\bar{\Omega}}$.
```

The algorithm 1 is an APG-based update procedure with closed-form solutions for the proposed problem (6), which improves the algorithm's efficiency. Since the TuckerAPP model is a nonconvex problem, we demonstrate the convergence properties of the algorithm using inexact block coordinate descent [43,44].

4.2. Convergence analysis

Theorem 1. *Let the bounded sequence generated by Algorithm 1 be denoted by $\Theta^k=\{\mathcal{G}^k,\{\mathbf{U}_n^k\}\}$. We can then guarantee that this sequence converges to a critical point, $\hat{\Theta}=\{\hat{\mathcal{G}},\{\hat{\mathbf{U}}_n\}\}$.*

We present a proof outline in three steps. First, we establish a square summable result, $\sum_{k=1}^{\infty}\left\|\Theta^{k-1}-\Theta^k\right\|_F^2<\infty$. Next, we verify that subsequence Θ^k converges to a stationary point $\hat{\Theta}$ [41]. Finally, the Kurdyka–Lojasiewicz (KL) property of objective function $\mathbb{F}$ guarantees that Θ^k converges to a critical point.

Proof. Square summable: We express (6) as $\mathbb{F}(\Theta)=\mathbb{F}_1(\Theta)+\mathbb{F}_2(\Theta)$, $\Theta=\{\{\mathbf{U}_n\},\mathcal{G}\}$, where $\mathbb{F}_1$ is either function ℓ or f and $\mathbb{F}_2$ is either the l_1 norm or a non-negative projector. The prox-linear updating rule indicates

$$\hat{\Theta}=\underset{\Theta}{\operatorname{argmin}}\ \left\langle\nabla_{\Theta}\mathbb{F}_1(\tilde{\Theta}),\Theta-\tilde{\Theta}\right\rangle+\frac{L_{\Theta}}{2}\|\Theta-\tilde{\Theta}\|_F^2+\mathbb{F}_2(\Theta),$$

where $\tilde{\Theta}$ is the extrapolation point. For any $\Theta^k=\{\{\mathbf{U}_n^k\},\mathcal{G}^k\}$ generated by Algorithm 1, it is worth noting that Algorithm 1 takes L_{Θ}^{k-1} as the Lipschitz constant of $\nabla_{\Theta}\mathbb{F}_1(\Theta^k)$, the (16) is satisfied.

$$\begin{aligned}\mathbb{F}_1(\Theta^k)\le{}&\mathbb{F}_1(\Theta^{k-1})+\left\langle\nabla_{\Theta}\mathbb{F}_1(\Theta^{k-1}),\Theta^k-\Theta^{k-1}\right\rangle\\&+\frac{L_{\Theta^{k-1}}}{2}\|\Theta^k-\Theta^{k-1}\|_F^2,\ \text{for any }k=1,\ldots,K.\end{aligned}\tag{16}$$

Considering the convexity of $\mathbb{F}_1,\mathbb{F}_2$, then the proximal gradient inequality assures that

$$\mathbb{F}(\Theta)-\mathbb{F}(\hat{\Theta})\ge\frac{L_{\Theta}}{2}\|\hat{\Theta}-\tilde{\Theta}\|_F^2+L_{\Theta}\left\langle\tilde{\Theta}-\Theta,\hat{\Theta}-\tilde{\Theta}\right\rangle.$$

Given by the results Propositions 1 and 2, we have $\nabla_{\Theta}\mathbb{F}_1(\Theta)$ is Lipschitz continuous, which has bounded Lipschitz constant. At the kth iteration of Algorithm 1, we perform a re-update when $\mathbb{F}\left(\Theta^k\right) < \mathbb{F}\left(\Theta^{k-1}\right)$. Then for three successive $\Theta^{k-2}, \Theta^{k-1}, \Theta^k$, we have

$$\begin{aligned}\mathbb{F}(\Theta^{k-1}) - \mathbb{F}(\Theta^k) \geq\ & \frac{L_{\Theta}^{k-1}}{2}\|\Theta^k - \tilde{\Theta}^{k-1}\|_F^2 + L_{\Theta}^{k-1}\left\langle \tilde{\Theta}^{k-1} - \Theta^{k-1}, \Theta^k - \tilde{\Theta}^{k-1}\right\rangle \\ \geq\ & \frac{L_{\Theta}^{k-1}}{2}\|\Theta^{k-1} - \Theta^k\|_F^2 - \frac{L_{\Theta}^{k-2}}{2}\|\Theta^{k-2} - \Theta^{k-1}\|_F^2, \\ \geq\ & \delta\|\Theta^k - \Theta^{k-1}\|_F^2,\ \ \delta > 0.\end{aligned}$$

It can be inferred that the sequence $\{\mathbb{F}(\Theta^k)\}$ is nonincreasing and lower bounded. Summing the above inequality over k from 1 to K, we have

$$\mathbb{F}(\Theta^0) - \mathbb{F}(\Theta^K) \geq \sum_{k=1}^{K} \text{const.}\ \|\Theta^{k-1} - \Theta^k\|_F^2.$$

Letting $K \to \infty$ and observing $\mathbb{F}$ is lower bounded, we have

$$\sum_{k=1}^{\infty}\left\|\Theta^{k-1} - \Theta^k\right\|_F^2 < \infty,\ \ \lim_{k\to\infty}\left\|\Theta^{k-1} - \Theta^k\right\|_F^2 = 0.$$

Subsequence convergence: We set $\hat{\Theta}$ as a limit point of $\{\Theta^k\}$ depending on the square summable property. Since $\{\mathbb{F}(\Theta^k)\}$ is bounded and continuous, the sequence $\{\Theta^k\}$ is bounded [45]. We ensure the subgradient of $\mathbb{F}$ has a lower bound to generate a stationary point $\hat{\Theta}$. For the given $\Theta^k = \{\{\mathbf{U}_n^k\}, \mathcal{G}^k\}$ and $\Theta^{k-1} = \{\{\mathbf{U}_n^{k-1}\}, \mathcal{G}^{k-1}\}$, we have

$$\mathcal{G}^k = \underset{\mathcal{G}}{\operatorname{argmin}}\ \left\langle \nabla_{\mathcal{G}} f(\mathcal{G}^{k-1}, \{\mathbf{U}_n^{k-1}\}), \mathcal{G} - \mathcal{G}^{k-1}\right\rangle + \frac{\tilde{L}_{\mathcal{G}}}{2}\|\mathcal{G} - \mathcal{G}^{k-1}\|_{\mathrm{F}}^2 + \alpha\|\operatorname{vec}(\mathcal{G})\|_1.$$

The optimality condition yields

$$\nabla_{\mathcal{G}} f(\mathcal{G}^{k-1}, \{\mathbf{U}_n^{k-1}\}) + \alpha\mathbb{P}_{\mathcal{G}^k} = \tilde{L}_{\mathcal{G}}\left(\mathcal{G}^{k-1} - \mathcal{G}^k\right),\ \text{for some } \mathbb{P}_{\mathcal{G}} \in \partial\|\operatorname{vec}(\mathcal{G})\|_1.$$

Since ∇f is Lipschitz continuous, it can be concluded that

$$\begin{aligned}\operatorname{dist}\left(\mathbf{0}, \partial_{\mathcal{G}}\mathbb{F}\left(\Theta^k\right)\right) &\leq \|\nabla_{\mathcal{G}} f(\Theta^k) - \nabla_{\mathcal{G}} f(\mathcal{G}^k, \mathbf{U}_n^{k-1})\|_{\mathrm{F}}^2 + \tilde{L}_{\mathcal{G}}\left(\mathcal{G}^k - \mathcal{G}^{k-1}\right) \\ &\leq \text{const.}\left\|\Theta^k - \Theta^{k-1}\right\|_{\mathrm{F}}^2.\end{aligned} \tag{17}$$

Similarly, we have for all $\mathbf{U}_n$ that

$$\begin{aligned}\operatorname{dist}\left(\mathbf{0}, \partial_{\mathbf{U}_n}\mathbb{F}\left(\Theta^k\right)\right) &\leq \|\nabla_{\mathbf{U}_n}\ell(\Theta^k) - \nabla_{\mathbf{U}_n}\ell(\mathbf{U}_n^{k-1}, \mathcal{G}^k)\|_{\mathrm{F}}^2 + \tilde{L}_{\mathbf{U}_n}\left(\mathbf{U}_n^k - \mathbf{U}_n^{k-1}\right) \\ &\leq \text{const.}\left\|\Theta^k - \Theta^{k-1}\right\|_{\mathrm{F}}^2.\end{aligned} \tag{18}$$

Then, combining the sub-differentiability property of $\mathbb{F}$ and (17)–(18), we have the subgradient lower bound

$$\left\|\mathbb{P}^k\right\|_{\mathrm{F}}^2 \leq \text{const.}\left\|\Theta^k - \Theta^{k-1}\right\|_{\mathrm{F}}^2,\quad \mathbb{P}^k \in \partial\mathbb{F}\left(\Theta^k\right),\quad \forall k = 1, 2, \ldots, K.$$

Referred by Lemma 5 in [46], the limit points set of the sequence Θ^k is compact. Then, the subsequence Θ^k converges to stationary point $\hat{\Theta}$.

Global convergence: It is straightforward to verify $\mathbb{F}_1, \mathbb{F}_2$ are semi-algebraic functions and then demonstrate that $\mathbb{F}$ satisfies the Kurdyka–Lojasiewicz (KL) property [46] at $\hat{\Theta}$, namely, there exists $\mu, \rho > 0, \eta \in [0, 1]$, and a neighborhood $\mathcal{B}(\hat{\Theta}, \rho) = \left\{\Theta : \|\Theta - \hat{\Theta}\|_F^2 \leq \rho\right\}$ such that

$$\|\mathbb{F}(\Theta) - \mathbb{F}(\hat{\Theta})\|^{\eta} \leq \mu \cdot \operatorname{dist}(\mathbf{0}, \partial\mathbb{F}(\Theta)),\ \text{for all } \Theta \in \mathcal{B}(\hat{\Theta}, \rho). \tag{19}$$

Combining the subsequence convergence and KL property, the sequence Θ^k converges to a critical point $\hat{\Theta}$ of Eq. (6). □

4.3. Computational complexity analysis

Suppose that $\mathcal{X}, \mathcal{G} \in \mathbb{R}^{I_1\times\cdots\times I_N}$. The basic computational costs can be summarized as follows: computing $\mathbf{U}_n^{\top}\mathbf{U}_n$ requires $\mathcal{O}(I_n^3)$; the mode-n product of tensor $\mathcal{G}$ with factor matrix $\mathbf{U}_n$ requires $\mathcal{O}\left(I_n\prod_{j=1}^{N} I_j\right)$; and reformulating the Kronecker product in $\mathbf{G}_{\mathbf{V}}^n = \mathbf{G}_{(n)}\mathbf{V}_n^{\top}$ incurs a cost of $\mathcal{O}\left(I_n\left(\prod_{j\neq n} I_j\right)^2\right)$. The computational complexity of our APG-based TuckerAPP algorithm can be analyzed by examining each update step in the iterative procedure:

- Factor matrix updates (10). For modes without additional regularization, the main costs arise from Kronecker product operations $\mathcal{O}\left(\prod_{i\neq n} I_i^2\right)$, matrix multiplications, and the projection step $\mathcal{O}(I_n^2)$. For regularized modes, the graph Laplacian regularization introduces an additional cost of $\mathcal{O}(k \cdot I_n^2)$, while the Toeplitz constraint contributes $\mathcal{O}(I_n^2 \log I_n)$.
- Core tensor update (11). The gradient computation involves mode-n tensor–matrix multiplications with complexity $\mathcal{O}\left(\prod_{i=1}^{N} I_i \cdot \sum_{j=1}^{N} I_j^2\right)$. The ℓ_1-regularization is addressed via soft-thresholding, which requires $\mathcal{O}\left(\prod_{i=1}^{N} I_i\right)$.
- Tensor reconstruction. Reconstructing $\mathcal{X} = \mathcal{G} \times_{n=1}^{N} \mathbf{U}_n$ requires $\mathcal{O}\left(\sum_{n=1}^{N} I_n^2 \prod_{i=1}^{N} I_i\right)$.

Collecting the above results, the per-iteration time complexity of the APG-based TuckerAPP algorithm can be expressed as

$$\mathcal{O}\left(\prod_{i=1}^{N} I_i \cdot \sum_{j=1}^{N} I_j^2 + \sum_{n=1}^{N} I_n \cdot \left(\prod_{i\neq n} I_i\right)^2 + \sum_{n=1}^{K} I_n^3 + \sum_{n=K+1}^{N} I_n^2 \log I_n\right), \tag{20}$$

which is dominated by the factor matrix updates. The overall per-iteration complexity is $\mathcal{O}\left(\sum_{n=1}^{N} I_n \cdot \left(\prod_{i\neq n} I_i\right)^2\right)$, which shows that the proposed algorithm is theoretically efficient [39].

5. Experiments

In this section, we conduct experiments on two traffic datasets to compare the TuckerAPP with baselines in different missing scenarios. The data and code used in the study are available at https://github.com/GongWenwuu/STRTD.

5.1. Experimental setup

5.1.1. Traffic datasets

We use the following two real world datasets for our experiment validation.

- (**G**): Guangzhou urban traffic speed dataset. The original data are of size 214 × 8784 in the form of a multivariate time series matrix. We select seven days for our model training and reshape it into a third-order tensor of size 214 × 144 × 7, that is, (sensors, time, day).
- (**H**): The Hangzhou dataset captures the flow of passengers in transit and consists of 80 metro stations. This dataset records passenger flow every 10 min from January 1 to January 25, 2019. It is structured as a third-order tensor with dimensions 80 × 108 × 25.

To analyze the spatiotemporal characteristics of these data sets, we first calculate the spatial correlations [6] between various pairs of rows in the traffic matrix **Y**. Fig. 2(a) shows the cumulative distribution function (CDF) of the correlation coefficient, indicating that more than 50% of the sensors in two traffic datasets exhibit strong correlations. This observation reveals that the sensor network in datasets **G** and **H** has strong spatial correlations. Fig. 2(b) shows the CDF of the traffic data with the increment rates (IRs) [19]. More than 50% of the IRs of the data vary between 0.1 and 2, indicating temporal correlation in the data. These results imply that the proposed spatiotemporal constraints are essential for our TDI problems.

5.1.2. Missing scenario

For a thorough verification of the TuckerAPP to TDI problem, we take into account three missing scenarios, i.e., random missing (RM), no random missing (NM), and blackout missing (BM). Generally, RM means missing data is uniformly distributed, and NM is performed by randomly selecting sensors and discarding consecutive hours. At the

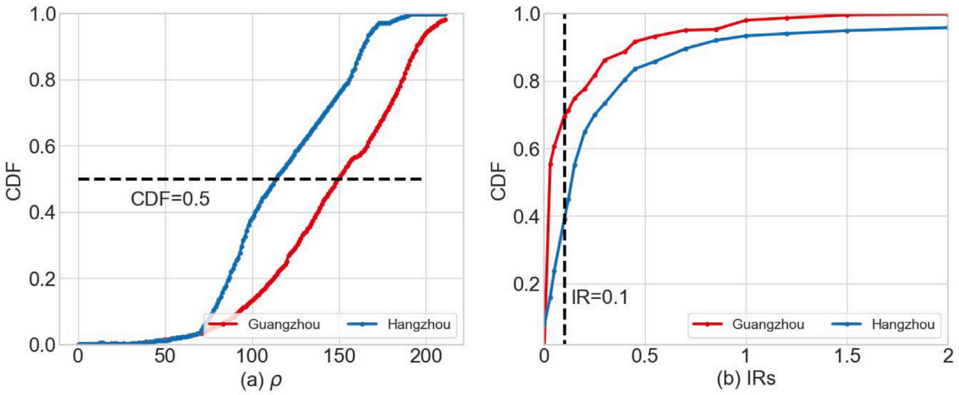

Fig. 2. Interpretation of the spatiotemporal characteristics for traffic datasets.

Table 2
Comparison of baseline models.

Baselines	Priors			Tensor
	Low-rank	Spatial	Temporal	
TuckerAPP	√	√	√	third-order
LR-SETD [10]	√		√	third-order
Hankel [4]	√	√	√	fourth-order
stTT [14]	√	√	√	third-order
LATC [8]	√		√	third-order
LSTC [21]	√		√	third-order
SPC [12]	√	√		third-order

√ denotes the mentioned method has considered the constraint.

same time, BM refers to all sensors that do not work for a certain period. We mask the observed index set Ω and use partial observations for model training according to the mechanisms.

5.1.3. Baseline models

We select six TDI methods for model comparison: the Tucker-based method LR-SETD [10], the CP-based method SPC [12], the SNN-based methods Hankel [4] and LATC [8], the TT-based method stTT [14], and the tubal-based method LSTC [21]. Table 2 shows the different model structure results for tensor-based imputation baselines.

5.1.4. Evaluation metrics

To measure the imputation performance, we adopt two criteria, including mean absolute percentage error (MAPE) and normalized mean absolute error (NMAE):

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100, \quad \text{NMAE} = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\sum_{i=1}^{n}|y_i|} \tag{21}$$

where y_i and $\hat{y}_i$ are actual values and imputed values, respectively.

5.2. Implementation details

5.2.1. Parameters setting

The hyperparameter α adjusts the strength of the sparsity term, that is, the low-rank tensor approximation. In our experiments, α is initially evaluated in a composite missing data scenario, which includes 99% RM, 99% NM, and 95% BM, resulting in a sample ratio (SR) of 0.1. As shown in Fig. 3, an optimal value of α achieves the best performance. Moreover, the impact of α on the imputation error is minimal, emphasizing the robustness of the proposed TuckerAPP model concerning hyperparameter tuning. Furthermore, we evaluated the performance of the dynamic correction parameter ϕ. Fig. 3 shows that setting $\phi = 0.6$ reduces the imputation error in high-level missing scenarios.

5.2.2. Convergence

Fig. 4 (left) illustrates the relative error against the iteration number for our proposed TuckerAPP model in the SR = 0.1 missing scenario. As the iterations progress, the value decreases and stabilizes after roughly 100 iterations. This indicates that the solving algorithm is not sensitive to the initial values and converges well to a critical point.

5.3. Comprehensive results analysis

This section will compare the TuckerAPP method with other baselines mentioned in Table 3. For better model comparison, the termination condition for all experiments is set to (15). Furthermore, the baseline parameters are optimally assigned or automatically chosen as described in the reference papers.

5.3.1. Ablation study

To verify the validity of the spatiotemporal regularization, we discuss the effect of the spatial and temporal constraints in the proposed TuckerAPP model with different RM ratios. In our experiments, we calculate the maximum singular value of spatiotemporal constraint matrices to deliver β_n, γ_n, i.e., $\beta_n = \left\|\mathbf{X}_{(n)}\right\|_2 / 2\sigma \|\mathbf{L}\|_2$ and $\gamma_n = \left\|\mathbf{X}_{(n)}\right\|_2 / 2\sigma \|\mathbf{T}\mathbf{T}^{\mathrm{T}}\|_2$. Fig. 5 illustrates the impact of spatiotemporal constraints on datasets **G** and **H**. The results show that both spatial and temporal regularizations enhance imputation accuracy. Specifically, spatial characteristics play a dominant role in the traffic speed dataset (**G**), while temporal characteristics are more critical in the traffic network dataset (**H**). Moreover, temporal regularization consistently contributes more significantly than spatial regularization. In comparison with other approaches, our method provides a clear and interpretable rationale for modeling spatiotemporal dependencies.

5.3.2. Overall performance comparison

To demonstrate the superiority of the TuckerAPP model, Table 3 presents the overall performance of the baseline models on the **G** and **H** datasets under various missing data scenarios, with the best error indicators highlighted in bold. In particular, TuckerAPP achieves the lowest MAPE and NMAE values, consistently outperforming other methods in all random missing (RM) scenarios. While reconstructing non-random missing (NM) and block missing (BM) scenarios is more challenging, TuckerAPP still delivers robust performance. These findings demonstrate that combining short-term traffic patterns with long-term trends, using low-rank structure and spatiotemporal constraints, dramatically enhances the imputation accuracy of the TuckerAPP model.

We calculate the MAPE and NMAE values in different RM scenarios (SR changes from 0.90 to 0.05) in Fig. 6. The results show that TuckerAPP has the lowest value, even for highly missing ones. Especially when the missing rate is 95%, imputing with the MAPE and NMAE values results in improvements higher than 3%. The results show

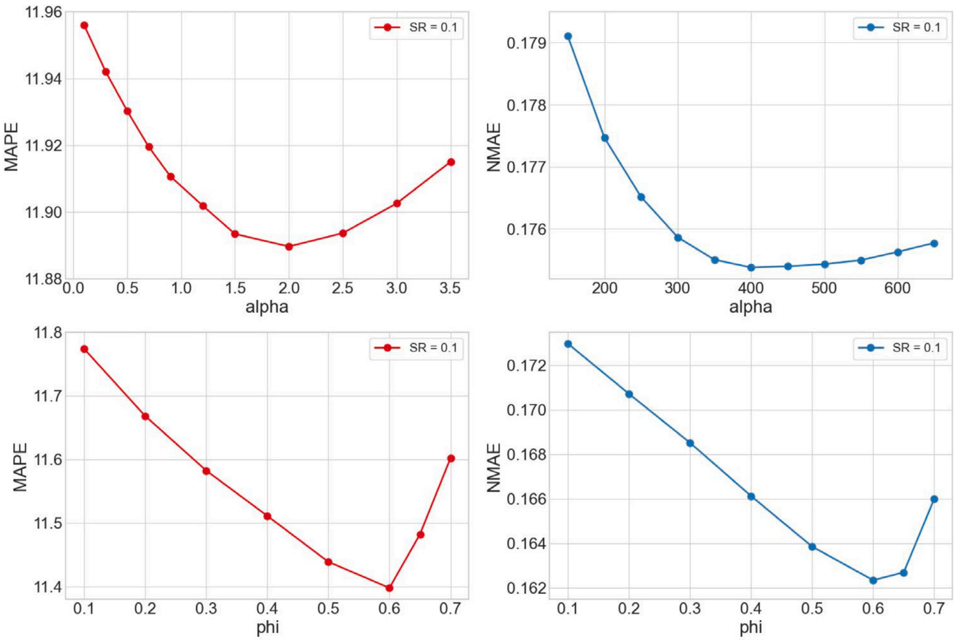

Fig. 3. Hyperparameter tuning on α and ϕ for the traffic tensor $\mathbf{G}$ (left) and $\mathbf{H}$ (right) respectively.

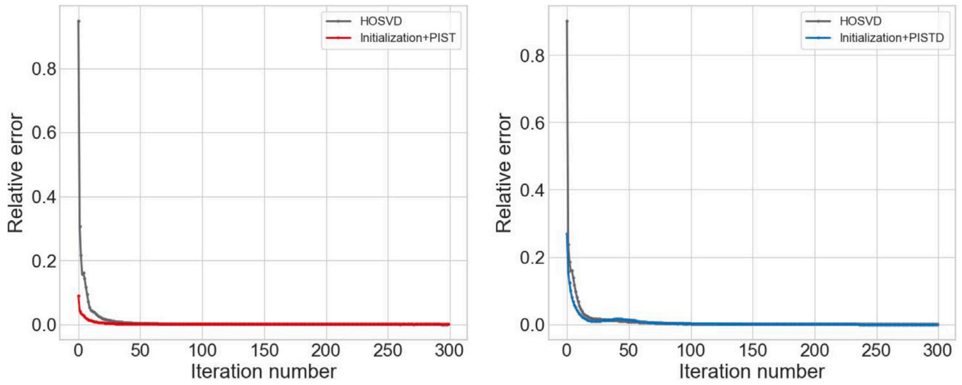

Fig. 4. The relative error for the iterations with SR = 0.1 for the traffic tensors $\mathbf{G}$ (left) and $\mathbf{H}$ (right), respectively.

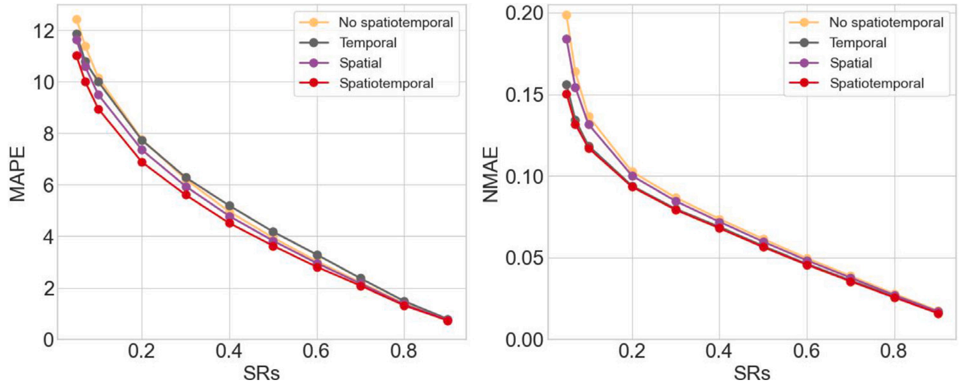

Fig. 5. Illustration of the influence of spatiotemporal constraints for $\mathbf{G}$ (left) and $\mathbf{H}$(right).

that TuckerAPP can impute traffic tensors more accurately with fewer observed data and within a reasonable computational time.

5.3.3. *Case study visualization*

We present several TuckerAPP imputation examples in different missing scenarios using the Guangzhou (**G**) dataset. For random missing (RM), Fig. 7 shows that the TuckerAPP model maintains consistent signal trends across various sample rates (SR), as indicated by the number of purple dots. The results show the model's ability to impute missing traffic accurately. Furthermore, the residuals shown in Fig. 8 highlight that the TuckerAPP model effectively reconstructs the traffic data with high precision, even in extreme cases where 90% of the data are

Table 3
Performance comparison of TuckerAPP and other baselines for RM, NM, and BM scenarios.

Data	Missing	TuckerAPP	LR-SETD	LATC	LSTC	stTT	SPC	Hankel
	RM-30%	**2.08**	2.73	6.46	7.13	10.96	2.17	7.79
	RM-70%	**5.61**	6.58	6.93	6.89	7.88	10.95	11.62
	RM-90%	**8.94**	11.17	10.05	10.44	10.35	11.28	15.11
G	RM-95%	**11.02**	13.55	12.60	16.22	12.25	12.84	17.69
(MAPE)	NM-70%	10.81	21.24	**10.37**	14.76	19.84	15.68	26.43
	NM-90%	**17.48**	57.33	20.89	25.16	22.23	28.97	49.35
	BM-30%	9.74	12.68	**9.23**	10.52	18.12	13.43	9.85
	RM-30%	**0.0497**	0.0501	0.1229	0.1159	0.1196	0.2120	0.3092
	RM-70%	**0.1435**	0.1753	0.1488	0.1935	0.1527	0.2251	0.3178
	RM-90%	**0.2175**	0.2274	0.2328	0.2356	0.2361	0.3764	0.3407
H	RM-95%	**0.2532**	0.2752	0.4809	0.2601	0.3636	0.5124	0.3576
(NMAE)	NM-70%	**0.1279**	0.2722	0.1451	0.2017	0.1297	0.1388	0.6261
	NM-90%	**0.2312**	0.3761	0.2816	0.3051	0.2667	0.3361	0.6928
	BM-30%	0.1071	0.2024	0.1562	0.1785	**0.0437**	0.0422	0.1993

The best results are highlighted in bold fonts.

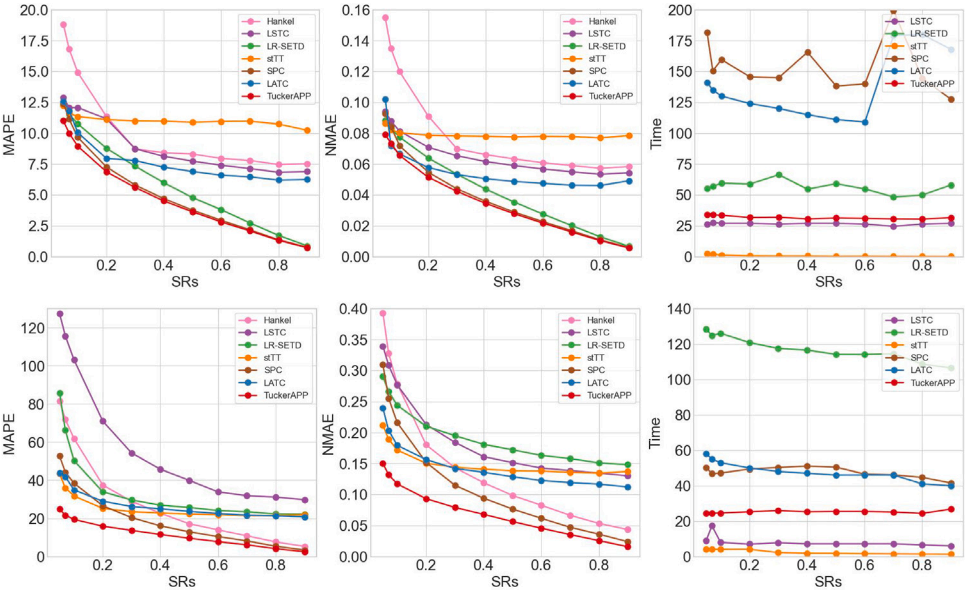

Fig. 6. MAPE, NMAE, and computation time for different sample ratios under RM scenarios for datasets **G** (Top) and **H** (Bottom).

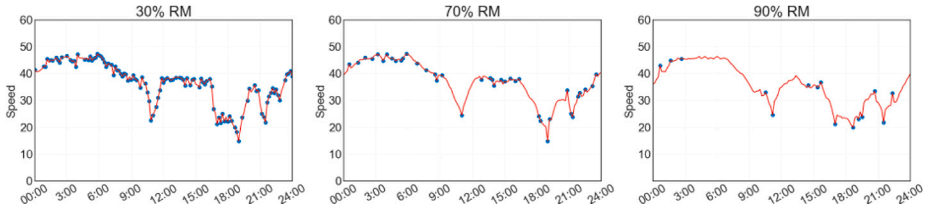

Fig. 7. Results of RM scenario on **G** dataset. This example corresponds to the 81st sensor and the 4th day of the dataset. Purple dots indicate the partially observed data, and red curves indicate the imputed values.

missing. To further validate the superiority of our TuckerAPP, we plot the result of the structurally missing scenarios (NM and BM) in Figs. 9 and 10. In this case, accurate imputation and traffic trend learning are achievable even in severe missing scenarios with TuckerAPP.

6. Conclusion

This paper introduces an enhanced Tucker decomposition framework for the traffic data imputation (TDI) problem. We propose the spatiotemporal regularized Tucker APProach (TuckerAPP) model, which integrates tensor sparsity and spatiotemporal constraints to enhance imputation accuracy under high missing data rates. TuckerAPP captures long-term traffic trends via sparse Tucker decomposition and models short-term dynamics through Laplacian-based spatial regularization and Toeplitz-constrained temporal factors. Furthermore, we develop an efficient and provably convergent alternating proximal gradient algorithm to solve the resulting multi-block separable nonconvex optimization problem. Extensive experiments on two real-world traffic datasets demonstrate that TuckerAPP consistently outperforms existing tensor-based baselines across diverse missing data scenarios, including random and structured missing patterns, as evidenced by Table 3 and Fig. 6.

Future research directions include: First, while TuckerAPP approximates low-rank structure through sparse core tensors and non-negative

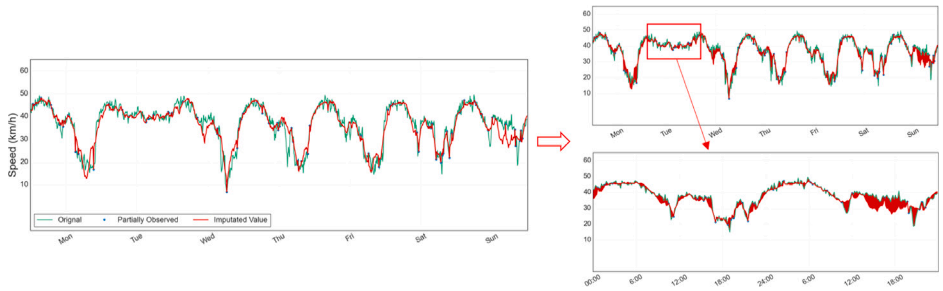

Fig. 8. Imputed values by TuckerAPP on **G** dataset under RM scenario with 90% missing. Note that the red area (residual area) is only used to express the estimation performance, which does not represent the cumulative residual.

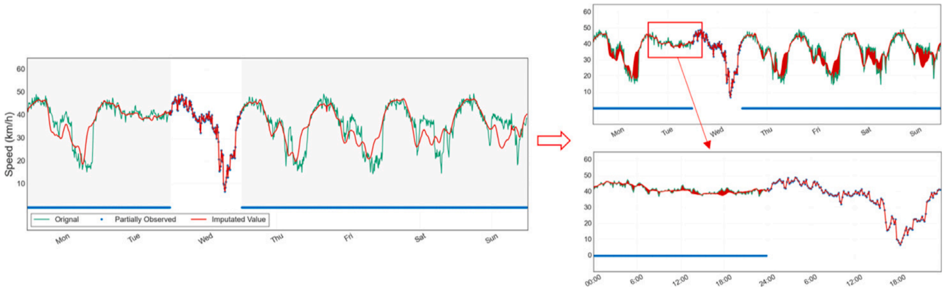

Fig. 9. Imputed values by TuckerAPP on **G** dataset under NM scenario with 70% missing. The gray rectangles indicate the missing area.

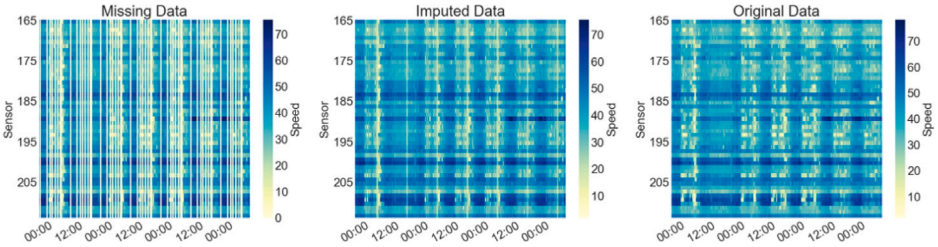

Fig. 10. The visualization of TuckerAPP on **G** dataset under BM scenario with 30% missing. The middle heat map is our TuckerAPP results.

factor matrices, it does not explicitly recover exact Tucker ranks. Future work could explore alternative rank characterizations, such as factor matrix rank products, to more precisely encode Tucker rank [30]. Second, extending TuckerAPP beyond imputation to tasks such as spatiotemporal traffic forecasting [36] and urban mobility pattern analysis [27] under missing data conditions would provide broader utility. In future work, we plan to extend TuckerAPP into a fully functional application platform that integrates automated data preprocessing, missing data recovery, and dynamic visualization for large-scale spatiotemporal datasets.

CRediT authorship contribution statement

Wenwu Gong: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Conceptualization. **Zhejun Huang:** Writing – original draft, Supervision, Funding acquisition. **Jiaxin Lu:** Investigation, Formal analysis. **Lili Yang:** Writing – original draft, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Gong Wenwu reports financial support was provided by Southern University of Science and Technology. Lili Yang reports a relationship with Southern University of Science and Technology that includes: employment. Wenwu Gong reports a relationship with Southern University of Science and Technology that includes: employment and funding grants. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the SUSTech Presidential Postdoctoral Fellowship, Southern Key Laboratory of Technology Finance, and Educational Commission of Guangdong Province (Grant No. 2021ZDZX1069).

Data availability

Data will be made available on request.

References

[1] X. Chen, Z. He, J. Wang, Spatial-temporal traffic speed patterns discovery and incomplete data recovery via SVD-combined tensor decomposition, Transp. Res. Part C Emerg. Technol. 86 (2018) 59–77.

[2] C. Lyu, Q. Lu, X. Wu, C. Antoniou, Tucker factorization-based tensor completion for robust traffic data imputation, Transp. Res. Part C Emerg. Technol. 160 (2024) 104502.

[3] X. Chen, Z. He, Y. Chen, Y. Lu, J. Wang, Missing traffic data imputation and pattern discovery with a Bayesian augmented tensor factorization model, Transp. Res. Part C Emerg. Technol. 104 (2019) 66–77.

[4] X. Wang, Y. Wu, D. Zhuang, L. Sun, Low-rank Hankel tensor completion for traffic speed estimation, IEEE Trans. Intell. Transp. Syst. 24 (5) (2023) 4862–4871.

[5] M.T. Bahadori, Q.R. Yu, Y. Liu, Fast multivariate spatio-temporal analysis via low rank tensor learning, in: Neural Information Processing Systems, NIPS, 2014, pp. 3491–3499.

[6] H. Tan, G. Feng, J. Feng, W. Wang, Y.-J. Zhang, F. Li, A tensor-based method for missing traffic data completion, Transp. Res. Part C Emerg. Technol. 28 (2013) 15–27.

[7] A.B. Said, A. Erradi, Spatiotemporal tensor completion for improved urban traffic imputation, IEEE Trans. Intell. Transp. Syst. (2021) 1–14.

[8] X. Chen, M. Lei, N. Saunier, L. Sun, Low-rank autoregressive tensor completion for spatiotemporal traffic data imputation, IEEE Trans. Intell. Transp. Syst. (2021) 1–10.

[9] B. Ran, H. Tan, Y. Wu, P.J. Jin, Tensor based missing traffic data completion with spatial–temporal correlation, Phys. A Stat. Mech. Appl. 446 (2016) 54–63.

[10] C. Pan, C. Ling, H. He, L. Qi, Y. Xu, A low-rank and sparse enhanced tucker decomposition approach for tensor completion, Appl. Math. Comput. 465 (2024) 128432.

[11] H. Tan, J. Feng, Z. Chen, F. Yang, W. Wang, Low multilinear rank approximation of tensors and application in missing traffic data, Adv. Mech. Eng. 6 (2014) 1575–1597.

[12] T. Yokota, Q. Zhao, A. Cichocki, Smooth PARAFAC decomposition for tensor completion, IEEE Trans. Signal Process. 64 (20) (2016) 5423–5436.

[13] X. Chen, Z. He, L. Sun, A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation, Transp. Res. Part C Emerg. Technol. 98 (2019) 73–84.

[14] Z. Zhang, Y. Chen, H. He, L. Qi, A tensor train approach for internet traffic data completion, Ann. Oper. Res. 06 (339) (2024) 1461–1479.

[15] H. He, C. Ling, W. Xie, Tensor completion via a generalized transformed tensor T-product decomposition without t-SVD, J. Sci. Comput. 93 (2) (2022) 1–35.

[16] X. Xu, M. Lin, X. Luo, Z. Xu, HRST-LR: A hessian regularization spatio-temporal low rank algorithm for traffic data imputation, IEEE Trans. Intell. Transp. Syst. 24 (10) (2023) 11001–11017.

[17] W. Gong, Z. Huang, L. Yang, LSPTD: Low-rank and spatiotemporal priors enhanced tucker decomposition for internet traffic data imputation, in: 2023 IEEE 26th International Conference on Intelligent Transportation Systems, ITSC, 2023, pp. 460–465.

[18] Y. Wu, H. Tan, Y. Li, J. Zhang, X. Chen, A fused CP factorization method for incomplete tensors, IEEE Trans. Neural Netw. Learn. Syst. 30 (3) (2019) 751–764.

[19] Y. Wang, Y. Zhang, X. Piao, H. Liu, K. Zhang, Traffic data reconstruction via adaptive spatial-temporal correlations, IEEE Trans. Intell. Transp. Syst. 20 (4) (2019) 1531–1543.

[20] H.-F. Yu, N. Rao, I.S. Dhillon, Temporal regularized matrix factorization for high-dimensional time series prediction, in: Neural Information Processing Systems, NIPS, 2016, pp. 847–855.

[21] X. Chen, Y. Chen, N. Saunier, L. Sun, Scalable low-rank tensor learning for spatiotemporal traffic data imputation, Transp. Res. Part C Emerg. Technol. 129 (2021) 103226.

[22] J.H. Goulart, A. Kibangou, G. Favier, Traffic data imputation via tensor completion based on soft thresholding of tucker core, Transp. Res. Part C Emerg. Technol. 85 (2017) 348–362.

[23] J. Wang, J. Wu, Z. Wang, F. Gao, X. Zhang, Understanding urban dynamics via context-aware tensor factorization with neighboring regularization, IEEE Trans. Knowl. Data Eng. 32 (11) (2020) 2269–2283.

[24] L. Su, J. Liu, J. Zhang, X. Tian, H. Zhang, C. Ma, Smooth low-rank representation with a grassmann manifold for tensor completion, Knowl.-Based Syst. 270 (2023) 110510.

[25] X. Chen, Z. He, L. Sun, A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation, Transp. Res. Part C Emerg. Technol. 98 (2019) 73–84.

[26] T.G. Kolda, B.W. Bader, Tensor decompositions and application, SIAM Rev. 5 (3) (2009) 455–500.

[27] X. Chen, C. Zhang, X. Chen, N. Saunier, L. Sun, Discovering dynamic patterns from spatiotemporal data with time-varying low-rank autoregression, IEEE Trans. Knowl. Data Eng. (2023).

[28] Q. Shi, J. Yin, J. Cai, A. Cichocki, T. Yokota, L. Chen, M. Yuan, J. Zeng, Block Hankel tensor ARIMA for multiple short time series forecasting, in: AAAI Conference on Artificial Intelligence, AAAI, 2020, pp. 5758–5766.

[29] J.H. de Morais Goulart, G. Favier, Low-rank tensor recovery using sequentially optimal modal projections in iterative hard thresholding, SIAM J. Sci. Comput. 39 (3) (2017) 860–889.

[30] Q. Xie, Q. Zhao, D. Meng, Z. Xu, Kronecker-basis-representation based tensor sparsity and its applications to tensor recovery, IEEE Trans. Pattern Anal. Mach. Intell. 40 (8) (2018) 1888–1902.

[31] X.J. Zhang, M.K. Ng, Sparse nonnegative tensor factorization and completion with noisy observations, IEEE Trans. Inform. Theory 68 (4) (2022) 2551–2572.

[32] N. Gillis, Introduction to nonnegative matrix factorization, 2017, (Preprint) arXiv abs/1703.00663.

[33] M. Roughan, Y. Zhang, W. Willinger, L. Qiu, Spatio-temporal compressive sensing and internet traffic matrices (extended version), IEEE ACM Trans. Netw. 20 (3) (2012) 662–676.

[34] A. Narita, K. Hayashi, R. Tomioka, H. Kashima, Tensor factorization using auxiliary information, Data Min. Knowl. Discov. 25 (2012) 298–324.

[35] Y. Qiu, G. Zhou, Y. Zhang, S. Xie, Graph regularized nonnegative tucker decomposition for tensor data representation, in: 2019 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, 2019, pp. 8613–8617.

[36] X. Chen, L. Sun, Bayesian temporal factorization for multidimensional time series prediction, IEEE Trans. Pattern Anal. Mach. Intell. 44 (9) (2022) 4659–4673.

[37] Q. Yu, X. Zhang, Y. Chen, L. Qi, Low tucker rank tensor completion using a symmetric block coordinate descent method, Numer. Linear Algebra Appl. 30 (3) (2023) e2464.

[38] X. Zhang, M.K. Ng, Sparse nonnegative tensor factorization and completion with noisy observations, IEEE Trans. Inform. Theory 68 (4) (2022) 2551–2572.

[39] W. Gong, Z. Huang, L. Yang, Accurate regularized tucker decomposition for image restoration, Appl. Math. Model. 123 (11) (2023) 75–86.

[40] T.K. Sinha, J. Naram, P. Kumar, Nonnegative low-rank tensor completion via dual formulation with applications to image and video completion, in: IEEE/CVF Winter Conference on Applications of Computer Vision, WACV, 2022, pp. 3732–3740.

[41] Y. Xu, Alternating proximal gradient method for sparse nonnegative tucker decomposition, Math. Program. Comput. 5 (3) (2015) 455–500.

[42] J. Liang, T. Luo, C.-B. Schönlieb, Improving "fast iterative shrinkage-thresholding algorithm": Faster, smarter, and greedier, SIAM J. Sci. Comput. 44 (3) (2022) A1069–A1091.

[43] Y. Xu, W. Yin, A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion, SIAM J. Imaging Sci. 6 (3) (2013) 1758–1789.

[44] Y. Yang, M. Pesavento, Z.-Q. Luo, B. Ottersten, Inexact block coordinate descent algorithms for nonsmooth nonconvex optimization, IEEE Trans. Signal Process. 68 (2020) 947–961.

[45] J. Li, W. Zhou, X. Li, Proximal alternating partially linearized minimization for perturbed compressive sensing, IEEE Trans. Signal Process. 71 (2023) 3373–3384.

[46] J. Bolte, S. Sabach, M. Teboulle, Proximal alternating linearized minimization for nonconvex and nonsmooth problems, Math. Program. 146 (7) (2014) 459–494.