POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Matrix and Tensor Models for Spatiotemporal Traffic Data Imputation and Forecasting

XINYU CHEN
Département de Département des génies civil, géologique et des mines

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*
Génie civil

Décembre 2023

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Cette thèse intitulée :

Matrix and Tensor Models for Spatiotemporal Traffic Data Imputation and Forecasting

présentée par **Xinyu CHEN**
en vue de l'obtention du diplôme de *Philosophiæ Doctor*
a été dûment acceptée par le jury d'examen constitué de :

Francesco CIARI, président
Nicolas SAUNIER, membre et directeur de recherche
Lijun SUN, membre et codirecteur de recherche
James GOULET, membre
Guillaume RABUSSEAU, membre externe

DEDICATION

To those who are dear to me

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Prof. Nicolas Saunier (advisor) and Prof. Lijun Sun (co-advisor) for their exceptional guidance during my PhD studies. I have learned a lot from both of them, not only about research and technical matters but also about life in general. Prof. Saunier and Prof. Sun have supervised my PhD research on spatiotemporal data modeling with machine learning. Throughout this process, they provided me with plenty of freedom to pursue my research interests while engaging in great discussions and offering insightful feedback on new ideas and attempts. Without their support, the thesis presented herein would not have materialized. Their exceptional supervision helped me complete my PhD, and I feel extremely fortunate to have had both of them as my advisors.

I also feel fortunate to have met and worked with many amazing people over the last few years. The discussions and collaborations with Xiaoxu Chen, Zhanhong Cheng, Mengying Lei, Xudong Wang, and Chengyuan Zhang at McGill University significantly improved the quality of my PhD research. Prof. HanQin Cai at the University of Central Florida provided great discussions for solving optimization problems and analyzing time complexity and convergence of algorithms. For formulating the traffic flow modeling problems with mathematical tools in my thesis, I would like to especially thank Prof. Xi-Le Zhao at the University of Electronic Science and Technology of China for engaging in many interesting conversations. He provided great support and ideas for solving complicated optimization problems. Without his help, the inherent methodological challenges would have made this research even harder to present in the current version. Additionally, I would like to acknowledge the great feedback and suggestions from anonymous reviewers. Though I do not know their names, they helped improve the quality of each small study submitted to journals and conferences. Of course, any remaining mistakes are entirely my own.

I would like to express my gratitude to IVADO and CIRRELT for generously providing scholarships, allowing me to pursue research topics without any financial burden. My wife, Fangfang Zheng, provided love, support, and encouragement throughout my PhD studies, helping me stay motivated during the many research attempts that yielded no results. Finally, I want to express my gratitude to my friends for consistently encouraging me to take breaks from the computer when I needed essential moments of enjoyment and relaxation!

RÉSUMÉ

De nos jours, les technologies de détection avancées et les systèmes d'information constituent la base pour recueillir des données de trafic spatiotemporel à partir des systèmes de transport, tels que les réseaux routiers urbains et autoroutiers. Ces données peuvent ensuite soutenir de nombreuses applications des systèmes de transport intelligents (STI). Cependant, la qualité des données reste insatisfaisante en raison des mécanismes de collecte de données et des problèmes liés aux systèmes de détection, par exemple des problèmes de données manquantes. Dans cette thèse, nous abordons les problèmes réels de données manquantes dans les systèmes de transport et visons de développer des méthodes d'imputer et de prévoir des données de trafic spatiotemporel en présence de valeurs manquantes grâce à l'apprentissage automatique. Au cours du processus de modélisation des données, nous proposons une série de modèles de complétion de matrices et de tenseurs à faible rang dans lesquels à la fois la propriété à faible rang et les corrélations temporelles des données de trafic sont correctement caractérisées.

Dans le cadre des tâches d'imputation de données de trafic, nous introduisons des techniques de modélisation temporelle spécifiques, telles que l'autorégression, le lissage temporel et la Hankelisation, dans les méthodes de matrices et de tenseurs à faible rang. Ces modèles résultants sont capables de découvrir des motifs à faible rang et les tendances globales et locales des séries temporelles des données de trafic spatiotemporelles. Tout d'abord, nous présentons un modèle de complétion de tenseur à faible rang qui intègre le processus d'autorégression dans la minimisation de la norme nucléaire du tenseur tronqué. Ce modèle peut caractériser à la fois les tendances non locales et locales des séries temporelles dans les données de trafic. Ensuite, nous introduisons un modèle de faible rang unifié qui repose sur la minimisation de la norme nucléaire de matrices circulantes. Nous proposons d'intégrer une régularisation temporelle sous forme de convolution circulaire dans le cadre à faible rang. En conséquence, ce modèle peut capturer à la fois les tendances globales et locales des données de trafic. Un avantage remarquable de ce modèle est sa mise en œuvre rapide à l'aide de la transformée de Fourier rapide, ce qui le rend adapté aux problèmes d'imputation de grandes quantités de données de trafic. Troisièmement, nous abordons le défi de l'imputation de données de trafic manquantes extrêmes à l'aide de la factorisation de tenseur de Hankel. Le modèle proposé avec une paramétrisation par convolution permet de capturer automatiquement les corrélations spatiotemporelles à partir de très faibles quantités de données de trafic. Enfin, nous évaluons les modèles proposés sur plusieurs ensembles de données de trafic du monde réel et confirmons leur efficacité par rapport aux modèles de référence.

Dans les tâches de prévision du trafic avec très peu de données, nous avons développé un cadre efficace de prévision de séries temporelles basé sur la factorisation matricielle temporelle. Pour capturer la non-stationnarité et les tendances des séries temporelles dans les données de trafic, nous introduisons une différenciation saisonnière dans la matrice de facteurs temporels de basse dimension et modélisons les corrélations des séries temporelles des facteurs temporels à l'aide de la vectorisation autorégressive. Le modèle proposé offre un cadre efficace pour l'apprentissage à partir de données de trafic partiellement observées et la prédiction. Nous évaluons le modèle à l'aide de deux ensembles de données sur l'état du trafic à l'échelle de la ville, démontrant ainsi sa pertinence pour la prévision de flux de trafic avec très peu de données et à grande dimension.

En résumé, cette thèse présente des méthodes de matrices/tenseurs à faible rang, dont les idées fondamentales établissent de nombreuses nouvelles connexions avec la modélisation des séries chronologiques (temporelles). Nous démontrons que les modèles d'imputation et de prévision proposés peuvent faire face aux défis des données manquantes dans les données de trafic spatiotemporel, ce qui est important pour les systèmes de transport intelligents pilotés par les données.

ABSTRACT

Nowadays, advanced sensing technologies and information systems provide the basis for gathering spatiotemporal traffic data from real-world transportation systems, such as urban road networks and highway networks. These data can further support numerous intelligent transportation system (ITS) applications. However, data quality is often unsatisfactory due to data collection mechanisms and issues with sensing systems. In this thesis, we address real-world missing data problems in transportation systems and aim to answer how to impute and forecast spatiotemporal traffic data in the presence of missing values with machine learning. During the data modeling process, we propose a series of low-rank matrix and tensor completion models in which both the low-rank property and temporal correlations of traffic data are properly characterized.

In the traffic data imputation tasks, we introduce specific temporal modeling techniques, such as autoregression, temporal smoothing, and Hankelization, into the low-rank matrix and tensor methods. These resulting models are capable of discovering low-rank patterns and global/local time series trends in spatiotemporal traffic data. First, we present a low-rank tensor completion model that integrates the autoregression process into truncated tensor nuclear norm minimization. This model can characterize both nonlocal and local time series trends in traffic data. Second, we introduce a unified low-rank model that builds on the circulant matrix nuclear norm minimization. We propose integrating temporal regularization in the form of circular convolution into the low-rank framework. As a result, this model can capture both global and local trends in traffic data. One remarkable advantage of this model is its fast implementation using fast Fourier transform, making it well-suited for large traffic data imputation problems. Third, we address the challenge of extreme missing traffic data imputation using the Hankel tensor factorization. The proposed model with convolutional parameterization allows for the automatic capture of spatiotemporal correlations from sparse traffic data. Finally, we evaluate the proposed models on several real-world traffic datasets and confirm their effectiveness compared to the baseline models.

In sparse traffic forecasting tasks, we have developed an efficient time series forecasting framework based on temporal matrix factorization. To capture the nonstationarity and time series trends in traffic data, we introduce seasonal differencing to the low-dimensional temporal factor matrix and model time series correlations of temporal factors using vector autoregression. Our proposed model offers an efficient framework for learning from partially observed traffic data and making predictions. We evaluate the model using two city-scale

traffic state datasets, demonstrating its suitability for high-dimensional and sparse traffic flow forecasting.

Overall, this thesis introduces low-rank matrix/tensor methods, whose core ideas establish many new connections with (temporal) time series modeling. We demonstrate that the proposed imputation and forecasting models can cope with the challenges of missing data in spatiotemporal traffic data, which are important for data-driven intelligent transportation systems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

ADMM	Alternating Direction Method of Multipliers
BM	Block-Out Missing
BTMF	Bayesian Temporal Matrix Factorization
CircNNM	Circulant Matrix Nuclear Norm Minimization
ConvNNM	Convolution Matrix Nuclear Norm Minimization
CP	CANDECOMP/PARAFAC
FFT	Fast Fourier Transform
GP	Gaussian Process
GPS	Global Positioning System
HaLRTC	High Accuracy Low-Rank Tensor Completion
HTF	Hankel Tensor Factorization
ITS	Intelligent Transportation System
LAMC	Low-Rank Autoregressive Matrix Completion
LATC	Low-Rank Autoregressive Tensor Completion
LRMC	Low-Rank Matrix Completion
LRTC	Low-Rank Tensor Completion
LRTC-TNN	Low-Rank Tensor Completion with Truncated Nuclear Norm
LCR	Laplacian Convolutional Representation
MAPE	Mean Absolute Percentage Error
MCMC	Markov chain Monte Carlo
NGSIM	Next Generation Simulation
NM	Non-Random Missing
NoTMF	Nonstationary Temporal Matrix Factorization
NYC	New York City
PeMS	Performance Measurement System
RM	Random Missing
RMSE	Root Mean Square Error
SMF	Smoothing Matrix Factorization
SPC	Smooth PARAFAC Tensor Completion
SVD	Singular Value Decomposition
TMF	Temporal Matrix Factorization
TRMF	Temporal Regularized Matrix Factorization
VAR	Vector Autoregression

CHAPTER 1 INTRODUCTION

1.1 Motivation

With recent advances in sensing technologies, large-scale, high-dimensional, and multidimensional spatiotemporal traffic data, including information on traffic volume and speed, are collected continuously. This data is gathered from both fixed traffic sensing systems (e.g., loop detectors, radar detectors, and video cameras) and crowdsourcing/floating sensing systems (e.g., trajectories from smartphone users). Because spatiotemporal traffic data are collected from various sensors and applications with specific time resolutions, they can be classified as a unique type of multivariate/multidimensional time series [1]. In the sensing system of transportation networks, sensors such as inductive loop detectors and video cameras can record the information of each passing vehicle or count the total passing vehicles with a certain time resolution as traffic measurements such as volume and speed. One example is the highway traffic sensing system, known as the Performance Measurement System (PeMS), administered by the California Department of Transportation.[1] It comprises more than 35,000 detectors and has been gathering traffic flow and speed information with a 30-second time resolution (i.e., 2,880 time steps per day) since 1999 [2]. The collected dataset is both large-scale and high-dimensional, and furthermore, PeMS provides the health report for the sensing system, including sensor malfunctioning and missing data problems.

Using mobile sensing systems, the development of global navigation satellite systems such as global positioning system (GPS) makes the positioning equipment available for recording vehicles' positions at each timestamp and constructing spatiotemporal trajectory/movement data as the profiles of vehicles. On transportation networks, it has become common to record the position information of probe vehicles, including taxis, buses, and ridesharing vehicles, and send their position information to a traffic data center at fixed time intervals. With an acceptable penetration of probe vehicles, it is possible to estimate city-wide traffic states from trajectory data of taxis or ridesharing vehicles. In the Uber Movement project[2], the trajectory dataset is an important source for estimating hourly traffic speed information. Take the London Movement Speed dataset created by the Uber Movement project as an example; the dataset includes the hourly traffic speed information of about 220,000 road segments in

[1] PeMS system is available at `http://pems.dot.ca.gov/`.

[2] The project is publicly available at `https://movement.uber.com/`. The project provides the average speed on a given road segment for each hour of each day in the specific month. To account for the issue of insufficient sampling, the speed data are aggregated and computed when road segments have at least 5 unique trips within that hour.

urban areas. As another example, the New York City (NYC) Movement Speed dataset covers the hourly traffic speed information of about 100,000 road segments. Analyzing these data requires us to address the issue of high dimensionality.

Due to the availability of these traffic data, spatiotemporal traffic data modeling has become a meaningful but technically challenging research direction in transportation science over the past few decades. These advances could bring many benefits to real-world transportation systems [3,4]. Traffic data have provided us with unprecedented opportunities for sensing road traffic flow dynamics and human mobility patterns, allowing for the development of reliable and advanced Intelligent Transportation Systems (ITS). For instance, forecasting the demand and states (e.g., volume and speed) of transportation networks is essential for a wide range of ITS applications [5], including traffic state monitoring, travel time estimation, trip/route planning, and traffic signal control, to name just a few. Data-driven approaches in the machine learning field provide appropriate solutions to harness the power and value of traffic data. Despite the vast body of literature on spatiotemporal traffic data modeling in recent years, it is still necessary to address several emerging challenges arising from spatiotemporal correlations, complex traffic flow dynamics, and multiple data behaviors.

First, *missing data is one of the most longstanding and representative problems in spatiotemporal traffic data.* Spatiotemporal traffic data collected from transportation systems are inevitably incomplete due to several reasons. The missing data problem may arise from sensor malfunctioning, communication failures, maintenance issues, and sparse sensing. Another cause of missing data problems is insufficient sensor coverage in both spatial and temporal dimensions. For example, floating cars in an urban transportation network can provide a good sample of real-time traffic information. However, the data is naturally sparse and far from satisfactory for supporting city-wide and fine-resolution traffic speed and travel time monitoring. In such cases, the missing data problem presents both methodological and practical challenges, making it difficult to extract accurate signals from the data. Moreover, various missing data scenarios (e.g., complicated patterns and high missing rates) also pose challenges for missing data reconstruction.

Second, *it is technically challenging and computationally expensive to process large-scale and high-dimensional traffic datasets.* Nowadays, the development of sensing technologies makes it easy to collect large-scale spatiotemporal traffic data. However, processing and analyzing large-scale traffic data is challenging. In some ITS applications (e.g., online traffic forecasting) with real-time constraints, there is an emphasis on efficiency over accuracy.

Third, *it is worth modeling spatiotemporal correlations, multiple data behaviors, and complex temporal correlations in traffic data using a data-driven framework.* Spatiotemporal traffic

data are, by nature, matrices and tensors specified by spatial, temporal, and domain-specific dimensions. In transportation systems, traffic data exhibit strong spatial dependencies; for instance, there is a robust spatial structure (e.g., related to the transportation networks) underlying the observed traffic data. Simultaneously, traffic measurement data often evolve over time and display strong temporal dependencies and patterns. Therefore, utilizing a matrix or tensor structure allows for the interpretation of spatiotemporal traffic data with their intrinsic characteristics. However, since multiple data behaviors include incompleteness, high dimensionality, multidimensionality, and noisy measurements, discovering a unified framework to account for these data behaviors is challenging. Thus, we consider addressing these challenges in spatiotemporal traffic data imputation and forecasting tasks. This thesis proposes data-driven solutions for spatiotemporal traffic data imputation and forecasting.

1.2 Problem Definition

In this section, we summarize the scientific problems addressed in this thesis, which include univariate/multivariate traffic time series imputation and traffic forecasting in the presence of sparse data.

1.2.1 Univariate Traffic Time Series Imputation

In transportation systems, sensors collect traffic measurement data from each spatial location on a continuous basis, thereby generating a time series sequence that exhibits both global and local trends due to the intrinsic traffic flow dynamics and human mobility patterns. The global trends include daily rhythms and seasonality, while the local trends imply the consistency of data points within consecutive time steps. The concern is how to find a unified framework that takes into account both the global and local trends of traffic time series.

Formally, for any partially observed time series $\boldsymbol{y} \in \mathbb{R}^T$ (i.e., involving T time steps[3]) with observed index set Ω, the goal is to reconstruct the missing values, namely, $\mathcal{P}_{\Omega}^{\perp}(\boldsymbol{y})$, from the partial observations $\mathcal{P}_{\Omega}(\boldsymbol{y})$. Herein, $\mathcal{P}_{\Omega} : \mathbb{R}^T \to \mathbb{R}^T$ denotes the orthogonal projection supported on the observed index set Ω, while $\mathcal{P}_{\Omega}^{\perp} : \mathbb{R}^T \to \mathbb{R}^T$ denotes the orthogonal projection supported on the complement of Ω. On the vector $\boldsymbol{y} \in \mathbb{R}^T$ with observed index set Ω, the

[3]Time step in this thesis refers to the time interval between measurements, also called time resolution (e.g., 5-min or hourly).

operator $\mathcal{P}_\Omega(\cdot)$ can be described as follows,

$$[\mathcal{P}_\Omega(\boldsymbol{y})]_t = \begin{cases} y_t, & \text{if } t \in \Omega, \\ 0, & \text{otherwise}, \end{cases} \tag{1.1}$$

where $t = 1, 2, \ldots, T$.

1.2.2 Multivariate Traffic Data Imputation

In transportation systems, if the sensor network collects traffic data from N spatial locations, then the generated traffic data is indeed a multivariate time series in the form of a matrix. Formally, for any partially observed time series $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ consisting of N variables and T time steps, if its observed index set is denoted by Ω, then the goal is to reconstruct the missing values, namely, $\mathcal{P}_\Omega^\perp(\boldsymbol{Y})$, from the partial observations $\mathcal{P}_\Omega(\boldsymbol{Y})$. Herein, $\mathcal{P}_\Omega : \mathbb{R}^{N \times T} \to \mathbb{R}^{N \times T}$ denotes the orthogonal projection supported on the observed index set Ω, while $\mathcal{P}_\Omega^\perp : \mathbb{R}^{N \times T} \to \mathbb{R}^{N \times T}$ denotes the orthogonal projection supported on the complement of Ω.

For instance, on the data matrix $\boldsymbol{Y}$ with observed index set Ω, we have the following definition:

$$[\mathcal{P}_\Omega(\boldsymbol{Y})]_{n,t} = \begin{cases} y_{n,t}, & \text{if } (n,t) \in \Omega, \\ 0, & \text{otherwise}, \end{cases} \tag{1.2}$$

where $n = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T$.

On such an imputation task, the model is required to achieve an efficient learning process from the limited observations. The aim is to perform accurate and efficient imputation on incomplete traffic data with different missing patterns and different degrees of missing data. Essentially, the difficulty of the imputation depends on different factors such as missing patterns, missing rate, and data scale. In the following, we consider solving these practical issues and develop some matrix and tensor completion solutions.

1.2.3 Traffic Forecasting on Sparse Data

In this work, we focus on sparse traffic state forecasting and address the following data behaviors simultaneously:

- High-dimensionality: The data is of high dimension and large scale, encompassing thousands of road segments.
- Sparsity: Only a small fraction of data is observed due to the nature of the data

collection process, such as insufficient sampling.

To represent the traffic state measurements collected from N road segments over T time steps, we use a data matrix $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ consisting of the column vectors $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T \in \mathbb{R}^N$. Here, N is the number of road segments, and T is the number of time steps. The set Ω denotes the index set of observed entries in the data matrix $\boldsymbol{Y}$. The objective is to forecast traffic state vectors $\boldsymbol{y}_{T+1}, \ldots, \boldsymbol{y}_{T+\delta}, \delta \in \mathbb{N}^+$ for a given forecasting time horizon $\delta \in \mathbb{N}^+$, using the observations $\mathcal{P}_\Omega(\boldsymbol{Y})$. For better illustration, refer to Figure 1.1, which depicts the forecasting task on traffic time series.

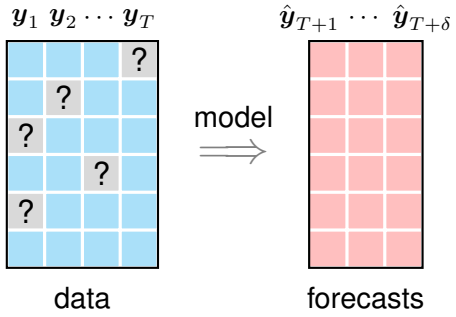


Figure 1.1 Illustration of the traffic state forecasting task on the incomplete data $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T$. The goal is to produce the future traffic state vectors $\hat{\boldsymbol{y}}_{T+1}, \ldots, \hat{\boldsymbol{y}}_{T+\delta}$ accurately.

To obtain satisfactory forecasting performance, we need to address the following concerns:

- How to perform traffic state forecasting with a small fraction of observations? It requires us to overcome the sparsity issue in traffic state data and present a well-suited learning and forecasting mechanism on sparse inputs.
- How to apply the matrix factorization framework to a large-scale and high-dimensional setting? While matrix factorization with temporal modeling is well-suited to time series forecasting with missing values [1, 6], it still requires us to develop an efficient implementation of the proposed matrix factorization model on large traffic state data.

Unlike the classical traffic forecasting solutions which require fully observed data as inputs, this research focuses on addressing traffic state forecasting on imperfect and sparse data. Recent research has recognized the importance and challenges of analyzing sparse time series data [1, 6]. In the following, we consider a data-driven approach by applying the low-rank matrix factorization framework with proper temporal modeling, expecting to simultaneously address the high-dimensionality and sparsity issues in traffic state data.

1.3 Contributions

This research presents several low-rank matrix and tensor models for spatiotemporal traffic data imputation and traffic forecasting in the presence of sparse data. Despite existing efforts to introduce low-rank structures for traffic data in the literature, the proposed matrix and tensor models are tailored to specific domain settings. This includes introducing the day dimension to construct tensor data and incorporating autoregression and local trend smoothing to appropriately model time series trends. These concepts offer valuable insights for both methodology and practical applications, rather than merely extending previous models to a specific domain. The study also contributes by developing rigorous low-rank matrix and tensor algorithms for spatiotemporal traffic data. As shown in Figure 1.2, we propose to reconstruct partially observed traffic data $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ by some low-rank matrix and tensor models in which both low-rank property and temporal correlations of traffic data are characterized simultaneously. As can be seen, these models demonstrate certain modeling purposes and scenarios, ranging from imputation to forecasting.

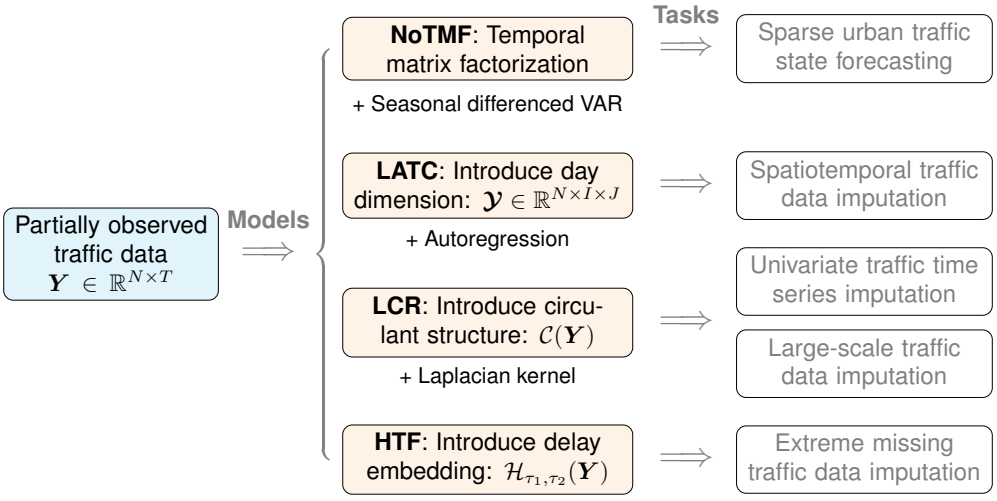


Figure 1.2 Brief summary of the proposed models for spatiotemporal traffic data modeling.

The primary contributions of this research are summarized as follows,

- To efficiently impute and forecast sparse traffic data, we propose discovering the low-rank property of traffic data through matrix factorization and characterizing the time series trends of sparse observations using seasonal differenced vector autoregression (VAR) based on the low-dimensional (latent) temporal factors. For detailed development of the nonstationary temporal matrix factorization (NoTMF) model, please refer to Section 3.1.

- To leverage the multidimensionality of traffic data and achieve accurate imputation, we propose a unified tensor completion framework. This framework constructs the traffic tensor by introducing a day dimension and strengthens the model using temporal autoregression. For detailed development of low-rank autoregressive tensor completion (LATC), please refer to Section 3.2.

- To find an efficient imputation solution for large-scale traffic data, we propose a low-rank completion framework that employs circulant matrix/tensor nuclear norm minimization and temporal regularization with Laplacian kernels. The resulting model can be efficiently solved using the fast Fourier transform (FFT) due to the circulant structure and circular convolution property. For a detailed development of Laplacian convolutional representation (LCR), please refer to Section 3.3.

- To address extreme missing data scenarios, we introduce a Hankel tensor structure for sparse traffic data and propose an efficient convolutional parameterization in the tensor factorization formula. For detailed development of memory-efficient Hankel tensor factorization (HTF), please refer to Section 3.4.

- Finally, in Chapter 4, we evaluate the aforementioned imputation and forecasting models on various spatiotemporal data and empirically verify the effectiveness of these models. It is worth noting that these models show different modeling purposes and application scenarios.

1.4 Organization

In the following chapters, we cover the content of this thesis as follows,

- Chapter 2 gives a summary of the related work to this study.

- Chapter 3 introduces a sequence of low-rank matrix and tensor models that are proposed for spatiotemporal traffic data imputation and forecasting, including NoTMF, LATC, LCR, and HTF.

- Chapter 4 conducts extensive experiments on spatiotemporal traffic datasets with the proposed models and evaluates their performance for traffic data imputation and forecasting.

- Chapter 5 concludes the whole thesis with remarks, limitations, and future research directions.

The work composing this thesis has resulted in the following publications/preprints:

- Xinyu Chen, Mengying Lei, Nicolas Saunier, Lijun Sun. Low-rank autoregressive tensor completion for spatiotemporal traffic data imputation. IEEE Transactions on Intelligent Transportation Systems, 2022, 23 (8): 12301–12310.
- Xinyu Chen, Lijun Sun. Bayesian temporal factorization for multidimensional time series prediction. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, 44 (9): 4659–4673.
- Xinyu Chen, Zhanhong Cheng, Nicolas Saunier, Lijun Sun (2022). Laplacian convolutional representation for traffic time series imputation. arXiv: 2212.01529.

It has also been presented at the following conferences/workshops:

- Xinyu Chen, Mengying Lei, Nicolas Saunier, Lijun Sun. Low-rank autoregressive tensor completion for spatiotemporal traffic data imputation. The 7th SIGKDD Workshop on Mining and Learning from Time Series (MiLeTS), 2021.
- Xinyu Chen, Chengyuan Zhang, Lijun Sun, Nicolas Saunier. Nonstationary temporal matrix factorization for sparse traffic time series forecasting. The 102nd Annual Meeting of Transportation Research Board (TRB24), 2023.
- Xinyu Chen, Zhanhong Cheng, Nicolas Saunier, Lijun Sun. Laplacian convolutional representation for traffic time series imputation. In Proceedings of the World Conference on Transportation Research (WCTR), 2023.

Details on related work lying outside the scope of this thesis can be found here:

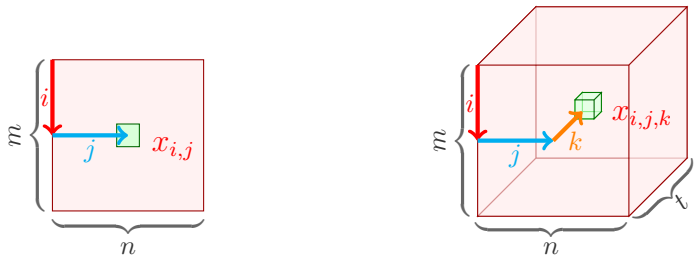
- Xinyu Chen, Yixian Chen, Nicolas Saunier, Lijun Sun. Scalable low-rank tensor learning for spatiotemporal traffic data imputation. Transportation Research Part C: Emerging Technologies, 2020, 117: 102673.
- Xinyu Chen, Chengyuan Zhang, Xiaoxu Chen, Nicolas Saunier, Lijun Sun. Discovering dynamic patterns from spatiotemporal data with time-varying low-rank autoregression. IEEE Transactions on Knowledge and Data Engineering, 2023, early access.

1.5 Use of Notation

Throughout this thesis, we follow the notation of matrix/tensor computations in [7, 8]. Essentially, we have the following symbols across the whole thesis:

- $\mathbb{R}$ denotes the set of real numbers, e.g., $\mathbb{R}^d$ is the set of d-dimensional vector over $\mathbb{R}$.
- $\mathbb{N}$ denotes the set of natural numbers, while $\mathbb{N}^+$ denotes the set of positive natural numbers.

We use boldface uppercase letters to denote matrices, e.g., $\boldsymbol{X} \in \mathbb{R}^{m\times n}$, boldface lowercase letters to denote vectors, e.g., $\boldsymbol{x} \in \mathbb{R}^n$, and lowercase letters to denote scalars, e.g., $x \in \mathbb{R}$. In particular, we denote a third-order tensor by $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{m\times n\times t}$ and the tensor unfoldings of $\boldsymbol{\mathcal{X}}$ by $\boldsymbol{\mathcal{X}}_{(1)} \in \mathbb{R}^{m\times(nt)}$, $\boldsymbol{\mathcal{X}}_{(2)} \in \mathbb{R}^{n\times(mt)}$, and $\boldsymbol{\mathcal{X}}_{(3)} \in \mathbb{R}^{t\times(mn)}$, following the rules as detailed in [7]. As shown in Figure 1.3, it is not hard to intuitively distinguish the difference between a matrix and a (third-order) tensor. While a matrix has rows and columns, a tensor has at least three dimensions. In this case, the (i, j)-th entry of the matrix $\boldsymbol{X}$ is written as $x_{i,j}$ (or sometimes x_{ij}), as the (i, j, k)-th entry of the tensor $\boldsymbol{\mathcal{X}}$ is written as $x_{i,j,k}$.



(a) $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ (b) $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{m\times n\times t}$

Figure 1.3 Illustration of matrix and tensor. (a) The matrix $\boldsymbol{X}$ is of size $m \times n$, in which each entry is denoted by $x_{i,j}$. (b) The third-order tensor $\boldsymbol{\mathcal{X}}$ is of size $m \times n \times t$, in which each entry is denoted by $x_{i,j,k}$.

CHAPTER 2 LITERATURE REVIEW

2.1 Spatiotemporal Traffic Data Imputation

Spatiotemporal traffic data record timestamped traffic state variables (e.g., traffic volume and speed) from various locations, typically collected through a network of sensors. These data are crucial for a wide range of ITS applications. Unfortunately, spatiotemporal traffic data often suffer from the challenge of missing data, which limits their utility in real-world applications. For example, PeMS has been collecting data from thousands of sensors, including inductive loop detectors, side-fire radar, and magnetometers [9]. These sensors can record traffic state parameters like speed and volume every 30 seconds, but at any time steps, there are about 69.59% of the sensors that not are in working conditions due to various types of failures [9].

Recent research has witnessed a growing interest in the development of efficient missing data imputation methods for spatiotemporal traffic data. Some key references in this area include [10–16]. The fundamental challenge in missing data imputation is how to effectively utilize partial observations while capturing higher-order correlations and dependencies in the data. Several important frameworks have emerged in this context: 1) classical time series analysis methods such as VAR [17], 2) purely low-rank matrix factorization/completion, e.g., matrix factorization [12, 18], principal component analysis [19], and their variants, 3) purely low-rank tensor factorization/completion, e.g., Bayesian tensor factorization [14] and low-rank tensor completion (LRTC) based imputation [20, 21], and 4) low-rank temporal models, e.g., temporal regularized matrix factorization (TRMF) [6] and Bayesian temporal matrix factorization (BTMF) [1]. Recent studies have increasingly leveraged the low-rank property of spatiotemporal traffic data, allowing the application of data-driven techniques, including compressive sensing [10], matrix/tensor factorization [1, 6], and low-rank matrix/tensor completion [16, 21]. In the following sections, we provide a summary of related work that focuses on low-rank matrix and tensor models for spatiotemporal traffic data imputation.

2.1.1 Matrix and Tensor Factorization

Matrix/tensor factorization models are widely used and handy mathematical tools for reconstructing missing data in spatiotemporal traffic measurements. For example, Zhu *et al.* impose the low-rank assumption on a traffic state matrix from a list of road segments over a certain time and convert the missing data imputation into a matrix completion problem [10]. Based on probabilistic principal component analysis, Qu *et al.* and Li *et al.* develop an

improved model to better characterize the complicated spatiotemporal dependencies [11, 19]. For the urban traffic states collected through probe vehicles, Yu *et al.* develop a city-wide traffic data estimation framework by performing low-rank matrix completion.

Recall that traffic time series data is unique in the sense that there exist strong daily patterns and day-to-day similarity [22]. To better capture the correlations and utilize the global time series trends across different days, the existing studies consider dividing the temporal dimension into day and time of day, and then apply tensor factorization techniques to impute missing values [12,13,23]. In general, these models fold a long traffic time series collected from a spatial location/sensor over several days as a matrix by stacking the vectors corresponding to different days; as a result, such operation converts the original traffic time series matrix (for several locations) into a third-order tensor by introducing the day dimension. In such cases, traffic data can be represented as a tensor of dimensions (spatial location/sensor $\times$ time of day $\times$ day). Although this is a special operation for constructing the traffic data tensor, many real-world time series data generated by human mobility and activities (e.g., energy and electricity consumption) also exhibit similar daily and weekly patterns. Notably, it is also possible to introduce the week dimension in the tensor completion task, but experiments in [14] demonstrate that the day dimension is more helpful than the week dimension. Given that most traffic datasets are inherently low-rank or at least the important spatiotemporal patterns of traffic data are in the low-dimensional spaces, tensor factorization models usually show better imputation performance against matrix factorization models. There are two main specific reasons:

- From an algebraic perspective, the third-order tensor representation preserves more complicated spatiotemporal correlations than the matrix representation, given that the multivariate traffic time series matrix is indeed one of the unfoldings of the tensor data.
- In the spatiotemporal context, the tensor representation not only preserves spatial dependencies but also shows both global and local time series trends (e.g., traffic speed data at the morning rush hour on Monday might be similar to that on Tuesday).

Overall, matrix/tensor factorization models can approximate the incomplete spatiotemporal matrix/tensor by using linear factorization with a certain rank.

2.1.2 Low-Rank Matrix/Tensor Completion

In the past decade, substantial progress has been made in developing state-of-the-art low-rank matrix and tensor completion methods. On the rank minimization of low-rank matrix/tensor

completion, given that the rank operator is neither convex nor continuous, the nuclear norm is introduced as a convex surrogate of the rank function to be minimized [24–26]. Following that idea, some studies have shown the advantages of using nonconvex surrogate functions to approximate the rank for matrices (see e.g., truncated nuclear norm in [27, 28], weighted nuclear norm in [29], and generalized nonconvex nonsmooth low-rank minimization in [30, 31]). It is shown that nonconvex surrogate regularization can work better than nuclear norm minimization in some real-world applications (e.g., image inpainting).

In practice, low-rank models on the matrix provide insight into low-rank tensor models. LRTC, taking the sum of nuclear norms of tensor unfoldings, has attracted considerable attention since it was first introduced by Liu *et al.* in [32]. The existing studies also extend the truncated nuclear norm minimization to a tensor setting (e.g., [21, 33]), showing that nonconvex truncated nuclear norm minimization is more capable than convex nuclear norm minimization on tensor completion tasks.

2.1.3 Low-Rank Models with Temporal Modeling

A large body of existing studies has leveraged temporal dynamics in low-rank models for time series imputation, and the algorithms are expected to discover low-rank patterns and temporal dynamics simultaneously. A common assumption within these models is that time series and thus their low-rank factors show local temporal dependencies.

In terms of local smoothness, Chen and Cichocki propose a Toeplitz matrix-based regularization to impose temporal smoothness in matrix factorization [34]; the regularization penalizes the difference between the low-rank factors of two consecutive time steps. A similar regularization based on the Toeplitz matrix is used by [35] for traffic data reconstruction. Chen *et al.* apply a quadratic variation to a traffic tensor completion problem to ensure temporal smoothness [36]. Many studies also consider using Gaussian process (GP) priors to characterize latent spatial/temporal smoothness [37, 38], e.g., kernelized matrix/tensor factorization. Lei *et al.* propose a fully Bayesian solution by an efficient Markov chain Monte Carlo (MCMC) sampling algorithm for kernelized matrix factorization [39]. The maximum a posteriori estimation of using a proper GP prior yields the same form as a Laplacian regularization, where the Laplacian matrix is the inverse of the covariance matrix of the GP prior [38, 40]. As Laplacian regularization is of broad use in graph modeling, it is also applicable to temporal modeling. For example, Rao *et al.* present a scalable collaborative filtering algorithm with spatial/temporal Laplacian regularization for matrix data imputation [41].

In terms of modeling temporal dynamics of low-rank factors, Xiong *et al.* formulate a Bayesian tensor factorization with first-order Markovian assumptions on the temporal fac-

tors [42]. Yu *et al.* integrate the time series autoregression of latent temporal factors as a regularization term into matrix factorization [6]. This work assumes the independent autoregression for each latent temporal factor. In contrast, Chen and Sun apply a VAR on the latent temporal factors and develop a fully Bayesian solution for multidimensional time series prediction in the presence of missing values [1]. The introduction of generative mechanisms such as autoregression not only offers better interpolation/imputation accuracy but also enables the factorization models to perform time series forecasting.

2.1.4 Low-Rank Models with Algebraic Structures

Recent studies also show great potential for low-rank completion with certain algebraic structures such as Hankel matrix/tensor and circulant matrix/tensor. These approaches overcome some limitations of pure low-rank matrix completion (LRMC) and LRTC models, for example not handling when an entire row/column is missing and invariant to the permutation of rows/columns. For example, the model developed by Yokota *et al.* can recover the missing slices of tensor using Hankelization [43]. Sedighin *et al.* apply tensor train decomposition to tensors obtained by extended multi-way delay embedded transform and show its superiority over LRTC models in the missing data imputation tasks [44].

A critical property of the circulant matrix is that its nuclear norm can be efficiently obtained via FFT. Using this property, Yamamoto *et al.* propose a fast tensor completion in the delay embedding space [45]; Liu and Zhang use the nuclear norm minimization of circulant matrices for missing data recovery and time series forecasting [46]. Despite the fast algorithm on the circulant matrix, circulant matrix-based models are inadequate for capturing the local trend/continuity in time series. Therefore, Liu and Zhang propose a convolution matrix nuclear norm minimization (ConvNNM) model for better local trends modeling if the window length of the convolution matrix is set to be relatively small [46]. Further, Liu proposes a learnable and orthonormal transformation for ConvNNM to reinforce its modeling ability when the convolutional low-rank condition is not satisfied [47].

Although there are several low-rank matrix and tensor models as mentioned above for traffic data imputation, playing as the primary tools for capturing spatiotemporal patterns in data and achieving efficient reconstruction, these models are not well-suited to extreme missing data imputation scenarios (e.g., 90% missing rate) on traffic data. The delay embedding transform is a classical data augmentation technique that enhances the performance of matrix and tensor completion algorithms (e.g., circulant/anti-circulant matrix, convolution matrix, delay embedding matrix, and Hankel matrix) [43, 45, 48–50]. In the literature, the Hankel matrix/tensor has been demonstrated to be an efficient algebraic structure for

signal reconstruction [44, 49–51], image inpainting [43, 52, 53], and spatiotemporal data modeling [54, 55]. By stacking the vectors/matrices/tensors with shifted positions in their entries, the Hankel structure implicitly captures the sequential dependencies in the data and enables the matrix/tensor factorization algorithms to recover missing rows/columns. For example, Ying *et al.* developed a signal recovery method using CANDECOMP/PARAFAC (CP) factorization and carried out Hankel matrix nuclear norm minimization for factorized components/vectors [49]. Zhang and Wang introduce the low-rank completion problem of a Hankel matrix mapped from a multichannel time series and develop a fast algorithm for robust Hankel matrix completion [56]. Wang *et al.* propose a Hankel matrix factorization model to recover missing values (in particular for the missing data of the whole columns) in multivariate time series [57].

Since the Hankel operation often increases the dimensionality and size of data significantly, the high memory consumption and computational costs are the bottlenecks for the wider use of the Hankel matrix/tensor models [48]. Other algebraic structures have been used to address the computational issues. For example, an existing study presents fast circulant matrix nuclear norm minimization (CircNNM) methods via the use of the FFT [46]. It has demonstrated that the ConvNNM is more accurate than CircNNM for modeling local trends in time series [46]. However, ConvNNM cannot be efficiently solved by using FFT. The accuracy of ConvNNM was further improved with a learnable and orthonormal transformation in [47], but the computational issues remained. Yamamoto *et al.* develop a fast approximation algorithm via FFT for Tucker tensor decomposition on the delay embedding tensor [45].

2.2 Time Series Forecasting on Sparse Data

In practice, for small-scale datasets, VAR is a widely-used solution for time series forecasting. However, on high-dimensional time series data, the classical VAR model severely suffers from the over-parameterization issue [58–61]. This means that the number of parameters vastly exceeds the number of observations. In this case, prior knowledge and assumptions on the parameter space, including the low-rank property of data and the sparsity of the coefficient matrices, become essential for developing efficient algorithms. For example, in multivariate reduced-rank regression [58, 59], the coefficient matrices are assumed to be low-rank, and the variants of this method include high-dimensional VAR with both matrix factorization (e.g., [62, 63]) and tensor factorization (e.g., [61]). Imposing sparsity on the coefficients also provides appropriate algorithms, such as using ℓ_1-norm regularized estimates [60, 64] and using the combination of nuclear norm and lasso penalties [65].

On incomplete time series data, a central challenge is how to learn temporal dynamics from partially observed data efficiently. A naive solution is to first impute the missing data and then make predictions on the reconstructed data (e.g., [66]). However, such methods may produce substantial estimation biases due to the separation of imputation and forecasting, not to mention that making accurate imputation itself is not a trivial task as discussed in the previous section. For natural high-dimensional datasets with missing values, the low-rank assumption has been demonstrated to be very effective for imputation [1,6]. In literature, dimensionality reduction techniques such as matrix and tensor factorization models have been used to model high-dimensional and incomplete time series data [1, 6, 42, 67]. The underlying assumption in these models is that the incomplete high-dimensional dataset can be well characterized by a few latent factors, which evolve over time following certain temporal processes (e.g., VAR). This assumption is similar to the dynamic factor model with a state-space representation [68]. However, modeling the incomplete data in a matrix factorization framework is more computationally efficient due to the simplified isotropic covariance structure on errors [6].

In addition to high-dimensionality and sparsity, nonstationarity is also a unique feature of real-world time series data. The properties of a stationary time series do not depend on time. Most existing works in the literature simply assume that the data and the underlying latent temporal factors are stationary when applying VAR for modeling temporal dynamics [1, 6, 67, 69]. Only a few matrix and tensor factorization-based time series models address the nonstationarity issue, mainly by explicitly modeling the periodic/seasonal structure [6, 70, 71]. As mentioned in [70], to overcome the limitation of matrix factorization for seasonal modeling, one classical approach for temporal modeling with periodic patterns is converting matrix factorization into tensor factorization according to the season information. In TRMF [6], seasonality is considered by using a well-designed time lag set for the autoregression model. Shifting seasonal matrix factorization proposed by [71] can learn multiple seasonal patterns (or regimes) from multi-viewed data, i.e., matrix-variate time series. These works demonstrate the effectiveness of matrix/tensor models for time series data with diverse periodic patterns.

CHAPTER 3 METHODOLOGY

Spatiotemporal traffic time series are often high-dimensional, incomplete, sparse, and nonstationary, showing both global and local trends due to the daily and weekly rhythm of human mobility and dynamic patterns of traffic flow. These characteristics hinder the development of efficient data-driven ITS and pose great practical challenges. The whole methodology aims at answering how to characterize low-rank patterns and spatiotemporal global and local trends in the modeling process, and consequently solving real-world problems that include traffic data imputation and forecasting in the presence of missing values.

An outline of this chapter is as follows,

- **Spatiotemporal traffic data imputation and forecasting with matrix factorization**. Section 3.1 introduces a NoTMF model for traffic time series imputation and forecasting in the presence of sparse data. This model characterizes the nonlocal trends and local trends of latent temporal factors via the seasonal differenced VAR process. The unified framework reveals both low-rank properties of sparse traffic data modeled by matrix factorization and time series correlations modeled by VAR.
- **Spatiotemporal traffic data imputation with tensor completion**. Section 3.2 presents a missing data imputation approach that utilizes the day dimension of traffic data and builds a tensor completion task through truncated nuclear norm minimization. To characterize the time series trends, the proposed LATC model takes into account univariate autoregressive processes along the temporal dimension, allowing one to reinforce the low-rank tensor model.
- **Large-scale traffic data imputation with an efficient implementation**. Section 3.3 shows a scalable and fast imputation approach implemented by using the property of the circulant matrix/tensor and circular convolution and their relationship with discrete Fourier transform. The global trends can be characterized by the circulant matrix/tensor nuclear norm, while the local trends can be characterized by the Laplacian kernelized regularization. For implementation, the proposed framework can be solved by using the FFT, which is shown to be well-suited to large-scale traffic data imputation tasks.
- **Extreme missing traffic data imputation with tensor factorization**. Section 3.4 focuses on the extreme missing traffic data imputation task. We present a memory-efficient HTF that takes a Hankelization process on spatiotemporal traffic data. Each

slice of the Hankel tensor takes a convolutional matrix factorization in which convolutional parameterization allows one to capture spatiotemporal correlations of traffic data automatically.

3.1 Nonstationary Temporal Matrix Factorization

In this section, we propose an NoTMF model, in which matrix factorization leverages a spatial factor matrix and a temporal factor matrix to recover the incomplete time series data, and a higher-order VAR process is imposed on a properly differenced copy of the temporal factor matrix in the latent space. This approach not only reveals the low-rank property of the sparse traffic data but also preserves consistent temporal dynamics. Training NoTMF involves the optimization problem with respect to the two factor matrices and a collection of VAR coefficient matrices. To efficiently solve the optimization problem, we derive an alternating minimization framework in which a complicated optimization is raised when estimating the temporal factor matrix because it involves both partially observed matrix factorization and seasonal differenced VAR. In our framework, we propose to find the approximated solution to the subproblem on the temporal factor matrix through the conjugate gradient method.

3.1.1 Matrix Factorization

Low-rank matrix factorization is a classical approach for reconstructing missing values from a partially observed data matrix [24,72], usually in the form of matrix multiplication between two matrices. Traffic states on a large urban road network have similar temporal patterns (e.g., peak hours and off-peak hours), and the patterns would repeat at both daily and weekly levels, referring to the daily and weekly rhythm of human mobility and essential dynamic patterns of traffic flow. Such correlated periodicity/seasonality among a large number of time series further enhances the low-rank property of the data, which becomes the primary reason that low-dimensional factorization models perform extremely well on traffic data imputation tasks [15].

Given a traffic state data matrix $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ with observed index set Ω, as shown in Figure 3.1, one can approximate the data by factorizing it into a spatial factor matrix $\boldsymbol{W} \in \mathbb{R}^{R \times N}$ and a temporal factor matrix $\boldsymbol{X} \in \mathbb{R}^{R \times T}$ with rank $R < \min\{N, T\}$ such that

$$\mathcal{P}_{\Omega}(\boldsymbol{Y}) \approx \mathcal{P}_{\Omega}(\boldsymbol{W}^{\top}\boldsymbol{X}), \tag{3.1}$$

or element-wise, we have the following form:

$$y_{n,t} \approx \boldsymbol{w}_n^\top \boldsymbol{x}_t, \forall (n,t) \in \Omega, \tag{3.2}$$

where $\boldsymbol{w}_n, \boldsymbol{x}_t \in \mathbb{R}^R$ are the n-th and t-th columns of the factor matrices $\boldsymbol{W}$ and $\boldsymbol{X}$, respectively. Note that there are R latent factors in $\boldsymbol{W}$ (and $\boldsymbol{X}$).

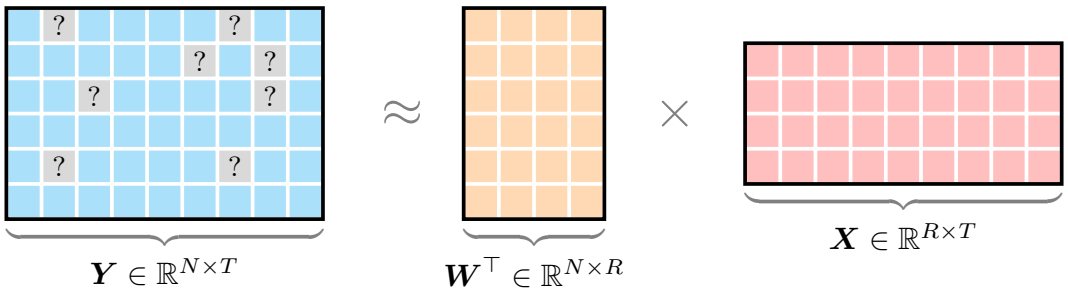


Figure 3.1 Illustration of matrix factorization whose components include the spatial factor matrix $\boldsymbol{W}$ with R rows and temporal factor matrix with R rows. The multiplication between $\boldsymbol{W}^\top$ and $\boldsymbol{X}$ results in a complete N-by-T matrix, usually playing as the reconstruction of sparse traffic time series data. The symbols "?" in the data matrix $\boldsymbol{Y}$ refer to the missing entries.

To achieve such approximation, one fundamental problem is how to specify the latent factor matrices, so that the partially observed data matrix $\boldsymbol{Y}$ matches $\boldsymbol{W}^\top \boldsymbol{X}$ as closely as possible [72]. The optimization problem of matrix factorization is given by

$$\min_{\boldsymbol{W},\boldsymbol{X}} \frac{1}{2} \|\mathcal{P}_\Omega(\boldsymbol{Y} - \boldsymbol{W}^\top \boldsymbol{X})\|_F^2 + \frac{\rho}{2}(\|\boldsymbol{W}\|_F^2 + \|\boldsymbol{X}\|_F^2), \tag{3.3}$$

where ρ is the weight parameter for regularization terms. The symbol $\|\cdot\|_F$ denotes the Frobenius norm of the matrix, and accordingly $\|\cdot\|_F^2$ denotes the sum of squared entries of the matrix.

3.1.2 Temporal Matrix Factorization

Temporal matrix factorization (TMF) is an important variant in the large family of matrix factorization models. Thanks to temporal modeling techniques such as the autoregressive process, TMF is extremely useful for multivariate time series forecasting in the presence of missing values [1, 6]. The basic idea behind the model is that matrix factorization can learn low-rank temporal patterns from partially observed time series data, while autoregression

models can capture the time-evolving coefficients. The goal is to capture time series correlations and patterns within the framework of low-rank matrix factorization. Typically, these patterns may offer valuable insights into the spatiotemporal structure and assist data processing tasks such as dimensionality reduction, missing data imputation [41, 42], and time series forecasting [1,6]. Existing studies on TMF have shown that introducing temporal modeling within the matrix factorization framework can achieve superior performance in traffic time series forecasting and imputation tasks [1, 6]. Formally, the optimization problem of TMF [1, 6] can be summarized as follows,

$$\begin{aligned}\min_{\boldsymbol{W},\boldsymbol{X},\boldsymbol{A}_1,\ldots,\boldsymbol{A}_d} \; & \frac{1}{2}\|\mathcal{P}_{\Omega}(\boldsymbol{Y}-\boldsymbol{W}^{\top}\boldsymbol{X})\|_F^2 \\ & + \frac{\gamma}{2}\sum_{t=d+1}^{T}\Big\|\boldsymbol{x}_t-\sum_{k=1}^{d}\boldsymbol{A}_k\boldsymbol{x}_{t-k}\Big\|_2^2 \\ & + \frac{\rho}{2}(\|\boldsymbol{W}\|_F^2+\|\boldsymbol{X}\|_F^2),\end{aligned} \tag{3.4}$$

where $\boldsymbol{A}_k \in \mathbb{R}^{R\times R}, k = 1,\ldots,d$ represent the coefficient matrices for characterizing the temporal dynamics of $\boldsymbol{X} \in \mathbb{R}^{R\times T}$ in the VAR process, and $\boldsymbol{x}_t \in \mathbb{R}^R, \forall t \in \{1,\ldots,T\}$ denotes the t-th column of temporal factor matrix $\boldsymbol{X}$. The symbol $\|\cdot\|_2$ denotes the ℓ_2-norm of vector, and accordingly $\|\cdot\|_2^2$ denotes the sum of squared entries of vector. Hyperparameters $\{\gamma,\rho\}$ are the weight parameters for regularization terms.

Remark 1. *As shown in Eq.* (3.4)*, VAR takes the autoregression on the temporal factor matrix* $\boldsymbol{X} \in \mathbb{R}^{R\times T}$ *in which* R *is the number of variables and* T *is the length of time. VAR builds the correlations among column vectors* $\boldsymbol{x}_t \in \mathbb{R}^R$ *with coefficient matrices, namely,*

$$\boldsymbol{x}_t = \sum_{k=1}^{d}\boldsymbol{A}_k\boldsymbol{x}_{t-k}+\boldsymbol{\epsilon}_t,\ \forall t \in \{d+1,\ldots,T\}, \tag{3.5}$$

where $\boldsymbol{A}_k \in \mathbb{R}^{R\times R}$*,* $k = 1,\ldots,d$ *are the coefficient matrices, and* $\boldsymbol{\epsilon}_t \in \mathbb{R}^R \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{\Sigma})$*,* $\forall t$ *is an independent and identically distributed (i.i.d.) error term. Since* $d \in \mathbb{N}^+$ *is the order of VAR, the above formula is also called a* d*th-order VAR.*

On the high-dimensional time series (i.e., with a large N *as the number of variables), the number of coefficients in VAR would be* dN^2*, possibly leading to over-parameterization and overfitting issues. The TMF framework only possesses* dR^2 *coefficients on the low-dimensional latent space, preventing the model from over-parameterization in the high-dimensional case.*

In Eq. (3.4), the first term quantifies the low-rank reconstruction error, the second term

ensures that the temporal variation of $\boldsymbol{X}$ can be approximated by a higher-order VAR process, and the third term is the regularization for factor matrices. Although the TMF model has been presented in the literature, the solution algorithm presented here is different due to the complicated optimization problem of the temporal factors. In particular, time series modeling on $\boldsymbol{X}$ is the cornerstone of TMF. For instance, the previous studies present TMF models by taking into account both univariate autoregressive process [6] (i.e., $\boldsymbol{A}_k$ is diagonal) and multivariate VAR process [1].

3.1.3 Time Series Differencing

In the time series analysis, one classical approach to achieve stationarity on the time series is performing differencing operations [73]. For example, one can perform first-order differencing and seasonal differencing to transform the original time series data into a stationary copy for model estimation. For a season-m differencing operation on the time series $\boldsymbol{x}_t \in \mathbb{R}^R, t = 1, \ldots, T$, the transformed time series are given by

$$\tilde{\boldsymbol{x}}_t = \boldsymbol{x}_t - \boldsymbol{x}_{t-m}, \forall t \in \{m+1, \ldots, T\}. \tag{3.6}$$

Accordingly, the dth-order VAR process on the differenced copy of time series can be written as follows,

$$\boldsymbol{x}_t - \boldsymbol{x}_{t-m} = \sum_{k=1}^{d} \boldsymbol{A}_k(\boldsymbol{x}_{t-k} - \boldsymbol{x}_{t-m-k}) + \boldsymbol{\epsilon}_t, \, \forall t \in \{d+m+1, \ldots, T\}, \tag{3.7}$$

where $\boldsymbol{A}_k \in \mathbb{R}^{R \times R}, k = 1, \ldots, d$ are the coefficient matrices, and $\boldsymbol{\epsilon}_t \in \mathbb{R}^R \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}), \forall t$ is an i.i.d. error term.

3.1.4 Nonstationary Temporal Matrix Factorization

In this study, we aim at simultaneously handling the following emerging issues in real-world traffic time series datasets: 1) **High-dimensionality** (i.e., large N): Dataset is of large scale with thousands of spatial locations. 2) **Sparsity and missing values**: Dataset involves missing values, and sometimes only a small fraction of data is observed due to the data collection mechanism. 3) **Nonstationarity**: Real-world traffic time series often show strong trends and seasonality. Solving these issues can help better support data-driven ITS. For instance, the Uber movement speed dataset registers traffic speed data from thousands of road segments with strong daily and weekly periodic patterns. However, due to the insufficient sampling and the limited penetration of ridesharing vehicles, we only have access to a small

fraction of observed traffic states even with an hourly time resolution.

Model Description

In the TMF framework as discussed in Section 3.1.2 and illustrated in Figure 3.2, a central assumption of VAR is that the modeled time series is stationary. However, the aforementioned periodicity/seasonality in real-world data $\boldsymbol{Y}$ will be characterized by the latent factor matrix $\boldsymbol{X}$, and such nonstationarity property contradicts the stationarity assumptions in VAR. In this case, we would expect the temporal dynamics to be time-dependent/time-varying, and the estimated coefficient matrix will become sub-optimal and produce biased estimation. To characterize the nonstationarity of traffic state data, we propose a parsimonious NoTMF by integrating seasonal differencing and other differencing operations into the VAR process, consequently leading to the design of the temporal loss in Eq. (3.9). If we use the season-m differenced VAR (see Eq. (3.7)) in NoTMF, then the optimization problem takes the form:

$$\begin{aligned}\min_{\boldsymbol{W},\boldsymbol{X},\boldsymbol{A}_1,\ldots,\boldsymbol{A}_d} \; & \frac{1}{2}\|\mathcal{P}_{\Omega}(\boldsymbol{Y}-\boldsymbol{W}^{\top}\boldsymbol{X})\|_F^2 \\ & + \frac{\gamma}{2}\sum_{t=d+m+1}^{T}\Big\|(\boldsymbol{x}_t-\boldsymbol{x}_{t-m})-\sum_{k=1}^{d}\boldsymbol{A}_k(\boldsymbol{x}_{t-k}-\boldsymbol{x}_{t-m-k})\Big\|_2^2 \\ & + \frac{\rho}{2}(\|\boldsymbol{W}\|_F^2+\|\boldsymbol{X}\|_F^2),\end{aligned} \tag{3.8}$$

where the second term is the transformed temporal loss through seasonal differencing. Let $\mathcal{R}(\boldsymbol{X})$ denote the regularization of temporal loss, then we have

$$\begin{aligned}\mathcal{R}(\boldsymbol{X}) &= \frac{1}{2}\sum_{t=d+m+1}^{T}\Big\|(\boldsymbol{x}_t-\boldsymbol{x}_{t-m})-\sum_{k=1}^{d}\boldsymbol{A}_k(\boldsymbol{x}_{t-k}-\boldsymbol{x}_{t-m-k})\Big\|_2^2 \\ &= \frac{1}{2}\Big\|\boldsymbol{X}\boldsymbol{\Psi}_0^{\top}-\sum_{k=1}^{d}\boldsymbol{A}_k\boldsymbol{X}\boldsymbol{\Psi}_k^{\top}\Big\|_F^2,\end{aligned} \tag{3.9}$$

where we rewrite the vector-form temporal loss in the form of a matrix by utilizing the temporal operator matrices $\{\boldsymbol{\Psi}_0,\boldsymbol{\Psi}_1,\ldots,\boldsymbol{\Psi}_d\}$ as described in Definition 3.1.1. By doing so, the matrix-form temporal loss can facilitate the matrix computations and enable fast algorithms.

Definition 3.1.1 (Temporal Operator Matrices)**.** *According to the expression of dth-order VAR on the latent temporal factors $\boldsymbol{x}_t$, $\forall t$, the temporal operator matrix $\boldsymbol{\Psi}_k$ associated with*

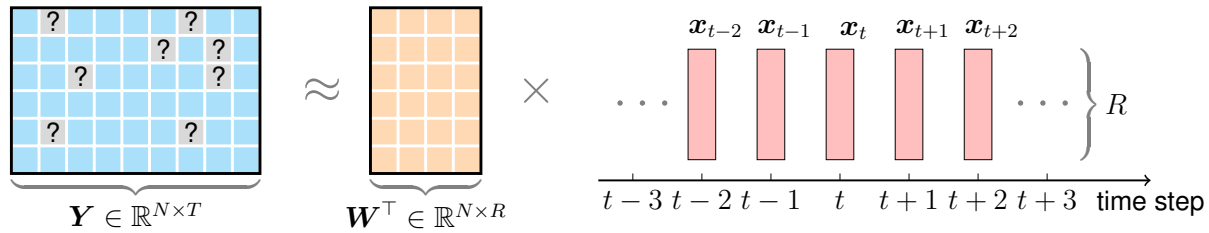


Figure 3.2 Illustration of TMF on traffic time series data. For spatiotemporal traffic states, the data can be factorized into a spatial factor matrix and a temporal factor matrix in which the temporal factor matrix is indeed a multivariate time series. The symbols "?" in the data matrix $\boldsymbol{Y}$ refer to the missing entries.

the season m is defined as follows,

$$\begin{aligned}\boldsymbol{\Psi}_k \triangleq & \begin{bmatrix}\mathbf{0}_{(T-d-m)\times(d-k)} & -\boldsymbol{I}_{T-d-m} & \mathbf{0}_{(T-d-m)\times(k+m)}\end{bmatrix} \\ & + \begin{bmatrix}\mathbf{0}_{(T-d-m)\times(d+m-k)} & \boldsymbol{I}_{T-d-m} & \mathbf{0}_{(T-d-m)\times k}\end{bmatrix} \\ & \in \mathbb{R}^{(T-d-m)\times T}, \forall k \in \{0, 1, \ldots, d\},\end{aligned} \tag{3.10}$$

where without loss of generality, $\mathbf{0}_{m\times n}$ denotes the m-by-n matrix of zeros, and $\boldsymbol{I}_m$ denotes the m-by-m identity matrix.

As a result, the season-m differenced time series can be written in the form of a matrix:

$$\begin{aligned}\boldsymbol{X}\boldsymbol{\Psi}_0^\top &= \begin{bmatrix} | & & | \\ \boldsymbol{x}_{d+m+1} - \boldsymbol{x}_{d+1} & \cdots & \boldsymbol{x}_T - \boldsymbol{x}_{T-m} \\ | & & | \end{bmatrix}, \\ \boldsymbol{X}\boldsymbol{\Psi}_1^\top &= \begin{bmatrix} | & & | \\ \boldsymbol{x}_{d+m} - \boldsymbol{x}_d & \cdots & \boldsymbol{x}_{T-1} - \boldsymbol{x}_{T-m-1} \\ | & & | \end{bmatrix}, \\ &\vdots \\ \boldsymbol{X}\boldsymbol{\Psi}_d^\top &= \begin{bmatrix} | & & | \\ \boldsymbol{x}_{m+1} - \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_{T-d} - \boldsymbol{x}_{T-m-d} \\ | & & | \end{bmatrix},\end{aligned} \tag{3.11}$$

of size $R \times (T - d - m)$.

Remark 2. *Since the temporal operator matrices $\{\boldsymbol{\Psi}_0, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_d\}$ are a sequence of sparse matrices, utilizing such sparse structure allows one to reduce the memory consumption and*

develop fast algorithms in the case of long time series (i.e., with a large T).

Thus, if we use the season-m differencing in NoTMF, then the optimization problem becomes

$$\begin{aligned}\min_{\boldsymbol{W},\boldsymbol{X},\boldsymbol{A}_1,\ldots,\boldsymbol{A}_d} \; & \frac{1}{2}\|\mathcal{P}_{\Omega}(\boldsymbol{Y}-\boldsymbol{W}^{\top}\boldsymbol{X})\|_F^2 \\ & + \frac{\gamma}{2}\Big\|\boldsymbol{X}\boldsymbol{\Psi}_0^{\top}-\sum_{k=1}^{d}\boldsymbol{A}_k\boldsymbol{X}\boldsymbol{\Psi}_k^{\top}\Big\|_F^2 \\ & + \frac{\rho}{2}(\|\boldsymbol{W}\|_F^2+\|\boldsymbol{X}\|_F^2),\end{aligned} \tag{3.12}$$

in which the factorized components $\{\boldsymbol{W},\boldsymbol{X}\}$ and the VAR coefficient matrices $\{\boldsymbol{A}_k\}$ need to be estimated. If one has both $\boldsymbol{W}$ and $\boldsymbol{X}$, then the missing entries of the data matrix $\boldsymbol{Y}$ can be reconstructed accordingly. In particular, since the temporal factor matrix also involves a VAR process, the estimated $\boldsymbol{X}$ allows one to capture temporal dynamics of traffic data.

Seasonal differencing can help explain seasonality and periodicity underlying the traffic time series data. As a special case, when $m=1$, the temporal modeling operators correspond to first-order differencing. Accordingly, the temporal loss in such case can be written as

$$\mathcal{R}(\boldsymbol{X})=\frac{1}{2}\sum_{t=d+2}^{T}\Big\|(\boldsymbol{x}_t-\boldsymbol{x}_{t-1})-\sum_{k=1}^{d}\boldsymbol{A}_k(\boldsymbol{x}_{t-k}-\boldsymbol{x}_{t-1-k})\Big\|_2^2. \tag{3.13}$$

Alternatively, based on both seasonal differencing and first-order differencing, the temporal loss of NoTMF takes the form:

$$\mathcal{R}(\boldsymbol{X})=\frac{1}{2}\Big\|\boldsymbol{X}\boldsymbol{\Psi}_0^{\top}\boldsymbol{\Phi}^{\top}-\sum_{k=1}^{d}\boldsymbol{A}_k\boldsymbol{X}\boldsymbol{\Psi}_k^{\top}\boldsymbol{\Phi}^{\top}\Big\|_F^2, \tag{3.14}$$

where

$$\boldsymbol{\Phi}\triangleq\begin{bmatrix}\boldsymbol{0}_{h\times 1} & \boldsymbol{I}_h\end{bmatrix}-\begin{bmatrix}\boldsymbol{I}_h & \boldsymbol{0}_{h\times 1}\end{bmatrix}\in\mathbb{R}^{h\times(h+1)}, h=T-d-m-1, \tag{3.15}$$

refers to the first-order differencing. For complicated time series data with both trend and seasonality, achieving stationarity may require more complicated operations such as higher-order differencing and differencing with multiple seasonality (e.g., both daily and weekly). The temporal loss in Eq. (3.14) can be adapted for certain time series modeling purposes, and the resulting NoTMF is flexible for handling nonstationarity.

We next introduce an alternating minimization framework to solve the optimization problem in NoTMF, in which we denote the objective function of Eq. (3.12) by f.

Estimating the Spatial Factor Matrix $\boldsymbol{W}$

To estimate the spatial factor matrix $\boldsymbol{W}$, we first write down the partial derivative of f with respect to $\boldsymbol{W}$ as follows,

$$\frac{\partial f}{\partial \boldsymbol{W}} = -\boldsymbol{X}\mathcal{P}_{\Omega}^{\top}(\boldsymbol{Y} - \boldsymbol{W}^{\top}\boldsymbol{X}) + \rho\boldsymbol{W}. \tag{3.16}$$

Let $\frac{\partial f}{\partial \boldsymbol{W}} = \boldsymbol{0}$, then we have the following formula:

$$\boldsymbol{X}\mathcal{P}_{\Omega}^{\top}(\boldsymbol{W}^{\top}\boldsymbol{X}) + \rho\boldsymbol{W} = \boldsymbol{X}\mathcal{P}_{\Omega}^{\top}(\boldsymbol{Y}). \tag{3.17}$$

In fact, Eq. (3.17) represents a system of linear equations, and it is not difficult to obtain the closed-form solution. A natural solution could be pursued through least squares of each column vector of $\boldsymbol{W}$ [72]:

$$\boldsymbol{w}_n = \Big(\sum_{t:(n,t)\in\Omega} \boldsymbol{x}_t\boldsymbol{x}_t^{\top} + \rho\boldsymbol{I}_R\Big)^{-1} \sum_{t:(n,t)\in\Omega} \boldsymbol{x}_t y_{n,t}, \tag{3.18}$$

where $n = 1, 2, \ldots, N$. The notation $\sum_{t:(n,t)\in\Omega}$ implies the sum over all t in the observed index set Ω with the fixed index n.

Estimating the Temporal Factor Matrix $\boldsymbol{X}$

We assume a seasonal differencing on the nonstationary temporal factor matrix and assert that such a process can promote the modeling capability of the matrix factorization framework on traffic time series data. However, the challenge may arise because the VAR process on the temporal factor matrix with seasonal differencing complicates the optimization problem. With respect to the temporal factor matrix $\boldsymbol{X}$, the partial derivative of f is given by

$$\frac{\partial f}{\partial \boldsymbol{X}} = -\boldsymbol{W}\mathcal{P}_{\Omega}(\boldsymbol{Y} - \boldsymbol{W}^{\top}\boldsymbol{X}) + \gamma\sum_{k=0}^{d}\boldsymbol{A}_k^{\top}\Big(\sum_{h=0}^{d}\boldsymbol{A}_h\boldsymbol{X}\boldsymbol{\Psi}_h^{\top}\Big)\boldsymbol{\Psi}_k + \rho\boldsymbol{X}, \tag{3.19}$$

where $\boldsymbol{A}_0 \triangleq -\boldsymbol{I}_R \in \mathbb{R}^{R\times R}$ is introduced as a negative identity matrix.

Let $\frac{\partial f}{\partial \boldsymbol{X}} = \boldsymbol{0}$, then we have

$$\boldsymbol{W}\mathcal{P}_{\Omega}(\boldsymbol{W}^{\top}\boldsymbol{X}) + \gamma\sum_{k=0}^{d}\boldsymbol{A}_k^{\top}\Big(\sum_{h=0}^{d}\boldsymbol{A}_h\boldsymbol{X}\boldsymbol{\Psi}_h^{\top}\Big)\boldsymbol{\Psi}_k + \rho\boldsymbol{X} = \boldsymbol{W}\mathcal{P}_{\Omega}(\boldsymbol{Y}). \tag{3.20}$$

In fact, Eq. (3.20) is a generalized Sylvester equation with multiple terms [74], involving d^2+2 terms in the left-hand side. Since both orthogonal projection and VAR process complicate this matrix equation, if we convert the formula into a standard system of linear equations, then finding the exact solution to the matrix equation would take $\mathcal{O}(R^3T^3)$ time, infeasible if the dimension RT is large, as shown in Lemma 3.1.2.

Lemma 3.1.1 (Mixed-Product Property of Kronecker Product [74]). *Suppose $\boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{X} \in \mathbb{R}^{n\times p}, \boldsymbol{B} \in \mathbb{R}^{p\times q}$ be three matrices commensurate from multiplication in that order, then it holds that*

$$\operatorname{vec}(\boldsymbol{AXB}) = (\boldsymbol{B}^\top \otimes \boldsymbol{A}) \operatorname{vec}(\boldsymbol{X}), \tag{3.21}$$

where $\operatorname{vec}(\cdot)$ *denotes the vectorization operator, and* $\otimes$ *denotes the Kronecker product (see Definition 3.1.2).*

Definition 3.1.2 (Kronecker Product [8]). *For any matrices $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{Y} \in \mathbb{R}^{p\times q}$, the Kronecker product between $\boldsymbol{X}$ and $\boldsymbol{Y}$ is given by*

$$\boldsymbol{X} \otimes \boldsymbol{Y} = \begin{bmatrix} x_{11}\boldsymbol{Y} & x_{12}\boldsymbol{Y} & \cdots & x_{1n}\boldsymbol{Y} \\ x_{21}\boldsymbol{Y} & x_{22}\boldsymbol{Y} & \cdots & x_{2n}\boldsymbol{Y} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1}\boldsymbol{Y} & x_{m2}\boldsymbol{Y} & \cdots & x_{mn}\boldsymbol{Y} \end{bmatrix} \in \mathbb{R}^{(mp)\times(nq)}, \tag{3.22}$$

where the resulting matrix is of size $(mp)\times(nq)$ with $m \times n$ blocks.

Lemma 3.1.2. *Given a partially observed data matrix $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$ with the observed index set Ω, the closed-form solution to Eq.* (3.20) *in the form of vectorized $\boldsymbol{X}$ is given by*

$$\operatorname{vec}(\boldsymbol{X}) = \left(\boldsymbol{S} + \gamma \sum_{k=0}^{d}\sum_{h=0}^{d} (\boldsymbol{\Psi}_k \otimes \boldsymbol{A}_k)^\top (\boldsymbol{\Psi}_h \otimes \boldsymbol{A}_h) + \rho \boldsymbol{I}_{RT}\right)^{-1} \operatorname{vec}(\boldsymbol{W}\mathcal{P}_\Omega(\boldsymbol{Y})), \tag{3.23}$$

where the block matrix

$$\boldsymbol{S} \triangleq \begin{bmatrix} \boldsymbol{S}_1 & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S}_2 & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{S}_T \end{bmatrix} \in \mathbb{R}^{(RT)\times(RT)}, \tag{3.24}$$

has a sequence of blocks on the diagonal such that

$$\boldsymbol{S}_t \triangleq \sum_{n:(n,t)\in\Omega} \boldsymbol{w}_n \boldsymbol{w}_n^\top \in \mathbb{R}^{R\times R},\ t = 1, 2, \ldots, T, \tag{3.25}$$

where the notation $\sum_{n:(n,t)\in\Omega}$ *implies the sum over all* n *in the observed index set* Ω *with the fixed index* t*.*

Proof. If we let the first term of the left-hand side in Eq. (3.20) as

$$\boldsymbol{C} \triangleq \boldsymbol{W}\mathcal{P}_{\Omega}(\boldsymbol{W}^{\top}\boldsymbol{X}) \in \mathbb{R}^{R\times T}, \tag{3.26}$$

then the t-th column of $\boldsymbol{C}$ is given by

$$\boldsymbol{c}_t = \sum_{n:(n,t)\in\Omega} \boldsymbol{w}_n \boldsymbol{w}_n^{\top}\boldsymbol{x}_t = \boldsymbol{S}_t\boldsymbol{x}_t \quad \Rightarrow \quad \operatorname{vec}(\boldsymbol{C}) = \boldsymbol{S}\operatorname{vec}(\boldsymbol{X}). \tag{3.27}$$

In the meanwhile, if we let the VAR components in Eq. (3.20) be

$$\boldsymbol{D} \triangleq \sum_{k=0}^{d} \boldsymbol{A}_k^{\top}\Big(\sum_{h=0}^{d}\boldsymbol{A}_h\boldsymbol{X}\boldsymbol{\Psi}_h^{\top}\Big)\boldsymbol{\Psi}_k \in \mathbb{R}^{R\times T}, \tag{3.28}$$

then according to the mixed-product property of the Kronecker product (see Lemma 3.1.1), we get

$$\begin{aligned}
\operatorname{vec}(\boldsymbol{D}) &= \sum_{k=0}^{d}(\boldsymbol{\Psi}_k^{\top}\otimes\boldsymbol{A}_k^{\top})\operatorname{vec}\Big(\sum_{h=0}^{d}\boldsymbol{A}_h\boldsymbol{X}\boldsymbol{\Psi}_h^{\top}\Big) \\
&= \sum_{k=0}^{d}\sum_{h=0}^{d}(\boldsymbol{\Psi}_k^{\top}\otimes\boldsymbol{A}_k^{\top})(\boldsymbol{\Psi}_h\otimes\boldsymbol{A}_h)\operatorname{vec}(\boldsymbol{X}) \\
&= \sum_{k=0}^{d}\sum_{h=0}^{d}(\boldsymbol{\Psi}_k\otimes\boldsymbol{A}_k)^{\top}(\boldsymbol{\Psi}_h\otimes\boldsymbol{A}_h)\operatorname{vec}(\boldsymbol{X}).
\end{aligned} \tag{3.29}$$

As a consequence, Eq. (3.20) is equivalent to

$$\Big(\boldsymbol{S} + \gamma\sum_{k=0}^{d}\sum_{h=0}^{d}(\boldsymbol{\Psi}_k\otimes\boldsymbol{A}_k)^{\top}(\boldsymbol{\Psi}_h\otimes\boldsymbol{A}_h) + \rho\boldsymbol{I}_{RT}\Big)\operatorname{vec}(\boldsymbol{X}) = \operatorname{vec}(\boldsymbol{W}\mathcal{P}_{\Omega}(\boldsymbol{Y})), \tag{3.30}$$

and thus leading to Eq. (3.23) as claimed. □

However, this vectorized solution is memory-consuming and computationally expensive for large problems. In large applications, as suggested by [41], conjugate gradient-based alternating minimization scheme for matrix factorization provides a solution that is both efficient and scalable. The conjugate gradient method allows one to search for the solution to a system of linear equations with a relatively small number of iterations (e.g., 5 or 10). Therefore, to estimate the temporal factor matrix, we develop an efficient conjugate gradient [74] routine.

We first define the left-hand side of Eq. (3.30) as the following function:

$$\theta_x(\boldsymbol{X}) \triangleq \Big(\boldsymbol{S} + \gamma \sum_{k=0}^{d}\sum_{h=0}^{d}(\boldsymbol{\Psi}_k \otimes \boldsymbol{A}_k)^\top(\boldsymbol{\Psi}_h \otimes \boldsymbol{A}_h) + \rho \boldsymbol{I}_{RT}\Big)\,\mathrm{vec}(\boldsymbol{X}), \tag{3.31}$$

which is equivalent to

$$\theta_x(\boldsymbol{X}) = \mathrm{vec}\Big(\boldsymbol{W}\mathcal{P}_\Omega(\boldsymbol{W}^\top\boldsymbol{X}) + \gamma \sum_{k=0}^{d}\boldsymbol{A}_k^\top\Big(\sum_{h=0}^{d}\boldsymbol{A}_h\boldsymbol{X}\boldsymbol{\Psi}_h^\top\Big)\boldsymbol{\Psi}_k + \rho\boldsymbol{X}\Big), \tag{3.32}$$

where $\boldsymbol{S}$ is a sparse matrix as defined in Lemma 3.1.2. Then, the right-hand side of Eq. (3.30) is given by

$$\psi_x \triangleq \mathrm{vec}(\boldsymbol{W}\mathcal{P}_\Omega(\boldsymbol{Y})). \tag{3.33}$$

Since Eq. (3.30) is in the form of a standard system of linear equations with a symmetric positive-definite matrix, the conjugate gradient method is well-suited to such Kronecker structured linear equations. Algorithm 1 summarizes the estimation procedure for approximating the least squares solution to $\boldsymbol{X}$ in Eq. (3.20). Alternatively, we can also use the conjugate gradient to solve Eq. (3.17) when estimating the spatial factor matrix $\boldsymbol{W}$.

Algorithm 1 Conjugate Gradient for Approximating $\boldsymbol{X}$

Input: Data $\boldsymbol{Y}$ with the observed index set Ω, spatial factor matrix $\boldsymbol{W}$, initialized temporal factor matrix $\boldsymbol{X}$, coefficient matrix $\boldsymbol{A}$, and maximum iteration $\ell_{\max}$ (e.g., $\ell_{\max} = 5$).
Output: Estimated temporal factor matrix $\boldsymbol{X}$.

Initialize $\boldsymbol{x}_0$ by the vectorized $\boldsymbol{X}$.
Compute residual vector $\boldsymbol{r}_0 = \psi_x - \theta_x(\boldsymbol{X})$, and let $\boldsymbol{d}_0 = \boldsymbol{r}_0$.
for $\ell = 0$ to $\ell_{\max} - 1$ **do**
  Convert vector $\boldsymbol{d}_\ell$ into matrix $\boldsymbol{D}_\ell$.
  Compute $\alpha_\ell = \frac{\boldsymbol{r}_\ell^\top \boldsymbol{r}_\ell}{\boldsymbol{d}_\ell^\top \theta_x(\boldsymbol{D}_\ell)}$.
  Update $\boldsymbol{x}_{\ell+1} = \boldsymbol{x}_\ell + \alpha_\ell \boldsymbol{d}_\ell$.
  Update $\boldsymbol{r}_{\ell+1} = \boldsymbol{r}_\ell - \alpha_\ell \theta_x(\boldsymbol{D}_\ell)$.
  Compute $\beta_\ell = \frac{\boldsymbol{r}_{\ell+1}^\top \boldsymbol{r}_{\ell+1}}{\boldsymbol{r}_\ell^\top \boldsymbol{r}_\ell}$.
  Update $\boldsymbol{d}_{\ell+1} = \boldsymbol{r}_{\ell+1} + \beta_\ell \boldsymbol{d}_\ell$.
end for
Convert vector $\boldsymbol{x}_{\ell_{\max}}$ into matrix $\boldsymbol{X}$.
return Temporal factor matrix $\boldsymbol{X}$.

Estimating the Coefficient Matrix $\boldsymbol{A}$

In Eq. (3.9), Definition 3.1.1 allows one to rewrite the temporal loss in the form of a matrix. According to the property of Kronecker product and bilinear linear equations [74], it always

holds that

$$\begin{aligned}\sum_{k=1}^{d}\boldsymbol{A}_k\boldsymbol{X}\boldsymbol{\Psi}_k^\top =&\boldsymbol{A}_1\boldsymbol{X}\boldsymbol{\Psi}_1^\top+\boldsymbol{A}_2\boldsymbol{X}\boldsymbol{\Psi}_2^\top+\cdots+\boldsymbol{A}_d\boldsymbol{X}\boldsymbol{\Psi}_d^\top\\ =&\begin{bmatrix}\boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_d\end{bmatrix}\begin{bmatrix}\boldsymbol{X} & \boldsymbol{0} & \cdots & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{X} & \cdots & \boldsymbol{0}\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{X}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Psi}_1^\top\\ \boldsymbol{\Psi}_2^\top\\ \vdots\\ \boldsymbol{\Psi}_d^\top\end{bmatrix}\\ =&\begin{bmatrix}\boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_d\end{bmatrix}(\boldsymbol{I}_d\otimes\boldsymbol{X})\begin{bmatrix}\boldsymbol{\Psi}_1^\top\\ \boldsymbol{\Psi}_2^\top\\ \vdots\\ \boldsymbol{\Psi}_d^\top\end{bmatrix}\\ =&\boldsymbol{A}(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^\top,\end{aligned}\tag{3.34}$$

and consequently, we get

$$\mathcal{R}(\boldsymbol{X})=\frac{1}{2}\|\boldsymbol{X}\boldsymbol{\Psi}_0^\top-\boldsymbol{A}(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^\top\|_F^2,\tag{3.35}$$

where $\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_1 & \cdots & \boldsymbol{A}_d\end{bmatrix}\in\mathbb{R}^{R\times(dR)}$ is an augmented matrix, while $\boldsymbol{\Psi}=\begin{bmatrix}\boldsymbol{\Psi}_1 & \cdots & \boldsymbol{\Psi}_d\end{bmatrix}\in\mathbb{R}^{(T-d-m)\times(dT)}$ is the augmented matrix that comprises the temporal operator matrices. $\boldsymbol{I}_d\in\mathbb{R}^{d\times d}$ is the identity matrix.

Therefore, the optimization problem of NoTMF now becomes

$$\begin{aligned}\min_{\boldsymbol{W},\boldsymbol{X},\boldsymbol{A}}\ &\frac{1}{2}\|\mathcal{P}_\Omega(\boldsymbol{Y}-\boldsymbol{W}^\top\boldsymbol{X})\|_F^2\\ &+\frac{\gamma}{2}\|\boldsymbol{X}\boldsymbol{\Psi}_0^\top-\boldsymbol{A}(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^\top\|_F^2\\ &+\frac{\rho}{2}(\|\boldsymbol{W}\|_F^2+\|\boldsymbol{X}\|_F^2).\end{aligned}\tag{3.36}$$

With respect to the coefficient matrix $\boldsymbol{A}$, the partial derivative of f can be written as follows,

$$\begin{aligned}\frac{\partial f}{\partial\boldsymbol{A}}=&-\gamma(\boldsymbol{X}\boldsymbol{\Psi}_0^\top-\boldsymbol{A}(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^\top)\boldsymbol{\Psi}(\boldsymbol{I}_d\otimes\boldsymbol{X})^\top\\ =&-\gamma\boldsymbol{X}\boldsymbol{\Psi}_0^\top\boldsymbol{\Psi}(\boldsymbol{I}_d\otimes\boldsymbol{X})^\top+\gamma\boldsymbol{A}(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^\top\boldsymbol{\Psi}(\boldsymbol{I}_d\otimes\boldsymbol{X})^\top.\end{aligned}\tag{3.37}$$

Let $\frac{\partial f}{\partial\boldsymbol{A}}=\boldsymbol{0}$, then we have a least squares solution to the coefficient matrix:

$$\begin{aligned}\boldsymbol{A}=&\gamma\boldsymbol{X}\boldsymbol{\Psi}_0^\top\boldsymbol{\Psi}(\boldsymbol{I}_d\otimes\boldsymbol{X})^\top\left(\gamma(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^\top\boldsymbol{\Psi}(\boldsymbol{I}_d\otimes\boldsymbol{X})^\top\right)^{-1}\\ \equiv&\boldsymbol{X}\boldsymbol{\Psi}_0^\top[(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^\top]^\dagger,\end{aligned}\tag{3.38}$$

where $\cdot^{\dagger}$ denotes the Moore-Penrose pseudo-inverse.

Solution Algorithm

Algorithm 2 NoTMF($\boldsymbol{Y}, \Omega, d, R, \gamma, \rho, m$)

Input: Data $\boldsymbol{Y}$ with the observed index set Ω, order d for the VAR, and rank R.
Output: Spatial factor matrix $\boldsymbol{W}$, temporal factor matrix $\boldsymbol{X}$, and coefficient matrix $\boldsymbol{A}$.

```
Initialize the variables {W, X, A}.
Generate temporal operator matrices {Ψ_k}.
repeat
    Compute W as the least squares solution (or conjugate gradient).
    Compute X from Eq. (3.30) with conjugate gradient.
    Compute A by the least squares solution.
until convergence
return W, X, A.
```

We summarize the implementation of NoTMF in Algorithm 2. Here, we analyze the convergence of NoTMF as follows. Recall that the three-block cost function:

$$\begin{aligned} f(\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{A}) \triangleq & \frac{1}{2}\|\mathcal{P}_{\Omega}(\boldsymbol{Y} - \boldsymbol{W}^{\top}\boldsymbol{X})\|_F^2 \\ & + \frac{\gamma}{2}\|\boldsymbol{X}\boldsymbol{\Psi}_0^{\top} - \boldsymbol{A}(\boldsymbol{I}_d \otimes \boldsymbol{X})\boldsymbol{\Psi}^{\top}\|_F^2 \\ & + \frac{\rho}{2}(\|\boldsymbol{W}\|_F^2 + \|\boldsymbol{X}\|_F^2), \end{aligned} \tag{3.39}$$

is the (nonconvex) objective function of NoTMF, and the expected optimal solutions are given by

$$(\hat{\boldsymbol{W}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{A}}) = \underset{\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{A}}{\arg\min} \ f(\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{A}). \tag{3.40}$$

Our alternating minimization is the block coordinate minimization, allowing one to write down the updates at the iteration $\ell + 1$ as follows,

$$\begin{cases} \boldsymbol{W}_{\ell+1} = \underset{\boldsymbol{W}}{\arg\min} \ f(\boldsymbol{W}, \boldsymbol{X}_{\ell}, \boldsymbol{A}_{\ell}), \\ \boldsymbol{X}_{\ell+1} = \underset{\boldsymbol{X}}{\arg\min} \ f(\boldsymbol{W}_{\ell+1}, \boldsymbol{X}, \boldsymbol{A}_{\ell}), \\ \boldsymbol{A}_{\ell+1} = \underset{\boldsymbol{A}}{\arg\min} \ f(\boldsymbol{W}_{\ell+1}, \boldsymbol{X}_{\ell+1}, \boldsymbol{A}). \end{cases} \tag{3.41}$$

If one intends to justify that our algorithm decreases the cost function monotonically, i.e.,

$$f(\boldsymbol{W}_{\ell+1}, \boldsymbol{X}_{\ell+1}, \boldsymbol{A}_{\ell+1}) \leq f(\boldsymbol{W}_{\ell}, \boldsymbol{X}_{\ell}, \boldsymbol{A}_{\ell}), \tag{3.42}$$

then it essentially follows a three-block problem by referring to Eq. (3.41):

$$\begin{cases} \text{find } \boldsymbol{W}_{\ell+1} \text{ such that } f(\boldsymbol{W}_{\ell+1}, \boldsymbol{X}_{\ell}, \boldsymbol{A}_{\ell}) \leq f(\boldsymbol{W}_{\ell}, \boldsymbol{X}_{\ell}, \boldsymbol{A}_{\ell}), \\ \text{find } \boldsymbol{X}_{\ell+1} \text{ such that } f(\boldsymbol{W}_{\ell+1}, \boldsymbol{X}_{\ell+1}, \boldsymbol{A}_{\ell}) \leq f(\boldsymbol{W}_{\ell+1}, \boldsymbol{X}_{\ell}, \boldsymbol{A}_{\ell}), \\ \text{find } \boldsymbol{A}_{\ell+1} \text{ such that } f(\boldsymbol{W}_{\ell+1}, \boldsymbol{X}_{\ell+1}, \boldsymbol{A}_{\ell+1}) \leq f(\boldsymbol{W}_{\ell+1}, \boldsymbol{X}_{\ell+1}, \boldsymbol{A}_{\ell}). \end{cases} \tag{3.43}$$

As can be seen, these conditions can be satisfied given that the cost function of each subproblem is nonincreasing in the alternating minimization scheme. As a consequence, Eq. (3.42) can be satisfied. Then one can prove that the cost function is lower bounded by following typical procedures as [75]. In such a way, it is guaranteed that NoTMF converges.

3.1.5 Rolling Forecasting Mechanism

We next introduce the forecasting scheme for NoTMF. As the variables $\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{A}$ are estimated through Algorithm 2, one can first perform seasonal differencing on $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$ (i.e., columns of temporal factor matrix) as $\boldsymbol{v}_1 = \boldsymbol{x}_{m+1} - \boldsymbol{x}_1, \ldots, \boldsymbol{v}_{T-m} = \boldsymbol{x}_T - \boldsymbol{x}_{T-m}$, then estimate the forecasts $\hat{\boldsymbol{v}}_{T-m+1}, \ldots, \hat{\boldsymbol{v}}_{T-m+\delta}$ with the coefficient matrix $\boldsymbol{A}$ at the time horizon $\delta \in \mathbb{N}^{+}$. Finally, we can generate the forecasts $\hat{\boldsymbol{y}}_{T+1}, \ldots, \hat{\boldsymbol{y}}_{T+\delta}$ sequentially by

$$\begin{aligned} \hat{\boldsymbol{y}}_{T+1} &= \boldsymbol{W}^{\top}(\boldsymbol{x}_{T-m} + \hat{\boldsymbol{v}}_{T-m}), \\ &\vdots \\ \hat{\boldsymbol{y}}_{T+\delta} &= \boldsymbol{W}^{\top}(\hat{\boldsymbol{x}}_{T-m-1+\delta} + \hat{\boldsymbol{v}}_{T-m-1+\delta}). \end{aligned} \tag{3.44}$$

In our implementation, we consider a rolling forecasting scheme for spatiotemporal traffic state data with sparse inputs. We denote the incremental data at the κ-th rolling forecasting with the time horizon δ by $\boldsymbol{Y}_{\kappa} \in \mathbb{R}^{N \times (T+(\kappa-1)\cdot\delta)}$. NoTMF can process the entire data at once, but it is impractical to accommodate data arriving sequentially. Therefore, we take into account the dictionary learning with a fixed spatial factor matrix (also examined by [69]). Such a collection of spatial factors enables the system to reformat the incremental sparse data to lower-dimensional temporal factors. In each rolling scheme, we first learn the temporal factor matrix through conjugate gradient, which is computationally efficient. Then we update the coefficient matrix accordingly to capture time-evolving patterns. Algorithm 3 shows the implementation of rolling forecasting on the multivariate traffic states data.

Algorithm 3 NoTMF Rolling Forecasting with Missing Values

Input: Data $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ with the observed index set Ω, order d of VAR, rank R, K rolling times, and forecasting time horizon δ.

Output: Forecasts $\hat{\boldsymbol{Y}} \in \mathbb{R}^{N \times \delta \cdot K}$.

1: Train NoTMF on data matrix $\boldsymbol{Y}$ as shown in Algorithm 2 and return the result: $\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{A} = \text{NoTMF}(\boldsymbol{Y}, \Omega, d, R, \gamma, \rho, m)$.
2: Generate the forecasts $\hat{\boldsymbol{y}}_{T+1}, \ldots, \hat{\boldsymbol{y}}_{T+\delta}$ by Eq. (3.44) and stack them as the first δ columns of $\hat{\boldsymbol{Y}}$.
3: **for** $\kappa = 2$ to K **do**
4: Input the incremental data $\boldsymbol{Y}_\kappa \in \mathbb{R}^{N \times (T+(\kappa-1)\cdot\delta)}$ in which the last δ columns are the newly arriving data.
5: Generate temporal modeling operators $\{\boldsymbol{\Psi}_k\}$.
6: Compute $\boldsymbol{X}_\kappa$ from data $\boldsymbol{Y}_\kappa$ (with the observed index set Ω_κ) with conjugate gradient.
7: Update the coefficient matrix $\boldsymbol{A}$.
8: Generate the forecasts: $\hat{\boldsymbol{y}}_{T+(\kappa-1)\cdot\delta+1}, \ldots, \hat{\boldsymbol{y}}_{T+(\kappa-1)\cdot\delta+\delta}$ and stack them to $\hat{\boldsymbol{Y}}$.
9: **end for**
10: **return** $\hat{\boldsymbol{Y}}$.

3.2 Low-Rank Autoregressive Tensor Completion

In this section, we propose an LATC framework by introducing temporal variation as a regularization term into the completion of a third-order tensor of the dimensions (spatial location/sensor $\times$ time of day $\times$ day). The third-order tensor structure allows one to better capture the global consistency of traffic data, such as inherent seasonality and day-to-day similarity. To achieve local consistency, we design the temporal variation by imposing an autoregression model for each time series with coefficients as learnable parameters. Different from previous spatial and temporal regularization terms, the minimization of temporal variation can better characterize temporal generative mechanisms beyond local smoothness, allowing one to deal with more challenging scenarios such as block-out missing data. To solve the optimization problem in LATC, we introduce an alternating minimization scheme that estimates the low-rank tensor and autoregression coefficients iteratively.

3.2.1 Tensorization for Global Consistency

We denote the spatiotemporal traffic data collected from N sensors over J days by $\boldsymbol{Y}$, whose columns correspond to time steps and rows correspond to spatial locations:

$$\boldsymbol{Y} = \begin{bmatrix} | & | & & | \\ \boldsymbol{y}_1 & \boldsymbol{y}_2 & \cdots & \boldsymbol{y}_{IJ} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{N \times (IJ)}, \tag{3.45}$$

where I is the number of time steps per day. In the form of a matrix, the data has IJ columns, corresponding to IJ time steps in total. The partially observed matrix can be written as $\mathcal{P}_{\Omega}(\boldsymbol{Y})$ with observed entries supported on the observed index set Ω.

We next introduce the forward tensorization operator $\mathcal{Q} : \mathbb{R}^{N \times (IJ)} \to \mathbb{R}^{N \times I \times J}$ that converts the multivariate time series matrix into a third-order tensor. The temporal dimension of the traffic time series is divided into two dimensions, i.e., time of day and day. Formally, a third-order tensor can be generated by the forward tensorization operator as $\boldsymbol{\mathcal{X}} = \mathcal{Q}(\boldsymbol{Y}) \in \mathbb{R}^{N \times I \times J}$. Conversely, the resulting tensor can be converted into the original matrix by $\boldsymbol{Y} = \mathcal{Q}^{-1}(\boldsymbol{\mathcal{X}}) \in \mathbb{R}^{N \times (IJ)}$ where $\mathcal{Q}^{-1}(\cdot)$ denotes the inverse operator of $\mathcal{Q}(\cdot)$.

The tensorization step transforms a matrix-based imputation problem into a tensor completion problem. Global consistency can be achieved by minimizing tensor rank. In practice, tensor rank is often approximated by using the sum of nuclear norms $\|\boldsymbol{\mathcal{X}}\|_*$ [32] or truncated nuclear norms $\|\boldsymbol{\mathcal{X}}\|_{r,*}$ [21], in which r is the truncation parameter. Our motivation for doing so is that the spatiotemporal traffic data can be characterized by both long-term global trends and short-term local trends. The long-term trends refer to certain periodic, seasonal, and cyclical patterns. Traffic flow data over 24 hours on a typical weekday often shows a systematic "M" shape resulting from human behavioral rhythms, with two peaks during morning and evening rush hours [76]. The pattern also exists at the weekly level with substantial differences from weekdays to weekends. The short-term trends capture certain temporary volatility/perturbation that deviates from the global patterns (e.g., due to an incident or special event). The short-term trends seem to be more random, but they are ubiquitous in reality. LATC leverages both global and local patterns by using matrix and tensor representations simultaneously.

3.2.2 Temporal Variation for Local Consistency

In the time series analysis, autoregression is a classical approach to capture sequential correlations of time series. For the time series data matrix $\boldsymbol{Z} \in \mathbb{R}^{N \times (IJ)}$, suppose $\mathcal{H} =$

$\{h_1, h_2, \ldots, h_d\}$ be the time lag set (time lags are integers), then the autoregression takes the following expression:

$$z_{n,t} = \sum_{k=1}^{d} a_{n,k} z_{n,t-h_k} + \epsilon_{n,t}, \; \forall t \in \{h_d + 1, \ldots, T\}, \tag{3.46}$$

where $a_{n,k}, \forall n, k$ is the (n,k)-th entry of the coefficient matrix $\boldsymbol{A} \in \mathbb{R}^{N \times d}$, and $\epsilon_{n,t} \sim \mathcal{N}(0, \sigma^2)$, $\forall n, t$ is an i.i.d. error term.

Accordingly, we define the temporal variation as reconstruction loss of the autoregression:

$$\|\boldsymbol{Z}\|_{\boldsymbol{A},\mathcal{H}} = \sum_{n=1}^{N} \sum_{t=h_d+1}^{IJ} \Big(z_{n,t} - \sum_{k=1}^{d} a_{n,k} z_{n,t-h_k} \Big)^2, \tag{3.47}$$

where the specification of time lags $h_1, h_2, \ldots, h_d$ depends on the need for modeling local and nonlocal trends of the given time series.

As can be seen, $\|\boldsymbol{Z}\|_{\boldsymbol{A},\mathcal{H}}$ quantifies the total squared error when fitting each time series $\boldsymbol{z}_n \in \mathbb{R}^{IJ}$ with an autoregression model parameterized by the coefficients $\boldsymbol{a}_n \in \mathbb{R}^d$. Given an estimated $\boldsymbol{A}$, minimizing the temporal variation will encourage the time series data $\boldsymbol{Z}$ to show stronger temporal consistency. This implies that the multivariate time series matrix $\boldsymbol{Z}$ would be better explained by a sequence of autoregression models parameterized by $\boldsymbol{A}$. It should be noted that both $\boldsymbol{Z}$ and $\boldsymbol{A}$ are variables in the proposed temporal variation term.

3.2.3 Low-Rank Temporal Tensor Model

Model Description

To ensure both global consistency and local consistency in a low-rank modeling framework, we propose an LATC model that integrates temporal variation of the time series into the truncated nuclear norm minimization of tensor. For any partially observed traffic data matrix $\boldsymbol{Y} \in \mathbb{R}^{N \times (IJ)}$ with observed index set Ω, we have the following optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{Z},\boldsymbol{A}} \; \|\mathcal{Q}(\boldsymbol{Z})\|_{r,*} + \frac{\gamma}{2} \|\boldsymbol{Z}\|_{\boldsymbol{A},\mathcal{H}} \\ &\text{s.t. } \mathcal{P}_{\Omega}(\boldsymbol{Z}) = \mathcal{P}_{\Omega}(\boldsymbol{Y}), \end{aligned} \tag{3.48}$$

where $\|\cdot\|_{r,*}$ denotes the truncated nuclear norm of tensor, and $r \in \mathbb{N}^+$ is the truncation parameter which satisfies $r < \min\{N, I, J\}$. In the objective function, γ is the weight parameter that controls the trade-off between truncated nuclear norm and temporal variation. Figure 3.3 shows that $\boldsymbol{Y}$ can be reconstructed with both low-rank property and time series

dynamics due to the tensorization and the autoregression.

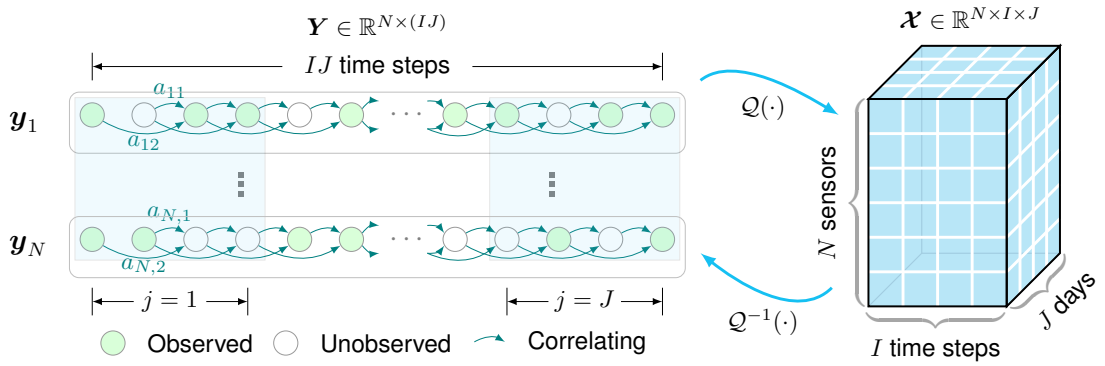


Figure 3.3 Illustration of the proposed LATC framework for spatiotemporal traffic data imputation with time lags $\mathcal{H} = \{1, 2\}$. Each time series $\boldsymbol{y}_n, \forall n \in \{1, 2, \ldots, N\}$ is modeled by the autoregression coefficients $\{a_{n,1}, a_{n,2}\}$.

Most nuclear norm-based tensor completion models employ the ADMM algorithm to solve the optimization problem. However, due to the introduction of autoregressive process, we can no longer apply the default ADMM algorithm to solve the optimization problem in Eq. (3.48). In this case, we consider an alternating minimization scheme that separates the original optimization problem into two subproblems:

$$\begin{cases} \boldsymbol{Z} := \underset{\{\boldsymbol{Z} \,|\, \mathcal{P}_\Omega(\boldsymbol{Z}) = \mathcal{P}_\Omega(\boldsymbol{Y})\}}{\arg\min} \|\mathcal{Q}(\boldsymbol{Z})\|_{r,*} + \dfrac{\gamma}{2}\|\boldsymbol{Z}\|_{\boldsymbol{A},\mathcal{H}}, \\ \boldsymbol{A} := \underset{\boldsymbol{A}}{\arg\min}\ \dfrac{1}{2}\|\boldsymbol{Z}\|_{\boldsymbol{A},\mathcal{H}}. \end{cases} \tag{3.49}$$

Estimating the Variable $\boldsymbol{Z}$

When the coefficient matrix $\boldsymbol{A}$ is fixed as a known variable, the optimization problem associated with the variable $\boldsymbol{Z}$ becomes a general low-rank tensor completion problem. Therefore, this subproblem can be solved by using the alternating direction method of multipliers (ADMM) in a similar way as in [32] and [28]. We introduce the constraint $\boldsymbol{\mathcal{X}} = \mathcal{Q}(\boldsymbol{Z})$ that is closely related to the partially observed matrix $\boldsymbol{Y}$. The current optimization involves the variable $\boldsymbol{Z}$ and the auxiliary variable $\boldsymbol{\mathcal{X}}$, and which can be formally written as follows,

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{X}},\boldsymbol{Z}} \|\boldsymbol{\mathcal{X}}\|_{r,*} + \frac{\gamma}{2}\|\boldsymbol{Z}\|_{\boldsymbol{A},\mathcal{H}} \\ &\text{s.t.} \begin{cases} \boldsymbol{\mathcal{X}} = \mathcal{Q}(\boldsymbol{Z}), \\ \mathcal{P}_\Omega(\boldsymbol{Z}) = \mathcal{P}_\Omega(\boldsymbol{Y}). \end{cases} \end{aligned} \tag{3.50}$$

The augmented Lagrangian function of the optimization problem in Eq. (3.50) can be written as follows,

$$\begin{aligned}\mathcal{L}(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{W}}) =& \|\boldsymbol{\mathcal{X}}\|_{r,*} + \frac{\gamma}{2}\|\boldsymbol{\mathcal{Z}}\|_{\boldsymbol{\mathcal{A}},\mathcal{H}} + \frac{\lambda}{2}\|\boldsymbol{\mathcal{X}} - \mathcal{Q}(\boldsymbol{\mathcal{Z}})\|_F^2 \\ &+ \langle \boldsymbol{\mathcal{X}} - \mathcal{Q}(\boldsymbol{\mathcal{Z}}), \boldsymbol{\mathcal{W}} \rangle + \pi(\boldsymbol{\mathcal{Z}}),\end{aligned} \tag{3.51}$$

where $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{N\times I\times J}$ is the Lagrange multiplier, and λ is the hyperparameter. The symbol $\langle \cdot, \cdot \rangle$ denotes the inner product of tensors of the same size. Note that the constraint $\boldsymbol{\mathcal{X}} = \mathcal{Q}(\boldsymbol{\mathcal{Z}})$ in the optimization problem is relaxed by the Lagrange multiplier. We define $\pi(\cdot)$, corresponding to the constraint $\mathcal{P}_\Omega(\boldsymbol{\mathcal{Z}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{Y}})$, which is given by

$$\pi(\boldsymbol{\mathcal{Z}}) = \begin{cases} 0, & \text{if } \mathcal{P}_\Omega(\boldsymbol{\mathcal{Z}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{Y}}), \\ +\infty, & \text{otherwise.} \end{cases} \tag{3.52}$$

Thus, the ADMM scheme can be summarized as follows,

$$\begin{cases} \boldsymbol{\mathcal{X}} := \arg\min\limits_{\boldsymbol{\mathcal{X}}} \ \mathcal{L}(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{W}}), \\ \boldsymbol{\mathcal{Z}} := \arg\min\limits_{\boldsymbol{\mathcal{Z}}} \ \mathcal{L}(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{W}}), \\ \boldsymbol{\mathcal{W}} := \boldsymbol{\mathcal{W}} + \lambda(\boldsymbol{\mathcal{X}} - \mathcal{Q}(\boldsymbol{\mathcal{Z}})). \end{cases} \tag{3.53}$$

In what follows, we discuss the detailed solution to the variable $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Z}}$. The optimization problem over the variable $\boldsymbol{\mathcal{X}}$ is a truncated nuclear norm minimization. The truncated nuclear norm of any given tensor is the weighted sum of truncated nuclear norms on the unfolding matrices of the tensor, which takes the form:

$$\|\boldsymbol{\mathcal{X}}\|_{r,*} = \sum_{s=1}^{3} \alpha_s \|\boldsymbol{\mathcal{X}}_{(s)}\|_{r,*}, \tag{3.54}$$

for tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{N\times I\times J}$ with $\sum\limits_{s=1}^{3} \alpha_s = 1$. Herein, we consider $\alpha_s = \frac{1}{3}, \forall s$ as the default value. For the minimization of the truncated nuclear norm on tensor, the aforementioned formula is not in its appropriate form because unfolding a tensor in different modes/dimensions cannot guarantee the dependencies of the variables [32]. Therefore, we introduce a sequence of auxiliary tensors $\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \boldsymbol{\mathcal{X}}_3 \in \mathbb{R}^{N\times I\times J}$, corresponding to the unfolding matrices of $\boldsymbol{\mathcal{X}}$.

Accordingly, it is possible to obtain the closed-form solution to each $\boldsymbol{\mathcal{X}}_s$:

$$\begin{aligned}\boldsymbol{\mathcal{X}}_s &:= \arg\min_{\boldsymbol{\mathcal{X}}} \ \alpha_s\|\boldsymbol{\mathcal{X}}_{(s)}\|_{r,*} + \frac{\lambda}{2}\|\mathcal{Q}^{-1}(\boldsymbol{\mathcal{X}}) - \boldsymbol{\mathcal{Z}}\|_F^2 + \langle \mathcal{Q}^{-1}(\boldsymbol{\mathcal{X}}) - \boldsymbol{\mathcal{Z}}, \mathcal{Q}^{-1}(\boldsymbol{\mathcal{W}})\rangle \\ &= \arg\min_{\boldsymbol{\mathcal{X}}} \ \alpha_s\|\boldsymbol{\mathcal{X}}_{(s)}\|_{r,*} + \frac{\lambda}{2}\|\boldsymbol{\mathcal{X}} - (\mathcal{Q}(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{W}}/\lambda)\|_F^2 \\ &= \text{fold}_s\Big(\mathcal{D}_{r,\alpha_s/\lambda}(\mathcal{Q}(\boldsymbol{\mathcal{Z}})_{(s)} - \boldsymbol{\mathcal{W}}_{(s)}/\lambda)\Big),\end{aligned} \tag{3.55}$$

where $\mathcal{D}_{\cdot}(\cdot)$ denotes the generalized singular value thresholding that is associated with truncated nuclear norm minimization as shown in Lemma 3.2.1.

Lemma 3.2.1. *For any matrix $\boldsymbol{Z} \in \mathbb{R}^{m\times n}$, let $\alpha, \lambda > 0$ be the weight parameters and $r \in \mathbb{N}^+$ ($r < \min\{m, n\}$) be the truncation parameter, an optimal solution to the truncated nuclear norm minimization problem*

$$\min_{\boldsymbol{X}} \ \alpha\|\boldsymbol{X}\|_{r,*} + \frac{\lambda}{2}\|\boldsymbol{X} - \boldsymbol{Z}\|_F^2, \tag{3.56}$$

is given by the generalized singular value thresholding [30, 77, 78]:

$$\hat{\boldsymbol{X}} = \mathcal{D}_{r,\alpha/\lambda}(\boldsymbol{Z}) = \boldsymbol{U}\,\text{diag}([\boldsymbol{\sigma} - \mathbb{1}_{\{>r\}}\cdot\alpha/\lambda]_+)\boldsymbol{V}^\top, \tag{3.57}$$

where $\boldsymbol{U}\,\text{diag}(\boldsymbol{\sigma})\boldsymbol{V}^\top$ is the singular value decomposition (SVD) of $\boldsymbol{Z}$. $[\cdot]_+$ denotes the positive truncation at 0 which satisfies $[\sigma - \alpha/\lambda]_+ = \max\{\sigma - \alpha/\lambda, 0\}$. $\mathbb{1}_{\{>r\}} \in \{0,1\}^{\min\{m,n\}}$ is a binary indicator vector whose first r entries are 0 and other entries are 1.

Gathering the results of $\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \boldsymbol{\mathcal{X}}_3$ in Eq. (3.55), we can update the variable $\boldsymbol{\mathcal{X}}$ by

$$\boldsymbol{\mathcal{X}} := \sum_{s=1}^{3}\alpha_s\boldsymbol{\mathcal{X}}_s. \tag{3.58}$$

Given that $\boldsymbol{\mathcal{X}} = \mathcal{Q}(\boldsymbol{\mathcal{Z}})$, we can rewrite the optimization problem with respect to the variable $\boldsymbol{\mathcal{Z}}$ as follows,

$$\begin{aligned}\boldsymbol{\mathcal{Z}} &:= \arg\min_{\boldsymbol{\mathcal{Z}}} \ \frac{\gamma}{2}\|\boldsymbol{\mathcal{Z}}\|_{\boldsymbol{A},\mathcal{H}} + \frac{\lambda}{2}\|\boldsymbol{\mathcal{X}} - \mathcal{Q}(\boldsymbol{\mathcal{Z}})\|_F^2 - \langle\mathcal{Q}(\boldsymbol{\mathcal{Z}}), \boldsymbol{\mathcal{W}}\rangle \\ &= \arg\min_{\boldsymbol{\mathcal{Z}}} \ \frac{\gamma}{2}\|\boldsymbol{\mathcal{Z}}\|_{\boldsymbol{A},\mathcal{H}} + \frac{\lambda}{2}\|\boldsymbol{\mathcal{Z}} - \mathcal{Q}^{-1}(\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{W}}/\lambda)\|_F^2.\end{aligned} \tag{3.59}$$

We use Lemma 3.2.2 to solve this optimization problem.

Lemma 3.2.2. *For any multivariate time series $\boldsymbol{Z} \in \mathbb{R}^{N\times T}$ which consists of N time series*

over T consecutive time steps, the autoregressive process for any (n,t)-th entry of $\boldsymbol{Z}$ takes

$$z_{n,t} \approx \sum_{k=1}^{d} a_{n,k} z_{n,t-h_k}, \tag{3.60}$$

with autoregression coefficient matrix $\boldsymbol{A} \in \mathbb{R}^{N\times d}$ and time lag set $\mathcal{H} = \{h_1, h_2, \ldots, h_d\}$. This autoregressive process also takes the following general formula:

$$\begin{aligned} \boldsymbol{\Psi}_0 \boldsymbol{Z}^\top \approx & \sum_{k=1}^{d} \boldsymbol{\Psi}_k (\boldsymbol{a}_k^\top \odot \boldsymbol{Z}^\top) \\ = & \sum_{k=1}^{d} \boldsymbol{\Psi}_k \boldsymbol{Z}^\top \operatorname{diag}(\boldsymbol{a}_k) \\ = & \boldsymbol{\Psi}(\boldsymbol{A}^\top \odot \boldsymbol{Z}^\top), \end{aligned} \tag{3.61}$$

and for each time series $\boldsymbol{z}_n \in \mathbb{R}^T, \forall n$, we have

$$\begin{aligned} \boldsymbol{\Psi}_0 \boldsymbol{z}_n \approx & \sum_{k=1}^{d} a_{n,k} \boldsymbol{\Psi}_k \boldsymbol{z}_n \\ = & \boldsymbol{\Psi}(\boldsymbol{a}_n \otimes \boldsymbol{z}_n) \\ = & \boldsymbol{\Psi}(\boldsymbol{I}_d \otimes \boldsymbol{z}_n)\boldsymbol{a}_n, \end{aligned} \tag{3.62}$$

where $\odot$ and $\otimes$ denote the Khatri-Rao product (see Definition 3.2.1) and the Kronecker product (see Definition 3.1.2), respectively. The temporal operator matrix $\boldsymbol{\Psi}_k$ associated with the time lag set $\mathcal{H}$ is defined as follows,

$$\boldsymbol{\Psi}_k \triangleq \begin{bmatrix} \boldsymbol{0}_{(T-h_d)\times(h_d-h_k)} & \boldsymbol{I}_{T-h_d} & \boldsymbol{0}_{(T-h_d)\times h_k} \end{bmatrix} \in \mathbb{R}^{(T-h_d)\times T}, \forall k \in \{0, 1, \ldots, d\}, \tag{3.63}$$

where $h_0 = 0$. On these temporal operator matrices,

$$\boldsymbol{\Psi} = \begin{bmatrix} \boldsymbol{\Psi}_1 & \boldsymbol{\Psi}_2 & \cdots & \boldsymbol{\Psi}_d \end{bmatrix} \in \mathbb{R}^{(T-h_d)\times(dT)}, \tag{3.64}$$

is introduced as the augmented matrix.

Definition 3.2.1 (Khatri-Rao Product [8])**.** *For any matrices $\boldsymbol{X} \in \mathbb{R}^{m\times d}$ and $\boldsymbol{Y} \in \mathbb{R}^{n\times d}$ with the same number of columns such that*

$$\boldsymbol{X} = \begin{bmatrix} | & | & & | \\ \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_d \\ | & | & & | \end{bmatrix}, \qquad \boldsymbol{Y} = \begin{bmatrix} | & | & & | \\ \boldsymbol{y}_1 & \boldsymbol{y}_2 & \cdots & \boldsymbol{y}_d \\ | & | & & | \end{bmatrix}, \tag{3.65}$$

the Khatri-Rao product between $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *is given by*

$$\boldsymbol{X} \odot \boldsymbol{Y} = \begin{bmatrix} | & | & & | \\ \boldsymbol{x}_1 \otimes \boldsymbol{y}_1 & \boldsymbol{x}_2 \otimes \boldsymbol{y}_2 & \cdots & \boldsymbol{x}_d \otimes \boldsymbol{y}_d \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{(mn)\times d}, \tag{3.66}$$

where the resulting matrix has d *columns, and each column is of length* mn*.*

According to Lemma 3.2.2, we have the following optimization problem with respect to the variable $\boldsymbol{Z}$:

$$\begin{aligned} \boldsymbol{Z} := \arg\min_{\boldsymbol{Z}} \frac{\gamma}{2}\Big\|\boldsymbol{Z}\boldsymbol{\Psi}_0^\top - \sum_{k=1}^{d} \operatorname{diag}(\boldsymbol{a}_k)\boldsymbol{Z}\boldsymbol{\Psi}_k^\top\Big\|_F^2 \\ + \frac{\lambda}{2}\|\boldsymbol{Z} - \mathcal{Q}^{-1}(\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{W}}/\lambda)\|_F^2. \end{aligned} \tag{3.67}$$

Let f be the objective function, then we can write down the partial derivative of f with respect to the variable $\boldsymbol{Z}$ as follows,

$$\begin{aligned} \frac{\partial f}{\partial \boldsymbol{Z}} =& \gamma\Big(\boldsymbol{Z}\boldsymbol{\Psi}_0^\top - \sum_{h=1}^{d} \operatorname{diag}(\boldsymbol{a}_h)\boldsymbol{Z}\boldsymbol{\Psi}_h^\top\Big)\boldsymbol{\Psi}_0 \\ & - \gamma \sum_{k=1}^{d} \operatorname{diag}(\boldsymbol{a}_k)\Big(\boldsymbol{Z}\boldsymbol{\Psi}_0^\top - \sum_{h=1}^{d} \operatorname{diag}(\boldsymbol{a}_h)\boldsymbol{Z}\boldsymbol{\Psi}_h^\top\Big)\boldsymbol{\Psi}_k \\ & + \lambda\boldsymbol{Z} - \lambda\mathcal{Q}^{-1}(\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{W}}/\lambda) \\ =& \gamma \sum_{k=0}^{d}\sum_{h=0}^{d} \operatorname{diag}(\boldsymbol{a}_k)\operatorname{diag}(\boldsymbol{a}_h)\boldsymbol{Z}\boldsymbol{\Psi}_h^\top\boldsymbol{\Psi}_k + \lambda\boldsymbol{Z} \\ & - \mathcal{Q}^{-1}(\lambda\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{W}}), \end{aligned} \tag{3.68}$$

where $\boldsymbol{a}_0 = -\mathbb{1}_d \in \mathbb{R}^d$ is the vector of negative ones.

Let $\frac{\partial f}{\partial \boldsymbol{Z}} = \boldsymbol{0}$, then we have a generalized Sylvester equation:

$$\gamma \sum_{k=0}^{d}\sum_{h=0}^{d} \operatorname{diag}(\boldsymbol{a}_k)\operatorname{diag}(\boldsymbol{a}_h)\boldsymbol{Z}\boldsymbol{\Psi}_h^\top\boldsymbol{\Psi}_k + \lambda\boldsymbol{Z} = \mathcal{Q}^{-1}(\lambda\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{W}}), \tag{3.69}$$

which can be solved by using the conjugate gradient method. Let

$$\begin{cases} \theta_z(\boldsymbol{Z}) \triangleq \operatorname{vec}\Big(\gamma \sum_{k=0}^{d}\sum_{h=0}^{d} \operatorname{diag}(\boldsymbol{a}_k)\operatorname{diag}(\boldsymbol{a}_h)\boldsymbol{Z}\boldsymbol{\Psi}_h^\top\boldsymbol{\Psi}_k + \lambda\boldsymbol{Z}\Big), \\ \psi_z \triangleq \operatorname{vec}(\mathcal{Q}^{-1}(\lambda\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{W}})), \end{cases} \tag{3.70}$$

be the vectorization of the left-hand side and the right-hand side of Eq. (3.69), respectively.

The conjugate gradient method for solving the variable $\boldsymbol{Z}$ is given by Algorithm 4.

Algorithm 4 Conjugate Gradient for Approximating $\boldsymbol{Z}$

Input: Variables $\{\boldsymbol{\mathcal{X}}, \boldsymbol{Z}, \boldsymbol{\mathcal{W}}\}$, coefficient matrix $\boldsymbol{A}$, temporal operator matrices $\{\boldsymbol{\Psi}_k\}$, hyperparameter $\{\gamma, \lambda\}$, and maximum iteration $\ell_{\max}$ (e.g., $\ell_{\max} = 5$).
Output: Estimated matrix $\boldsymbol{Z}$.

Initialize $\boldsymbol{z}_0$ by the vectorized $\boldsymbol{Z}$.
Compute residual vector $\boldsymbol{r}_0 = \psi_z - \theta_z(\boldsymbol{Z})$, and let $\boldsymbol{d}_0 = \boldsymbol{r}_0$.
for $\ell = 0$ **to** $\ell_{\max} - 1$ **do**
  Convert vector $\boldsymbol{d}_\ell$ into matrix $\boldsymbol{D}_\ell$.
  Compute $\alpha_\ell = \frac{\boldsymbol{r}_\ell^\top \boldsymbol{r}_\ell}{\boldsymbol{d}_\ell^\top \theta_z(\boldsymbol{D}_\ell)}$.
  Update $\boldsymbol{z}_{\ell+1} = \boldsymbol{z}_\ell + \alpha_\ell \boldsymbol{d}_\ell$.
  Update $\boldsymbol{r}_{\ell+1} = \boldsymbol{r}_\ell - \alpha_\ell \theta_z(\boldsymbol{D}_\ell)$.
  Compute $\beta_\ell = \frac{\boldsymbol{r}_{\ell+1}^\top \boldsymbol{r}_{\ell+1}}{\boldsymbol{r}_\ell^\top \boldsymbol{r}_\ell}$.
  Update $\boldsymbol{d}_{\ell+1} = \boldsymbol{r}_{\ell+1} + \beta_\ell \boldsymbol{d}_\ell$.
end for
Convert $\boldsymbol{z}_{\ell_{\max}}$ into matrix $\boldsymbol{Z}$.
return $\boldsymbol{Z}$.

Estimating the Coefficient Matrix $\boldsymbol{A}$

As mentioned above, $\boldsymbol{A} \in \mathbb{R}^{N \times d}$ is the coefficient matrix in the defined temporal variation term. For the univariate autoregression, we can compute with the rows of the coefficient matrix $\boldsymbol{A}$ independently from the optimization problem in Eq. (3.49). As a consequence, the optimization problem associated with the coefficients shows a closed-form solution, which is given by

$$\begin{aligned} \boldsymbol{a}_n :=& \arg\min_{\boldsymbol{a}_n} \ \frac{1}{2}\Big\| \boldsymbol{\Psi}_0 \boldsymbol{z}_n - \sum_{k=1}^{d} a_{n,k} \boldsymbol{\Psi}_k \boldsymbol{z}_n \Big\|_2^2 \\ =& \arg\min_{\boldsymbol{a}_n} \ \frac{1}{2}\| \boldsymbol{\Psi}_0 \boldsymbol{z}_n - \boldsymbol{\Psi}(\boldsymbol{I}_d \otimes \boldsymbol{z}_n) \boldsymbol{a}_n \|_2^2 \\ =& [\boldsymbol{\Psi}(\boldsymbol{I}_d \otimes \boldsymbol{z}_n)]^\dagger \boldsymbol{\Psi}_0 \boldsymbol{z}_n, \ \forall n. \end{aligned} \tag{3.71}$$

Solution Algorithm

Algorithm 5 shows the implementation of the proposition solution to estimate the LATC model. In the whole procedure, the inner iterative process for solving the variable $\boldsymbol{Z}$ is the ADMM scheme, requiring a relatively small number of iterations (e.g., $K = 3$). In the ADMM scheme, we update the variables $\boldsymbol{\mathcal{X}}$, $\boldsymbol{Z}$, and $\boldsymbol{\mathcal{W}}$, iteratively, and let $\mathcal{P}_\Omega(\boldsymbol{Z}) = \mathcal{P}_\Omega(\boldsymbol{Y})$ after each iteration. For optimizing the coefficient matrix $\boldsymbol{A} \in \mathbb{R}^{N \times d}$, we implement the least squares update for each row independently. Doing the whole procedure of LATC, it allows

one to capture both low-rank patterns and time series correlations of spatiotemporal traffic data.

Algorithm 5 Low-Rank Autoregressive Tensor Completion

Input: Data $\boldsymbol{Y} \in \mathbb{R}^{N\times(IJ)}$ with the observed index set Ω, time lag set $\mathcal{H} = \{h_1, h_2, \ldots, h_d\}$, truncation parameter $r \in \mathbb{N}^+$, and hyperparameters $\{\gamma, \lambda, \lambda_{\max}\}$.
Output: Reconstructed matrix $\hat{\boldsymbol{Y}}$ and coefficient matrix $\boldsymbol{A}$.

Initialize the variable $\boldsymbol{\mathcal{W}}$ as zeros and $\boldsymbol{A}$ as small random values.
Generate temporal operator matrices $\{\boldsymbol{\Psi}_k\}$.
repeat
  for $\nu = 1$ **to** K (e.g., $K = 3$) **do**
    Compute $\lambda = \min\{1.05 \times \lambda, \lambda_{\max}\}$.
    for $s = 1$ **to** 3 **do**
      Compute $\boldsymbol{\mathcal{X}}_s$ by Eq. (3.55).
    end for
    Compute $\boldsymbol{\mathcal{X}}$ by Eq. (3.58).
    Compute $\boldsymbol{Z}$ from Eq. (3.69) with conjugate gradient (see Algorithm 4).
    Compute $\boldsymbol{\mathcal{W}}$ by Eq. (3.53).
    Transform observation information by letting $\mathcal{P}_\Omega(\boldsymbol{Z}) = \mathcal{P}_\Omega(\boldsymbol{Y})$.
  end for
  for $n = 1$ **to** N **do**
    Compute $\boldsymbol{a}_n$ by Eq. (3.71).
  end for
until convergence
Compute the reconstructed matrix by $\hat{\boldsymbol{Y}} := \mathcal{Q}^{-1}(\boldsymbol{\mathcal{X}})$.
return $\hat{\boldsymbol{Y}}, \boldsymbol{A}$.

3.3 Low-Rank Laplacian Convolutional Representation

In this section, we introduce a low-rank LCR model for imputing both univariate and multivariate traffic time series. The cornerstone of LCR includes the circulant matrix/tensor nuclear norm minimization and the Laplacian kernelized regularization. The circulant matrix/tensor nuclear norm minimization allows one to reveal the underlying low-rank property of sparse traffic data, while the Laplacian kernelized regularization is capable of characterizing the local trends of traffic time series. In particular, we propose to formulate the Laplacian kernelized regularization in the form of circular convolution and give a fast implementation of the involved optimization through FFT in the frequency domain, instead of the time domain.

3.3.1 Laplacian Kernel

While graph modeling brings new insights into relational data analysis and machine learning, we propose to characterize the temporal dependencies of time series through undirected and circulant graphs. Recall that the Laplacian matrix is defined as

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A} \in \mathbb{R}^{T \times T}, \tag{3.72}$$

for T nodes/observations in a graph, in which $\boldsymbol{D} \in \mathbb{R}^{T \times T}$ and $\boldsymbol{A} \in \mathbb{R}^{T \times T}$ are the (diagonal) degree matrix and adjacency matrix, respectively [79]. As exemplified by Figure 3.4, the Laplacian matrices of these two graphs are

$$\boldsymbol{L} = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{bmatrix}, \tag{3.73}$$

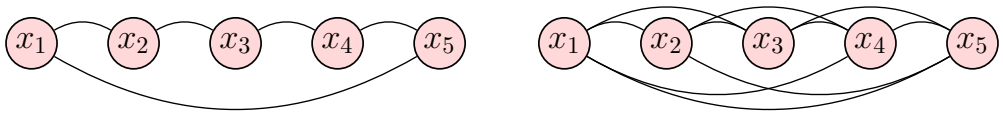
and

$$\boldsymbol{L} = \begin{bmatrix} 4 & -1 & -1 & -1 & -1 \\ -1 & 4 & -1 & -1 & -1 \\ -1 & -1 & 4 & -1 & -1 \\ -1 & -1 & -1 & 4 & -1 \\ -1 & -1 & -1 & -1 & 4 \end{bmatrix}, \tag{3.74}$$

respectively. Note that Laplacian matrices are symmetric.

Recall that the circulant matrix is an important structure that shows broad use in the field of signal processing [48, 80]. We give a concise description of the circulant matrix in Definition 3.3.1. Notably, Laplacian matrices as mentioned in Eqs. (3.73) and (3.74) are circulant matrices, and their first columns are given by $\boldsymbol{\ell} = (2, -1, 0, 0, -1)^\top$ and $\boldsymbol{\ell} = (4, -1, -1, -1, -1)^\top$, respectively.

Definition 3.3.1 (Circulant Matrix)**.** *For any vector* $\boldsymbol{x} = (x_1, x_2, \cdots, x_T)^\top \in \mathbb{R}^T$*, the cir-*



(a) Circulant graph with degree 2. (b) circulant graph with degree 4.

Figure 3.4 Undirected and circulant graphs on the relational data samples $\{x_1, x_2, \ldots, x_5\}$ with certain degrees.

culant matrix can be written as

$$\mathcal{C}(\boldsymbol{x}) \triangleq \begin{bmatrix} x_1 & x_T & x_{T-1} & \cdots & x_2 \\ x_2 & x_1 & x_T & \cdots & x_3 \\ x_3 & x_2 & x_1 & \cdots & x_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_T & x_{T-1} & x_{T-2} & \cdots & x_1 \end{bmatrix} \in \mathbb{R}^{T\times T}, \tag{3.75}$$

where $\mathcal{C} : \mathbb{R}^T \to \mathbb{R}^{T\times T}$ denotes the circulant operator. The first column of $\mathcal{C}(\boldsymbol{x})$ is the vector $\boldsymbol{x}$, and the diagonal entries of $\mathcal{C}(\boldsymbol{x})$ are all equal to x_1.

Following the definition of the circulant matrix, we introduce a Laplacian kernel as described in Definition 3.3.2, which allows one to characterize temporal dependencies of time series. In the aforementioned cases, the first column of the Laplacian matrix $\boldsymbol{L}$ is indeed a simple example of the Laplacian kernel.

Definition 3.3.2 ((Circular) Laplacian Kernel)**.** *Given any time series $\boldsymbol{x} = (x_1, \ldots, x_T)^\top \in \mathbb{R}^T$, suppose $\tau \in \mathbb{N}^+$ ($\tau \leq \frac{1}{2}(T-1)$) be the kernel size of an undirected and circulant graph, then the Laplacian kernel is defined as*

$$\boldsymbol{\ell} \triangleq (2\tau, \underbrace{-1, \cdots, -1}_{\tau}, 0, \cdots, 0, \underbrace{-1, \cdots, -1}_{\tau})^\top \in \mathbb{R}^T, \tag{3.76}$$

which is also the first column of the Laplacian matrix and the inherent degree matrix is diagonalized with entries 2τ.

Remark 3. *Here, it is possible to define the Laplacian kernel as a kind of dictionary in the convolutional representation [81–83] for temporal modeling. As a special case, if we consider the directed and circulant graphs in Figure 3.4, then the Laplacian kernels would be $\boldsymbol{\ell} = (1, 0, 0, 0, -1)^\top$ and $\boldsymbol{\ell} = (2, 0, 0, -1, -1)^\top$, respectively. In fact, this case leads to*

the temporal regularization in the form of random walk [42], which can be demonstrated to reinforce the local trends in time series reconstruction.

Essentially, according to Definition 3.3.2, the corresponding Laplacian matrix $\boldsymbol{L} \in \mathbb{R}^{T\times T}$ can be characterized as a circulant matrix. To reformulate the temporal regularization, one can build the connection between circular convolution (see Definition 3.3.3) and circulant matrix. We begin with circular convolution on the vectors $\boldsymbol{x} \in \mathbb{R}^T$ and $\boldsymbol{y} \in \mathbb{R}^{\tilde{\tau}}$ (see Definition 3.3.3). According to the rule of circular convolution, we have the following expression to compute $\boldsymbol{x} \star \boldsymbol{y}$ by introducing matrix-vector multiplication, i.e.,

$$\boldsymbol{x} \star \boldsymbol{y} \equiv \mathcal{C}_{\tilde{\tau}}(\boldsymbol{x})\boldsymbol{y}, \tag{3.77}$$

for any $\boldsymbol{x} \in \mathbb{R}^T$ and $\boldsymbol{y} \in \mathbb{R}^{\tilde{\tau}}$, in which $\mathcal{C}_{\tilde{\tau}} : \mathbb{R}^T \to \mathbb{R}^{T\times\tilde{\tau}}$ denotes the convolution operator and the resultant convolution matrix [46, 47] is given by

$$\mathcal{C}_{\tilde{\tau}}(\boldsymbol{x}) \triangleq \begin{bmatrix} x_1 & x_T & x_{T-1} & \cdots & x_{T-\tilde{\tau}+2} \\ x_2 & x_1 & x_T & \cdots & x_{T-\tilde{\tau}+3} \\ x_3 & x_2 & x_1 & \cdots & x_{T-\tilde{\tau}+4} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_T & x_{T-1} & x_{T-2} & \cdots & x_{T-\tilde{\tau}+1} \end{bmatrix} \in \mathbb{R}^{T\times\tilde{\tau}}, \tag{3.78}$$

where $\tilde{\tau}$ is the window length. In fact, $\mathcal{C}_{\tilde{\tau}}(\boldsymbol{x})$ consists of the first $\tilde{\tau}$ columns of the circulant matrix $\mathcal{C}(\boldsymbol{x})$.

Let $\boldsymbol{x} = (x_1, x_2, \cdots, x_T)^\top \in \mathbb{R}^T$ and $\boldsymbol{y} = (y_1, y_2, \cdots, y_T)^\top \in \mathbb{R}^T$, namely, in the case of $\tilde{\tau} = T$, then we have the following property for the circular convolution:

$$\boldsymbol{x} \star \boldsymbol{y} \equiv \mathcal{C}(\boldsymbol{x})\boldsymbol{y} \equiv \mathcal{C}(\boldsymbol{y})\boldsymbol{x}, \tag{3.79}$$

where $\mathcal{C}(\boldsymbol{x})$ is the circulant matrix as defined in Eq. (3.75), showing to be a special case of the convolution matrix. Accordingly, we can also define the circulant matrix $\mathcal{C}(\boldsymbol{y})$ on the vector $\boldsymbol{y}$.

Definition 3.3.3 (Circular Convolution)**.** *For any vectors*

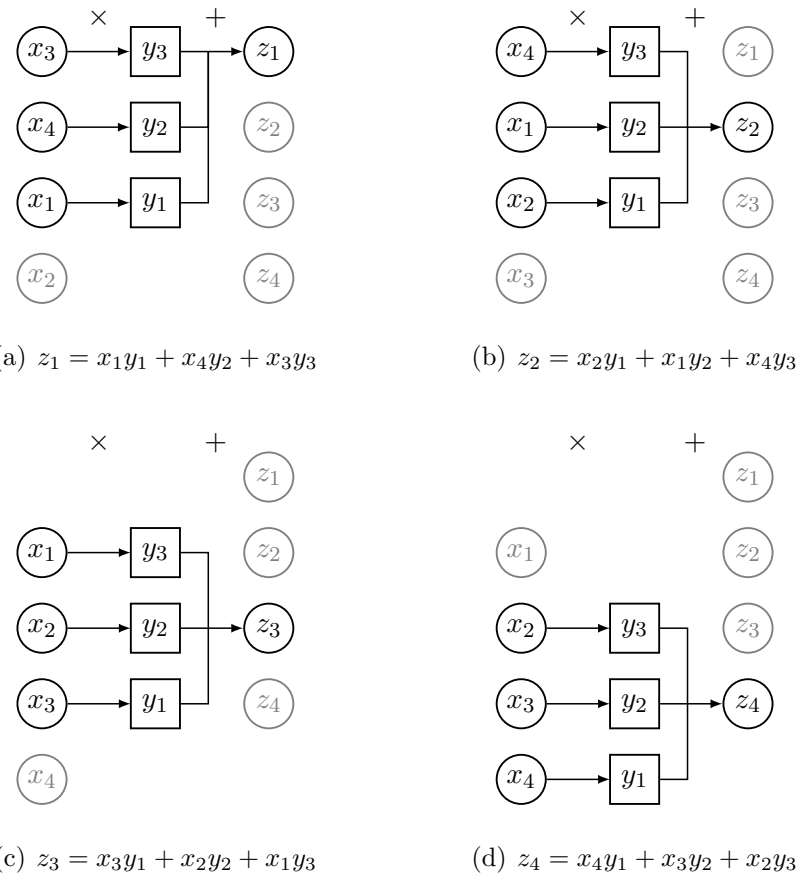
$$\boldsymbol{x} = (x_1, x_2, \cdots, x_T)^\top \in \mathbb{R}^T, \quad \boldsymbol{y} = (y_1, y_2, \cdots, y_{\tilde{\tau}})^\top \in \mathbb{R}^{\tilde{\tau}}, \tag{3.80}$$

with $\tilde{\tau} \leq T$, the circular convolution of two vectors is $\boldsymbol{z} = \boldsymbol{x} \star \boldsymbol{y} \in \mathbb{R}^T$ [84] (see e.g.,

Figure 3.5), denoting the operator with the symbol $\star$; element-wise, we have

$$z_t = \sum_{k=1}^{\tilde{\tau}} x_{t-k+1} y_k, \; \forall t \in \{1, 2, \ldots, T\}, \tag{3.81}$$

where z_t is the t-th entry of $\boldsymbol{z}$ and $x_{t-k+1} = x_{t-k+1+T}$ for $t + 1 \leq k$.



(a) $z_1 = x_1y_1 + x_4y_2 + x_3y_3$ (b) $z_2 = x_2y_1 + x_1y_2 + x_4y_3$

(c) $z_3 = x_3y_1 + x_2y_2 + x_1y_3$ (d) $z_4 = x_4y_1 + x_3y_2 + x_2y_3$

Figure 3.5 Circular convolution between vectors $\boldsymbol{x} \in \mathbb{R}^4$ and $\boldsymbol{y} \in \mathbb{R}^3$.

Therefore, according to the definitions of Laplacian kernel and circulant matrix and the relationship between circular convolution and circulant matrix (see Eq. (3.79)), we can reformulate the temporal regularization as follows,

$$\mathcal{R}_\tau(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{L}\boldsymbol{x}\|_2^2 = \frac{1}{2}\|\mathcal{C}(\boldsymbol{\ell})\boldsymbol{x}\|_2^2 = \frac{1}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2. \tag{3.82}$$

In what follows, we utilize the Convolution Theorem (see Theorem 3.3.1) to reformulate the temporal regularization in the frequency domain.

Theorem 3.3.1 (Convolution Theorem [84]). *For any vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^T$, a circular convolution in the time domain is a product in the frequency domain, formally, it always holds that*

$$\boldsymbol{x} \star \boldsymbol{y} = \mathcal{F}^{-1}(\mathcal{F}(\boldsymbol{x}) \circ \mathcal{F}(\boldsymbol{y})), \tag{3.83}$$

where $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ denote the discrete Fourier transform and the inverse discrete Fourier transform, respectively. $\mathcal{F}(\boldsymbol{x}), \mathcal{F}(\boldsymbol{y}) \in \mathbb{C}^T$ are the results of discrete Fourier transform on $\boldsymbol{x}, \boldsymbol{y}$ with $\mathbb{C}$ denoting the set of complex numbers. The symbol $\circ$ denotes the Hadamard product, namely, the element-wise product.

Typically, Theorem 3.3.1 gives a concise description of the relationship between circular convolution and discrete Fourier transform, and it shows that circular convolution can be implemented in the frequency domain. Circulant matrix is advantageous because the required matrix-vector product can be done efficiently by leveraging the structure. Since the Laplacian kernel in this case holds the property of circulant matrix (see Eq. (3.79)), the temporal regularization can be therefore reformulated as follows,

$$\mathcal{R}_\tau(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2 = \frac{1}{2T}\|\mathcal{F}(\boldsymbol{\ell}) \circ \mathcal{F}(\boldsymbol{x})\|_2^2. \tag{3.84}$$

As can be seen, the temporal regularization through the Laplacian matrix is converted into the formula that is associated with the Laplacian kernel in the frequency domain. It seems that the length-T Laplacian kernel $\boldsymbol{\ell}$ can represent the graphical relationship of T-by-T Laplacian matrix $\boldsymbol{L}$ in the temporal regularization, showing no need for constructing the matrix $\boldsymbol{L}$. In the temporal regularization, the coefficients are governed by the Laplacian kernel.

Remark 4. *For Eq.* (3.84)*, we can prove the statement as follows. Let*

$$\begin{cases} \boldsymbol{\alpha} = \boldsymbol{\ell} \star \boldsymbol{x}, \\ \boldsymbol{\beta} = \mathcal{F}(\boldsymbol{\ell}) \circ \mathcal{F}(\boldsymbol{x}), \end{cases} \tag{3.85}$$

then it always holds that

$$\boldsymbol{\alpha} = \mathcal{F}^{-1}(\boldsymbol{\beta}) \quad or \quad \mathcal{F}(\boldsymbol{\alpha}) = \boldsymbol{\beta}. \tag{3.86}$$

Furthermore, according to the Parseval's Theorem [84], we get

$$\|\boldsymbol{\alpha}\|_2^2 = \frac{1}{T}\|\mathcal{F}(\boldsymbol{\alpha})\|_2^2 = \frac{1}{T}\|\boldsymbol{\beta}\|_2^2, \tag{3.87}$$

as claimed in Eq. (3.84).

3.3.2 Univariate Time Series Imputation Model

Model Description

In this work, we are aiming to incorporate both global and local consistency in the modeling process of incomplete traffic time series via the use of ConvNNM and CircNNM. We propose a low-rank completion model, i.e., LCR, in which we utilize circulant matrix nuclear norm to pursue the global trends and use the temporal regularization via Laplacian kernel to characterize the local trends in time series (see Figure 3.6 for an illustration). Formally, the LCR model can be formulated as follows,

$$\begin{aligned} &\min_{\boldsymbol{x}} \|\mathcal{C}(\boldsymbol{x})\|_* + \gamma \cdot \mathcal{R}_\tau(\boldsymbol{x}) \\ &\text{s.t. } \mathcal{P}_\Omega(\boldsymbol{x}) = \mathcal{P}_\Omega(\boldsymbol{y}), \end{aligned} \tag{3.88}$$

where the notation $\|\cdot\|_*$ denotes the nuclear norm of the matrix, which is defined as the sum of singular values of the matrix. The vector $\boldsymbol{x} \in \mathbb{R}^T$ is the reconstructed time series corresponding to the partially observed time series $\boldsymbol{y}$. In the objective function, γ is the weight parameter.

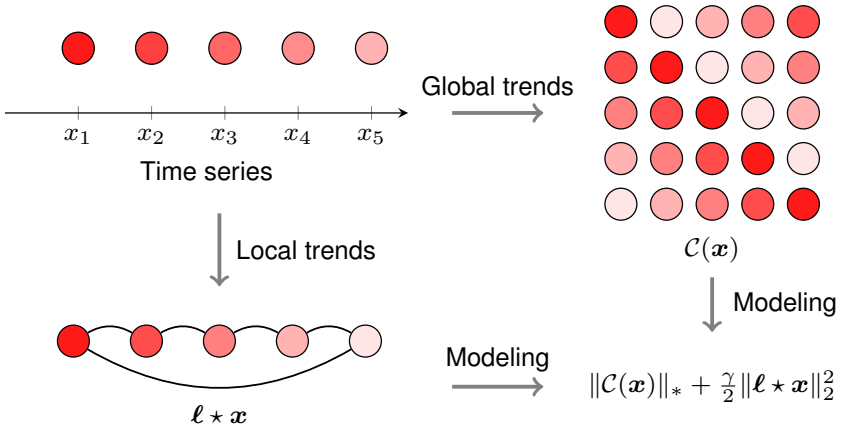


Figure 3.6 Illustration of the proposed LCR model. The global trends and local trends are characterized by the circulant matrix and Laplacian kernel, respectively. The whole framework is expected to capture both global low-rank patterns and local time series trends in traffic data.

Since traffic time series data are usually noisy, the strong observation constraint in Eq. (3.88) should be replaced by $\|\mathcal{P}_\Omega(\boldsymbol{z}-\boldsymbol{y})\|_2 \leq \epsilon$ in which $\epsilon \geq 0$ is the tolerance. Now, the optimization

problem of LCR is given by

$$\begin{aligned} \min_{\boldsymbol{x}} \ & \|\mathcal{C}(\boldsymbol{x})\|_* + \gamma \cdot \mathcal{R}_\tau(\boldsymbol{x}) \\ \text{s.t.} \ & \|\mathcal{P}_\Omega(\boldsymbol{x}-\boldsymbol{y})\|_2 \leq \epsilon. \end{aligned} \tag{3.89}$$

Our LCR model stems from ConvNNM [46, 47], and it can be solved by the ADMM framework. To resolve the convex optimization problem of LCR in Eq. (3.89), we introduce an auxiliary variable $\boldsymbol{z}$ to preserve the observation information. Thus, the optimization problem now becomes

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{z}} \ & \|\mathcal{C}(\boldsymbol{x})\|_* + \gamma \cdot \mathcal{R}_\tau(\boldsymbol{x}) + \eta \cdot \pi(\boldsymbol{z}) \\ \text{s.t.} \ & \boldsymbol{x} = \boldsymbol{z}, \end{aligned} \tag{3.90}$$

where η is the weight parameter. We define $\pi(\cdot)$ corresponding to the reconstructed error between $\boldsymbol{z}$ and $\boldsymbol{y}$ in the set Ω, which is given by

$$\pi(\boldsymbol{z}) = \frac{1}{2}\|\mathcal{P}_\Omega(\boldsymbol{z}-\boldsymbol{y})\|_2^2. \tag{3.91}$$

To reinforce both global and local trends in the reconstructed time series, the observation constraint can be related to the noisy version as shown in Eq. (3.89), thus leading to the denoised and smooth time series in $\boldsymbol{x}$.

Accordingly, the augmented Lagrangian function of Eq. (3.90) can be written as follows,

$$\begin{aligned} \mathcal{L}(\boldsymbol{x},\boldsymbol{z},\boldsymbol{w}) = & \|\mathcal{C}(\boldsymbol{x})\|_* + \gamma \cdot \mathcal{R}_\tau(\boldsymbol{x}) + \frac{\lambda}{2}\|\boldsymbol{x}-\boldsymbol{z}\|_2^2 \\ & + \langle \boldsymbol{w}, \boldsymbol{x}-\boldsymbol{z}\rangle + \eta \cdot \pi(\boldsymbol{z}), \end{aligned} \tag{3.92}$$

where $\boldsymbol{w} \in \mathbb{R}^T$ is the Lagrange multiplier, and λ is the hyperparameter. The symbol $\langle \cdot, \cdot \rangle$ denotes the inner product, e.g., $\langle \boldsymbol{w}, \boldsymbol{x}-\boldsymbol{z}\rangle = \boldsymbol{w}^\top(\boldsymbol{x}-\boldsymbol{z}) \in \mathbb{R}$. Note that the constraint $\boldsymbol{x} = \boldsymbol{z}$ in the optimization problem is relaxed by the Lagrange multiplier.

Thus, the ADMM scheme can be summarized as follows,

$$\begin{cases} \boldsymbol{x} := \arg\min_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x},\boldsymbol{z},\boldsymbol{w}), \\ \boldsymbol{z} := \arg\min_{\boldsymbol{z}} \mathcal{L}(\boldsymbol{x},\boldsymbol{z},\boldsymbol{w}), \\ \boldsymbol{w} := \boldsymbol{w} + \lambda(\boldsymbol{x}-\boldsymbol{z}). \end{cases} \tag{3.93}$$

Estimating the Variable $\boldsymbol{x}$

In particular, with respect to the variable $\boldsymbol{x}$, if we rewrite the regularization terms in Eq. (3.93) as follows,

$$f = \gamma \cdot \mathcal{R}_\tau(\boldsymbol{x}) + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{w}/\lambda\|_2^2, \tag{3.94}$$

then the formula for $\mathcal{R}_\tau(\boldsymbol{x})$ can be converted into a discrete Fourier transformed copy (see Eq. (3.84)), and the Parseval's Theorem is applicable to the remaining term of f. Thus, the above formula for f is equivalent to

$$\begin{aligned} f =& \frac{\gamma}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{w}/\lambda\|_2^2 \\ =& \frac{\gamma}{2T}\|\mathcal{F}(\boldsymbol{\ell}) \circ \mathcal{F}(\boldsymbol{x})\|_2^2 + \frac{\lambda}{2T}\|\mathcal{F}(\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{w}/\lambda)\|_2^2 \\ =& \frac{\gamma}{2T}\|\hat{\boldsymbol{\ell}} \circ \hat{\boldsymbol{x}}\|_2^2 + \frac{\lambda}{2T}\|\hat{\boldsymbol{x}} - \hat{\boldsymbol{z}} + \hat{\boldsymbol{w}}/\lambda\|_2^2, \end{aligned} \tag{3.95}$$

where we let $\hat{\boldsymbol{\ell}} = \mathcal{F}(\boldsymbol{\ell})$ be the discrete Fourier transformed Laplacian kernel, and we introduce the variables $\{\hat{\boldsymbol{x}}, \hat{\boldsymbol{z}}, \hat{\boldsymbol{w}}\} = \{\mathcal{F}(\boldsymbol{x}), \mathcal{F}(\boldsymbol{z}), \mathcal{F}(\boldsymbol{w})\}$ referring to the variables $\{\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{w}\}$ in the frequency domain after discrete Fourier transform.

Accordingly, the partial derivative of the function f with respect to the variable $\hat{\boldsymbol{x}}$ is given by

$$\begin{aligned} \frac{\partial f}{\partial \hat{\boldsymbol{x}}} =& \frac{\gamma}{T}\hat{\boldsymbol{\ell}} \circ \hat{\boldsymbol{\ell}}^* \circ \hat{\boldsymbol{x}} + \frac{\lambda}{T}(\hat{\boldsymbol{x}} - \hat{\boldsymbol{z}} + \hat{\boldsymbol{w}}/\lambda) \\ =& \frac{1}{T}(\gamma\hat{\boldsymbol{\ell}}^* \circ \hat{\boldsymbol{\ell}} + \lambda\mathbb{1}_T) \circ \hat{\boldsymbol{x}} - \frac{1}{T}(\lambda\hat{\boldsymbol{z}} - \hat{\boldsymbol{w}}), \end{aligned} \tag{3.96}$$

where $\mathbb{1}_T \in \mathbb{R}^T$ is the vector that is comprised of ones. The notation $\cdot^*$ represents the complex conjugate.

Let $\frac{\partial f}{\partial \hat{\boldsymbol{x}}} = \boldsymbol{0}$, then it produces a closed-form solution such that

$$\hat{\boldsymbol{x}} = (\lambda\hat{\boldsymbol{z}} - \hat{\boldsymbol{w}}) \oslash (\gamma\hat{\boldsymbol{\ell}}^* \circ \hat{\boldsymbol{\ell}} + \lambda\mathbb{1}_T), \tag{3.97}$$

where $\oslash$ denotes the Hadamard division.

Remark 5. *In this case, the function f can also be written as follows,*

$$f = \frac{\gamma}{2}\|\mathcal{C}(\boldsymbol{\ell})\boldsymbol{x}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{w}/\lambda\|_2^2, \tag{3.98}$$

via the use of a circulant matrix, and we therefore have

$$\begin{aligned}\frac{\partial f}{\partial \boldsymbol{x}} =&\gamma \mathcal{C}(\boldsymbol{\ell})^\top \mathcal{C}(\boldsymbol{\ell})\boldsymbol{x} + \lambda(\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{w}/\lambda)\\ =&\gamma\phi(\boldsymbol{\ell}) \star \boldsymbol{\ell} \star \boldsymbol{x} + \lambda\boldsymbol{x} - \lambda\boldsymbol{z} + \boldsymbol{w}\\ =&(\gamma\phi(\boldsymbol{\ell}) \star \boldsymbol{\ell} + \lambda\mathbb{I}_T) \star \boldsymbol{x} - \lambda\boldsymbol{z} + \boldsymbol{w},\end{aligned} \tag{3.99}$$

where $\phi(\boldsymbol{\ell}) = (\ell_1, \ell_T, \ell_{T-1}, \cdots, \ell_3, \ell_2)^\top \in \mathbb{R}^T$ *is the permutation of the vector* $\boldsymbol{\ell} = (\ell_1, \ell_2, \cdots, \ell_T)^\top \in \mathbb{R}^T$*, see Definition 3.3.4 for details. In addition, we introduce* $\mathbb{I}_T = (1, 0, 0, \cdots, 0, 0)^\top \in \mathbb{R}^T$ *as the first column of* T*-by-*T *identity matrix.*

Let $\frac{\partial f}{\partial \boldsymbol{x}} = \mathbf{0}$*, it gives that*

$$(\gamma\phi(\boldsymbol{\ell}) \star \boldsymbol{\ell} + \lambda\mathbb{I}_T) \star \boldsymbol{x} = \lambda\boldsymbol{z} - \boldsymbol{w} \tag{3.100}$$

$$\begin{aligned}\Longrightarrow \quad \hat{\boldsymbol{x}} =&\mathcal{F}(\boldsymbol{x})\\ =&\mathcal{F}(\lambda\boldsymbol{z} - \boldsymbol{w}) \oslash \mathcal{F}(\gamma\phi(\boldsymbol{\ell}) \star \boldsymbol{\ell} + \lambda\mathbb{I}_T)\\ =&(\lambda\mathcal{F}(\boldsymbol{z}) - \mathcal{F}(\boldsymbol{w})) \oslash (\gamma\mathcal{F}(\phi(\boldsymbol{\ell})) \circ \mathcal{F}(\boldsymbol{\ell}) + \lambda\mathcal{F}(\mathbb{I}_T))\\ =&(\lambda\hat{\boldsymbol{z}} - \hat{\boldsymbol{w}}) \oslash (\gamma\hat{\boldsymbol{\ell}}^* \circ \hat{\boldsymbol{\ell}} + \lambda\mathbb{1}_T),\end{aligned} \tag{3.101}$$

which is consistent with Eq. (3.97)*. Herein, according to the property of FFT, it always holds that* $\mathcal{F}(\phi(\boldsymbol{\ell})) = \mathcal{F}(\boldsymbol{\ell})^*$ *and* $\mathcal{F}(\mathbb{I}_T) = \mathbb{1}_T$*.*

In particular, recall that the Laplacian kernel as discussed in Definition 3.3.2:

$$\boldsymbol{\ell} = (2\tau, \underbrace{-1, \cdots, -1}_{\tau}, 0, \cdots, 0, \underbrace{-1, \cdots, -1}_{\tau})^\top \in \mathbb{R}^T, \tag{3.102}$$

leads to the following circulant matrix that is symmetric:

$$\mathcal{C}(\boldsymbol{\ell}) = \begin{bmatrix} 2\tau & -1 & \cdots & -1 & 0 & \cdots & 0 & -1 & \cdots & -1 \\ -1 & 2\tau & \cdots & -1 & -1 & \cdots & 0 & 0 & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & 2\tau & -1 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & -1 & 2\tau & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 2\tau & -1 & \cdots & -1 \\ -1 & 0 & \cdots & 0 & 0 & \cdots & -1 & 2\tau & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & 0 & 0 & \cdots & -1 & -1 & \cdots & 2\tau \end{bmatrix} \in \mathbb{R}^{T\times T}. \tag{3.103}$$

Thus, we have

$$\begin{aligned} & \mathcal{C}(\boldsymbol{\ell})^\top = \mathcal{C}(\boldsymbol{\ell}) \\ \Longrightarrow \quad & \phi(\boldsymbol{\ell}) = \boldsymbol{\ell} \\ \Longrightarrow \quad & \mathcal{F}(\phi(\boldsymbol{\ell})) = \mathcal{F}(\boldsymbol{\ell}) \\ \Longrightarrow \quad & \mathcal{F}(\boldsymbol{\ell})^* = \mathcal{F}(\boldsymbol{\ell}) \end{aligned} \tag{3.104}$$

implying that the complex conjugate of $\mathcal{F}(\boldsymbol{\ell})$ is equal to $\mathcal{F}(\boldsymbol{\ell})$ itself. The closed-form solution in Eq. (3.97) *can be rewritten as follows,*

$$\hat{\boldsymbol{x}} = (\lambda \hat{\boldsymbol{z}} - \hat{\boldsymbol{w}}) \oslash (\gamma \hat{\boldsymbol{\ell}} \circ \hat{\boldsymbol{\ell}} + \lambda \mathbb{1}_T). \tag{3.105}$$

Definition 3.3.4 (ϕ-Permutation)**.** *For any vector $\boldsymbol{x} = (x_1, x_2, \cdots, x_T)^\top \in \mathbb{R}^T$, the permutation operator $\phi(\cdot)$ takes the following form:*

$$\phi(\boldsymbol{x}) = (x_1, x_T, x_{T-1}, \cdots, x_3, x_2)^\top \in \mathbb{R}^T, \tag{3.106}$$

which is also the first row of the circulant matrix $\mathcal{C}(\boldsymbol{x})$.

Remark 6. *For any vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^T$, the partial derivative of $\|\boldsymbol{x} \star \boldsymbol{y}\|_2^2$ with respect to $\boldsymbol{y}$ is given by*

$$\frac{\partial(\|\boldsymbol{x} \star \boldsymbol{y}\|_2^2)}{\partial \boldsymbol{y}} = \frac{\partial(\|\mathcal{C}(\boldsymbol{x})\boldsymbol{y}\|_2^2)}{\partial \boldsymbol{y}} = 2\mathcal{C}(\boldsymbol{x})^\top \mathcal{C}(\boldsymbol{x})\boldsymbol{y} = 2\phi(\boldsymbol{x}) \star \boldsymbol{x} \star \boldsymbol{y},$$

which shows the importance of the ϕ-permutation operator when computing the partial derivative with circular convolution.

Going back to the ADMM scheme, the subproblem for optimizing the variable $\boldsymbol{x}$ in the time domain can be converted into the optimization over the variable $\hat{\boldsymbol{x}}$ in the frequency domain, stemming from the property of circulant matrix in Lemma 3.3.1, i.e.,

$$\begin{aligned} & \boldsymbol{x} := \arg\min_{\boldsymbol{x}} \ \|\mathcal{C}(\boldsymbol{x})\|_* + \frac{\gamma}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + \langle \boldsymbol{w}, \boldsymbol{x} \rangle \\ \Longrightarrow \quad & \hat{\boldsymbol{x}} := \arg\min_{\hat{\boldsymbol{x}}} \|\hat{\boldsymbol{x}}\|_1 + \frac{\gamma}{2T}\|\hat{\boldsymbol{\ell}} \circ \hat{\boldsymbol{x}}\|_2^2 + \frac{\lambda}{2T}\|\hat{\boldsymbol{x}} - \hat{\boldsymbol{z}} + \hat{\boldsymbol{w}}/\lambda\|_2^2. \end{aligned} \tag{3.107}$$

On each $\hat{x}_t$, the optimization problem is given by

$$\begin{aligned} \hat{x}_t &:= \arg\min_{\hat{x}_t} \ |\hat{x}_t| + \frac{\gamma}{2T}|\hat{\ell}_t \hat{x}_t|^2 + \frac{\lambda}{2T}|\hat{x}_t - \hat{z}_t + \hat{w}_t/\lambda|^2 \\ &= \arg\min_{\hat{x}_t} \ |\hat{x}_t| + \frac{\gamma|\hat{\ell}_t|^2 + \lambda}{2T}\left|\hat{x}_t - \frac{\lambda \hat{z}_t - \hat{w}_t}{\gamma|\hat{\ell}_t|^2 + \lambda}\right|^2, \end{aligned} \tag{3.108}$$

where $|\hat{\ell}_t \hat{x}_t|^2 = |\hat{\ell}_t|^2 \cdot |\hat{x}_t|^2$.

The resultant ℓ_1-norm minimization is memory efficient, easy to compute, and preserves the singular values of the circulant matrix that are due to the discrete Fourier transform. FFT is an efficient algorithm for computing the discrete Fourier transform in $\mathcal{O}(T \log T)$ time.

Lemma 3.3.1. *For any vector $\boldsymbol{x} \in \mathbb{R}^T$, the nuclear norm of the resultant circulant matrix $\mathcal{C}(\boldsymbol{x}) \in \mathbb{R}^{T \times T}$ is related to the discrete Fourier transform:*

$$\|\mathcal{C}(\boldsymbol{x})\|_* = \|\mathcal{F}(\boldsymbol{x})\|_1. \tag{3.109}$$

Proof. For any circulant matrix $\mathcal{C}(\boldsymbol{x})$ if and only if it is diagonalizable by the unitary matrix, the eigenvalue decomposition [80] can be written as follows,

$$\mathcal{C}(\boldsymbol{x}) = \boldsymbol{U} \mathrm{diag}(\mathcal{F}(\boldsymbol{x})) \boldsymbol{U}^H, \tag{3.110}$$

where $\cdot^H$ denotes the conjugate transpose. Since $\boldsymbol{U}$ is a unitary matrix, it always holds that

$$\begin{aligned} \|\mathcal{C}(\boldsymbol{x})\|_* =& \|\boldsymbol{U} \mathrm{diag}(\mathcal{F}(\boldsymbol{x})) \boldsymbol{U}^H\|_* \\ =& \|\mathrm{diag}(\mathcal{F}(\boldsymbol{x}))\|_* \\ =& \|\mathcal{F}(\boldsymbol{x})\|_1, \end{aligned} \tag{3.111}$$

and we can quickly calculate the singular values of $\mathcal{C}(\boldsymbol{x})$ from the FFT of $\boldsymbol{x}$, i.e., $\mathcal{F}(\boldsymbol{x})$ in $\mathcal{O}(T \log T)$ time. □

In Eq. (3.108), let

$$\hat{\boldsymbol{h}} \triangleq (\lambda \hat{\boldsymbol{z}} - \hat{\boldsymbol{w}}) \oslash (\gamma \hat{\boldsymbol{\ell}} \circ \hat{\boldsymbol{\ell}} + \lambda \mathbb{1}_T), \tag{3.112}$$

The closed-form solution to $\hat{\boldsymbol{x}}$ can be found in Lemma 3.3.2. As we have the closed-form solution as described in Eq. (3.117) (i.e., with respect to each $\hat{x}_t$) such that

$$\hat{x}_t := \frac{\hat{h}_t}{|\hat{h}_t|} \cdot \max\{0, |\hat{h}_t| - 1/\delta_t\}, \tag{3.113}$$

with

$$\begin{cases} \delta_t \triangleq (\gamma |\hat{\ell}_t|^2 + \lambda)/T, \\ \hat{h}_t \triangleq (\lambda \hat{z}_t - \hat{w}_t)/(\gamma |\hat{\ell}_t|^2 + \lambda), \end{cases} \tag{3.114}$$

we can therefore update the variable $\boldsymbol{x}$ by

$$\boldsymbol{x} := \mathcal{F}^{-1}(\hat{\boldsymbol{x}}). \tag{3.115}$$

Lemma 3.3.2. *Following Eq.* (3.108)*, for any optimization problem in the form of ℓ_1-norm minimization in complex space, i.e.,*

$$\min_{\hat{\boldsymbol{x}}} \ \|\hat{\boldsymbol{x}}\|_1 + \frac{\delta}{2}\|\hat{\boldsymbol{x}} - \hat{\boldsymbol{h}}\|_2^2, \tag{3.116}$$

with complex-valued $\hat{\boldsymbol{x}}, \hat{\boldsymbol{h}} \in \mathbb{C}^T$ and weight parameter $\delta \in \mathbb{R}$, element-wise, the solution is given by

$$\hat{x}_t := \frac{\hat{h}_t}{|\hat{h}_t|} \cdot \max\{0, |\hat{h}_t| - 1/\delta\}, t = 1, \ldots, T. \tag{3.117}$$

Remark 7. *Lemma 3.3.2 invokes the shrinkage operator in [46, 85, 86].*

Estimating the Variable $\boldsymbol{z}$

In the ADMM scheme (see Eq. (3.93)), the subproblem with respect to the variable $\boldsymbol{z}$ can be written as follows,

$$\min_{\boldsymbol{z}} \ \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{z} - \boldsymbol{w}/\lambda\|_2^2 + \frac{\eta}{2}\|\mathcal{P}_\Omega(\boldsymbol{z} - \boldsymbol{y})\|_2^2, \tag{3.118}$$

if we let g be the objective function, then the partial derivative with respect to the variable $\boldsymbol{z}$ can be formed by $\mathcal{P}_\Omega(\boldsymbol{z})$ and $\mathcal{P}_\Omega^\perp(\boldsymbol{z})$ simultaneously. Here, we have

$$\begin{aligned} \frac{\partial g}{\partial \mathcal{P}_\Omega(\boldsymbol{z})} =& \lambda\mathcal{P}_\Omega(\boldsymbol{z} - \boldsymbol{x} - \boldsymbol{w}/\lambda) + \eta\mathcal{P}_\Omega(\boldsymbol{z} - \boldsymbol{y}), \\ \frac{\partial g}{\partial \mathcal{P}_\Omega^\perp(\boldsymbol{z})} =& \lambda\mathcal{P}_\Omega^\perp(\boldsymbol{z} - \boldsymbol{x} - \boldsymbol{w}/\lambda). \end{aligned} \tag{3.119}$$

With these partial derivatives,

$$\frac{\partial g}{\partial \boldsymbol{z}} = \frac{\partial g}{\partial \mathcal{P}_\Omega(\boldsymbol{z})} + \frac{\partial g}{\partial \mathcal{P}_\Omega^\perp(\boldsymbol{z})} = \boldsymbol{0} \tag{3.120}$$

allows one to find a closed-form solution to the variable $\boldsymbol{z}$:

$$\boldsymbol{z} := \frac{1}{\lambda + \eta}\mathcal{P}_\Omega(\lambda\boldsymbol{x} + \boldsymbol{w} + \eta\boldsymbol{y}) + \mathcal{P}_\Omega^\perp(\boldsymbol{x} + \boldsymbol{w}/\lambda). \tag{3.121}$$

Remark 8. *If $\eta \to +\infty$, then the solution would be $\boldsymbol{z} := \mathcal{P}_\Omega(\boldsymbol{y}) + \mathcal{P}_\Omega^\perp(\boldsymbol{x} + \boldsymbol{w}/\lambda)$, corresponding to the LCR model with strong observation constraint in Eq.* (3.88)*. In terms of the parameter η, we can preferably set the default one as $\eta = c \cdot \lambda$ with $c \in \{10^2, 10^3\}$ to preserve the observation information.*

Solution Algorithm

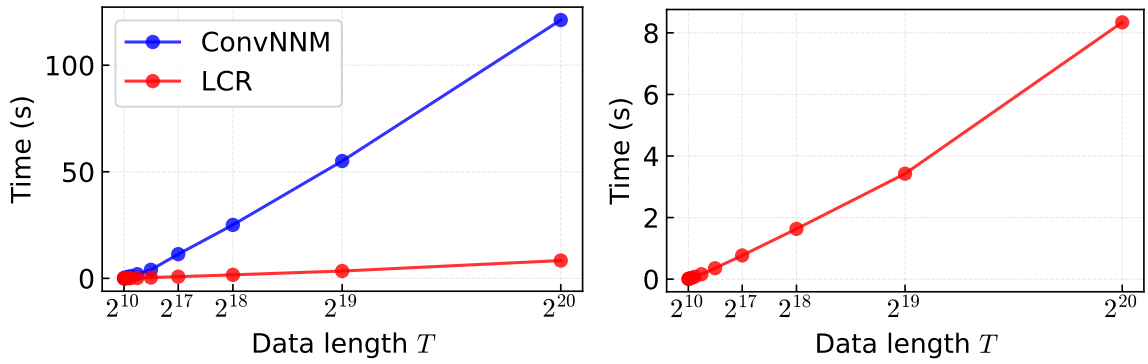
As mentioned above, our LCR model bridges the gap between the modeling processes of global low-rank property and local temporal trends underlying time series data, therefore reinforcing both global and local consistency. Since we utilize the circulant matrix and circular Laplacian kernel in our model, it is not hard to show the appealing properties of discrete Fourier transform and lead to an elegant and fast solution algorithm. Algorithm 6 summarizes the implementation of the proposed LCR model.

Algorithm 6 Laplacian Convolutional Representation

Input: Data $\boldsymbol{y} \in \mathbb{R}^T$ with the observed index set Ω, Laplacian kernel size $\tau \in \mathbb{N}^+$, and hyperparameters $\{\gamma, \lambda, \eta\}$.
Output: Reconstructed vector $\boldsymbol{x}$.

Initialize the variables $\{\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{w}\}$.
Construct the Laplacian kernel $\boldsymbol{\ell}$ with τ and perform FFT on it to get $\hat{\boldsymbol{\ell}}$.
repeat
  Perform FFT on $\{\boldsymbol{z}, \boldsymbol{w}\}$.
  Compute $\hat{\boldsymbol{h}}$ by Eq. (3.112).
  Compute $\hat{\boldsymbol{x}}$ by the shrinkage in Eq. (3.117).
  Compute $\boldsymbol{x}$ by $\boldsymbol{x} = \mathcal{F}^{-1}(\hat{\boldsymbol{x}})$ (see Eq. (3.115)).
  Compute $\boldsymbol{z}$ by Eq. (3.121).
  Compute $\boldsymbol{w} = \boldsymbol{w} + \lambda(\boldsymbol{x} - \boldsymbol{z})$ (see Eq. (3.93)).
until convergence
return $\boldsymbol{x}$.

To analyze the empirical time complexity of LCR (with default 50 iterations), Fig. 3.7 shows the running times of LCR on the artificially generated data with different data lengths (i.e., data $\boldsymbol{y} \in \mathbb{R}^T$ with $T \in \{2^{10}, 2^{11}, \ldots, 2^{20}\}$). As shown in Fig. 3.7(a), we compare LCR with ConvNNM (e.g., kernel size $\tilde{\tau} = 2^4$ in this case), demonstrating that LCR is more efficient than ConvNNM. Typically, ConvNNM can be converted into a standard nuclear norm minimization with singular value thresholding (i.e., of time complexity $\mathcal{O}(\tilde{\tau}^2 T)$) [25, 26, 46, 47, 87]. The computational cost of ConvNNM would be increased if a relative large $\tilde{\tau}$ is given on the convolution matrix $\mathcal{C}_{\tilde{\tau}}(\boldsymbol{y}) \in \mathbb{R}^{T \times \tilde{\tau}}$. In contrast to ConvNNM, both CircNNM and LCR have an efficient solution through FFT (i.e., of time complexity $\mathcal{O}(T \log T)$) [47].



(a) ConvNNM vs. LCR (b) LCR

Figure 3.7 Empirical time complexity. The model is tested 10 times on each generated data.

3.3.3 Multivariate Time Series Imputation Model

On the multivariate time series $\boldsymbol{Y}$, the first impulse is to follow the LCR model in the univariate case and formulate the multivariate model as follows,

$$\begin{aligned} &\min_{\boldsymbol{X}} \|\mathcal{C}(\boldsymbol{X})\|_* + \frac{\gamma}{2}\|\boldsymbol{L}\boldsymbol{X}^\top\|_F^2 \\ &\text{s.t. } \|\mathcal{P}_\Omega(\boldsymbol{X}-\boldsymbol{Y})\|_F \leq \epsilon, \end{aligned} \tag{3.122}$$

where we introduce the nuclear norm of the circulant tensor $\mathcal{C}(\boldsymbol{X})$ in Definition 3.3.5, which follows the tensor nuclear norm proposed in [88, 89]. The Laplacian matrix $\boldsymbol{L} \in \mathbb{R}^{T\times T}$ is basically given by $\mathcal{C}(\boldsymbol{\ell})$ with the prescribed Laplacian kernel $\boldsymbol{\ell} \in \mathbb{R}^T$ in Definition 3.3.2.

Definition 3.3.5 (Circulant Tensor Nuclear Norm)**.** *For any matrix $\boldsymbol{X} \in \mathbb{R}^{N\times T}$, the corresponding circulant tensor is $\mathcal{C}(\boldsymbol{X}) \in \mathbb{R}^{N\times N\times T\times T}$, which can be factorized in the Tucker format (i.e., higher-order singular value decomposition) [7]:*

$$\mathcal{C}(\boldsymbol{X}) = \boldsymbol{\mathcal{S}} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_1 \times_3 \boldsymbol{U}_2 \times_4 \boldsymbol{U}_2, \tag{3.123}$$

where $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{N\times N\times T\times T}$ is the core tensor (consisting of singular values [88, 89]), while $\boldsymbol{U}_1 \in \mathbb{R}^{N\times N}$ and $\boldsymbol{U}_2 \in \mathbb{R}^{T\times T}$ are unitary matrices. The notation $\times_k$, $\forall k \in \{1,2,3,4\}$ denotes the mode-k product between tensor and matrix [7]. The circulant tensor nuclear norm is defined as

$$\|\mathcal{C}(\boldsymbol{X})\|_* = \sum_{n=1}^{N}\sum_{t=1}^{T} s_{n,n,t,t}, \tag{3.124}$$

where $s_{n,n,t,t}$ is the (n,n,t,t)-th entry of the core tensor $\boldsymbol{\mathcal{S}}$.

As mentioned in Eq. (3.122), the Laplacian matrix in the temporal dimension works on each time series independently. If we aim to build the connection between the circulant matrix and FFT, then the problem would be reduced to

$$\begin{aligned} \min_{\boldsymbol{X}} \ & \sum_{n=1}^{N} \|\mathcal{C}(\boldsymbol{x}_n)\|_* + \frac{\gamma}{2} \sum_{n=1}^{N} \|\boldsymbol{\ell} \star \boldsymbol{x}_n\|_2^2 \\ \text{s.t.} \ & \|\mathcal{P}_{\Omega}(\boldsymbol{X} - \boldsymbol{Y})\|_F \leq \epsilon, \end{aligned} \tag{3.125}$$

where the vectors $\boldsymbol{x}_n \in \mathbb{R}^T$, $n = 1, 2, \ldots, N$ are the univariate time series or the rows of the data $\boldsymbol{X} \in \mathbb{R}^{N \times T}$, and $\boldsymbol{\ell} \in \mathbb{R}^T$ is the Laplacian kernel (see Definition 3.3.2). In this case, the whole problem is divided into N subproblems (defined as LCR_N in the following). The reconstruction of multivariate time series is achieved by reconstructing each time series independently.

To jointly characterize the spatial and temporal dependencies for traffic time series, we consider a separable kernel in the LCR model, namely,

$$\boldsymbol{K} \triangleq \boldsymbol{\ell}_s \boldsymbol{\ell}^\top \in \mathbb{R}^{N \times T}, \tag{3.126}$$

with the spatial kernel $\boldsymbol{\ell}_s = (1, 0, \cdots, 0)^\top \in \mathbb{R}^N$ (i.e., the first column of N-by-N identity matrix). In the case of spatial modeling, $\boldsymbol{\ell}_s$ can also be introduced as the Laplacian kernel in Definition 3.3.2. The optimization problem of two-dimensional LCR (LCR-2D) can be formulated as follows,

$$\begin{aligned} \min_{\boldsymbol{X}} \ & \|\mathcal{C}(\boldsymbol{X})\|_* + \frac{\gamma}{2} \|\boldsymbol{K} \star \boldsymbol{X}\|_F^2 \\ \text{s.t.} \ & \|\mathcal{P}_{\Omega}(\boldsymbol{X} - \boldsymbol{Y})\|_F \leq \epsilon, \end{aligned} \tag{3.127}$$

where the two-dimensional (2D) circular convolution is described in Definition 3.3.6. Although the spatial kernel $\boldsymbol{\ell}_s = (1, 0, \cdots, 0)^\top \in \mathbb{R}^N$ does not provide any spatial dependencies, the nuclear norm of circulant operator on $\boldsymbol{X}$ can achieve implicit spatial modeling. In the meanwhile, the spatiotemporal structure of data can be preserved. If applicable, one can characterize spatial correlations by using the Laplacian kernel.

Definition 3.3.6 (2D Circular Convolution [48, 90])**.** *For any matrices $\boldsymbol{X} \in \mathbb{R}^{N \times T}$ and $\boldsymbol{K} \in \mathbb{R}^{\nu_1 \times \nu_2}$ with $\nu_1 \leq N, \nu_2 \leq T$, the circular convolution of two matrices is defined as*

$$\boldsymbol{Z} = \boldsymbol{K} \star \boldsymbol{X} \in \mathbb{R}^{N \times T}, \tag{3.128}$$

or element-wise,

$$z_{n,t} = \sum_{i=1}^{\nu_1} \sum_{j=1}^{\nu_2} \kappa_{i,j} x_{n-i+1,t-j+1}, \tag{3.129}$$

where $n = 1, 2, \ldots, N$ *and* $t = 1, 2, \ldots, T$. $\kappa_{i,j}$ *is the* (i, j)*-th entry of* $\boldsymbol{K}$.

Remark 9. *In the field of signal processing, the 2D circular convolution also possesses the properties that are associated with the two-dimensional discrete Fourier transform. According to the Convolution Theorem and the Parseval's Theorem, we have*

$$\|\boldsymbol{K} \star \boldsymbol{X}\|_F^2 = \frac{1}{NT} \|\mathcal{F}(\boldsymbol{K}) \circ \mathcal{F}(\boldsymbol{X})\|_F^2, \tag{3.130}$$

for any matrices $\boldsymbol{K}, \boldsymbol{X} \in \mathbb{R}^{N \times T}$ *with special setting for* $\boldsymbol{K}$ *such that*

$$\boldsymbol{K} := \begin{bmatrix} \kappa_{1,1} & \cdots & \kappa_{1,\nu_2} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \kappa_{\nu_1,1} & \cdots & \kappa_{\nu_1,\nu_2} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{N \times T}. \tag{3.131}$$

As mentioned above, $\mathcal{F}(\cdot)$ *denotes the 2D discrete Fourier transform. Typically, 2D discrete Fourier transform can be computed by first transforming into each column vector (or row vector) and then each row vector (or column vector) of the matrix [48].*

As mentioned above, LCR in the multivariate case is also a convex problem that can be resolved by the ADMM framework. Following Eq. (3.93), the ADMM scheme is given by

$$\begin{cases} \boldsymbol{X} := \arg\min_{\boldsymbol{X}} \mathcal{L}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}), \\ \boldsymbol{Z} := \arg\min_{\boldsymbol{Z}} \mathcal{L}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}) \\ \quad\; = \dfrac{1}{\lambda + \eta} \mathcal{P}_\Omega(\lambda \boldsymbol{X} + \boldsymbol{W} + \eta \boldsymbol{Y}) + \dfrac{1}{\lambda} \mathcal{P}_\Omega^\perp(\lambda \boldsymbol{X} + \boldsymbol{W}), \\ \boldsymbol{W} := \boldsymbol{W} + \lambda(\boldsymbol{X} - \boldsymbol{Z}), \end{cases} \tag{3.132}$$

where $\mathcal{L}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W})$ is the augmented Lagrangian function:

$$\begin{aligned}\mathcal{L}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}) =& \|\mathcal{C}(\boldsymbol{X})\|_* + \frac{\gamma}{2}\|\boldsymbol{K} \star \boldsymbol{X}\|_F^2 \\ &+ \frac{\lambda}{2}\|\boldsymbol{X} - \boldsymbol{Z}\|_F^2 + \langle \boldsymbol{W}, \boldsymbol{X} - \boldsymbol{Z}\rangle \\ &+ \frac{\eta}{2}\|\mathcal{P}_\Omega(\boldsymbol{Z} - \boldsymbol{Y})\|_F^2.\end{aligned} \tag{3.133}$$

With respect to the variable $\boldsymbol{X}$, the subproblem is

$$\begin{aligned}\boldsymbol{X} := \arg\min_{\boldsymbol{X}} \ & \|\mathcal{C}(\boldsymbol{X})\|_* + \frac{\gamma}{2}\|\boldsymbol{K} \star \boldsymbol{X}\|_F^2 \\ &+ \frac{\lambda}{2}\|\boldsymbol{X} - \boldsymbol{Z} + \boldsymbol{W}/\lambda\|_2^2.\end{aligned} \tag{3.134}$$

Although the nuclear norm of circulant tensor in Definition 3.3.5 is more complicated than the nuclear norm of circulant matrix, 2D discrete Fourier transform is also applicable for finding the equivalent formula. In the frequency domain, it takes

$$\begin{aligned}\hat{\boldsymbol{X}} := \arg\min_{\hat{\boldsymbol{X}}} \ & \|\hat{\boldsymbol{X}}\|_1 + \frac{\gamma}{2NT}\|\hat{\boldsymbol{K}} \circ \hat{\boldsymbol{X}}\|_F^2 \\ &+ \frac{\lambda}{2NT}\|\hat{\boldsymbol{X}} - \hat{\boldsymbol{Z}} + \hat{\boldsymbol{W}}/\lambda\|_F^2,\end{aligned} \tag{3.135}$$

where we introduce the 2D discrete Fourier transformed $\{\hat{\boldsymbol{K}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{W}}\}$ referring to $\{\boldsymbol{K}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}\}$ in the frequency domain. Without loss of generality, this subproblem for ℓ_1-norm minimization in complex space can also be solved by the shrinkage operator in Eq. (3.117). Algorithm 7 summarizes the implementation of LCR-2D.

3.4 Memory-Efficient Hankel Tensor Factorization

In this section, we advance a memory-efficient HTF model specially designed to handle complicated spatiotemporal data modeling tasks, including the challenge of extreme missing traffic data imputation. Since the Hankelization process on spatiotemporal data allows one to build spatial and temporal correlations automatically, the resulting HTF is able to capture global low-rank patterns and spatiotemporal local trends simultaneously. To find a memory-efficient implementation of HTF, we introduce a novel Hankel indexing mechanism that eliminates the need for explicit construction of a Hankel tensor, yet preserving the properties of the Hankel structure. The proposed HTF employs a convolutional matrix factorization formula for each slice of the Hankel tensor, in which the formula stems from

Algorithm 7 Two-Dimensional Laplacian Convolutional Representation

Input: Data $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$ with the observed index set Ω, Laplacian kernel size $\tau \in \mathbb{N}^+$, and hyperparameters $\{\gamma, \lambda, \eta\}$.
Output: Reconstructed matrix $\boldsymbol{X} \in \mathbb{R}^{N\times T}$.
Initialize the variables $\{\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}\}$.
Construct the Laplacian kernel $\boldsymbol{\ell} \in \mathbb{R}^T$ with τ.
Construct the spatial kernel $\boldsymbol{\ell}_s = (1, 0, \cdots, 0) \in \mathbb{R}^N$ (or $\boldsymbol{\ell}_s \in \mathbb{R}^N$ with $\tau_s \in \mathbb{N}^+$) and build up a separable kernel $\boldsymbol{K} \triangleq \boldsymbol{\ell}_s \boldsymbol{\ell}^\top$.
repeat
  Perform FFT on $\{\boldsymbol{Z}, \boldsymbol{W}\}$.
  Compute $\hat{\boldsymbol{X}}$ by referring to the shrinkage in Eq. (3.117).
  Compute $\boldsymbol{X}$ by $\boldsymbol{X} = \mathcal{F}^{-1}(\hat{\boldsymbol{X}})$.
  Compute $\boldsymbol{Z}$ by Eq. (3.132).
  Compute $\boldsymbol{W}$ by Eq. (3.132).
until convergence
return $\boldsymbol{X}$.

the tensor-train factorization. To address the optimization problem of HTF, we present an alternating minimization scheme. Within this scheme, each subproblem can be converted into complicated but solvable linear equations, including generalized Sylvester equations with multiple terms and generalized systems of linear equations. The approximated solution to each subproblem is identified through the conjugate gradient method efficiently.

3.4.1 Hankel Structure

In this section, we begin with some preliminary definitions and descriptions for Hankelization on both vector and matrix data. Then, according to the property of Hankel tensors, we introduce a compact representation constructed from matrix data via Hankel indexing; such kind of representation through Hankel indexing is the cornerstone in developing memory-efficient HTF algorithms.

Hankel Matrix

The process of constructing the Hankel structure usually refers to as the notion of Hankelization [49]. Definition 3.4.1 gives a concise description of a Hankel matrix constructed from a given vector with a certain window length, in which each column of the Hankel matrix $\mathcal{H}_\tau(\boldsymbol{x})$ includes $T - \tau + 1$ entries of the original vector $\boldsymbol{x}$.

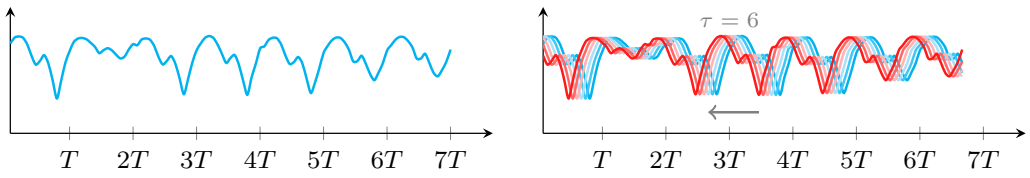
Definition 3.4.1 (Hankel Matrix)**.** *For any vector $\boldsymbol{x} = (x_1, x_2, \cdots, x_T)^\top \in \mathbb{R}^T$, let $\tau \in \mathbb{N}^+$ ($1 < \tau \leq \lfloor T/2 \rfloor$) be the window length of the Hankel structure, then the corresponding Hankel*

matrix can be written as follows,

$$\mathcal{H}_\tau(\boldsymbol{x}) = \begin{bmatrix} x_1 & x_2 & \cdots & x_\tau \\ x_2 & x_3 & \cdots & x_{\tau+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{T-\tau+1} & x_{T-\tau+2} & \cdots & x_T \end{bmatrix} \in \mathbb{R}^{(T-\tau+1)\times\tau},$$

where $\mathcal{H}_\tau : \mathbb{R}^T \to \mathbb{R}^{(T-\tau+1)\times\tau}$ *denotes the Hankel operator with the window length* τ*. The notation* $\lfloor\cdot\rfloor$ *denotes the floor function that maps the given real number to the greatest integer less than or equal to it.*

In the literature, Hankelization always shows important applications to time series modeling, e.g., singular spectrum analysis algorithm for time series decomposition [91]. Notably, one reason is that Hankelization can capture temporal correlations of time series data. Figure 3.8 shows the process of constructing the Hankel matrix on a given time series.



(a) Traffic time series. (b) Hankel matrix.

Figure 3.8 Illustration of constructing Hankel matrix on a given time series with a certain window length (e.g., $\tau = 6$). The Hankelization is preceded with certain steps backward.

Consider the following example: a time series $\boldsymbol{y} = (y_1, y_2, \cdots, y_T)^\top \in \mathbb{R}^T$ and a window length of $\tau = 3$. If the Hankel matrix $\mathcal{H}_\tau(\boldsymbol{y})$ can be approximated by a rank-one matrix (i.e., the outer product of two coordinate vectors), which is also relevant to the original vector $\boldsymbol{y}$, then there always exists

$$\mathcal{H}_\tau(\boldsymbol{y}) = \begin{bmatrix} y_1 & y_2 & y_3 \\ y_2 & y_3 & y_4 \\ \vdots & \vdots & \vdots \\ y_{T-2} & y_{T-1} & y_T \end{bmatrix} \approx \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{T-2} \end{bmatrix} \otimes_{\text{outer}} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \tag{3.136}$$

where the symbol $\otimes_{\text{outer}}$ denotes the outer product of vectors. As a consequence, the estimate

$\hat{\boldsymbol{y}}$ corresponding to the original vector $\boldsymbol{y}$ can be written as follows,

$$\hat{\boldsymbol{y}} = \mathcal{H}_{\tau}^{-1}\left(\begin{bmatrix} v_1x_1 & v_1x_2 & v_1x_3 \\ v_2x_1 & v_2x_2 & v_2x_3 \\ \vdots & \vdots & \vdots \\ v_{T-2}x_1 & v_{T-2}x_2 & v_{T-2}x_3 \end{bmatrix}\right) = \begin{bmatrix} v_1x_1 \\ (v_1x_2 + v_2x_1)/2 \\ (v_1x_3 + v_2x_2 + v_3x_1)/3 \\ \vdots \\ (v_{T-4}x_3 + v_{T-3}x_2 + v_{T-2}x_1)/3 \\ (v_{T-3}x_3 + v_{T-2}x_2)/2 \\ v_{T-2}x_3 \end{bmatrix}. \tag{3.137}$$

where $\mathcal{H}_{\tau}^{-1}(\cdot)$ denotes the inverse Hankelization operator, i.e., averaging over the anti-diagonal entries. It is evident that the process of automatically modeling temporal patterns occurs naturally during Hankel matrix factorization. For instance, the third entry of $\hat{\boldsymbol{y}}$, i.e., $\hat{y}_3 = (v_1x_3 + v_2x_2 + v_3x_1)/3$, is the $\boldsymbol{x}$-weighted estimate over the first three entries of $\boldsymbol{v}$.

Hankel Tensor

Hankel tensor stems from the concept of the Hankel matrix, which usually works on the input matrix or tensor [55]. Following Definition 3.4.1, Hankelization on any given matrix would produce a fourth-order tensor, carrying out the Hankelization processes over the N rows and the T columns independently (see Definition 3.4.2). The Hankel tensor $\boldsymbol{\mathcal{X}}$ has $\tau_1 \times \tau_2$ slices and each slice is a block of $\boldsymbol{X}$, only containing $(N - \tau_1 + 1) \times (T - \tau_2 + 1)$ entries of $\boldsymbol{X}$.

Definition 3.4.2 (Hankel Tensor)**.** *For any matrix $\boldsymbol{X} \in \mathbb{R}^{N\times T}$, let $\tau_1, \tau_2 \in \mathbb{N}^+$ ($1 < \tau_1 \leq \lfloor N/2 \rfloor$ and $1 < \tau_2 \leq \lfloor T/2 \rfloor$) be the window lengths of the Hankel structure, then the corresponding Hankel tensor is a fourth-order tensor whose size is $(N-\tau_1+1)\times\tau_1\times(T-\tau_2+1)\times\tau_2$. The Hankel tensor has $\tau_1 \times \tau_2$ slices in which each slice is an $(N - \tau_1 + 1)$-by-$(T - \tau_2 + 1)$ matrix, i.e.,*

$$\boldsymbol{\mathcal{X}}_{:,1,:,1} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1,T-\tau_2+1} \\ x_{21} & x_{22} & \cdots & x_{2,T-\tau_2+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N-\tau_1+1,1} & x_{N-\tau_1+1,2} & \cdots & x_{N-\tau_1+1,T-\tau_2+1} \end{bmatrix},$$

$$\vdots$$

$$\boldsymbol{\mathcal{X}}_{:,\tau_1,:,\tau_2} = \begin{bmatrix} x_{\tau_1,\tau_2} & x_{\tau_1,\tau_2+1} & \cdots & x_{\tau_1,T} \\ x_{\tau_1+1,\tau_2} & x_{\tau_1+1,\tau_2+1} & \cdots & x_{\tau_1+1,T} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,\tau_2} & x_{N,\tau_2+1} & \cdots & x_{N,T} \end{bmatrix},$$

with the specified indices over the second and fourth dimensions (see Figure 3.9). For the simplicity of notation, we let $\boldsymbol{\mathcal{X}} \triangleq \mathcal{H}_{\tau_1,\tau_2}(\boldsymbol{X})$ *be the Hankel tensor of* $\boldsymbol{X}$*. Note that the notation of tensor slices, i.e.,* $\boldsymbol{\mathcal{X}}_{:,k_1,:,k_2}$, $\forall k_1 \in \{1,2,\ldots,\tau_1\}, k_2 \in \{1,2,\ldots,\tau_2\}$*, follows [7].*

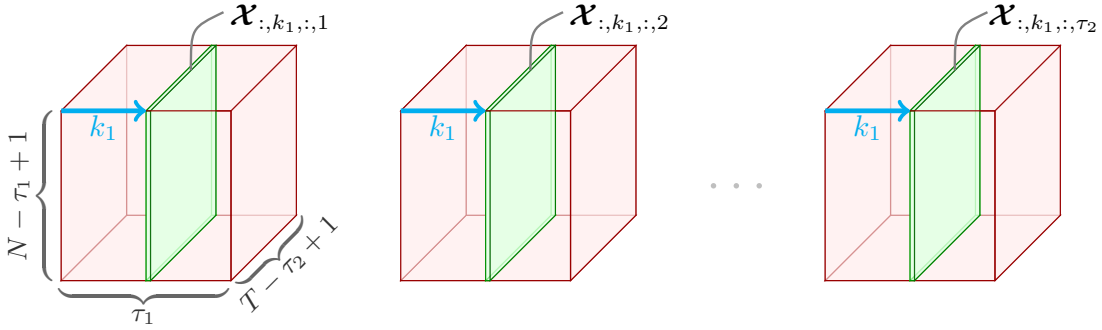


Figure 3.9 Hankel tensor $\boldsymbol{\mathcal{X}} = \mathcal{H}_{\tau_1,\tau_2}(\boldsymbol{X})$ built on the matrix $\boldsymbol{X} \in \mathbb{R}^{N\times T}$ with the window lengths $\tau_1, \tau_2 \in \mathbb{N}^+$. The fourth-order Hankel tensor consists of τ_2 third-order tensors, i.e., $\boldsymbol{\mathcal{X}}_{:,:,:,k_2}$, $k_2 = 1,2,\ldots,\tau_2$. Here we illustrate three third-order tensors $\boldsymbol{\mathcal{X}}_{:,:,:,1}$, $\boldsymbol{\mathcal{X}}_{:,:,:,2}$, and $\boldsymbol{\mathcal{X}}_{:,:,:,\tau_2}$ in red cubes. For each third-order tensor, we highlight the k_1-th slices (i.e., $(N-\tau_1+1)$-by-$(T-\tau_2+1)$ matrices) in green.

Typically, when performing Hankelization on a large matrix with long window lengths, the resulting Hankel tensor tends to be relatively large. This poses a practical challenge due to the high memory consumption it entails. Specifically, in Definition 3.4.2, the Hankel tensor is

$$\frac{1}{4}\tau_1\tau_2 < \frac{\tau_1\tau_2(N-\tau_1+1)(T-\tau_2+1)}{NT} < \tau_1\tau_2 \tag{3.138}$$

times of the size of the original matrix because

$$\begin{cases} \tau_1 \leq \lfloor \frac{N}{2} \rfloor \leq \frac{N}{2} < \frac{N}{2}+1 & \Rightarrow \quad N-\tau_1+1 > \frac{N}{2}, \\ \tau_2 \leq \lfloor \frac{T}{2} \rfloor \leq \frac{T}{2} < \frac{T}{2}+1 & \Rightarrow \quad T-\tau_1+1 > \frac{T}{2}, \end{cases}$$

and

$$\begin{cases} N-\tau_1+1 < N, \\ T-\tau_2+1 < T. \end{cases}$$

3.4.2 Hankel Indexing

To address the high memory consumption issue and preserve the property of Hankelization, we introduce a sequence of operators as the memory-efficient Hankel indexing. Specifically, according to the definition of a Hankel tensor, slices of $\boldsymbol{\mathcal{X}}$ are blocks of $\boldsymbol{X}$ with certain shifting

rules. Thus, a slice of $\boldsymbol{\mathcal{X}}$ can be represented by

$$\boldsymbol{\mathcal{X}}_{:,k_1,:,k_2} = \theta_{k_1,k_2}(\boldsymbol{X}), \tag{3.139}$$

for any $k_1 \in \{1, 2, \ldots, \tau_1\}$ and $k_2 \in \{1, 2, \ldots, \tau_2\}$. Herein, $\theta_{k_1,k_2} : \mathbb{R}^{N\times T} \to \mathbb{R}^{(N-\tau_1+1)\times(T-\tau_2+1)}$ denotes the operator of Hankel indexing (see Figure 3.10 for an intuitive illustration). In other words, Hankel indexing is a compact and memory-efficient way to represent slices of the Hankel tensor from the original matrix without explicitly constructing the Hankel tensor.

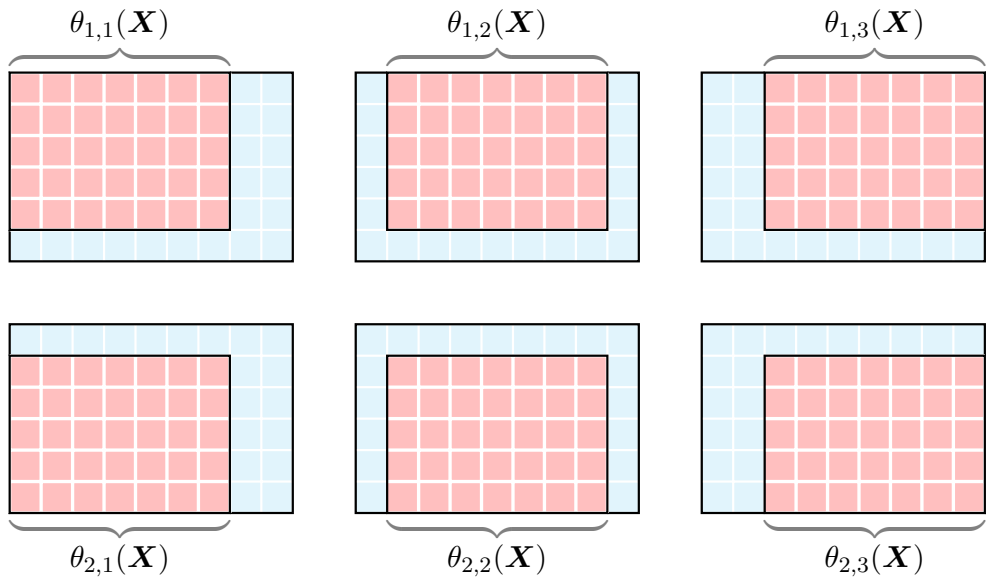


Figure 3.10 Illustration of Hankel indexing on the example matrix $\boldsymbol{X} \in \mathbb{R}^{6\times 9}$ (blue rectangle) with window lengths $\tau_1 = 2$ and $\tau_2 = 3$. Each slice (red rectangle) is represented by using the operator of Hankel indexing. The resulting Hankel tensor is of size $5 \times 2 \times 7 \times 3$, showing 2×3 slices along the second and fourth dimensions, i.e., a sequence of 5-by-7 matrices. Notably, it demonstrates that all entries of these slices are from the matrix $\boldsymbol{X}$.

When performing tensor factorization on the Hankel tensor $\boldsymbol{\mathcal{X}}$, suppose $\hat{\boldsymbol{X}}_{(k_1,k_2)}, \forall k_1, k_2$ be the low-rank approximation of $\theta_{k_1,k_2}(\boldsymbol{X})$, then the matrix $\boldsymbol{X}$ can be reconstructed by

$$\hat{\boldsymbol{X}} = \Big(\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \theta_{k_1,k_2}^{-1}(\hat{\boldsymbol{X}}_{(k_1,k_2)})\Big) \oslash \Big(\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \boldsymbol{\Gamma}_{(k_1,k_2)}\Big), \tag{3.140}$$

where the symbol $\oslash$ denotes the Hadamard division. Correspondingly, $\theta_{k_1,k_2}^{-1} : \mathbb{R}^{(N-\tau_1+1)\times(T-\tau_2+1)} \to \mathbb{R}^{N\times T}$ is introduced as the inverse operator of Hankel indexing with the specific indices k_1

and k_2. Formally, the inverse operator of Hankel indexing takes

$$\theta_{k_1,k_2}^{-1}(\hat{\boldsymbol{X}}_{(k_1,k_2)}) = \begin{bmatrix} \boldsymbol{\omega}_1 & \boldsymbol{\omega}_4 & \boldsymbol{\omega}_6 \\ \boldsymbol{\omega}_2 & \hat{\boldsymbol{X}}_{(k_1,k_2)} & \boldsymbol{\omega}_7 \\ \boldsymbol{\omega}_3 & \boldsymbol{\omega}_5 & \boldsymbol{\omega}_8 \end{bmatrix} \in \mathbb{R}^{N\times T}, \tag{3.141}$$

with the following blocks (filling in the entries with zeros):

$$\begin{aligned} &\boldsymbol{\omega}_1 = \mathbf{0}_{(k_1-1)\times(k_2-1)}, && \boldsymbol{\omega}_2 = \mathbf{0}_{(N-\tau_1+1)\times(k_2-1)}, \\ &\boldsymbol{\omega}_3 = \mathbf{0}_{(\tau_1-k_1)\times(k_2-1)}, && \boldsymbol{\omega}_4 = \mathbf{0}_{(k_1-1)\times(T-\tau_2+1)}, \\ &\boldsymbol{\omega}_5 = \mathbf{0}_{(\tau_1-k_1)\times(T-\tau_2+1)}, && \boldsymbol{\omega}_6 = \mathbf{0}_{(k_1-1)\times(\tau_2-k_2)}, \\ &\boldsymbol{\omega}_7 = \mathbf{0}_{(T-\tau_1+1)\times(\tau_2-k_2)}, && \boldsymbol{\omega}_8 = \mathbf{0}_{(\tau_1-k_1)\times(\tau_2-k_2)}, \end{aligned} \tag{3.142}$$

where the remaining entries of $\theta_{k_1,k_2}^{-1}(\hat{\boldsymbol{X}}_{(k_1,k_2)})$ except the block $\hat{\boldsymbol{X}}_{(k_1,k_2)}$ are all zeros. The auxiliary matrix $\boldsymbol{\Gamma}_{(k_1,k_2)} \in \{0,1\}^{N\times T}, \forall k_1, k_2$ is introduced as a binary matrix such that

$$[\boldsymbol{\Gamma}_{(k_1,k_2)}]_{n,t} = \begin{cases} 1, & \text{if } (n,t) \in \Theta, \\ 0, & \text{otherwise}, \end{cases} \tag{3.143}$$

for any (n,t)-th entry of $\boldsymbol{\Gamma}_{(k_1,k_2)}$ with $n \in \{1,2,\ldots,N\}$ and $t \in \{1,2,\ldots,T\}$. Herein, Θ is the index set of entries of the block $\hat{\boldsymbol{X}}_{(k_1,k_2)}$ in the inverse Hankel matrix $\theta_{k_1,k_2}^{-1}(\hat{\boldsymbol{X}}_{(k_1,k_2)})$ (see Eq. (3.141)). In the process of inverse Hankelization, the Hankel indexing allows one to perform averaging over the $\tau_1 \times \tau_2$ slices without constructing Hankel tensor explicitly. Thus, our method can achieve efficient estimation for the original matrix $\boldsymbol{X}$ across the $\tau_1 \times \tau_2$ slices of Hankel tensor.

In this work, Hankel indexing is of great significance if the Hankel tensor has a sequence of factorized components (e.g., factor matrices in tensor factorization). This approach eliminates the need to explicitly build a reconstructed Hankel tensor and perform the inverse Hankelization operation. Instead, the technical path would be: 1) reconstructing each slice of the Hankel tensor from the factorized components, e.g., factor matrices in tensor factorization, 2) performing inverse operators of Hankel indexing, and 3) leveraging Eq. (3.140) to compute the estimates sequentially. Since these factorized components are of much lower dimensionality, the memory consumption of the whole process is significantly reduced.

3.4.3 Hankel Tensor Factorization

In this section, we first introduce a tensor-train factorization formula for the constructed (fourth-order) Hankel tensor and then elaborate on how to solve the involved optimization problem without high memory consumption in the Hankelization process. The proposed HTF is totally equivalent to the original HTF but instead becomes memory-efficient in the algorithmic implementation. In particular, to reinforce the parameterization of tensor factorization, we propose to reformulate the tensor-train factorization in the form of circular convolution. As a result, each slice of the Hankel tensor takes a convolutional matrix factorization formula.

Optimization Problem of HTF

As mentioned above, the Hankelization of any matrix allows one to replicate the entries to additional dimensions, resulting in a fourth-order tensor. On the spatiotemporal traffic data $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$ collected from N spatial locations with T consecutive time steps, possibly involving partially or sparsely observed entries, the resulting Hankel tensor would show spatiotemporal correlations of $\boldsymbol{Y}$ in the low-rank modeling process. In particular, one advantage of such Hankelization is that low-rank tensor completion algorithms on the Hankel tensor can substantially address the reconstruction from sparse data (e.g., missing columns/rows [56] and extreme missing scenarios [55]) through characterizing low-rank patterns and time series trends. Therefore, the Hankel tensor is a well-suited structure for spatiotemporal data modeling.

Typically, one can establish a tensor factorization model on the Hankel tensor to reconstruct missing entries in $\boldsymbol{Y}$. Since the Hankel tensor, in this case, is a higher-order tensor, we take a tensor factorization formula in the tensor-train format [92], and the optimization problem of HTF can be written as follows,

$$\begin{aligned}\min_{\boldsymbol{\mathcal{Q}},\boldsymbol{\mathcal{S}},\boldsymbol{U},\boldsymbol{\mathcal{V}}} \ & \frac{1}{2}\Big\|\mathcal{P}_{\tilde{\Omega}}\Big(\mathcal{H}_{\tau_1,\tau_2}(\boldsymbol{Y}) - (\boldsymbol{\mathcal{Q}} \otimes_{2,2} \boldsymbol{\mathcal{S}}) \otimes_{3,3} (\boldsymbol{U} \otimes_{2,2} \boldsymbol{\mathcal{V}})\Big)\Big\|_F^2 \\ & + \frac{\rho}{2}(\|\boldsymbol{\mathcal{Q}}\|_F^2 + \|\boldsymbol{\mathcal{S}}\|_F^2 + \|\boldsymbol{U}\|_F^2 + \|\boldsymbol{\mathcal{V}}\|_F^2),\end{aligned} \tag{3.144}$$

where $\tilde{\Omega}$ denotes the observed index set of the Hankel tensor $\mathcal{H}_{\tau_1,\tau_2}(\boldsymbol{Y})$. Accordingly, the operator $\mathcal{P}_{\tilde{\Omega}} : \mathbb{R}^{(N-\tau_1+1)\times\tau_1\times(T-\tau_2+1)\times\tau_2} \to \mathbb{R}^{(N-\tau_1+1)\times\tau_1\times(T-\tau_2+1)\times\tau_2}$ denotes the orthogonal projection supported on the observed index set $\tilde{\Omega}$. In the tensor-train factorization with a certain rank $R \in \mathbb{N}^+$, the model is parameterized by the variables $\boldsymbol{\mathcal{Q}} \in \mathbb{R}^{(N-\tau_1+1)\times R}$, $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{\tau_1\times R\times R}$, $\boldsymbol{U} \in \mathbb{R}^{(T-\tau_2+1)\times R}$, and $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{\tau_2\times R\times R}$. These variables are factorized components of the Hankel tensor. The symbol $\otimes_{2,2}$ denotes the Einstein summation over the second

dimension of the left-hand side matrix and the right-hand side tensor, e.g.,

$$\begin{aligned} \boldsymbol{Q} \otimes_{2,2} \boldsymbol{\mathcal{S}} &\in \mathbb{R}^{(N-\tau_1+1)\times\tau_1\times R}, \\ \boldsymbol{U} \otimes_{2,2} \boldsymbol{\mathcal{V}} &\in \mathbb{R}^{(T-\tau_2+1)\times\tau_2\times R}, \end{aligned} \tag{3.145}$$

resulting in the third-order tensor, while the symbol $\otimes_{3,3}$ denotes the Einstein summation over the third dimension of the left-hand tensor and the right-hand tensor. Element-wise, we have

$$\begin{aligned} [\boldsymbol{Q} \otimes_{2,2} \boldsymbol{\mathcal{S}}]_{n,k_1,r} &= \sum_{i=1}^{R} q_{n,i} s_{k_1,i,r}, \\ [\boldsymbol{U} \otimes_{2,2} \boldsymbol{\mathcal{V}}]_{t,k_2,r} &= \sum_{i=1}^{R} u_{t,i} v_{k_2,i,r}. \end{aligned} \tag{3.146}$$

with $n \in \{1, 2, \ldots, N-\tau_1+1\}$, $k_1 \in \{1, 2, \ldots, \tau_1\}$, $t \in \{1, 2, \ldots, T-\tau_2+1\}$, and $k_2 \in \{1, 2, \ldots, \tau_2\}$. If we assume that

$$\boldsymbol{\mathcal{W}} \triangleq \boldsymbol{Q} \otimes_{2,2} \boldsymbol{\mathcal{S}}, \quad \boldsymbol{\mathcal{X}} \triangleq \boldsymbol{U} \otimes_{2,2} \boldsymbol{\mathcal{V}}, \tag{3.147}$$

then the operation $\otimes_{3,3}$ takes

$$[\boldsymbol{\mathcal{W}} \otimes_{3,3} \boldsymbol{\mathcal{X}}]_{n,k_1,t,k_2} = \sum_{r=1}^{R} w_{n,k_1,r} x_{t,k_2,r}. \tag{3.148}$$

Figure 3.11 shows the graphical network of such tensor-train factorization formula in which the connection is built by the rank R. To avoid overfitting when optimizing the variables $\{\boldsymbol{Q}, \boldsymbol{\mathcal{S}}, \boldsymbol{U}, \boldsymbol{\mathcal{V}}\}$, ρ is the weight parameter for the regularization terms.

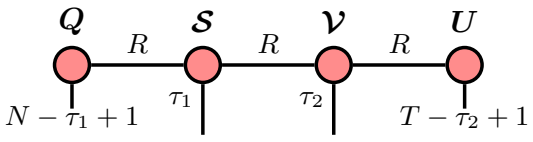


Figure 3.11 Illustration of the tensor network of HTF in Eq. (3.144).

Recall that the Hankel tensor $\mathcal{H}_{\tau_1,\tau_2}(\boldsymbol{Y})$ constructed according to Eq. (3.144) takes the space $\mathcal{O}(\tau_1\tau_2(N-\tau+1)(T-\tau_2+1))$, giving rise to high memory consumption. Notably, $\mathcal{H}_{\tau_1,\tau_2}(\boldsymbol{Y})$ consists of $\tau_1 \times \tau_2$ slices with each of size $(N-\tau_1+1) \times (T-\tau_2+1)$, and these slices can be expressed by using the operators of Hankel indexing (see Eq. (3.139)). Therefore, according to the property of tensor-train factorization [7,92], the tensor factorization in Eq. (3.144) can

be reformulated as follows,

$$\begin{aligned}&\left\|\mathcal{P}_{\tilde{\Omega}}\Big(\mathcal{H}_{\tau_1,\tau_2}(\boldsymbol{Y})-(\boldsymbol{Q}\otimes_{2,2}\boldsymbol{\mathcal{S}})\otimes_{3,3}(\boldsymbol{U}\otimes_{2,2}\boldsymbol{\mathcal{V}})\Big)\right\|_F^2\\=&\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2}\left\|\mathcal{P}_{\Omega_{k_1,k_2}}(\theta_{k_1,k_2}(\boldsymbol{Y})-\boldsymbol{Q}\boldsymbol{S}_{k_1}\boldsymbol{V}_{k_2}\boldsymbol{U}^\top)\right\|_F^2,\end{aligned}\tag{3.149}$$

where Ω_{k_1,k_2} is the observed index set of the matrix $\theta_{k_1,k_2}(\boldsymbol{Y})$. Both two variables

$$\begin{aligned}\boldsymbol{\mathcal{S}}&=\{\boldsymbol{S}_1,\boldsymbol{S}_2,\ldots,\boldsymbol{S}_{\tau_1}\}\in\mathbb{R}^{\tau_1\times R\times R},\\\boldsymbol{\mathcal{V}}&=\{\boldsymbol{V}_1,\boldsymbol{V}_2,\ldots,\boldsymbol{V}_{\tau_2}\}\in\mathbb{R}^{\tau_2\times R\times R},\end{aligned}\tag{3.150}$$

have a sequence of slices as $\boldsymbol{S}_{k_1}\in\mathbb{R}^{R\times R}$, $k_1=1,2,\ldots,\tau_1$ and $\boldsymbol{V}_{k_2}\in\mathbb{R}^{R\times R}$, $k_2=1,2,\ldots,\tau_2$, respectively. They have $(\tau_1+\tau_2)R^2$ parameters in total. If the slices $\boldsymbol{S}_{k_1},\forall k_1$ and $\boldsymbol{V}_{k_2},\forall k_2$ are diagonal matrices, then the HTF model would be reduced to the CP factorization [7].

In this study, we consider compressing parameters in $\boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{V}}$ by introducing circular convolution. First, we assume that both $\boldsymbol{S}_{k_1}$ and $\boldsymbol{V}_{k_2}$ are circulant matrices. Then, we define two kernels $\boldsymbol{s}_{k_1}\in\mathbb{R}^R$ and $\boldsymbol{v}_{k_2}\in\mathbb{R}^R$ corresponding to $\boldsymbol{S}_{k_1}$ and $\boldsymbol{V}_{k_2}$, respectively. By doing so, we propose to rewrite the formula of Eq. (3.144) as

$$\begin{aligned}\min_{\boldsymbol{Q},\boldsymbol{\mathcal{S}},\boldsymbol{U},\boldsymbol{\mathcal{V}}}\ &\frac{1}{2}\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2}\left\|\mathcal{P}_{\Omega_{k_1,k_2}}\Big(\theta_{k_1,k_2}(\boldsymbol{Y})-(\boldsymbol{Q}\star_r\boldsymbol{s}_{k_1}^\top)(\boldsymbol{U}\star_r\boldsymbol{v}_{k_2}^\top)^\top\Big)\right\|_F^2\\&+\frac{\rho}{2}(\|\boldsymbol{Q}\|_F^2+\|\boldsymbol{\mathcal{S}}\|_F^2+\|\boldsymbol{U}\|_F^2+\|\boldsymbol{\mathcal{V}}\|_F^2),\end{aligned}\tag{3.151}$$

where the vector $\boldsymbol{s}_{k_1}\in\mathbb{R}^R$ is the k_1-th column of $\boldsymbol{\mathcal{S}}\in\mathbb{R}^{R\times\tau_1}$, as the vector $\boldsymbol{v}_{k_2}\in\mathbb{R}^R$ is the k_2-th column of $\boldsymbol{\mathcal{V}}\in\mathbb{R}^{R\times\tau_2}$. We introduce this formula as the convolutional matrix factorization on each slice of the Hankel tensor. Definition 3.4.3 gives a concise description of the row-wise circular convolution, i.e., denoted by $\star_r$. We build the connection between $\boldsymbol{Q}$ and $\boldsymbol{s}_{k_1}$ (or between $\boldsymbol{U}$ and $\boldsymbol{v}_{k_2}$) by using circular convolution, playing as a certain kind of parameterization. On these slices of the Hankel tensor, they share the same parameters $\boldsymbol{Q}$ and $\boldsymbol{U}$, while showing differences over the parameters $\{\boldsymbol{s}_{k_1},\boldsymbol{v}_{k_2}\}$.

Definition 3.4.3 (Row-Wise Circular Convolution)**.** *For any matrix $\boldsymbol{X}\in\mathbb{R}^{m\times n}$ and vector $\boldsymbol{y}\in\mathbb{R}^n$, the row-wise circular convolution is defined as*

$$\boldsymbol{Z}\triangleq\boldsymbol{X}\star_r\boldsymbol{y}^\top\in\mathbb{R}^{m\times n},\tag{3.152}$$

element-wise,

$$z_{i,j}=\sum_{k=1}^{n}x_{i,j-k+1}y_k,\tag{3.153}$$

where $i \in \{1, 2, \ldots, m\}$ *and* $j \in \{1, 2, \ldots, n\}$. *By definition, it always holds that* $x_{i,j-k+1} = x_{i,j-k+1+n}$ *for* $j + 1 \leq k$. *Alternatively, for each row of* $\boldsymbol{Z}$, *there exists* $\boldsymbol{z}_i = \boldsymbol{x}_i \star \boldsymbol{y}$ *with* $\star$ *denoting the operator of circular convolution (see Definition 3.3.3).*

As shown in Eq. (3.151), the proposed HTF model has

$$\begin{aligned} &(N - \tau_1 + 1)R + \tau_1 R + (T - \tau_2 + 1)R + \tau_2 R \\ =&(N + T + 2)R, \end{aligned}$$

parameters in the factorized components. Compared to the matrix factorization such that $\boldsymbol{Y} \approx \boldsymbol{Q}\boldsymbol{U}^\top$ with $(N + T)R$ parameters in the factor matrices, the proposed HTF model can capture spatiotemporal correlations without massive parameters. For comparison, we list several HTF models with typical architectures that stem from Eq. (3.144):

- HTF (`dense`): Both $\{\boldsymbol{S}_{k_1}, \boldsymbol{V}_{k_2}\}$ are dense matrices without any structural constraint, referring to the tensor-train factorization.
- HTF (`diag`): Both $\{\boldsymbol{S}_{k_1}, \boldsymbol{V}_{k_2}\}$ are diagonal matrices, referring to the CP factorization.
- HTF (`circ`): Both $\{\boldsymbol{S}_{k_1}, \boldsymbol{V}_{k_2}\}$ are circulant matrices as mentioned above.

Alternating Minimization Algorithm

Although the optimization problem in Eq. (3.151) is nonconvex, it can be addressed by employing the classical alternating minimization algorithm [7, 8, 93]. In the iterative process, one needs to first write down the closed-form solution (e.g., least squares solution) or find the approximated solution to each variable and then implement the updates of the involved variables. In what follows, we give the optimization scheme for finding the approximated solution to each variable from a generalized system of linear equations via the use of the conjugate gradient method [8, 93].

Estimating the Variable $\boldsymbol{Q}$

Suppose f be the objective function of the optimization problem in Eq. (3.151), then the partial derivative of f with respect to the factor matrix $\boldsymbol{Q}$ can be written as follows,

$$\frac{\partial f}{\partial \boldsymbol{Q}} = -\sum_{k_1=1}^{\tau_1} \sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}} \Big(\theta_{k_1,k_2}(\boldsymbol{Y}) - \boldsymbol{Q}\boldsymbol{W}_{k_1,k_2} \Big) \boldsymbol{W}_{k_1,k_2}^\top + \rho \boldsymbol{Q}, \tag{3.154}$$

where we introduce auxiliary variables such that

$$\boldsymbol{W}_{k_1,k_2} \triangleq \mathcal{C}(\boldsymbol{s}_{k_1})^\top \mathcal{C}(\boldsymbol{v}_{k_2})\boldsymbol{U}^\top, \forall k_1, k_2, \tag{3.155}$$

and $\mathcal{C}(\cdot)$ denotes the operator of circulant matrix (see Definition 3.3.1). The transpose of the circulant matrix is also a circulant matrix, depending on the choice of notation. The circular convolution takes the properties associated with the circulant matrix and discrete Fourier transform (see Convolution Theorem in Theorem 3.3.1).

Letting $\frac{\partial f}{\partial \boldsymbol{Q}} = \boldsymbol{0}$ allows one to infer the solution to $\boldsymbol{Q}$ from the following generalized Sylvester equation:

$$\begin{aligned} &\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}\Big(\boldsymbol{Q}\boldsymbol{W}_{k_1,k_2}\Big)\boldsymbol{W}_{k_1,k_2}^\top + \rho\boldsymbol{Q} \\ =&\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}(\theta_{k_1,k_2}(\boldsymbol{Y}))\boldsymbol{W}_{k_1,k_2}^\top. \end{aligned} \tag{3.156}$$

However, some technical challenges arise from this matrix equation because 1) the whole matrix equation is in the Sylvester form with multiple terms, namely, $\tau_1\tau_2 + 1$ terms in the left-hand side and $\tau_1\tau_2$ terms in the right-hand side, and 2) the terms that involve the variable $\boldsymbol{Q}$ are mostly associated with the orthogonal projection (i.e., $\mathcal{P}_{\Omega_{k_1,k_2}}(\cdot)$). It is too complicated to write down the least squares solution and presents a substantial challenge in the development of feasible algorithms. To resolve the above matrix equation, our inference procedure is built on top of conjugate gradient [8], which is often used to approximate the solution to a system of linear equations. We utilize the conjugate gradient method to find the approximated solution to the variable $\boldsymbol{Q}$. Let

$$\begin{cases} \theta_q(\boldsymbol{Q}) \triangleq \mathrm{vec}\Big(\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}\Big(\boldsymbol{Q}\boldsymbol{W}_{k_1,k_2}\Big)\boldsymbol{W}_{k_1,k_2}^\top + \rho\boldsymbol{Q}\Big), \\ \psi_q \triangleq \mathrm{vec}\Big(\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}(\theta_{k_1,k_2}(\boldsymbol{Y}))\boldsymbol{W}_{k_1,k_2}^\top\Big), \end{cases} \tag{3.157}$$

be the vectorization of the left-hand side and the right-hand side of Eq. (3.156), respectively. The conjugate gradient method for solving the variable $\boldsymbol{Q}$ is given by Algorithm 8.

Algorithm 8 Conjugate Gradient for Approximating $\boldsymbol{Q}$

Input: Data $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$ with the observed index set Ω, variables $\{\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{U}, \boldsymbol{V}\}$, window lengths $\tau_1, \tau_2 \in \mathbb{N}^+$, hyperparameter ρ, and maximum iteration $\ell_{\max}$ (e.g., $\ell_{\max} = 5$).
Output: Estimated matrix $\boldsymbol{Q}$.

Initialize $\boldsymbol{q}_0$ by the vectorized $\boldsymbol{Q}$.
Compute residual vector $\boldsymbol{r}_0 = \psi_q - \theta_q(\boldsymbol{Q})$, and let $\boldsymbol{d}_0 = \boldsymbol{r}_0$.
for $\ell = 0$ **to** $\ell_{\max} - 1$ **do**
  Convert vector $\boldsymbol{d}_\ell$ into matrix $\boldsymbol{D}_\ell$.
  Compute $\alpha_\ell = \frac{\boldsymbol{r}_\ell^\top \boldsymbol{r}_\ell}{\boldsymbol{d}_\ell^\top \theta_q(\boldsymbol{D}_\ell)}$.
  Update $\boldsymbol{q}_{\ell+1} = \boldsymbol{q}_\ell + \alpha_\ell \boldsymbol{d}_\ell$.
  Update $\boldsymbol{r}_{\ell+1} = \boldsymbol{r}_\ell - \alpha_\ell \theta_q(\boldsymbol{D}_\ell)$.
  Compute $\beta_\ell = \frac{\boldsymbol{r}_{\ell+1}^\top \boldsymbol{r}_{\ell+1}}{\boldsymbol{r}_\ell^\top \boldsymbol{r}_\ell}$.
  Update $\boldsymbol{d}_{\ell+1} = \boldsymbol{r}_{\ell+1} + \beta_\ell \boldsymbol{d}_\ell$.
end for
Convert $\boldsymbol{q}_{\ell_{\max}}$ into matrix $\boldsymbol{Q}$.
return $\boldsymbol{Q}$.

Estimating the Variables $\{\boldsymbol{s}_{k_1}\}$

According to the property of circular convolution, we have the following equivalent form for the tensor factorization:

$$\begin{aligned} &\Big\|\mathcal{P}_{\Omega_{k_1,k_2}}\Big(\theta_{k_1,k_2}(\boldsymbol{Y}) - (\boldsymbol{Q} \star_r \boldsymbol{s}_{k_1}^\top)(\boldsymbol{U} \star_r \boldsymbol{v}_{k_2}^\top)^\top\Big)\Big\|_F^2 \\ =& \sum_{n=1}^{N-\tau_1+1} \Big\|\mathcal{P}_{\Omega_{k_1,k_2}^n}\Big([\theta_{k_1,k_2}(\boldsymbol{Y})]_n - \boldsymbol{U}\mathcal{C}(\boldsymbol{v}_{k_2})^\top(\boldsymbol{q}_n \star \boldsymbol{s}_{k_1})\Big)\Big\|_2^2, \end{aligned} \tag{3.158}$$

where $[\theta_{k_1,k_2}(\boldsymbol{Y})]_n \in \mathbb{R}^{T-\tau_2+1}$ is the n-th row of the matrix $\theta_{k_1,k_2}(\boldsymbol{Y})$, and correspondingly, Ω_{k_1,k_2}^n is the observed index set of the observed entries. The vector $\boldsymbol{q}_n \in \mathbb{R}^R$ is the n-th row of the matrix $\boldsymbol{Q}$.

As a result, the partial derivative of f with respect to the variable $\boldsymbol{s}_{k_1}$ can be written as follows,

$$\frac{\partial f}{\partial \boldsymbol{s}_{k_1}} = -\sum_{k_2=1}^{\tau_2}\sum_{n=1}^{N-\tau_1+1}\Big(\boldsymbol{H}_{k_2}^\top\Big(\mathcal{P}_{\Omega_{k_1,k_2}^n}\Big([\theta_{k_1,k_2}(\boldsymbol{Y})]_n - \boldsymbol{H}_{k_2}(\boldsymbol{q}_n \star \boldsymbol{s}_{k_1})\Big)\Big)\Big) \star \phi(\boldsymbol{q}_n) + \rho \boldsymbol{s}_{k_1}, \tag{3.159}$$

where we introduce auxiliary variables such that

$$\boldsymbol{H}_{k_2} \triangleq \boldsymbol{U}\mathcal{C}(\boldsymbol{v}_{k_2})^\top, \tag{3.160}$$

and $\phi(\cdot)$ denotes the operator of permutation (see Definition 3.3.4).

Letting $\frac{\partial f}{\partial \boldsymbol{s}_{k_1}} = \boldsymbol{0}$ allows one to infer the solution to $\boldsymbol{s}_{k_1}$ from the following system of linear equations:

$$\begin{aligned}&\sum_{k_2=1}^{\tau_2}\sum_{n=1}^{N-\tau_1+1}\Big(\boldsymbol{H}_{k_2}^{\top}\Big(\mathcal{P}_{\Omega_{k_1,k_2}^n}\Big(\boldsymbol{H}_{k_2}(\boldsymbol{q}_n \star \boldsymbol{s}_{k_1})\Big)\Big)\Big) \star \phi(\boldsymbol{q}_n) + \rho \boldsymbol{s}_{k_1} \\ =&\sum_{k_2=1}^{\tau_2}\sum_{n=1}^{N-\tau_1+1}\Big(\boldsymbol{H}_{k_2}^{\top}\Big(\mathcal{P}_{\Omega_{k_1,k_2}^n}\Big([\theta_{k_1,k_2}(\boldsymbol{Y})]_n\Big)\Big)\Big) \star \phi(\boldsymbol{q}_n).\end{aligned} \tag{3.161}$$

This system is still linear as circular convolution is a kind of linear operation. However, it is difficult to write down the closed-form solution to the variable $\boldsymbol{s}_{k_1}$ because 1) the whole system has multiple terms, namely, $\tau_2(N-\tau_1+1)+1$ terms in the left-hand side and $\tau_2(N-\tau_1+1)$ terms in the right-hand side, 2) the terms that involve the variable $\boldsymbol{s}_{k_1}$ are mostly associated with the orthogonal projection, and 3) the terms in both sides are associated with the operator of circular convolution. Thus, we consider utilizing the conjugate gradient method to find the approximated solution to the variable $\boldsymbol{s}_{k_1}$. Let

$$\begin{cases}\theta_s(\boldsymbol{s}_{k_1}) \triangleq \displaystyle\sum_{k_2=1}^{\tau_2}\sum_{n=1}^{N-\tau_1+1}\Big(\boldsymbol{H}_{k_2}^{\top}\Big(\mathcal{P}_{\Omega_{k_1,k_2}^n}\Big(\boldsymbol{H}_{k_2}(\boldsymbol{q}_n \star \boldsymbol{s}_{k_1})\Big)\Big)\Big) \star \phi(\boldsymbol{q}_n) + \rho \boldsymbol{s}_{k_1}, \\ \psi_s \triangleq \displaystyle\sum_{k_2=1}^{\tau_2}\sum_{n=1}^{N-\tau_1+1}\Big(\boldsymbol{H}_{k_2}^{\top}\Big(\mathcal{P}_{\Omega_{k_1,k_2}^n}\Big([\theta_{k_1,k_2}(\boldsymbol{Y})]_n\Big)\Big)\Big) \star \phi(\boldsymbol{q}_n),\end{cases} \tag{3.162}$$

be the left-hand side and the right-hand side of Eq. (3.161), respectively. The conjugate gradient method for solving the variable $\boldsymbol{s}_{k_1}$ is given by Algorithm 9. When computing with circular convolution, the algorithm takes an FFT implementation over the $N-\tau_1+1$-by-R matrices and then makes a summation over the $N-\tau_1+1$ rows.

Algorithm 9 Conjugate Gradient for Approximating $\boldsymbol{s}_{k_1}$

Input: Data $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$ with the observed index set Ω, variables $\{\boldsymbol{Q}, \boldsymbol{s}_{k_1}, \boldsymbol{U}, \boldsymbol{V}\}$, window lengths $\tau_1, \tau_2 \in \mathbb{N}^+$, hyperparameter ρ, and maximum iteration $\ell_{\max}$ (e.g., $\ell_{\max}=5$).
Output: Estimated vector $\boldsymbol{s}_{k_1}$.

Initialize $\boldsymbol{s}_0$ by $\boldsymbol{s}_{k_1}$.
Compute residual vector $\boldsymbol{r}_0 = \psi_s - \theta_s(\boldsymbol{s}_{k_1})$, and let $\boldsymbol{d}_0 = \boldsymbol{r}_0$.
for $\ell = 0$ **to** $\ell_{\max}-1$ **do**
  Compute $\alpha_\ell = \frac{\boldsymbol{r}_\ell^{\top}\boldsymbol{r}_\ell}{\boldsymbol{d}_\ell^{\top}\theta_s(\boldsymbol{d}_\ell)}$.
  Update $\boldsymbol{s}_{\ell+1} = \boldsymbol{s}_\ell + \alpha_\ell \boldsymbol{d}_\ell$.
  Update $\boldsymbol{r}_{\ell+1} = \boldsymbol{r}_\ell - \alpha_\ell \theta_s(\boldsymbol{d}_\ell)$.
  Compute $\beta_\ell = \frac{\boldsymbol{r}_{\ell+1}^{\top}\boldsymbol{r}_{\ell+1}}{\boldsymbol{r}_\ell^{\top}\boldsymbol{r}_\ell}$.
  Update $\boldsymbol{d}_{\ell+1} = \boldsymbol{r}_{\ell+1} + \beta_\ell \boldsymbol{d}_\ell$.
end for
return $\boldsymbol{s}_{k_1} := \boldsymbol{s}_{\ell_{\max}}$.

Estimating the Variable $\boldsymbol{U}$

With respect to the variable $\boldsymbol{U}$, the partial derivative of f is given by

$$\frac{\partial f}{\partial \boldsymbol{U}} = -\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}^{\top}\Big(\theta_{k_1,k_2}(\boldsymbol{Y}) - \boldsymbol{E}_{k_1,k_2}\boldsymbol{U}^{\top}\Big)\boldsymbol{E}_{k_1,k_2} + \rho\boldsymbol{U}, \tag{3.163}$$

where we introduce auxiliary variables such that

$$\boldsymbol{E}_{k_1,k_2} \triangleq \boldsymbol{Q}\mathcal{C}(\boldsymbol{s}_{k_1})^{\top}\mathcal{C}(\boldsymbol{v}_{k_2}),\ \forall k_1, k_2. \tag{3.164}$$

We can also infer the solution to $\boldsymbol{U}$ via the use of the conjugate gradient method from the following generalized Sylvester equation by letting $\frac{\partial f}{\partial \boldsymbol{U}} = \boldsymbol{0}$:

$$\begin{aligned} &\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}^{\top}(\boldsymbol{E}_{k_1,k_2}\boldsymbol{U}^{\top})\boldsymbol{E}_{k_1,k_2} + \rho\boldsymbol{U} \\ =&\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}^{\top}(\theta_{k_1,k_2}(\boldsymbol{Y}))\boldsymbol{E}_{k_1,k_2}, \end{aligned} \tag{3.165}$$

which involves $\tau_1\tau_2 + 1$ terms in the left-hand side and $\tau_1\tau_2$ terms in the right-hand side. Let

$$\begin{cases} \theta_u(\boldsymbol{U}) \triangleq \mathrm{vec}\Big(\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}^{\top}(\boldsymbol{E}_{k_1,k_2}\boldsymbol{U}^{\top})\boldsymbol{E}_{k_1,k_2}^{\top} + \rho\boldsymbol{U}\Big), \\ \psi_u \triangleq \mathrm{vec}\Big(\sum_{k_1=1}^{\tau_1}\sum_{k_2=1}^{\tau_2} \mathcal{P}_{\Omega_{k_1,k_2}}^{\top}(\theta_{k_1,k_2}(\boldsymbol{Y}))\boldsymbol{E}_{k_1,k_2}^{\top}\Big), \end{cases} \tag{3.166}$$

be the vectorization of the left-hand side and the right-hand side of Eq. (3.165), respectively. The conjugate gradient method for solving the variable $\boldsymbol{U}$ is given by Algorithm 10.

Estimating the Variables $\{\boldsymbol{v}_{k_2}\}$

According to the property of circular convolution, we have the following equivalent form:

$$\begin{aligned} &\Big\|\mathcal{P}_{\Omega_{k_1,k_2}}\Big(\theta_{k_1,k_2}(\boldsymbol{Y}) - (\boldsymbol{Q} \star_r \boldsymbol{s}_{k_1}^{\top})(\boldsymbol{U} \star_r \boldsymbol{v}_{k_2}^{\top})^{\top}\Big)\Big\|_F^2 \\ =&\sum_{t=1}^{T-\tau_2+1} \Big\|\mathcal{P}_{\Omega_{k_1,k_2}^t}\Big([\theta_{k_1,k_2}(\boldsymbol{Y})]_t - \boldsymbol{Q}\mathcal{C}(\boldsymbol{s}_{k_1})^{\top}(\boldsymbol{u}_t \star \boldsymbol{v}_{k_2})\Big)\Big\|_2^2, \end{aligned} \tag{3.167}$$

where $[\theta_{k_1,k_2}(\boldsymbol{Y})]_t \in \mathbb{R}^{N-\tau_1+1}$ is the t-th column of the matrix $\theta_{k_1,k_2}(\boldsymbol{Y})$, and correspondingly, Ω_{k_1,k_2}^t is the observed index set of the observed entries. The vector $\boldsymbol{u}_t \in \mathbb{R}^R$ is the t-th row of the matrix $\boldsymbol{U}$.

Algorithm 10 Conjugate Gradient for Approximating $\boldsymbol{U}$

Input: Data $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$ with the observed index set Ω, variables $\{\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{U}, \boldsymbol{V}\}$, window lengths $\tau_1, \tau_2 \in \mathbb{N}^+$, hyperparameter ρ, and maximum iteration $\ell_{\max}$ (e.g., $\ell_{\max} = 5$).
Output: Estimated matrix $\boldsymbol{U}$.

```
Initialize u_0 by the vectorized U.
Compute residual vector r_0 = ψ_u − θ_u(U), and let d_0 = r_0.
for ℓ = 0 to ℓ_max − 1 do
    Convert vector d_ℓ into matrix D_ℓ.
    Compute α_ℓ = (r_ℓ^⊤ r_ℓ) / (d_ℓ^⊤ θ_u(D_ℓ)).
    Update u_{ℓ+1} = u_ℓ + α_ℓ d_ℓ.
    Update r_{ℓ+1} = r_ℓ − α_ℓ θ_u(D_ℓ).
    Compute β_ℓ = (r_{ℓ+1}^⊤ r_{ℓ+1}) / (r_ℓ^⊤ r_ℓ).
    Update d_{ℓ+1} = r_{ℓ+1} + β_ℓ d_ℓ.
end for
Convert u_{ℓ_max} into matrix U.
return U.
```

Thus, the partial derivative of f with respect to the variable $\boldsymbol{v}_{k_2}$ is

$$\frac{\partial f}{\partial \boldsymbol{v}_{k_2}} = -\sum_{k_1=1}^{\tau_1}\sum_{t=1}^{T-\tau_2+1}\Big(\boldsymbol{G}_{k_1}^\top\Big(\mathcal{P}_{\Omega_{k_1,k_2}^t}\Big([\theta_{k_1,k_2}(\boldsymbol{Y})]_t - \boldsymbol{G}_t(\boldsymbol{u}_t \star \boldsymbol{v}_{k_2})\Big)\Big)\Big) \star \phi(\boldsymbol{u}_t) + \rho\boldsymbol{v}_{k_2}, \tag{3.168}$$

where we introduce auxiliary variables such that

$$\boldsymbol{G}_{k_1} \triangleq \boldsymbol{Q}\mathcal{C}(\boldsymbol{s}_{k_1})^\top. \tag{3.169}$$

Let $\frac{\partial f}{\partial \boldsymbol{v}_{k_2}} = \boldsymbol{0}$, we have the following system of linear equations:

$$\begin{aligned}&\sum_{k_1=1}^{\tau_1}\sum_{t=1}^{T-\tau_2+1}\Big(\boldsymbol{G}_{k_1}^\top\Big(\mathcal{P}_{\Omega_{k_1,k_2}^t}\Big(\boldsymbol{G}_{k_1}(\boldsymbol{u}_t \star \boldsymbol{v}_{k_2})\Big)\Big)\Big) \star \phi(\boldsymbol{u}_t) + \rho\boldsymbol{v}_{k_2}\\ =&\sum_{k_1=1}^{\tau_1}\sum_{t=1}^{T-\tau_2+1}\Big(\boldsymbol{G}_{k_1}^\top\Big(\mathcal{P}_{\Omega_{k_1,k_2}^t}\Big([\theta_{k_1,k_2}(\boldsymbol{Y})]_t\Big)\Big)\Big) \star \phi(\boldsymbol{u}_t),\end{aligned} \tag{3.170}$$

which can also be solved by the conjugate gradient method. Let

$$\begin{cases}\theta_v(\boldsymbol{v}_{k_2}) \triangleq \sum_{k_1=1}^{\tau_1}\sum_{t=1}^{T-\tau_2+1}\Big(\boldsymbol{G}_{k_1}^\top\Big(\mathcal{P}_{\Omega_{k_1,k_2}^t}\Big(\boldsymbol{G}_{k_1}(\boldsymbol{u}_t \star \boldsymbol{v}_{k_2})\Big)\Big)\Big) \star \phi(\boldsymbol{u}_t) + \rho\boldsymbol{v}_{k_2},\\ \psi_v \triangleq \sum_{k_1=1}^{\tau_1}\sum_{t=1}^{T-\tau_2+1}\Big(\boldsymbol{G}_{k_1}^\top\Big(\mathcal{P}_{\Omega_{k_1,k_2}^t}\Big([\theta_{k_1,k_2}(\boldsymbol{Y})]_t\Big)\Big)\Big) \star \phi(\boldsymbol{u}_t),\end{cases} \tag{3.171}$$

be the left-hand side and the right-hand side of Eq. (3.170), respectively. The conjugate gradient method for solving the variable $\boldsymbol{v}_{k_2}$ is given by Algorithm 11. When computing with circular convolution, the algorithm takes an FFT implementation over the $T - \tau_2 + 1$-by-R matrices and then makes a summation over the $T - \tau_2 + 1$ rows.

Algorithm 11 Conjugate Gradient for Approximating $\boldsymbol{v}_{k_2}$

Input: Data $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ with the observed index set Ω, variables $\{\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{U}, \boldsymbol{v}_{k_2}\}$, window lengths $\tau_1, \tau_2 \in \mathbb{N}^+$, hyperparameter ρ, and maximum iteration $\ell_{\max}$ (e.g., $\ell_{\max} = 5$).
Output: Estimated vector $\boldsymbol{v}_{k_2}$.

Initialize $\boldsymbol{v}_0$ by $\boldsymbol{v}_{k_2}$.
Compute residual vector $\boldsymbol{r}_0 = \psi_v - \theta_v(\boldsymbol{v}_{k_2})$, and let $\boldsymbol{d}_0 = \boldsymbol{r}_0$.
for $\ell = 0$ **to** $\ell_{\max} - 1$ **do**
  Compute $\alpha_\ell = \frac{\boldsymbol{r}_\ell^\top \boldsymbol{r}_\ell}{\boldsymbol{d}_\ell^\top \theta_v(\boldsymbol{d}_\ell)}$.
  Update $\boldsymbol{v}_{\ell+1} = \boldsymbol{v}_\ell + \alpha_\ell \boldsymbol{d}_\ell$.
  Update $\boldsymbol{r}_{\ell+1} = \boldsymbol{r}_\ell - \alpha_\ell \theta_v(\boldsymbol{d}_\ell)$.
  Compute $\beta_\ell = \frac{\boldsymbol{r}_{\ell+1}^\top \boldsymbol{r}_{\ell+1}}{\boldsymbol{r}_\ell^\top \boldsymbol{r}_\ell}$.
  Update $\boldsymbol{d}_{\ell+1} = \boldsymbol{r}_{\ell+1} + \beta_\ell \boldsymbol{d}_\ell$.
end for
return $\boldsymbol{v}_{k_2} := \boldsymbol{v}_{\ell_{\max}}$.

Solution Algorithm

As mentioned above, we obtain the matrix equations with respect to the factor matrices $\boldsymbol{Q}$ and $\boldsymbol{U}$, in the meanwhile, we have the generalized systems of linear equations to the vectors $\boldsymbol{s}_{k_1}$ and $\boldsymbol{v}_{k_2}$. Thus, we can summarize the alternating minimization scheme of the proposed HTF model as follows,

$$\begin{cases} \boldsymbol{Q} := \{\boldsymbol{Q} \mid \dfrac{\partial f}{\partial \boldsymbol{Q}} = \boldsymbol{0}\}, \\ \boldsymbol{s}_{k_1} := \{\boldsymbol{s}_{k_1} \mid \dfrac{\partial f}{\partial \boldsymbol{s}_{k_1}} = \boldsymbol{0}\},\ k_1 \in \{1, 2, \ldots, \tau_1\}, \\ \boldsymbol{U} := \{\boldsymbol{U} \mid \dfrac{\partial f}{\partial \boldsymbol{U}} = \boldsymbol{0}\}, \\ \boldsymbol{v}_{k_2} := \{\boldsymbol{v}_{k_2} \mid \dfrac{\partial f}{\partial \boldsymbol{v}_{k_2}} = \boldsymbol{0}\},\ k_2 \in \{1, 2, \ldots, \tau_2\}, \end{cases} \tag{3.172}$$

where the least squares solution to each subproblem can be approximated by the conjugate gradient method.

Algorithm 12 shows the implementation of the proposed HTF model. In the whole procedure, it does not require building a large Hankel tensor of size $(N - \tau_1 + 1) \times \tau_1 \times (T - \tau_2 + 1) \times \tau_2$

explicitly. The operator of Hankel indexing allows one to maintain the memory consumption within $\mathcal{O}(NT)$. This step toward Hankel tensor completion can address the most critical issue of a manifold of Hankel models (e.g., [43, 55]). In particular, solving the matrix equations or the systems of linear equations via the use of the conjugate gradient method can also reduce both space complexity and time complexity.

In the HTF model, if we have the factors $\{\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{U}, \boldsymbol{V}\}$ estimated from Eq. (3.172), then making use of the inverse operators of Hankel indexing in Eq. (3.140), we put forward the following reconstructed matrix $\hat{\boldsymbol{Y}} \in \mathbb{R}^{N \times T}$ for HTF:

$$\hat{\boldsymbol{Y}} = \Big(\sum_{k_1=1}^{\tau_1} \sum_{k_2=1}^{\tau_2} \theta_{k_1,k_2}^{-1} \big((\boldsymbol{Q} \star_r \boldsymbol{s}_{k_1}^\top)(\boldsymbol{U} \star_r \boldsymbol{v}_{k_2}^\top) \big) \Big) \oslash \Big(\sum_{k_1=1}^{\tau_1} \sum_{k_2=1}^{\tau_2} \boldsymbol{\Gamma}_{(k_1,k_2)} \Big), \tag{3.173}$$

where $\boldsymbol{\Gamma}_{(k_1,k_2)} \in \{0,1\}^{N \times T}, \forall k_1, k_2$ is the binary matrix defined in the same way as Eq. (3.143), consisting of 0 and 1 for indicating the observation information.

Therefore, HTF can not only discover the low-rank patterns from Hankel tensor data but also show appealing properties for producing spatiotemporal reconstruction outcomes that are both globally and locally consistent. Since HTF takes into account side information brought by spatial/temporal modeling via Hankelization, it is capable of learning from very sparse data.

Algorithm 12 Memory-Efficient Hankel Tensor Factorization

Input: Data $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ with the observed index set Ω, window lengths $\tau_1, \tau_2 \in \mathbb{N}^+$, rank R, and hyperparameter ρ.
Output: Reconstructed matrix $\hat{\boldsymbol{Y}} \in \mathbb{R}^{N \times T}$ and variables $\{\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{U}, \boldsymbol{V}\}$.

Randomly initialize the variables $\{\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{U}, \boldsymbol{V}\}$.
repeat
  Compute $\boldsymbol{Q}$ from Eq. (3.156) with conjugate gradient (see Algorithm 8).
  for $k_1 = 1$ **to** τ_1 **do**
    Compute $\boldsymbol{s}_{k_1}$ from Eq. (3.161) with conjugate gradient (see Algorithm 9).
  end for
  Compute $\boldsymbol{U}$ from Eq. (3.165) with conjugate gradient (see Algorithm 10).
  for $k_2 = 1$ **to** τ_2 **do**
    Compute $\boldsymbol{v}_{k_2}$ from Eq. (3.170) with conjugate gradient (see Algorithm 11).
  end for
  Compute the reconstructed matrix $\hat{\boldsymbol{Y}}$ (see Eq. (3.173)).
until convergence
return $\hat{\boldsymbol{Y}}, \boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{U}, \boldsymbol{V}$.

3.5 Concluding Remark

We have presented four low-rank temporal models for characterizing the low-rank property and the time series trends of sparse traffic data, showing temporal modeling advantages on the basis of low-rank matrix and tensor methods. These models can impute missing traffic time series data and perform efficient traffic prediction in the presence of sparse input data. More concretely, we summarize these models as follows,

- (**NoTMF**) NoTMF is an extension of TMF by integrating differencing operations on the latent temporal factors. We develop an approach to model the temporal loss using matrix operations, and this allows one to solve the optimization problem through conjugate gradient. The conjugate gradient method is an efficient routine for estimating the approximated solution to the factor matrix from the generalized Sylvester equation. Therefore, the proposed algorithms can support efficient modeling/forecasting on high-dimensional and large-scale traffic time series data, making efficient and reliable predictions for real-time applications.

- (**LATC**) In the LATC model, we introduce temporal variation as a new regularization term into the completion of a third-order tensor of dimensions (spatial location/sensor $\times$ time of day $\times$ day). The proposed model preserves both global low-rank structure (i.e., by minimizing the truncated nuclear norm) and local temporal structure (i.e., by minimizing the temporal variation) in multivariate traffic time series.

- (**LCR**) To model the local trends in traffic time series, we introduce a Laplacian kernel for temporal regularization in the form of circular convolution. Following that definition, we propose an LCR model that integrates the temporal regularization into a circulant-matrix-based low-rank model for characterizing both global and local trends in traffic time series, bridging the gap between low-rank models and graph Laplacian models. When developing the solution algorithm, we borrow the properties of circulant matrix and circular convolution and prove that the proposed LCR model has a fast implementation through FFT. To be specific, the nuclear norm minimization with Laplacian kernelized temporal regularization can be converted into an ℓ_1-norm minimization in complex space. Beyond univariate time series imputation, LCR can be easily adapted to multivariate or even multidimensional time series imputation tasks. Due to the efficient implementation and relatively low time complexity, LCR demonstrates strong generalization to large-scale and high-dimensional problems.

- (**HTF**) To address the high memory consumption issue in the Hankel tensor comple-

tion algorithms, we propose reformulating HTF by using Hankel indexing, which is an efficient representation for reformulating HTF without explicitly building Hankel tensors. On each slice of the Hankel tensor, we present a tensor factorization formula instead of the tensor-train format and CP format, imposing a convolutional parameterization process. The whole optimization problem of HTF is solved by using alternating minimization in which each subproblem is indeed a (generalized) system of linear equations and can be solved efficiently by the conjugate gradient method. The automatic spatiotemporal modeling with Hankel structure allows one to handle extreme missing traffic data imputation.

CHAPTER 4 EXPERIMENTS

With remarkable advances in low-cost sensing technologies, an increasing amount of real-world traffic measurement data are collected with high spatial sensor coverage and fine temporal resolution. This research focuses on answering some critical scientific questions on spatiotemporal traffic data and provides some data-driven machine learning solutions. In this chapter, we consider several spatiotemporal traffic data imputation and forecasting tasks and evaluate the proposed models presented in Chapter 3. The specific tasks are summarized as follows,

- **Univariate traffic time series imputation**. Section 4.1 gives a preliminary discussion of traffic time series imputation in the univariate case. To reconstruct missing and corrupted traffic data, it requires us to utilize both global and local trends of partially observed traffic data. Recall that LCR leverages 1) the circulant matrix nuclear norm for characterizing the low-rank property of traffic data, and 2) the Laplacian kernel for characterizing local trends of time series. We are considering using LCR to implement this task.

- **Spatiotemporal traffic data imputation**. Section 4.2 introduces spatiotemporal missing traffic data imputation with various missing data patterns, including random missing, non-random missing, and block-out missing. Recall that LATC utilizes the day dimension of traffic data to construct a data tensor with dimensions (spatial location/sensor $\times$ time of day $\times$ day). Furthermore, LATC characterizes time series trends by autoregression, capable of handling complicated missing data patterns.

- **Large-scale traffic data imputation**. Section 4.3 discusses how to solve the large-scale traffic data imputation with the proposed model. Since LCR shows a fast implementation through FFT, it is well-suited to such a task.

- **Extreme missing traffic data imputation**. Section 4.4 introduces the challenging task of extreme missing traffic data imputation in which partially observed traffic data are very sparse. Recall that HTF provides a flexible framework that can make full use of spatiotemporal patterns and correlations of traffic data. We consider using HTF to implement this task and in the meanwhile compare the imputation performance with other models.

- (**Sparse urban traffic state forecasting**). Section 4.5 focuses on the network-wide traffic state forecasting with sparsely collected movement data of Uber ridesharing

vehicles. The missing pattern is very different from the random missing pattern and the sparsity degree depends on the sampling of ridesharing vehicles in total traffic. We use NoTMF to achieve the task of sparse traffic state forecasting.

To evaluate the imputation and forecasting performance, we use the mean absolute percentage error (MAPE) and the root mean square error (RMSE) as performance metrics:

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i} \times 100, \qquad \text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \tag{4.1}$$

where n is the total number of estimated values, and y_i and $\hat{y}_i$ are the actual value and its estimate, respectively.

4.1 Univariate Traffic Time Series Imputation

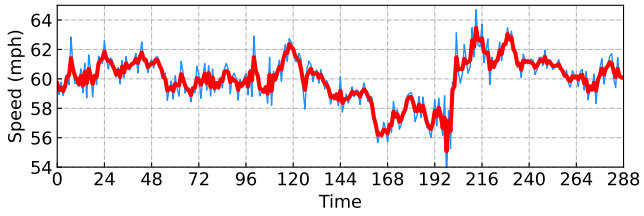
This section evaluates the reconstruction of univariate traffic time series from partial observations. The reconstruction results of the proposed LCR model are compared with several baseline models. Experiments are based on traffic speed time series (weak periodicity and strong noises) and traffic volume time series (strong periodicity). We focus on understanding how well the reconstructed time series preserves the global and local trends of the ground truth time series in the imputation task.

4.1.1 Traffic Speed

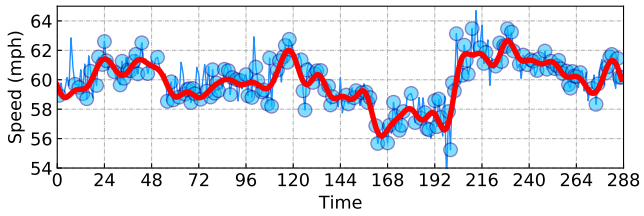
We begin with some preliminary experiments on univariate freeway traffic speed time series collected through dual-loop detectors on the highway network of Portland, USA.[1] The speed observations have a 15-minute time resolution (i.e., 96 expected data samples per day) over three days, and the data vector is of length 288. In particular, we consider the imputation scenarios on fully observed, 50%, 20%, 10%, and 5% observed data, respectively. To generate the partially observed time series, we randomly mask a certain number of observations as missing values with the prescribed missing rate.

Figure 4.1 shows the reconstruction/imputation results produced by our LCR model. Figure 4.1(a) demonstrates the reconstructed time series by LCR on the fully observed time series, in which the data noises can be smoothed out due to the existence of Laplacian kernelized temporal regularization and the relaxation of observation constraint $\|\mathcal{P}_\Omega(\boldsymbol{x}-\boldsymbol{y})\|_2 \leq \epsilon$ in which $\boldsymbol{y}$ and $\boldsymbol{x}$ are the partial observations and the reconstructed time series, respectively.

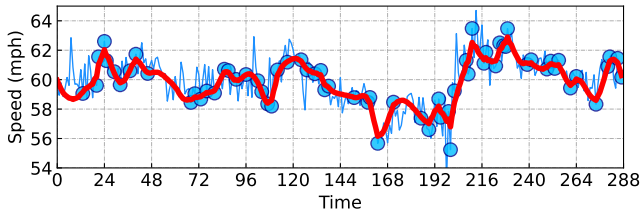
[1] `https://portal.its.pdx.edu/`



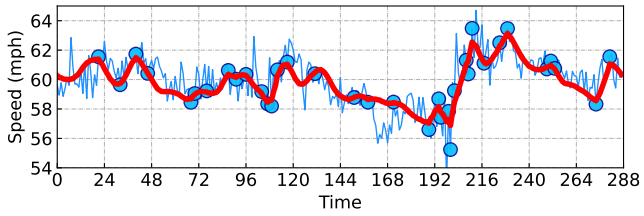
(a) Fully observed.



(b) 50% missing values.



(c) 80% missing values.

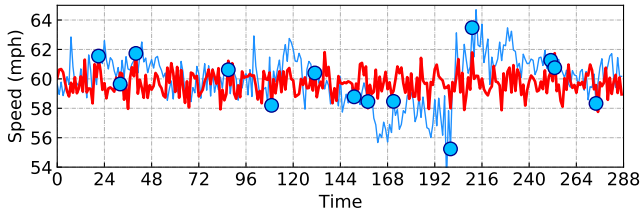


(d) 90% missing values.

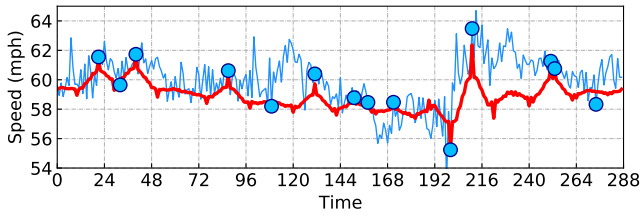
Figure 4.1 Univariate traffic time series imputation on the freeway traffic speed time series. The blue curve represents the ground truth time series, while the red curve refers to the reconstructed time series produced by LCR. Here, partial observations are illustrated as blue circles.

Figure 4.1(b), 4.1(c), and 4.1(d) show the imputation performance by LCR with partial observations. Of these results, the reconstructed time series can accurately approximate both partial observations and missing values while preserving the trends of the ground truth time series.

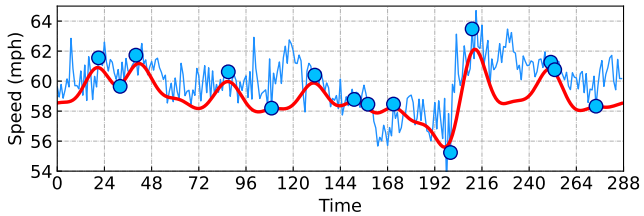
Next, we test a more challenging scenario in which we aim to reconstruct 95% missing values from 5% observations (i.e., reconstructing 274 missing values from only 14 data samples).



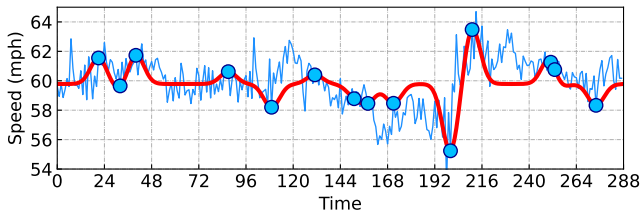
(a) CircNNM (red curve).



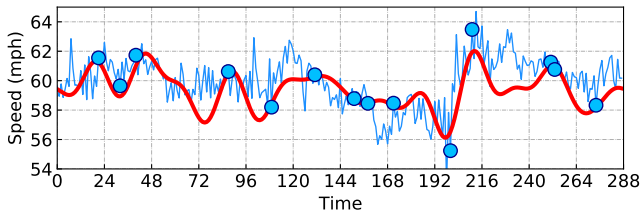
(b) ConvNNM (red curve).



(c) ConvNNM+ (red curve).



(d) GP (red curve).



(e) LCR (red curve) with $\tau = 2, \gamma = 5\lambda$.

Figure 4.2 Univariate traffic time series imputation on the freeway traffic speed time series. In this case, we mask 95% observations as missing values and only have 14 speed observations for training the model. The window length of ConvNNM and ConvNNM+ is set as $\tilde{\tau} = 48$.

We compare the time series imputation of LCR with the following baseline models:

- CircNNM [46]. This is a nuclear norm minimization of the circulant matrix via FFT, which can be regarded as a special case of our LCR model without temporal regularization.
- ConvNNM [46]. This is a standard nuclear norm minimization on the convolution matrix. In particular, we set up a baseline model as ConvNNM with Laplacian kernel (i.e., ConvNNM+) whose optimization problem is given by

$$\begin{aligned} \min_{\boldsymbol{x}} \ & \|\mathcal{C}_{\tilde{\tau}}(\boldsymbol{x})\|_* + \frac{\gamma}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2 \\ \text{s.t.} \ & \|\mathcal{P}_{\Omega}(\boldsymbol{x} - \boldsymbol{y})\|_2 \leq \epsilon, \end{aligned} \tag{4.2}$$

  where $\tilde{\tau} \in \mathbb{N}^+$ is the window length of the convolution matrix as discussed in Eq. (3.78).
- GP [94]. The GP regression with a squared exponential kernel. The hyperparameters are optimized via the maximum marginal likelihood method.

Essentially, comparing the proposed LCR model with these baseline models allows one to highlight a) the importance of global and local trends modeling in LCR, and b) the fast implementation of LCR via the use of FFT. The imputation results on the given time series are shown in Figure 4.2, we can summarize the following findings for the baseline models:

- The reconstructed time series produced by CircNNM shows high fluctuations due to the lack of local trend modeling. As a result, the reconstructed time series fails to reproduce trends of the ground truth time series under such a high missing rate.
- ConvNNM performs better than CircNNM. The reconstructed time series fits the observed values well, but the trend is still far from satisfactory compared to the ground truth time series. Unlike CircNNM, ConvNNM cannot employ a fast implementation via FFT. As a result, ConvNNM is not well-suited to large problems.
- ConvNNM+ outperforms ConvNNM. This demonstrates the significance of the Laplacian kernel for local time series trend modeling. Although ConvNNM+ and ConvNNM present very similar global time series trends, ConvNNM+ with Laplacian kernel shows a better approximation to the partial observations, as well as the local time series trend is highlighted. Unfortunately, both ConvNNM and ConvNNM+ require to implement the singular value thresholding on T-by-$\tilde{\tau}$ convolution matrices, consuming $\mathcal{O}(\tilde{\tau}T)$ space.

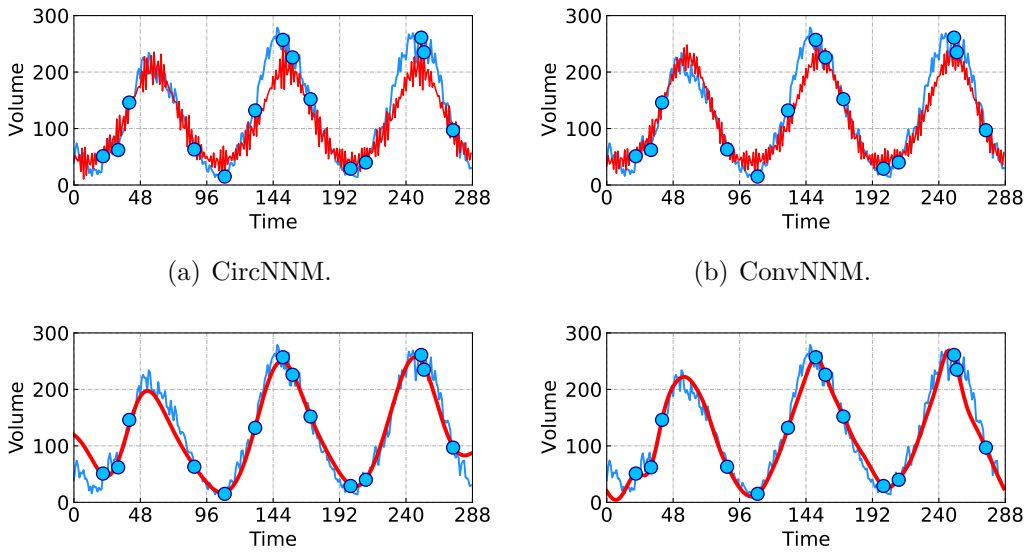
- The GP model reasonably reconstructs the traffic speed time series but is computationally costly. A common fact is that GP models are not well-suited to long time series and large problems due to the time complexity of $\mathcal{O}(T^3)$.

As shown in Figure 4.2(e), the reconstructed time series by LCR demonstrates consistent global and local trends with the trends of ground truth time series, and our LCR model clearly outperforms the baseline models in terms of either accuracy or computational efficiency. By comparing CircNNM with LCR, the imputation results emphasize the importance of temporal regularization with the Laplacian kernel. Notably, when producing the reconstructed time series $\boldsymbol{x}$, both CircNNM and LCR consume $\mathcal{O}(T)$ space, however, ConvNNM and ConvNNM+ take $\mathcal{O}(\tilde{\tau}T)$ space. In terms of temporal modeling, both LCR and GP can produce accurate reconstruction. Nevertheless, the proposed temporal regularization via Laplacian kernel presents an FFT solution in $\mathcal{O}(T \log T)$ time, which is much more computationally efficient than GP with $\mathcal{O}(T^3)$ time.

4.1.2 Traffic Volume

In addition to the traffic speed time series, we evaluate the LCR model on the freeway traffic volume time series collected through loop detectors on the highway network of Portland, USA. The volume observations are with 15-minute time resolution over three days, and the data vector is of length 288. The time series is characterized by a strong daily rhythm (see Figure 4.3). The time series shows relatively low traffic volume at night, and the traffic volume reaches a peak during rush hours, demonstrating typical traveler's behavioral rhythms. As shown in Figure 4.3, the time series possesses strong seasonality and three remarkable peaks over three days.

The task is reconstructing 95% missing values from 5% observations (i.e., reconstructing 274 missing values from only 14 data samples). As can be seen, due to the lack of explicit temporal modeling, both CircNNM and ConvNNM models cannot produce smooth time series as LCR. Observing the reconstructed time series, it is clear that LCR produces much more accurate reconstruction results than both CircNNM and ConvNNM. Yet, In contrast to the traffic speed time series, due to the strong seasonality (i.e., daily rhythm in traffic flow) in the traffic volume time series, both CircNNM and ConvNNM can produce reasonable time series, empirically demonstrating to be consistent with the results in [47].



(a) CircNNM.

(b) ConvNNM.

(c) GP.

(d) LCR with $\tau = 2, \gamma = 5\lambda$.

Figure 4.3 Univariate time series imputation on the freeway traffic volume time series. In this case, we randomly remove 95% observations as missing values, and we only have 14 volume observations for the reconstruction.

4.2 Spatiotemporal Traffic Data Imputation

In this section, we evaluate the proposed imputation model on two real-world traffic flow datasets with different missing patterns as shown in Figure 4.4, including random missing (RM), nonrandom missing (NM), and block-out missing (BM). The RM pattern is created when the observations of each spatial location are masked at completely random. According to the mechanism of the RM pattern, we mask a certain amount of observations as missing values (e.g., 30%, 70%, and 90%), and the remaining partial observations are input data for learning an imputation model. The NM pattern follows the mechanism that the observations of each spatial location are masked during several randomly selected continuous days, independently between sensors. The BM pattern is different from both RM and NM patterns, which masks observations of all spatial locations as missing values with a certain period. BM is a challenging scenario with complete column-wise missing data on the traffic data matrix $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$. We set the missing rate in the following experiments as 30%.

In this case, LATC can handle the BM missing data because 1) it folds the traffic data matrix as a data tensor with the day dimension, and 2) it takes autoregressive processes to characterize local time series trends of traffic data. We use LATC to evaluate the imputation scenarios over all three missing data patterns. For comparison, we take into account the

following baseline models:

- Low-rank autoregressive matrix completion (LAMC). This is a matrix-form variant of the LATC model.
- Low-rank tensor completion with truncated nuclear norm minimization (LRTC-TNN) [21]. This is a low-rank completion model in which truncated nuclear norm minimization can help maintain the most important low-rank patterns. Since the truncation in LRTC-TNN is defined as a rate parameter, we adapt LRTC-TNN to use integer truncation in order to make it consistent with LATC.
- BTMF [1]. This is a fully Bayesian temporal factorization framework that builds the correlation of temporal dynamics on latent factors by the VAR process. Due to the temporal modeling, it outperforms the standard matrix factorization in the missing data imputation tasks [1].
- Smooth PARAFAC tensor completion (SPC) [95]. This is a tensor decomposition-based completion model with total variation smoothness constraints.

4.2.1 On Seattle Freeway Traffic Speed Data

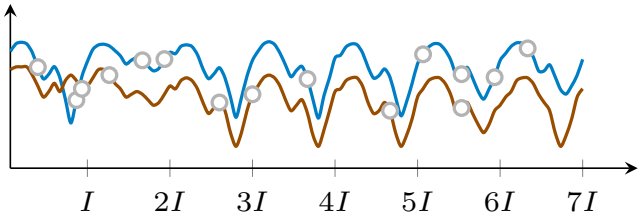
We first consider the Seattle freeway traffic speed dataset.[2] This dataset contains freeway traffic speed from 323 loop detectors on the freeway network with a 5-minute time resolution (i.e., 288 time steps per day) over the first four weeks of January, 2015 in Seattle, USA. The processed data is of size 323×8064 in the form of a multivariate time series matrix. Alternatively, introducing the day dimension would lead to a tensor data of size $323 \times 288 \times 28$.

On the generated missing data scenarios, one needs to validate a well-suited setting for LATC, prescribing the weight parameter γ, hyperparameter λ, truncation parameter r, and time lag set $\mathcal{H}$. We set the hyperparamter $\lambda = 10^{-5}$, validating the ratio γ/λ from a collection of given values, $\gamma/\lambda \in \{1/10, 1/5, 1, 5, 10\}$. The time lag set is given by $\mathcal{H} = \{1, 2, \ldots, 6\}$. We validate the truncation parameter from a collection of given values, $r \in \{5, 10, 15, 20, 25\}$.

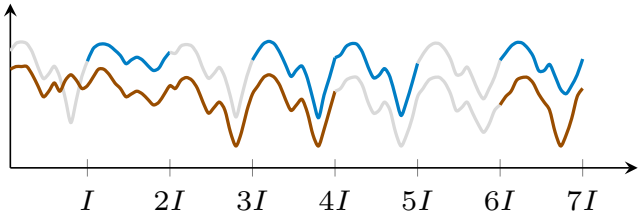
Table 4.1 summarizes the imputation performance achieved by LATC on the Seattle freeway traffic speed dataset. It demonstrates that:

- For all three missing patterns, LATC usually performs best as $\gamma = 10\lambda$. In terms of the RM data, the imputation results are consistently improved as the ratio γ/λ

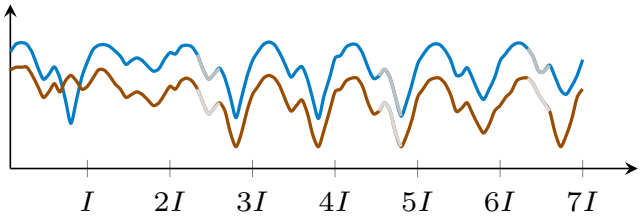
[2] `https://github.com/zhiyongc/Seattle-Loop-Data`



(a) RM pattern.



(b) NM pattern.



(c) BM pattern.

Figure 4.4 Illustration of three missing data patterns for spatiotemporal traffic data (e.g., traffic speed). Each time series represents the collected data from a given tensor. (a) Data are missing at random. Small circles indicate the missing values. (b) Data are missing continuously during a few time periods. (c) No sensors are available (i.e., block-out) over a certain time window.

changes from 1/10 to 10. In terms of the NM data, better imputation results are achieved when $\gamma = 10\lambda$ at the 70% missing rate. In terms of the BM data, LATC consistently performs better when the ratio γ/λ increases from 1/10 to 10. These results highlight the importance of temporal variation via autoregression, as the higher weight parameter γ over the hyperparameter λ is usually required for achieving better imputation performance.

- A well-suited truncation parameter r is required for achieving better imputation performance by LATC. The specification of r depends on both the missing pattern and the missing rate. In terms of the RM data, the best imputation results are achieved when the truncation parameter $r = 25$. In terms of the NM data, the best truncation

Table 4.1 Imputation performance (in MAPE/RMSE) of LATC with different truncation values and different ratios γ/λ on the Seattle freeway traffic speed dataset. Note that the best results are highlighted in bold fonts. The number next to the BM denotes the window length.

Missing rate	γ/λ	Truncation parameter				
		$r = 5$	$r = 10$	$r = 15$	$r = 20$	$r = 25$
30%, RM	1/10	5.72/3.48	5.44/3.38	5.24/3.30	5.09/3.24	4.98/3.19
	1/5	5.72/3.48	5.44/3.38	5.23/3.30	5.09/3.24	4.98/3.19
	1	5.70/3.47	5.42/3.37	5.22/3.29	5.08/3.23	4.97/3.19
	5	5.62/3.43	5.36/3.34	5.17/3.27	5.03/3.21	4.93/3.17
	10	5.55/3.40	5.30/3.32	5.12/3.25	4.99/3.19	**4.90**/**3.16**
70%, RM	1/10	7.47/4.36	6.89/4.12	6.53/3.96	6.31/3.85	6.14/3.79
	1/5	7.47/4.35	6.89/4.12	6.53/3.96	6.30/3.85	6.14/3.79
	1	7.45/4.34	6.87/4.11	6.51/3.95	6.29/3.84	6.12/3.78
	5	7.34/4.29	6.78/4.06	6.43/3.91	6.21/3.81	6.03/3.74
	10	7.22/4.23	6.69/4.02	6.36/3.87	6.14/3.78	**5.96**/**3.71**
90%, RM	1/10	9.36/5.29	8.44/4.91	8.06/4.74	7.95/4.70	7.89/4.71
	1/5	9.36/5.28	8.43/4.91	8.07/4.75	7.94/4.70	7.87/4.70
	1	9.34/5.28	8.41/4.89	8.03/4.73	7.88/4.68	7.81/4.67
	5	9.27/5.24	8.33/4.85	7.93/4.68	7.73/4.60	7.58/4.56
	10	9.21/5.20	8.26/4.82	7.85/4.64	7.63/4.55	**7.46**/**4.50**
30%, NM	1/10	7.96/4.65	7.51/4.48	7.31/4.39	7.18/4.36	**7.09**/4.36
	1/5	7.97/4.65	7.51/4.48	7.31/4.40	7.18/4.37	**7.09**/4.35
	1	7.97/4.66	7.52/4.49	7.32/4.40	7.18/4.36	**7.09**/4.35
	5	8.00/4.67	7.55/4.50	7.35/4.41	7.19/4.36	7.10/4.34
	10	8.04/4.68	7.59/4.51	7.37/4.42	7.22/4.37	7.10/**4.33**
70%, NM	1/10	9.90/5.58	9.45/5.43	9.47/5.51	9.50/5.54	9.50/5.60
	1/5	9.90/5.58	9.45/5.43	9.46/5.53	9.49/5.55	9.51/5.61
	1	9.91/5.59	9.44/5.43	9.46/5.51	9.48/5.54	9.50/5.60
	5	9.92/5.59	9.44/5.42	9.41/5.48	9.46/5.52	9.47/5.56
	10	9.94/5.59	**9.40**/**5.40**	**9.40**/5.46	9.45/5.50	**9.40**/5.50
30%, BM-12	1/10	10.01/5.58	9.59/5.47	9.85/5.58	10.07/5.67	9.94/5.63
	1/5	10.01/5.58	9.59/5.47	9.84/5.58	10.07/5.67	9.91/5.61
	1	10.00/5.57	9.57/5.45	9.81/5.55	10.00/5.64	9.89/5.59
	5	9.92/5.53	9.50/5.40	9.68/5.48	9.88/5.55	9.80/5.50
	10	9.86/5.49	**9.43**/**5.36**	9.60/5.42	9.84/5.50	9.80/5.47

parameters corresponding to 30% and 70% missing rates are $r = 25$ and $r = 10$, respectively. In terms of the BM data, LATC with $r = 10$ consistently performs better

than other settings.

In particular, as shown in Figure 4.5, we give some reconstructed time series achieved by LATC for all three missing patterns with different missing rates. On the RM data, LATC consistently produces accurate imputation. On the NM data with a 70% missing rate, LATC even reconstructs the missing time series over the whole day. On the BM data, LATC can capture both local trends and day-to-day dynamic patterns and make use of these for producing accurate imputation.

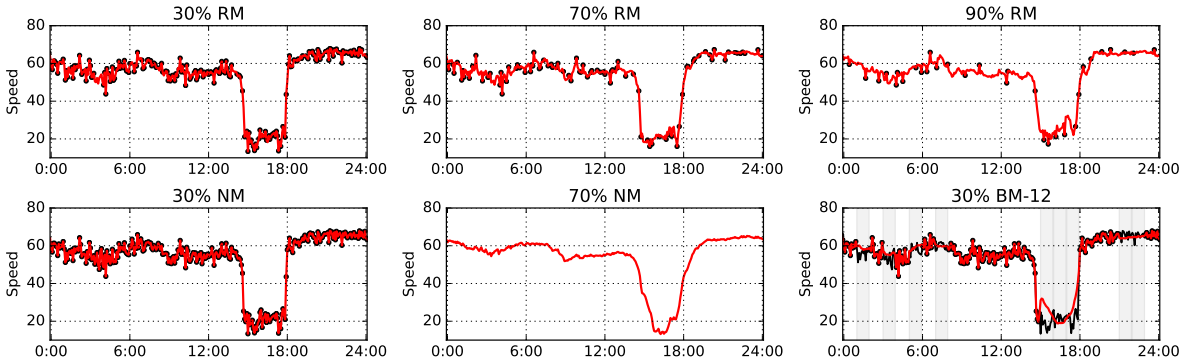


Figure 4.5 Time series imputation achieved by LATC on the Seattle freeway traffic speed dataset. This example corresponds to detector #3 and the 7th day of the dataset.

As mentioned above, despite the use of truncated nuclear norm built on tensor, the results also show the importance of temporal variation built on the multivariate time series matrix. Due to the temporal modeling, temporal variation can improve the imputation performance for missing traffic data imputation. Table 4.2 summarizes the overall imputation performance of LATC and baseline models on the dataset with various missing data scenarios. In most cases, LATC outperforms the baseline models. Comparing LATC with LAMC shows the advantage of tensor structure, i.e., LATC with tensor structure performs better than LAMC with matrix structure. This also implies that introducing a day dimension on traffic data is an appropriate operation for traffic data imputation. Comparing LATC with LRTC-TNN shows the advantage of temporal variation, i.e., temporal modeling with an autoregressive process has a positive influence on improving the imputation performance.

4.2.2 On Portland Highway Traffic Volume Data

In this study, we also consider the Portland highway traffic volume dataset. This dataset is collected from the highway network of the Portland-Vancouver Metropolitan region, which contains traffic volume from 1,156 loop detectors with a 15-minute time resolution (i.e., 96

Table 4.2 Imputation performance comparison (in MAPE/RMSE) of LATC and baseline models on the Seattle freeway traffic speed dataset. Note that the best results are highlighted in bold fonts.

Missing rate	LATC	LAMC	LRTC-TNN	BTMF	SPC
30%, RM	**4.90/3.16**	5.98/3.73	4.99/3.20	5.91/3.72	5.92/3.62
70%, RM	**5.96/3.71**	8.02/4.70	6.10/3.77	6.47/3.98	7.38/4.30
90%, RM	**7.46/4.50**	10.56/5.91	8.08/4.80	8.17/4.81	9.75/5.31
30%, NM	7.10/4.33	6.99/4.25	**6.85/4.21**	9.26/5.36	8.87/4.99
70%, NM	9.40/5.40	9.75/5.60	**9.23/5.35**	10.47/6.15	11.32/5.92
30%, BM-12	**9.43/5.36**	27.05/13.66	9.52/5.41	14.33/13.60	11.30/5.84

time steps per day) in January 2021. The processed data is of size 1156×2976 in the form of a multivariate time series matrix. Alternatively, introducing the day dimension would lead to a tensor data of size $1156 \times 96 \times 31$. To set up the LATC model for the generated missing data imputation tasks, we set the hyperparameter $\lambda = 10^{-5}$, validating the ratio γ/λ from a collection of given values, $\gamma/\lambda \in \{1/10, 1/5, 1, 5, 10\}$. The time lag set is given by $\mathcal{H} = \{1, 2, 3, 4\}$. We validate the truncation parameter from a collection of given values, $r \in \{5, 10, 15, 20, 25\}$.

Table 4.3 summarizes the imputation performance achieved by LATC on the Portland highway traffic volume dataset. It demonstrates that:

- The temporal variation is of great significance in the LATC model because an appropriate λ is required for the imputation in all three missing data patterns. In terms of the RM data, the best results are achieved when $\gamma = \lambda/5$ or $\gamma = \lambda$. In terms of the NM data, the best settings are $\gamma = \lambda/10$ and $\gamma = \lambda$ with respect to 30% and 70% missing rates, respectively. In terms of the BM data, a relatively large γ is required to achieve better imputation.

- An appropriate truncation parameter r of LATC is required for achieving better imputation in some cases. At the 70% missing rate on the NM data, LATC with the truncation parameter $r = 5$ produces the best result. On the BM data, LATC with a relatively small truncation parameter such as $r = 5$ produces better imputation results.

Figure 4.6 shows some reconstructed traffic volume time series achieved by LATC for all three missing data patterns with different missing rates. On the RM data, LATC produces consistently accurate imputation even at a 90% missing rate with limited observations. On the NM data, both two examples do now show any observations, but LATC can capture

local trends and day-to-day dynamic patterns and make use of these for producing accurate imputation. On the BM data, column-wise missing values can be reconstructed by LATC which utilizes tensor structure and autoregression.

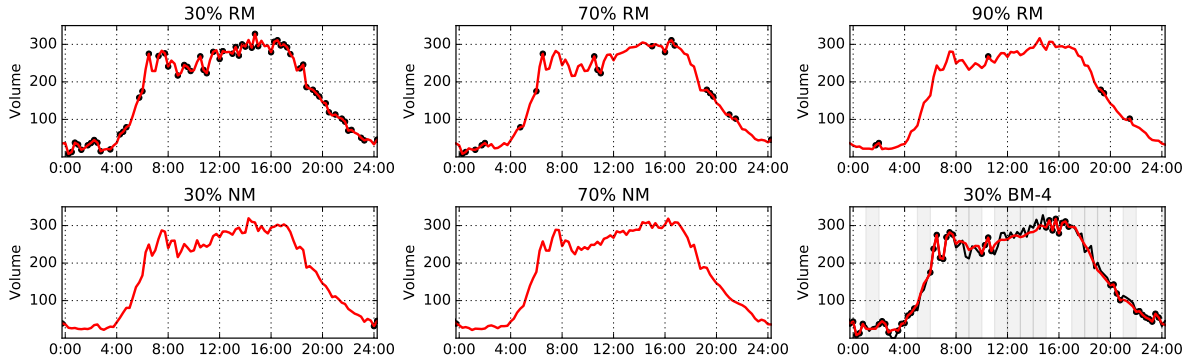


Figure 4.6 Time series imputation achieved by LATC on the Portland traffic volume dataset. This example corresponds to detector #3 and the 8th day of the dataset.

Table 4.4 summarizes the imputation performance of LATC and baseline models on the dataset. As can be seen, LATC consistently shows competitive imputation performance against the baseline models. In most cases, LATC outperforms LRTC-TNN, showing the importance of temporal variation (see 30%/70% RM data, 70% NM data, and BM data for example).

4.3 Large-Scale Traffic Data Imputation

Missing data problem in spatiotemporal traffic flow has long been challenging, particularly for large-scale and high-dimensional data with complicated missing mechanisms and diverse degree of missing rate. This section discusses the imputation problem on the large-scale traffic flow data. In particular, we consider a large-scale traffic dataset collected by the California Department of Transportation through their PeMS [2]. This dataset contains freeway traffic speed collected from 11,160 traffic measurement sensors over 4 weeks (the first 4 weeks in the year of 2018) with a 5-minute time resolution (288 time steps per day) in California, USA.[3] It can be arranged in a matrix of size 11160×8064. Note that this dataset contains about 90 million observations; some proposed imputation methods such as LATC and HTF are not well-suited to such a large-scale dataset.

To set up the imputation task, we randomly mask 30%, 50%, 70%, and 90% traffic speed observations as missing values, referring to 30%, 50%, 70%, and 90% missing rates, respec-

[3]The dataset is available at `https://doi.org/10.5281/zenodo.3939792`.

Table 4.3 Imputation performance (in MAPE/RMSE) of LATC with different truncation values and different ratios γ/λ on the Portland highway traffic volume dataset. Note that the best results are highlighted in bold fonts. The number next to the BM denotes the window length.

Missing rate	γ/λ	Truncation parameter				
		$r = 5$	$r = 10$	$r = 15$	$r = 20$	$r = 25$
30%, RM	1/10	17.73/16.49	17.45/15.90	17.21/15.99	17.11/16.29	17.04/16.50
	1/5	17.71/16.46	17.44/**15.88**	17.20/15.95	17.10/16.23	17.03/16.38
	1	17.64/16.36	17.39/15.90	17.15/15.90	17.02/15.97	**16.95**/15.99
	5	17.69/16.85	17.48/16.67	17.27/16.65	17.17/16.66	17.09/16.64
	10	17.95/17.70	17.78/17.63	17.61/17.64	17.53/17.66	17.47/17.65
70%, RM	1/10	21.05/19.85	20.47/19.15	20.07/19.16	19.86/19.17	19.78/19.06
	1/5	21.03/19.81	20.45/19.14	20.04/19.06	19.84/18.97	19.74/18.92
	1	20.91/19.73	20.34/19.03	19.94/18.80	19.71/18.72	**19.59**/**18.70**
	5	20.77/20.01	20.30/19.63	19.98/19.60	19.76/19.52	19.66/19.60
	10	20.72/20.50	20.36/20.32	20.07/20.38	19.91/20.38	19.85/20.46
90%, RM	1/10	24.92/24.25	23.83/23.42	23.51/23.12	23.42/22.93	23.55/23.06
	1/5	24.87/24.17	23.77/23.46	23.43/22.84	23.33/**22.75**	23.45/22.94
	1	24.83/24.25	23.64/23.28	23.28/22.90	**23.15**/22.83	23.25/23.08
	5	24.71/24.78	23.79/24.71	23.41/24.78	23.18/25.23	23.27/25.54
	10	24.51/25.30	23.80/25.80	23.53/26.27	23.37/26.80	23.50/27.28
30%, NM	1/10	19.86/19.65	19.48/**19.14**	19.18/21.93	18.95/22.57	**18.81**/23.42
	1/5	19.87/19.66	19.48/19.21	19.20/25.53	18.96/25.49	18.82/26.05
	1	19.95/19.77	19.53/19.40	19.25/27.39	19.91/35.00	18.93/32.18
	5	20.48/20.64	20.04/21.26	19.79/28.47	23.17/50.77	39.00/101.38
	10	21.13/21.52	20.75/25.55	20.43/29.72	35.43/84.99	51.70/123.28
70%, NM	1/10	28.21/52.73	41.93/95.99	59.93/124.12	60.97/122.74	51.94/109.76
	1/5	28.21/52.73	42.65/98.18	63.31/129.85	71.82/137.15	61.10/124.44
	1	**27.67**/**45.03**	40.22/93.06	61.35/127.13	74.46/145.18	77.71/151.28
	5	28.98/49.72	41.95/93.69	60.32/131.33	76.41/153.89	81.86/161.48
	10	29.90/50.56	41.03/90.24	58.69/127.20	71.55/150.34	83.41/164.87
30%, BM-4	1/10	26.43/39.50	34.23/83.26	45.25/106.58	47.06/110.22	48.19/112.29
	1/5	24.40/25.16	29.63/64.11	37.36/83.62	39.56/88.62	40.81/91.12
	1	24.01/**23.50**	24.76/36.87	25.81/42.58	26.38/45.33	26.93/47.59
	5	23.64/23.62	**23.01**/23.71	23.08/24.47	23.26/25.63	23.50/26.43
	10	23.56/23.82	23.13/24.02	23.27/24.57	23.50/25.39	23.80/26.00

tively. To assess the imputation performance, we use the actual values of the masked missing entries as the ground truth to compute MAPE and RMSE. For comparison, we choose the

Table 4.4 Imputation performance comparison (in MAPE/RMSE) of LATC and baseline models on the Portland highway traffic volume dataset. Note that the best results are highlighted in bold fonts.

Missing rate	LATC	LAMC	LRTC-TNN	BTMF	SPC
30%, RM	**16.95**/**15.99**	17.93/16.03	17.27/16.08	18.22/19.14	21.29/56.73
70%, RM	**19.59**/**18.70**	21.26/19.37	19.99/18.73	19.96/22.21	24.35/43.32
90%, RM	23.15/22.83	25.64/23.75	**22.90**/**22.68**	23.90/25.71	28.45/39.65
30%, NM	**19.48**/19.14	19.93/19.69	19.59/**18.91**	19.55/20.38	26.96/60.33
70%, NM	27.67/45.03	25.75/28.25	30.26/60.85	**23.86**/**26.74**	33.42/47.34
30%, BM-4	**24.01**/**23.50**	29.21/27.60	31.74/74.42	27.85/25.68	31.01/60.33

baseline models as CircNNM [46], LRMC [25], high-accuracy low-rank tensor completion (HaLRTC) [96], LRTC-TNN [21], and NoTMF. Through cross validation, we have the following settings:

- LCR-2D: We set the hyperparameters as $\lambda = 10^{-5}NT$, $\eta = 10^2\lambda$, and $\gamma = 10\lambda$. In particular, the kernel sizes are set as $\tau = 1$, $\tau = 2$, and $\tau = 3$ corresponding to the 30%/50% missing rate, 70% missing rate, and 90% missing rate, respectively.

- LCR$_N$: We set the hyperparameters as $\lambda = 10^{-3}T$, $\eta = 10^2\lambda$, and $\gamma = 10\lambda$. In particular, the kernel sizes are set as $\tau = 1$, $\tau = 2$, and $\tau = 3$ corresponding to the 30%/50% missing rate, 70% missing rate, and 90% missing rate, respectively.

- LCR: We consider the LCR via the vectorization on the multivariate time series [47], which is given by

$$\begin{aligned} \min_{\boldsymbol{X}} \ & \|\mathcal{C}(\text{vec}(\boldsymbol{X}^\top))\|_* + \frac{\gamma}{2}\|\boldsymbol{\ell} \star \text{vec}(\boldsymbol{X}^\top)\|_2^2 \\ \text{s.t.} \ & \|\mathcal{P}_\Omega(\boldsymbol{X} - \boldsymbol{Y})\|_F \leq \epsilon, \end{aligned} \tag{4.3}$$

  where the Laplacian kernel $\boldsymbol{\ell}$ is of length NT. We set the hyperparameters as $\lambda = 10^{-5}NT$, $\eta = 10^2\lambda$, and $\gamma = 10\lambda$. In particular, the kernel sizes are set as $\tau = 1$, $\tau = 2$, and $\tau = 3$ corresponding to the 30%/50% missing rate, 70% missing rate, and 90% missing rate, respectively.

- CTNNM: This is a special case of LCR-2D without a regularization term. We set the hyperparameters as $\lambda = 10^{-5}NT$ and $\eta = 10^2\lambda$.

- CircNNM: We follow the nuclear norm minimization of circulant matrix via the vectorization [46] and set the hyperparameters as $\lambda = 10^{-5}NT$ and $\eta = 10^2\lambda$.

As shown in Table 4.5, LCR-2D, LCR$_N$, and LCR achieve very competitive imputation accuracy and outperform baseline models. Among the baseline models in Table 4.5, CTNNM and CircNNM are the special cases of LCR-2D and LCR, respectively. Both models can be implemented by FFT like our LCR models. By comparing the imputation performance of CTNNM (or CircNNM) and LCR-2D (or LCR), we can demonstrate the significant improvement of imputation achieved by LCR-2D over CTNNM, mainly due to the existence of the regularization term with Laplacian kernels. Therefore, introducing local trend modeling with Laplacian kernels in our LCR models is of great significance for traffic time series imputation. Despite the aforementioned comparison, our LCR models perform significantly better than some matrix/tensor completion algorithms such as LRMC, HaLRTC, and LRTC-TNN. Here, these matrix/tensor completion algorithms provide well-suited frameworks for reconstructing missing values in the data matrix/tensor. However, they involve high time complexity in the singular value thresholding process, e.g.,

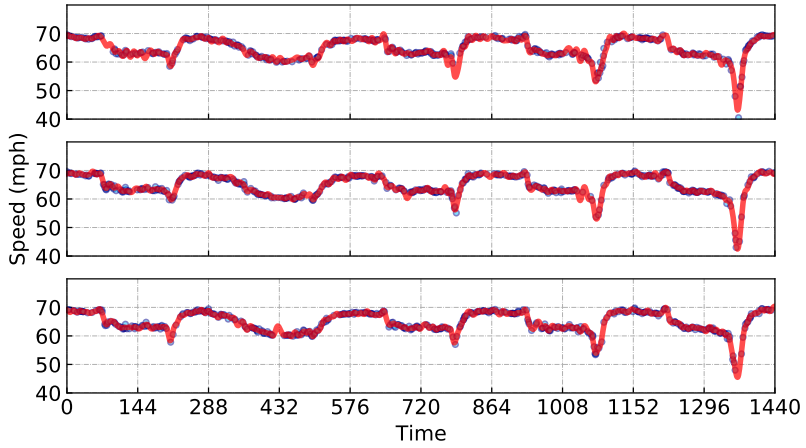
$$\mathcal{O}(\min\{N^2T, NT^2\}) \quad \text{(SVD)} \qquad \text{vs.} \qquad \mathcal{O}(NT\log(NT)) \quad \text{(FFT)},$$

making it extremely costly to work on large-scale problems. In contrast, our LCR models convert the nuclear norm minimization as an ℓ_1-norm minimization in the frequency domain via the use of FFT. NoTMF jointly characterizes global and local trends by a unified and efficient temporal matrix factorization framework, but as can be seen, it is inferior to the LCR models.

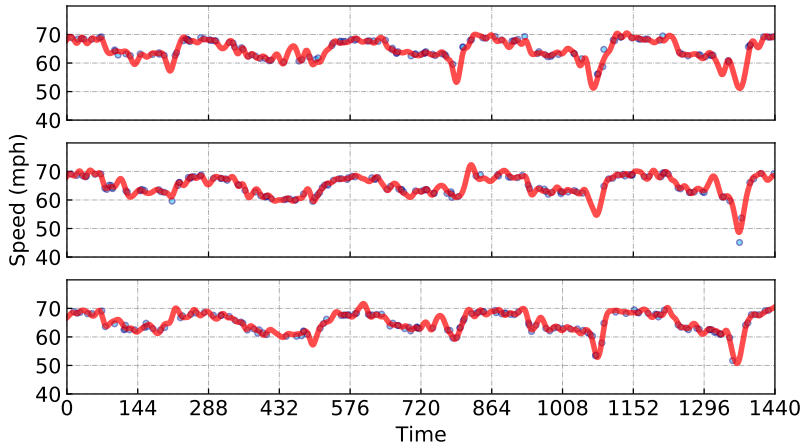
Table 4.5 Imputation performance (MAPE/RMSE) on the PeMS-4W traffic speed dataset. Note that the best results are highlighted in bold fonts.

Model	Missing rate			
	30%	50%	70%	90%
LCR-2D	1.50/**1.49**	1.76/**1.69**	**2.07**/**2.06**	**3.19**/**3.05**
LCR$_N$	**1.48**/1.50	**1.73**/1.73	**2.07**/2.12	3.24/3.22
LCR	1.50/**1.49**	1.76/**1.69**	2.08/2.07	3.21/3.06
CTNNM	2.26/1.84	2.67/2.14	3.40/2.66	5.22/3.90
CircNNM	2.26/1.84	2.69/2.15	3.43/2.67	5.34/3.96
LRMC	2.04/1.80	2.43/2.12	3.08/2.66	6.05/4.43
HaLRTC	1.98/1.73	2.22/1.98	2.84/2.49	4.39/3.66
LRTC-TNN	1.68/1.55	1.93/1.77	2.33/2.14	3.40/3.10
NoTMF	2.95/2.65	3.05/2.73	3.33/2.97	5.22/4.71

Figure 4.7 shows the speed imputation results of three road segments at 70% and 90% missing rates. In the extreme case with high missing rates, we can see that the reconstructed time



(a) 70% missing rate.



(b) 90% missing rate.

Figure 4.7 Multivariate traffic time series imputation by LCR on the PeMS-4W dataset. We visualize the traffic speed time series of the first three road segments during the first five days. Note that blue circles and red curves indicate the partial observations and the reconstructed time series, respectively.

series curves achieved by LCR preserve the overall trend of the original speed time series. Notably, the reconstructed curves even correctly depict the sharp decrease of the speed during congestion periods, demonstrating the effectiveness of LCR in capturing local information of the traffic speed time series.

4.4 Extreme Missing Traffic Data Imputation

In this section, we evaluate the proposed HTF model on some spatiotemporal traffic data imputation tasks, including speed field reconstruction of road traffic flow and freeway traf-

fic speed data imputation. We consider the extreme missing data scenarios to show the importance of spatial and temporal Hankelization on the sparse traffic data.

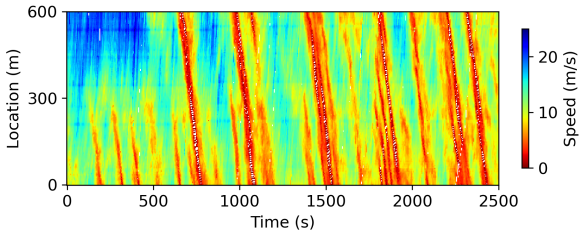
4.4.1 Speed Field Reconstruction of Road Traffic Flow

Speed field reconstruction is a critical problem in road traffic flow modeling as the data collection could often be far from ideal and only a small fraction of trajectories is gathered [3]. The task is to reconstruct the traffic speed distribution along a road segment during a period using the motion states (i.e., time, location, and speed values) of a small portion of vehicles [97], e.g., to estimate the network-wide traffic states by using only the motion states of taxis. We use the next generation simulation (NGSIM) data[4] for the experiment. Figure 4.8(a) shows the ground truth speed field, which is in the form of a matrix (of size 200×500). The speed field is with the 3-meter spatial resolution (i.e., $N = 200$) and the 5-second time resolution (i.e., $T = 500$), demonstrating spatiotemporal patterns of road traffic flow.

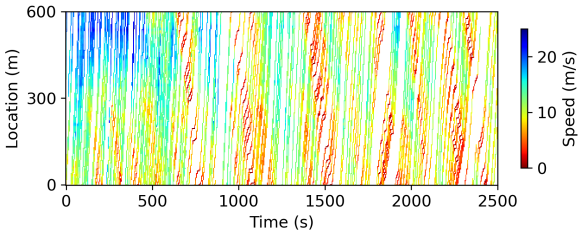
The speed field reconstruction problem, in this case, can be regarded as a short time series imputation task that allows one to highlight the importance of both spatial and temporal modeling via Hankelization. Figure 4.8(b) is the speed field produced from the trajectories of 20% of the vehicles chosen randomly, showing to be rather sparse. We evaluate our LCR and HTF models on Figure 4.8(b) and reconstruct a sparse speed field, while LCR plays as a baseline model. Through testing the LCR model, we have the following setting: $\lambda = 10^{-5}NT$, $\eta = 10^2\lambda$, and $\gamma = 5\lambda$. Through testing the HTF model, we have the following setting: $\tau_1 = 2, \tau_2 = 10$ (window lengths), $R = 10$ (rank), and $\rho = 10$ (hyperparameter). For comparison, we choose a baseline model as LCR that can be demonstrated to be efficient for speed field reconstruction due to the effective spatiotemporal modeling and the fast implementation via FFT. As shown in Figure 4.8, the proposed HTF model with circulant structure outperforms both HTF in the tensor-train format and HTF in the CP format. We empirically demonstrate the importance of circular convolution for the parameterization of tensor factorization. In addition, the proposed HTF model performs better than the LCR model, showing a strong ability for spatiotemporal modeling.

Furthermore, we consider a more challenging scenario on the NGSIM speed field data, i.e., trajectories of only 5% of the vehicles chosen randomly. As shown in Figure 4.9(a), the speed field is very sparse and seems to be hard to reconstruct without certain domain knowledge in

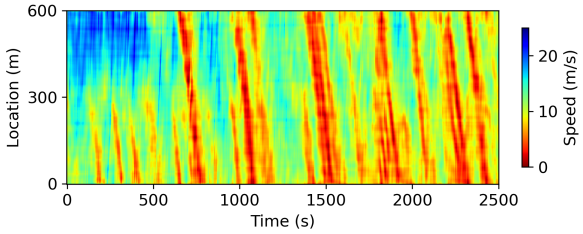
[4]U.S. Department of Transportation Federal Highway Administration. (2016). NGSIM Vehicle Trajectories and Supporting Data. [Dataset]. Provided by ITS DataHub through Data.transportation.gov. Accessed 2023-01-01 from `http://doi.org/10.21949/1504477`.



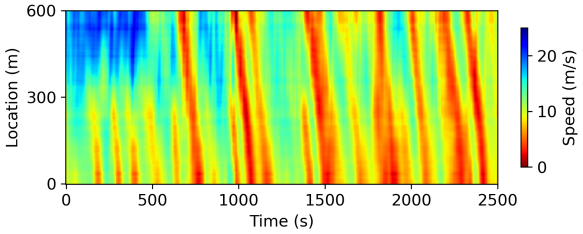
(a) Original speed field.



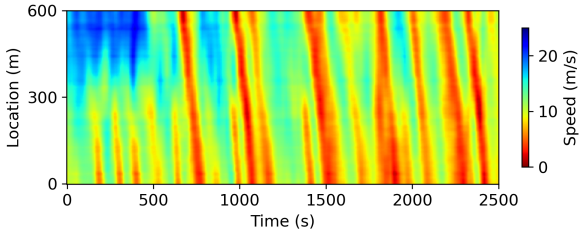
(b) Incomplete speed field.



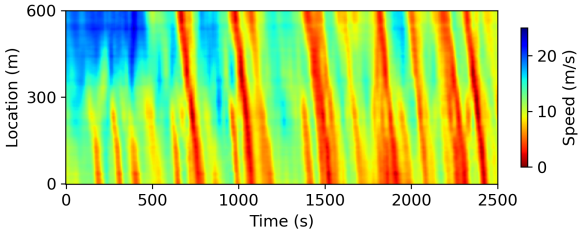
(c) LCR (MAPE = 41.29% & RMSE = 1.55 m/s).



(d) HTF (`dense`) (MAPE = 48.05% & RMSE = 1.57 m/s).



(e) HTF (`diag`) (MAPE = 50.23% & RMSE = 1.49 m/s).



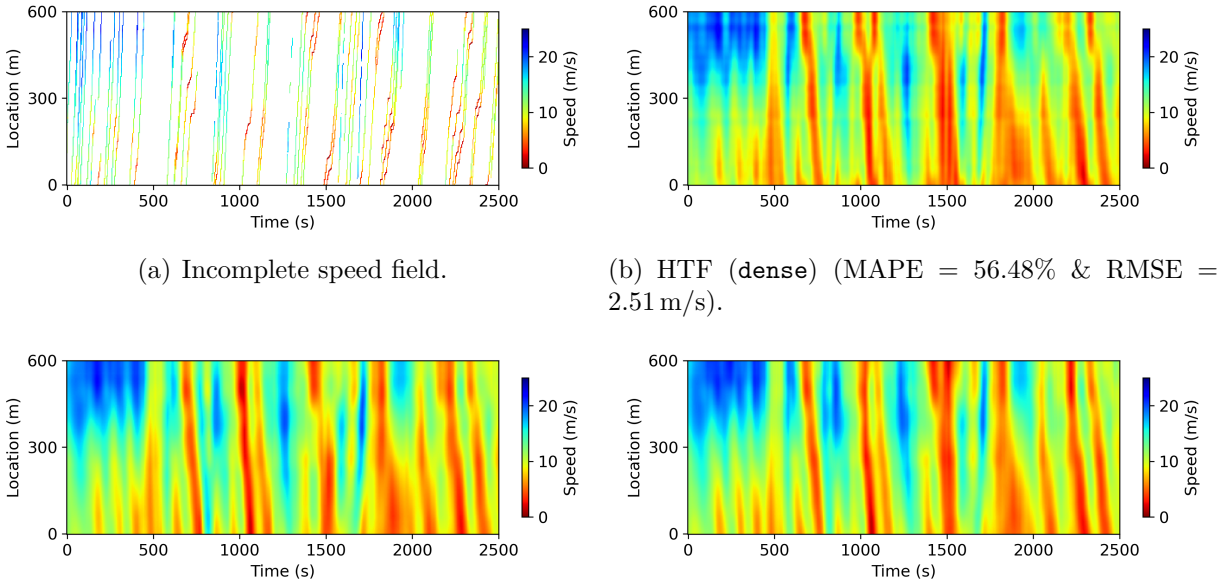
(f) HTF (`circ`) (MAPE = **41.16%** & RMSE = **1.39 m/s**).

Figure 4.8 Speed field reconstruction of road traffic flow on the 80% masked trajectories. The smaller the MAPE and RMSE, the better the reconstruction quality. The LCR model in this case refers to LCR-2D.

the spatiotemporal context. Through evaluation of separate data, we set the proposed HTF model with $\tau_1 = 15, \tau_2 = 20$ (window lengths), $R = 6$ (rank), and $\rho = 10$ (hyperparameter). As shown in Figure 4.9, HTF can identify spatiotemporal patterns from such very sparse data through Hankelization and tensor factorization and therefore produce consistent reconstruction as the ground truth data. Notably, HTF with the circulant structure outperforms both tensor-train factorization and CP factorization.

4.4.2 Freeway Traffic Speed Imputation

To demonstrate the performance of the proposed HTF model, we evaluate HTF and the baseline models on the Seattle freeway traffic speed data. This dataset can be represented



(a) Incomplete speed field.

(b) HTF (`dense`) (MAPE = 56.48% & RMSE = 2.51 m/s).

(c) HTF (`diag`) (MAPE = 53.93% & RMSE = 2.53 m/s).

(d) HTF (`circ`) (MAPE = **51.92%** & RMSE = **2.42 m/s**).

Figure 4.9 Speed field reconstruction of road traffic flow on the 95% masked trajectories.

as a time series matrix of size 323×8064 (see Section 4.2.1). To highlight the importance of spatiotemporal modeling via Hankelization, we consider some extreme missing data scenarios, i.e., 80%, 85%, 90%, and 95% missing values randomly generated from the original data.

To make a thorough comparison, we consider the following state-of-the-art baseline models: 1) LATC, 2) LRTC-TNN [98], 3) LCR, 4) smoothing matrix factorization (SMF), and 5) BTMF [1]. The parameter setting of HTF on the Seattle freeway traffic speed dataset is summarized as: $\tau_1 = 2$ (window length, see Table 4.6 for evaluating HTF with different $\tau_1 \in \{2, 3, 4\}$), $R = 100$ (rank), and $\rho = 10^{-1}$ (hyperparameter). In particular, we adjust the value of window length τ_2 according to the data sparsity: 1) for 80% and 85% missing rates, we set $\tau_2 = 6$; 2) for 90% missing rate, we set $\tau_2 = 12$; and 3) for 95% missing rate, we set $\tau_2 = 24$.

As shown in Table 4.7, the better imputation performance achieved by the HTF model is with the convolutional parameterization, against tensor-train factorization and CP factorization. The proposed HTF model can characterize both spatial and temporal dependencies of traffic data, while the circular convolution operator reinforces the parameterization process. The LCR model takes a circulant tensor (instead of Hankel tensor) and consequently shows a fast implementation through FFT, but the nuclear norm of circulant tensor would overly emphasize the spatiotemporal correlations in traffic data. Observing these results, HTF out-

Table 4.6 Imputation performance (in MAPE/RMSE) of HTF (`circ`) with $\tau_1 \in \{2, 3, 4\}$ on the Seattle freeway traffic speed dataset. Note that the best results are highlighted in bold fonts.

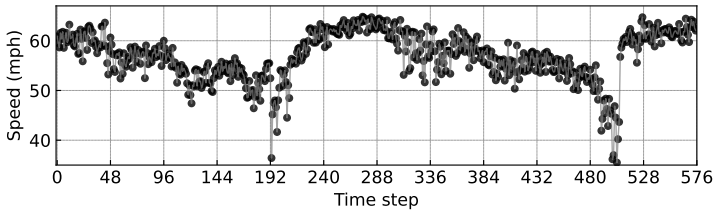
Window length	Missing rate			
	80%	85%	90%	95%
$\tau_1 = 2$	**6.21/3.88**	**6.51/4.06**	**6.98/4.30**	8.02/4.84
$\tau_1 = 3$	6.36/3.94	6.61/4.09	7.05/4.32	**7.97/4.77**
$\tau_1 = 4$	6.52/4.01	6.77/4.15	7.17/4.35	8.02/**4.77**

Table 4.7 Performance comparison (in MAPE/RMSE) for imputation tasks on the Seattle freeway traffic speed dataset. Note that the best results are highlighted in bold fonts.

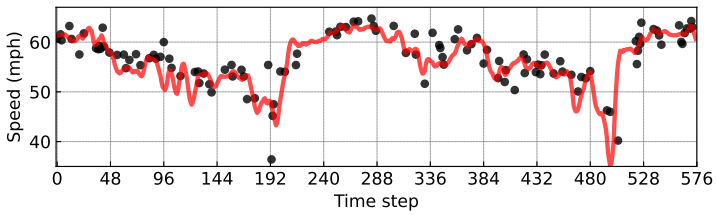
Model	Missing rate			
	80%	85%	90%	95%
HTF (`circ`)	**6.21/3.88**	**6.51/4.06**	**6.98/4.30**	**8.02/4.84**
HTF (`dense`)	8.75/5.16	9.86/5.76	9.24/5.36	9.89/5.70
HTF (`diag`)	7.25/4.33	7.93/4.66	8.61/4.96	9.25/5.20
LATC	6.50/4.00	6.90/4.21	7.47/4.51	8.75/5.05
LRTC-TNN	6.97/4.24	7.43/4.43	8.19/4.81	9.60/5.55
LCR	6.75/4.15	7.31/4.38	7.96/4.71	9.78/5.39
SMF	6.44/3.99	6.90/4.22	7.68/4.59	9.25/5.25
BTMF	6.85/4.17	7.36/4.42	8.13/4.79	9.63/5.48

performs LCR, though HTF does not have any regularization terms for local trend modeling like LCR. The comparison between HTF and SMF shows the flexibility of Hankelization over the smoothing regularization, in the meanwhile demonstrating the importance of Hankelization for modeling spatiotemporal data. Overall, The proposed HTF model outperforms all the baseline models, showing to be well-suited to the extreme missing data scenarios.

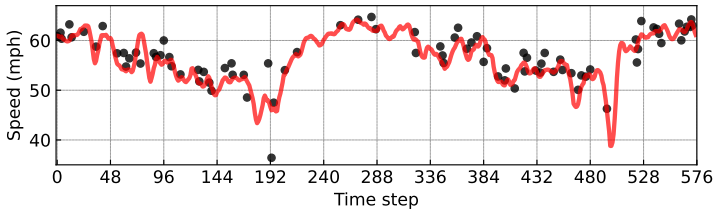
Figure 4.10 shows an example time series selected from the Seattle freeway traffic speed data in which Figure 4.10(a) presents the ground truth time series without missing values and the remaining ones show the reconstructed time series. In this case, we visualize the reconstruction results produced by HTF at different missing rates. As can be seen, HTF can achieve accurate imputation and learn the true time series trends from partial observations even with severe missing data (e.g., see Figure 4.10(e) for 95% missing rate).



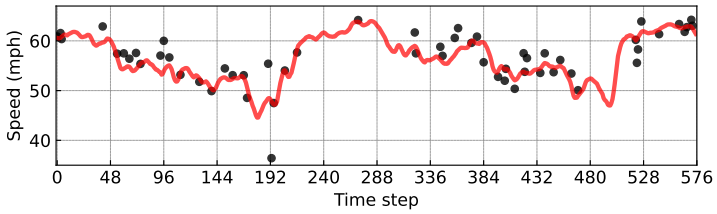
(a) Original time series.



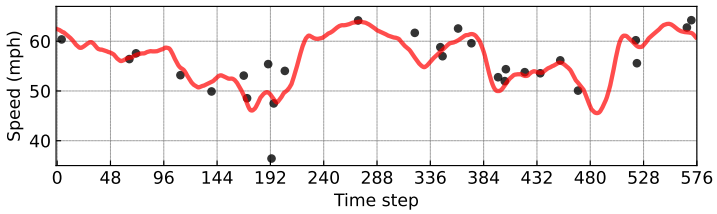
(b) 80% missing rate.



(c) 85% missing rate.



(d) 90% missing rate.



(e) 95% missing rate.

Figure 4.10 Reconstructed time series achieved by HTF on the Seattle freeway traffic speed dataset. The example corresponds to the detector #1 and 5th-6th days (i.e., 576 time steps). Black dots indicate the partially observed traffic speed data, while red curves indicate the imputed traffic speed values.

4.5 Traffic Forecasting From Sparse Data

In this section, we evaluate the performance of NoTMF on two large-scale traffic speed datasets collected from urban road networks—the NYC Uber movement speed dataset and the Seattle Uber movement speed dataset.

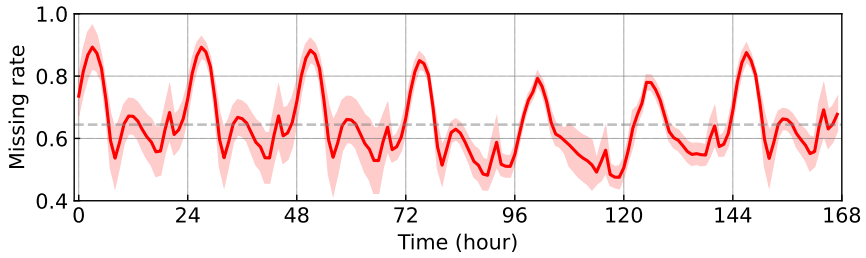
4.5.1 Data and Experiment Setup

The data is taken from the Uber movement project.[5] The project provides data and tools for cities to understand and address urban transportation problems and challenges in a data-driven fashion. Uber movement speed data measure the average speed on a given road segment in urban areas for each hour of each day over the study period, and they are representative examples for demonstrating both high-dimensionality and sparsity issues in real-world traffic data. Figure 4.11 shows two cases of Uber movement speed data in NYC and Seattle, USA, respectively. The high-dimensional and sparse data inevitably result in difficulties in analyzing traffic states or supporting data-driven city planning and decision-making.

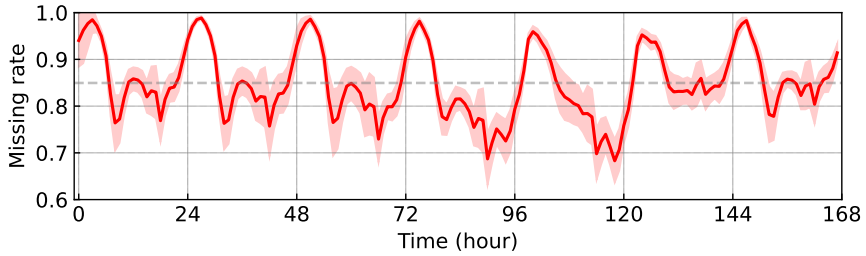
We use the NYC Uber movement speed data collected from 98,210 road segments during the first ten weeks of 2019 for the following experiments, and this dataset is of size 98210×1680. Due to the mechanism of insufficient sampling of ridesharing vehicles on urban road networks, the dataset contains 66.56% missing values. It can be seen that the time-evolving missing rates have periodicity patterns. At midnight, the missing rate can even reach $\sim$90%. This is due to limited ridesharing vehicles on the road network during the midnight. At some specific hours, only a small fraction of road segments have speed observations. Another dataset is the Seattle Uber movement speed dataset, which covers the hourly movement speed data from 63,490 road segments during the first ten weeks of 2019. The data matrix is of size 63490×1680 and contains 87.35% missing values, showing that the sparsity issue in this dataset is more difficult to handle than the NYC Uber movement speed dataset.

Remark 10. *Recall that matrix factorization fails to resolve the column-wise missing in the data matrix, the TMF framework reinforces the capability of matrix factorization for correlating the data columns along the temporal dimension. As shown in Figure 4.11, traffic state data at midnight are extremely sparse, and the standard matrix factorization would suffer from biased estimation. Therefore, the TMF framework demands appropriate temporal modeling techniques.*

[5] `https://movement.uber.com/`



(a) NYC Uber hourly movement speed data.



(b) Seattle Uber hourly movement speed data.

Figure 4.11 The missing rates of Uber movement speed data aggregated per week over the whole year of 2019. The red curve shows the average missing rates over all 52 weeks. The red area shows the standard deviation of missing rates in each hour over 52 weeks. The 168 time steps refer to 168 hours of Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday, and Monday. (a) The dataset has 98,210 road segments, and the whole missing rate is 64.43%. (b) The dataset has 63,490 road segments, and the whole missing rate is 84.95%.

Figure 4.12 shows the complementary histogram of the percentage of observed values in the NYC dataset. As can be seen, a large fraction of road segments only have very limited observations. This is especially interesting, as it shows that half of the road segments have less than 20% movement speed observations. Only 17% road segments have more than 80% movement speed observations. We can also see examples of incomplete movement speed observations from Figure 4.13.

In our experiment, we evaluate the model with 8-week data (from January 1st to February 25th) as the training set, one-week data (February 26th to March 4th) for validation, and one-week data (March 5th to March 11th) as the test set. We use rolling forecasting with time horizons $\delta = 1, 2, 3, 6$, corresponding to δ-hour-ahead forecasting. To confirm the importance of seasonal differencing and the advantage of the solution algorithm with alternating minimization and conjugate gradient, we evaluate the proposed NoTMF model and baseline models in the TMF framework. We consider the following TMF models in the literature:

- **TRMF** [6]: TRMF achieves temporal modeling on latent temporal factors by applying

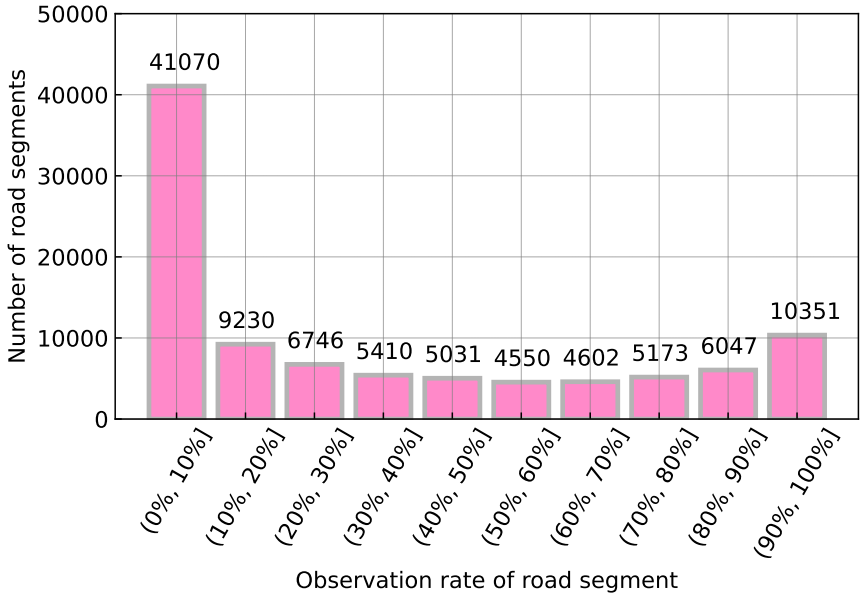


Figure 4.12 Histogram of observation rate of road segment in the NYC Uber movement speed dataset. Only a small fraction of road segments have an observation rate greater than 50%, i.e., 30723/98210 $\approx$ 31%. For the observation rates greater than 20% and 80%, there are about 49% and 17% of road segments, respectively.

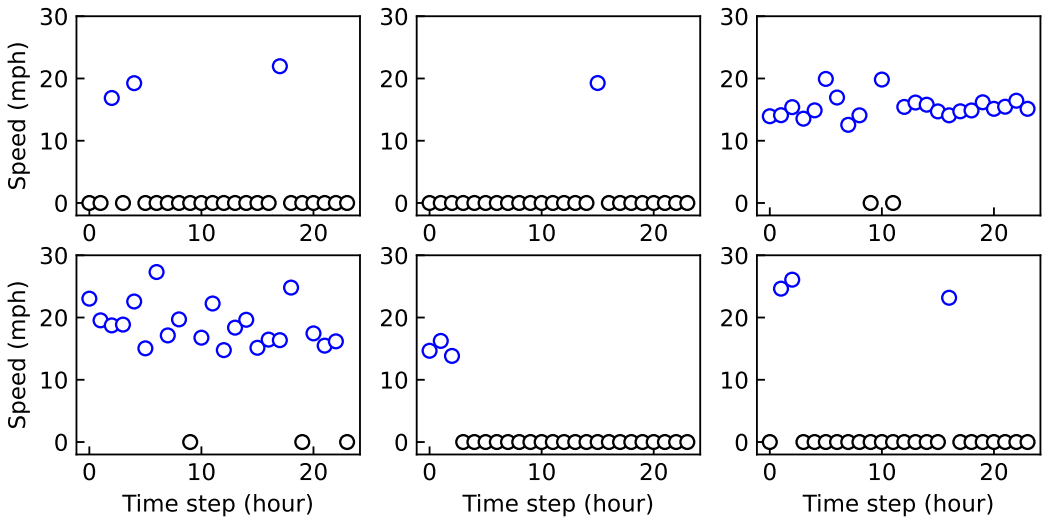


Figure 4.13 Movement speed of 6 road segments of January 1, 2019 (24 hours) in NYC. Blue points indicate the observed speed from the movement dataset, while black points indicate missing values (set to 0).

univariate autoregressive process.

- **BTMF** [1]: BTMF is a fully Bayesian TMF model with Gaussian assumption on

Eq. (3.4), which is solved by using MCMC algorithm. This model has been empirically demonstrated to be state-of-the-art against some baseline models (e.g., matrix/tensor factorization) for both time series imputation and forecasting on sparse data.

- **BTRMF** [1]: Bayesian TRMF is a fully Bayesian treatment for the TRMF model with Gaussian assumption.

4.5.2 Forecasting Performance

We evaluate the task of traffic time series forecasting in the presence of missing data on the NYC and Seattle Uber movement speed datasets. The preliminary experiment evaluates the NoTMF model with different ranks $R = 5, 10, 15, 20, 25, 30$. To set the hyperparameters $\{\gamma, \rho\}$, we first prescribe a collection of hyperparameters $\gamma \in \{10^2, 10^1, 10^0, 10^{-1}, 10^{-2}\}$ and $\rho \in \{10\gamma, 5\gamma, \gamma, 5\times 10^{-1}\gamma, 10^{-1}\gamma\}$ and evaluate the NoTMF with these settings on the training and validation sets. Then, we find the best hyperparameter pair for testing the model. In this case, the hyperparameters of NoTMF are properly given by $\gamma = 1$ and $\rho = 5$. As shown in Figure 4.14, NoTMF with a larger rank essentially provides higher accuracy but such improvement becomes marginal when $R > 15$. In the following experiments, we choose $R = 10$ for a good balance between performance and computational cost.

Table 4.8 shows the forecasting performance of NoTMF and baseline models on the NYC dataset. We summarize the following findings from the results:

- With the increase in forecasting time horizons, the forecasting errors of all models increase. For each time horizon, as the order becomes greater, the forecasting performance of NoTMF and TRMF is improved.

- The NoTMF models with different differencing operations demonstrate a significant improvement over TRMF with respect to forecasting accuracy. In contrast to the univariate autoregressive process, there is a clear benefit from temporal modeling with a multivariate VAR process.

- On this dataset, we have two choices for setting the season, one is $m = 24$, corresponding to the daily differencing; another is $m = 168$, corresponding to the weekly differencing. For both settings, NoTMF can achieve competitively accurate forecasts.

- NoTMF-1st represents the NoTMF model with both seasonal differencing and first-order differencing (see the VAR process in Eq. (3.14)), and it performs better than NoTMF ($m = 168$) when the order d is relatively small.

Table 4.8 Forecasting performance (in MAPE/RMSE) on the NYC movement speed dataset. The rolling forecasting tasks include different time horizons, i.e., $\delta = 1, 2, 3, 6$. We consider the rank as $R = 10$. After the cross validation for finding the best (γ, ρ), we set (γ, ρ) as $(1, 5)$ for NoTMF and TRMF. Note that the best results are highlighted in bold fonts.

δ	d	NoTMF ($m = 24$)	NoTMF ($m = 168$)	NoTMF-1st ($m = 168$)	TRMF	BTMF	BTRMF
1	1	13.63/2.88	13.53/2.86	**13.45**/**2.85**	14.50/3.12	14.94/3.13	15.93/3.33
	2	13.47/**2.84**	**13.41**/**2.84**	13.42/**2.84**	14.14/3.05	15.70/3.41	15.90/3.35
	3	13.46/2.84	**13.39**/**2.83**	13.43/2.84	13.87/2.96	15.80/3.34	16.08/3.43
	6	13.41/**2.83**	**13.39**/**2.83**	13.41/**2.83**	14.00/2.98	15.45/3.27	16.26/3.48
2	1	13.91/2.96	13.76/2.94	**13.70**/**2.92**	15.85/3.43	15.33/3.21	16.85/3.56
	2	13.77/2.92	**13.63**/**2.89**	13.72/2.92	15.04/3.31	15.87/3.32	17.27/3.71
	3	13.72/2.91	**13.61**/**2.89**	13.73/2.92	15.25/3.36	15.69/3.33	17.24/3.74
	6	13.59/**2.87**	**13.57**/2.88	13.68/2.91	14.92/3.24	15.91/3.39	18.18/3.97
3	1	14.30/3.05	14.06/3.02	**14.02**/**3.00**	17.52/3.83	15.86/3.32	18.61/3.91
	2	14.01/2.98	**13.84**/**2.94**	13.96/2.98	17.32/4.00	16.30/3.40	18.90/4.10
	3	13.95/2.97	**13.79**/**2.93**	13.98/2.98	16.91/3.71	16.56/3.49	18.68/4.05
	6	13.78/**2.92**	**13.73**/**2.92**	13.91/2.96	16.72/3.65	15.49/3.27	20.45/4.66
6	1	**14.61**/**3.11**	14.67/3.20	14.98/3.32	21.20/4.70	15.99/3.32	22.40/4.69
	2	**14.30**/**3.03**	14.33/3.09	14.90/3.28	20.87/5.01	16.04/3.33	23.56/5.63
	3	**14.26**/**3.03**	14.28/3.09	14.86/3.26	20.08/4.65	15.67/3.28	24.27/5.72
	6	**14.06**/**2.97**	14.16/3.06	14.80/3.23	20.40/4.35	16.38/3.50	26.34/6.60

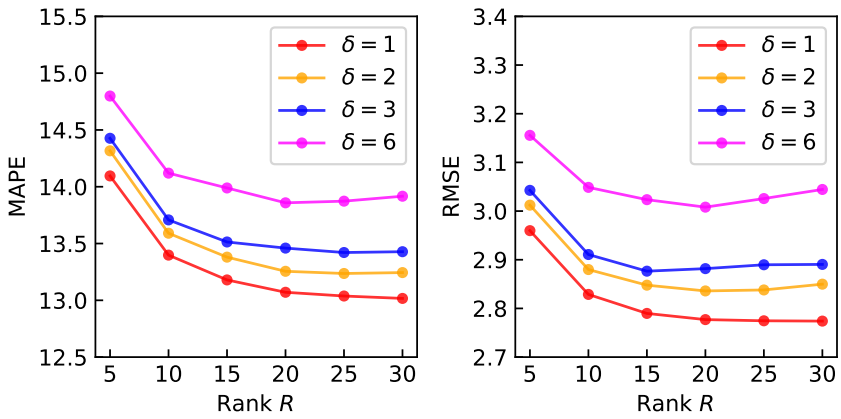


Figure 4.14 Performance of NoTMF with season-168 differencing (i.e., $m = 168$) and order $d = 6$ on the NYC dataset. The validated parameters are $\gamma = 1$ and $\rho = 5$.

Figure 4.17 shows the predicted time series and the partially observed speed values of some road segments. It demonstrates that the forecasts produced by NoTMF are consistent with the temporal patterns underlying partially observed time series. Thus, the NoTMF model can extract implicit temporal patterns (see Figure 4.18 for instance) from partially observed data and in the meanwhile perform forecasting.

On the Seattle dataset, we perform forecasting and compare the proposed NoTMF with some baseline models as shown in Table 4.9. Of these results, it seems that the VAR in the NoTMF and BTMF models helps produce steady forecasting performance with the increase of forecasting time horizons, while the univariate autoregression in the TRMF and BTRMF models fails to maintain the performance of the same level as the increase of forecasting time horizons. Notably, NoTMF consistently outperforms other models as shown in Table 4.9. We set NoTMF with both daily differencing (i.e., $m = 24$) and weekly differencing (i.e., $m = 168$), and it shows that the weekly differencing is superior to the daily differencing. Therefore, proper seasonal differencing in NoTMF is important for improving the forecasting performance.

Figure 4.15 and 4.16 show the speed distribution of the ground truth data versus the forecasts achieved by NoTMF on NYC and Seattle datasets, respectively. For each missing rate range, we group the speed observations of some road segments whose missing rate of the test set is in that range. For road segments with relatively lower missing rates, e.g., $(0, 10\%]$, the histograms present two peaks. For example on the Seattle dataset (see Figure 4.16(a-b)), one peak is around the speed of 20mph, while another is around the speed of 60mph. This implies that for the road segments with relatively complete speed observations, both lower speeds of

Table 4.9 Forecasting performance (MAPE/RMSE) on the Seattle movement speed dataset. The rolling forecasting tasks include different time horizons, i.e., $\delta = 1, 2, 3, 6$. We consider the rank as $R = 10$. After the cross validation for finding best (γ, ρ), we set (γ, ρ) as $(1, 5)$ for NoTMF and TRMF. Note that the best results are highlighted in bold fonts.

δ	d	NoTMF ($m = 24$)	NoTMF ($m = 168$)	NoTMF-1st ($m = 168$)	TRMF	BTMF	BTRMF
1	1	10.45/3.32	**10.26**/3.22	**10.26/3.21**	11.58/3.79	12.23/3.89	12.52/4.01
	2	10.53/3.34	10.29/3.23	**10.23/3.21**	10.92/3.51	12.95/4.18	13.16/4.31
	3	10.42/3.30	10.30/3.22	**10.25/3.21**	10.86/3.47	12.96/4.22	13.89/4.64
	6	10.50/3.32	**10.21/3.21**	10.27/3.22	10.99/3.51	12.91/4.18	13.90/4.67
2	1	10.90/3.55	10.32/3.25	**10.25/3.23**	12.07/4.02	12.74/4.06	13.31/4.32
	2	10.90/3.52	10.31/3.24	**10.25/3.23**	12.59/4.24	13.68/4.45	13.44/4.43
	3	10.81/3.49	10.31/3.24	**10.27/3.23**	12.01/3.96	13.55/4.46	13.66/4.56
	6	10.57/3.38	**10.25/3.23**	10.27/**3.23**	12.18/3.98	13.56/4.42	14.67/4.92
3	1	11.27/3.71	**10.41/3.29**	**10.41/3.29**	13.47/4.62	13.16/4.15	14.01/4.52
	2	11.26/3.71	**10.30/3.27**	10.34/**3.27**	14.48/5.19	13.63/4.37	14.39/4.76
	3	11.11/3.62	**10.35/3.28**	10.38/**3.28**	14.04/4.83	13.76/4.42	14.67/4.84
	6	10.96/3.55	**10.30/3.26**	**10.30/3.26**	13.32/4.51	13.28/4.29	15.64/5.31
6	1	11.88/3.97	10.63/3.43	**10.60/3.42**	15.59/5.32	13.63/4.30	16.39/5.28
	2	11.58/3.83	**10.55/3.40**	10.56/**3.40**	18.66/7.20	13.27/4.19	16.77/5.58
	3	11.54/3.81	10.57/3.39	**10.53/3.38**	17.94/6.32	13.88/4.36	17.35/5.70
	6	11.27/3.70	10.53/**3.35**	**10.50/3.35**	15.12/5.24	13.30/4.24	16.63/5.62

the congested traffic (possibly during peak hours) and higher speeds of the free-flow traffic (possibly during off-peak hours) are measured, revealing the bi-mode traffic states. Of these results, NoTMF can accurately forecast both congested and free-flow traffic speeds. With the increase in missing rates, the phenomena of bi-mode traffic states are weakened. As a whole, we can summarize from Figure 4.15 and 4.16 that the forecasts produced by NoTMF are accurate when compared to the ground truth data.

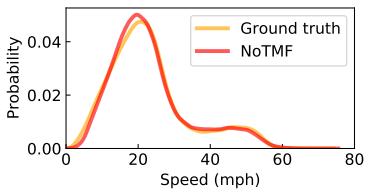
4.5.3 Nonstationarity Analysis

Nonstationarity is an important characteristic in real-world traffic time series data. Figure 4.18 illustrates some temporal factors of NoTMF with the following setting: $d = 1$, $R = 10$, and $m = 24$. As can be seen, temporal factors #3, #5, #6, #7, #8, and #9 show clear seasonality and periodicity. The long-term season is associated with the week, i.e., $7 \times 24 = 168$ time steps (hours), while the short-term season is associated with the day, i.e., 24 time steps (hours). For other temporal factors, they also show clear trends with weak seasonality and periodicity. Therefore, making use of seasonal differencing in nonstationary traffic state data can benefit the forecasting performance (as noted in Table 4.8 and 4.9). Our results further demonstrate the effectiveness of seasonal differencing in characterizing temporal process in the factor matrix $\boldsymbol{X}$.

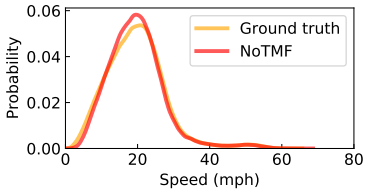
In particular, we visualize the coefficient matrix of NoTMF with the rank $R = 10$ and the order $d = 3$ in Figure 4.19. In these heatmaps, each is of size 10×10 and the diagonal entries represent the auto-correlations of each time series (i.e., temporal factor). In contrast to the NoTMF model without differencing operation, it demonstrates that the coefficient matrices of NoTMF with seasonal differencing show weak correlations. This implies the importance of seasonal differencing for stationarizing the time series and eliminating the correlations.

4.6 Concluding Remark

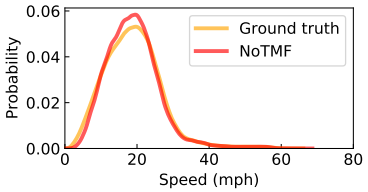
In this chapter, we evaluate the proposed imputation models presented in Chapter 3 on different spatiotemporal traffic data imputation and forecasting tasks. Regarding the imputation tasks, we consider univariate traffic time series, multivariate traffic time series, and high-dimensional traffic data with different missing data patterns. Regarding the forecasting task, we resolve the sparse urban traffic state forecasting problem in the presence of sparse data. These real-world tasks from transportation systems are meaningful for modeling spatiotemporal traffic data and supporting analysis and decision-making.



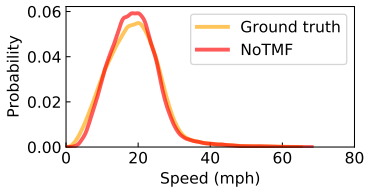
(a) Missing rate between 0 and 10%; 12,369 road segments.



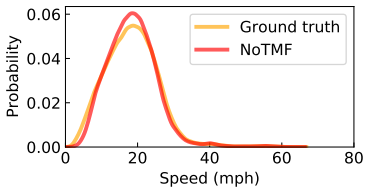
(b) Missing rate between 10% and 20%; 7,833 road segments.



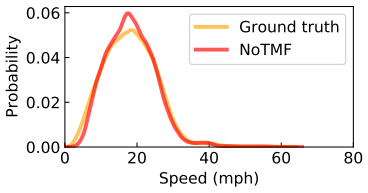
(c) Missing rate between 20% and 30%; 6,229 road segments.



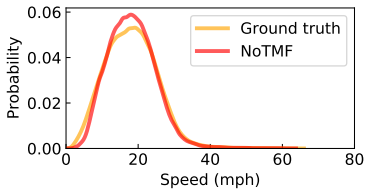
(d) Missing rate between 30% and 40%; 5,373 road segments.



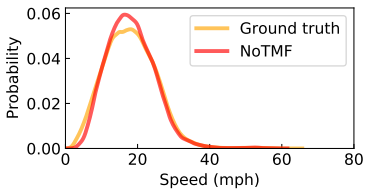
(e) Missing rate between 40% and 50%; 5,608 road segments.



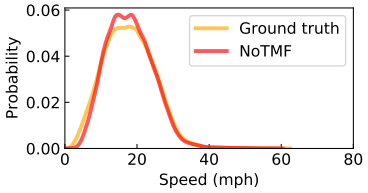
(f) Missing rate between 50% and 60%; 4,981 road segments.



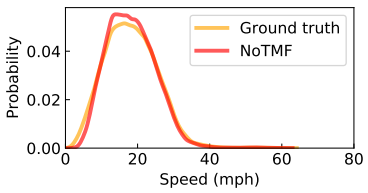
(g) Missing rate between 60% and 70%; 5,833 road segments.



(h) Missing rate between 70% and 80%; 6,633 road segments.

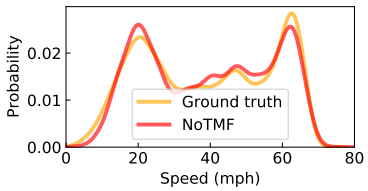


(i) Missing rate between 80% and 90%; 9,804 road segments.

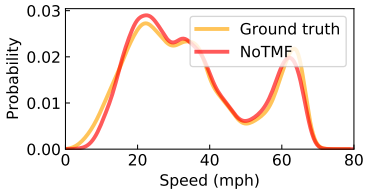


(j) Missing rate between 90% and 100%; 33,547 road segments.

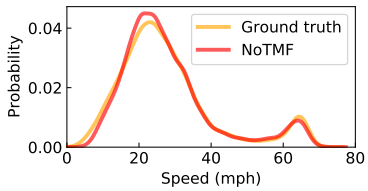
Figure 4.15 The histogram of ground truth data and forecasts achieved by NoTMF with $\delta = 6$ and $d = 6$ in the test set of the NYC dataset. The missing rate implies the ratio of missing values of road segments in the test set.



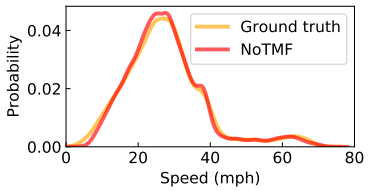
(a) Missing rate between 0 and 10%; 978 road segments.



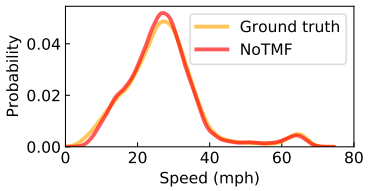
(b) Missing rate between 10% and 20%; 1,557 road segments.



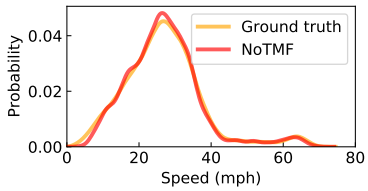
(c) Missing rate between 20% and 30%; 1,476 road segments.



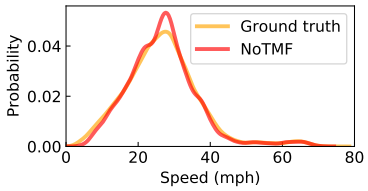
(d) Missing rate between 30% and 40%; 1,738 road segments.



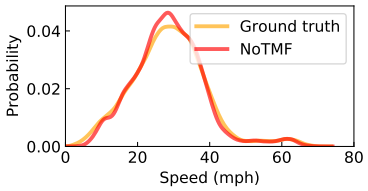
(e) Missing rate between 40% and 50%; 1,849 road segments.



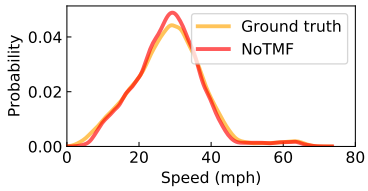
(f) Missing rate between 50% and 60%; 1,649 road segments.



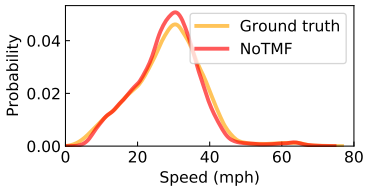
(g) Missing rate between 60% and 70%; 2,052 road segments.



(h) Missing rate between 70% and 80%; 3,225 road segments.



(i) Missing rate between 80% and 90%; 6,392 road segments.



(j) Missing rate between 90% and 100%; 42,574 road segments.

Figure 4.16 The histogram of ground truth data and forecasts achieved by NoTMF with $\delta = 6$ and $d = 6$ in the test set of the Seattle dataset. The missing rate implies the ratio of missing values of road segments in the test set.

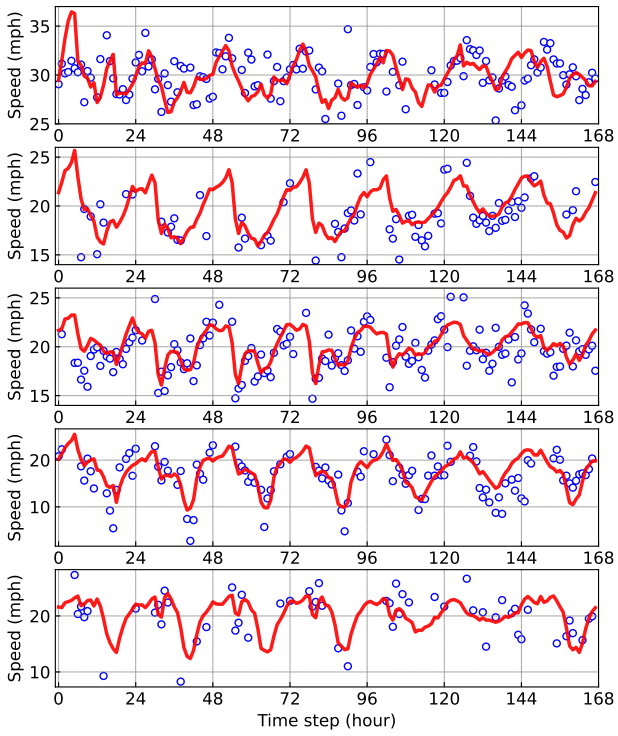


Figure 4.17 Five examples (corresponding to five road segments) for showing the forecasting results of the NoTMF model with time horizon $\delta = 6$. The red curves indicate the forecasts in the testing week, while the blue scatters indicate the ground truth speed data.

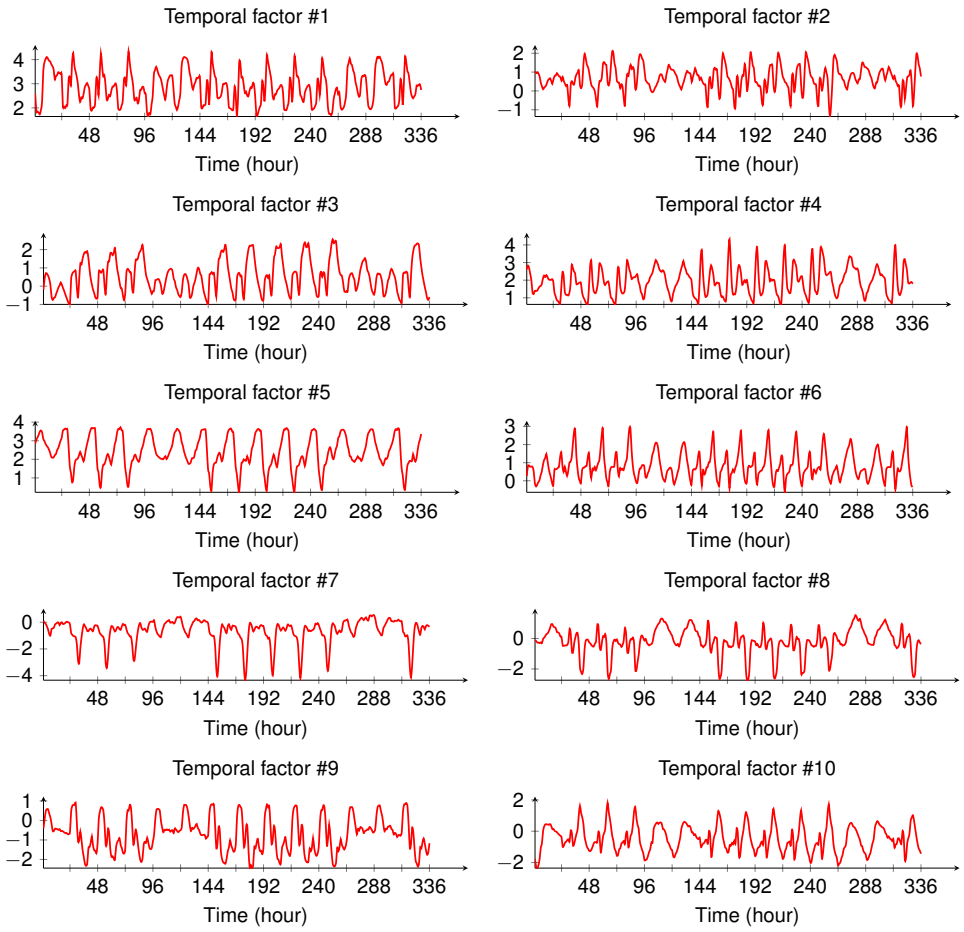
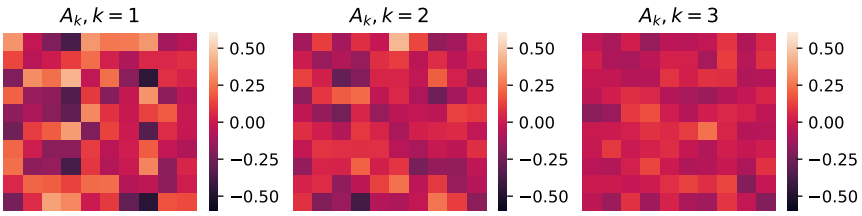
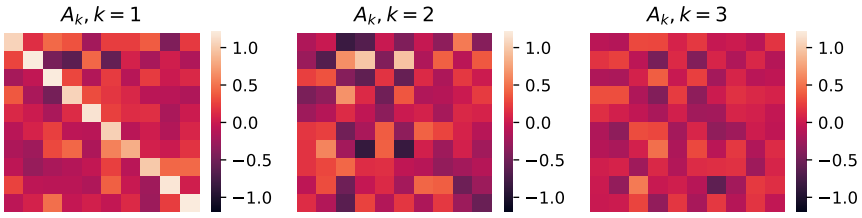


Figure 4.18 Temporal factors of NoTMF model ($R = 10$) on NYC movement speed data. The order and season of NoTMF are set as $d = 1$ and $m = 24$, respectively. The subfigures only show the temporal factors in the first two weeks.



(a) Coefficient matrices in NoTMF with seasonal differencing.



(b) Coefficient matrices in NoTMF without differencing operations.

Figure 4.19 Heatmap of coefficient matrices $\{\boldsymbol{A_k}\}$ in NoTMF. Note that we set the rank as $R = 10$ and the order as $d = 3$ for both models.

CHAPTER 5 CONCLUSION

This thesis shows several scientific contributions related to the development of low-rank matrix and tensor models and applications to spatiotemporal traffic data modeling. Some of them are direct contributions to the spatiotemporal modeling techniques on the basis of low-rank matrix and tensor models, whereas others make use of these proposed models to address real-world traffic data imputation and forecasting problems. In this chapter, we summarize the main findings and describe the limitations and future directions in which this work can be further extended.

5.1 Work Summary

In this thesis, we investigate some important scientific problems of spatiotemporal data modeling, including traffic data imputation and forecasting on sparse traffic data. Despite the vast body of literature on time series analysis from many scientific areas, most existing time series models require complete time series data as input. For performing reliable time series forecasting in the presence of missing data, a simple and natural solution is to adopt a two-stage approach: first applying imputation algorithms to fill in those missing entries, and then performing predictions based on the reconstructed time series. This simple two-stage approach has been used in a wide range of real-world applications [66]; however, by applying imputation first, the prediction task would suffer from accumulated errors resulting from the imputation algorithm. In this work, NoTMF provides a flexible framework for sparse time series forecasting. The model learns latent temporal dynamics from sparse time series directly and computes the forecasts with dictionary learning efficiently.

For performing reliable imputation on partially observed traffic data, we advance the low-rank matrix/tensor models by incorporating the temporal correlations. We summarize the features of the proposed low-rank matrix/tensor models in Section 3.5. We can also see from the experiments that:

- Both global low-rank property and local time series trends of traffic data are important for modeling partially observed traffic data. Low-rank methods such as matrix/tensor factorization allow one to characterize the low-frequency information of traffic data, while temporal modeling techniques such as autoregression and temporal smoothing make the time series locally consistently and remove the data noises.

- Introducing the domain knowledge of spatiotemporal traffic data to the modeling pro-

cess is critical. For example, traffic data represent human mobility behavior with daily and weekly rhythm, thus seasonal differencing and circulant matrix/tensor work well in the imputation and forecasting tasks. Another example is the local consistency of traffic data, which implies the meaningfulness of temporal modeling via the use of Hankel tensor.

5.2 Limitations

The thesis presents a sequence of low-rank matrix and tensor methods for traffic data with certain modeling purposes and application scenarios. However, there are some limitations:

- NoTMF in Section 3.1. The NoTMF model is efficient for learning from sparse traffic state data and computing the forecasts with a well-designed forecasting mechanism. However, the model can still produce estimation biases when some data columns (i.e., corresponding to the time steps at night) have limited observations. In our case, we care more about the forecasts of congested traffic states that are usually generated during rush hours, thus the estimation biases of traffic states at night do not hurt the forecasting model severely. But this is an unsolved issue in NoTMF. Another issue is that NoTMF cannot build nonlinear dependencies on spatiotemporal data, making it not flexible enough to capture the complex spatiotemporal dependencies in traffic data.

- LATC in Section 3.2. The LATC model utilizes univariate autoregression to characterize the time series correlations in traffic data. By doing so, the resulting temporal variation regularization can reinforce the modeling capability of low-rank tensor models, but the univariate autoregression cannot represent the interactions among different sequences of time series.

- LCR in Section 3.3. The LCR takes a convex optimization in which the objective function includes the circulant matrix nuclear norm and temporal regularization. According to the properties of the circulant matrix and circular convolution, it is not hard to present a very efficient solution algorithm through FFT. However, both the circulant matrix and (circulant) Laplacian kernel require us to draw the connection between data points x_1 (i.e., the first entry of the time series) and x_T (i.e., the last entry of time series). The underlying circulant structure still hurts the imputation performance because it does not always hold in real-world cases. Generally speaking, the first entries of time series are usually very different from the last entries of time series.

- HTF in Section 3.4. One great concern of memory-efficient HTF is the computational

cost despite the remarkable reduction of memory consumption. It is still necessary to develop fast Hankel tensor completion algorithms. As we know, building temporal correlations is important for both imputation and forecasting, evaluating local trend (e.g., first-order differencing) and nonlocal trend (e.g., seasonal differencing) modeling in HTF are still necessary.

- Experiments in Chapter 4. We evaluate the proposed models on some real-world traffic datasets, but it is still necessary to consider the impacts of different levels of data noises and realistic missing data patterns, which might be more complicated than the above-mentioned experiments. In addition, it is necessary to evaluate the proposed models on other traffic data, e.g., traffic accident data (crash numbers) and cycling data.

5.3 Future Research

5.3.1 Algorithm Development

There are several directions to advance the aforementioned algorithms.

- We can extend the proposed low-rank matrix/tensor factorization framework to account for spatial dependencies/correlations by incorporating tools such as spatial autoregression and GP structures. For example, NoTMF and LATC can be reinforced by adding graph regularization on the spatial dimension of traffic data (see e.g., [6,41]). Given an adjacency matrix that represents the spatial structure of traffic data, one can construct a Laplacian matrix $\boldsymbol{L}$ and build the regularization term.

- To overcome the impacts of outliers, the proposed matrix/tensor completion framework can also be adapted to robust models. The error matrix/tensor can be modeled by imposing an ℓ_1-norm (see e.g., robust tensor principal component analysis in [99]). By doing so, the model can also be used to detect outliers from partial observations.

- The circulant operation in LCR assumes that the start data points and the end data points are connected, which is a disadvantage in real-world data analysis. To overcome this issue on any time series $\boldsymbol{x} \in \mathbb{R}^T$, we can construct the following vector:

$$\boldsymbol{x}_{\text{new}} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{J}_T \boldsymbol{x} \end{bmatrix} = (x_1, \cdots, x_T, x_T, \cdots, x_1)^\top \in \mathbb{R}^{2T}, \tag{5.1}$$

where $\boldsymbol{J}_T \in \mathbb{R}^{T \times T}$ is the exchange matrix whose antidiagonal entries are one and other entries are zero. Introducing Laplacian kernel $\boldsymbol{\ell}$ to the concatenated vector $\boldsymbol{x}_{\text{new}}$ can

avoid the correlations between start and end data points. To eliminate correlations between the start data points and the end data points in our model, we introduce a $2N$-by-$2T$ block matrix (i.e., with four blocks) which flips the original matrix $\boldsymbol{Y}$, see Fig. 5.1. This matrix can be regarded as the input into LCR-2D and CircNNM. The model results are constructed by averaging the blocks according to the flipping operation.

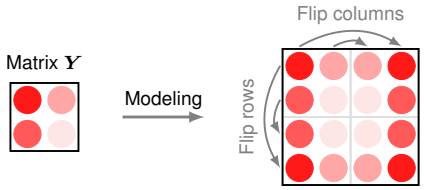


Figure 5.1 Constructing the matrix that flips the original matrix $\boldsymbol{Y}$ along rows and columns simultaneously. This operation can prevent the LCR-2D model from misleading values on the border rows and columns.

- The Laplacian kernel in LCR models takes the same weights for the correlated data points in the case of kernel size $\tau > 1$. A more reasonable setting could be using Gaussian kernels, demonstrating different weight values for the correlated data points.
- We can integrate recent advances in deep learning to better capture the complex and nonlinear dynamics of spatiotemporal data (see e.g., [100, 101]).

5.3.2 Traffic Flow Modeling

Advanced information systems provide great opportunities for approaching big data in transportation. In recent years, another trend of emerging technologies such as autonomous vehicles and connected autonomous vehicles also highlights the importance of data and algorithms. All of these require us to utilize multi-source big traffic data for developing solutions to transport modeling. Due to the availability of big traffic data and the development of machine learning, it is an opportunity to reformulate traffic flow modeling problems from a data-driven perspective. However, data behaviors and characteristics of traffic flow complicate the modeling process, posing both methodological and practical challenges. The following directions would be of great interest for the future research:

- **(Direction A)** High-resolution speed field reconstruction of vehicular traffic flow with multi-source data. It requires us to first represent multi-source traffic data onto the same data space. Then, the trajectory data collected from individual vehicles and the

traffic measurement data collected by fixed detectors are expected to be incorporated for reconstructing the high-resolution speed field of vehicular traffic flow.

- (**Direction B**) City-wide traffic state estimation using floating car data. Floating car data (e.g., collected from taxis/ridesharing vehicles) are important for monitoring urban traffic states. However, this kind of data usually suffers from insufficient sampling of floating cars in total traffic and low-resolution positioning information. With the great advances of the internet, positioning technologies, and connected vehicles, it is possible to gather high-resolution movement data of vehicles with very accurate positioning information. Thus, it is meaningful to take advantage of these data and find algorithmic solutions to urban traffic state estimation.

- (**Direction C**) Short-term traffic flow forecasting on the imbalanced and sparse data. Essentially, it requires us to handle imperfect data with relatively low data quality. For implementing traffic flow prediction, it demands us to find an efficient learning mechanism on the sparse inputs and in the meanwhile characterize the nonlinear behavior. TMF models presented above can provide a starting point, but the modeling capability (e.g., nonlinearity) should be reinforced in future studies.

Understanding traffic flow dynamics is a long-standing topic in transport modeling [3], it is meaningful for drawing connections among data, models, and simulation. This future research aims to establish efficient machine learning approaches for traffic flow modeling problems, as multi-source big traffic data are now available. The approaches are expected to bridge the gap between big traffic data and real-world transport applications, making transport systems more intelligent, i.e., really intelligent transportation systems. In addition, future research is also expected to bring fundamental research advances to the general field of spatiotemporal data modeling and promote its application to other domains such as geographic information systems.

REFERENCES

[1] X. Chen and L. Sun, "Bayesian temporal factorization for multidimensional time series prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 4659–4673, 2022.

[2] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: mining loop detector data," *Transportation Research Record*, vol. 1748, no. 1, pp. 96–102, 2001.

[3] M. Treiber and A. Kesting, "Traffic flow dynamics," *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*, pp. 983–1000, 2013.

[4] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: concepts, methodologies, and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 3, pp. 1–55, 2014.

[5] Y. Li and C. Shahabi, "A brief overview of machine learning methods for short-term traffic forecasting and future directions," *SIGSPATIAL Special*, vol. 10, no. 1, pp. 3–9, 2018.

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

[7] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[8] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013.

[9] T. Mallick, P. Balaprakash, E. Rask, and J. Macfarlane, "Transfer learning with graph neural networks for short-term highway traffic forecasting," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 10 367–10 374.

[10] Y. Zhu, Z. Li, H. Zhu, M. Li, and Q. Zhang, "A compressive sensing approach to urban traffic estimation with probe vehicles," *IEEE Transactions on Mobile Computing*, vol. 12, no. 11, pp. 2289–2302, 2012.

[11] L. Li, Y. Li, and Z. Li, "Efficient missing data imputing for traffic flow by considering temporal and spatial dependence," *Transportation research part C: emerging technologies*, vol. 34, pp. 108–120, 2013.

[12] M. T. Asif, N. Mitrovic, J. Dauwels, and P. Jaillet, "Matrix and tensor based methods for missing data estimation in large traffic networks," *IEEE Transactions on intelligent transportation systems*, vol. 17, no. 7, pp. 1816–1825, 2016.

[13] H. Tan, Y. Wu, B. Shen, P. J. Jin, and B. Ran, "Short-term traffic prediction based on dynamic tensor completion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 8, pp. 2123–2133, 2016.

[14] X. Chen, Z. He, and L. Sun, "A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation," *Transportation Research Part C: Emerging Technologies*, vol. 98, pp. 73 – 84, 2019.

[15] L. Sun and X. Chen, "Bayesian temporal factorization for multidimensional time series prediction," *arXiv preprint arXiv:1910.06366*, 2019.

[16] J. Yu, M. E. Stettler, P. Angeloudis, S. Hu, and X. M. Chen, "Urban network-wide traffic speed estimation with massive ride-sourcing gps traces," *Transportation Research Part C: Emerging Technologies*, vol. 112, pp. 136–152, 2020.

[17] F. Bashir and H.-L. Wei, "Handling missing data in multivariate time series using a vector autoregressive model-imputation (var-im) algorithm," *Neurocomput.*, vol. 276, no. C, p. 23–30, Feb. 2018.

[18] M. T. Asif, N. Mitrovic, L. Garg, J. Dauwels, and P. Jaillet, "Low-dimensional models for missing data imputation in road networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on.* IEEE, 2013, pp. 3527–3531.

[19] L. Qu, L. Li, Y. Zhang, and J. Hu, "Ppca-based missing data imputation for traffic flow volume: A systematical approach," *IEEE Transactions on intelligent transportation systems*, vol. 10, no. 3, pp. 512–522, 2009.

[20] B. Ran, H. Tan, Y. Wu, and P. J. Jin, "Tensor based missing traffic data completion with spatial–temporal correlation," *Physica A: Statistical Mechanics and its Applications*, vol. 446, pp. 54–63, 2016.

[21] X. Chen, J. Yang, and L. Sun, "A nonconvex low-rank tensor completion model for spatiotemporal traffic data imputation," *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102673, 2020.

[22] L. Li, X. Su, Y. Zhang, Y. Lin, and Z. Li, "Trend modeling for traffic time series analysis: An integrated study," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3430–3439, 2015.

[23] X. Chen, Z. He, and J. Wang, "Spatial-temporal traffic speed patterns discovery and incomplete data recovery via svd-combined tensor decomposition," *Transportation Research Part C: Emerging Technologies*, vol. 86, pp. 59–77, 2018.

[24] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[25] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[26] E. J. Candes and Y. Plan, "Matrix completion with noise," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 925–936, 2010.

[27] D. Zhang, Y. Hu, J. Ye, X. Li, and X. He, "Matrix completion by truncated nuclear norm regularization," in *2012 IEEE Conference on computer vision and pattern recognition*. IEEE, 2012, pp. 2192–2199.

[28] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He, "Fast and accurate matrix completion via truncated nuclear norm regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2117–2130, Sep. 2013.

[29] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 2862–2869.

[30] C. Lu, C. Zhu, C. Xu, S. Yan, and Z. Lin, "Generalized singular value thresholding," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

[31] Q. Yao, J. T. Kwok, T. Wang, and T.-Y. Liu, "Large-scale low-rank matrix learning with nonconvex regularizers," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 11, pp. 2628–2643, 2018.

[32] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208–220, 2013.

[33] S. Xue, W. Qiu, F. Liu, and X. Jin, "Low-rank tensor completion by truncated nuclear norm regularization," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 2600–2605.

[34] Z. Chen and A. Cichocki, "Nonnegative matrix factorization with temporal smoothness and/or spatial decorrelation constraints," *Laboratory for Advanced Brain Signal Processing, RIKEN, Tech. Rep*, vol. 68, 2005.

[35] Y. Wang, Y. Zhang, X. Piao, H. Liu, and K. Zhang, "Traffic data reconstruction via adaptive spatial-temporal correlations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1531–1543, 2018.

[36] X. Chen, Y. Chen, N. Saunier, and L. Sun, "Scalable low-rank tensor learning for spatiotemporal traffic data imputation," *Transportation research part C: emerging technologies*, vol. 129, p. 103226, 2021.

[37] J. Luttinen and A. Ilin, "Variational gaussian-process factor analysis for modeling spatio-temporal data," *Advances in neural information processing systems*, vol. 22, 2009.

[38] T. Zhou, H. Shan, A. Banerjee, and G. Sapiro, "Kernelized probabilistic matrix factorization: Exploiting graphs and side information," in *Proceedings of the 2012 SIAM international Conference on Data mining*. SIAM, 2012, pp. 403–414.

[39] M. Lei, A. Labbe, Y. Wu, and L. Sun, "Bayesian kernelized matrix factorization for spatiotemporal traffic data imputation and kriging," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[40] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning theory and kernel machines*. Springer, 2003, pp. 144–158.

[41] N. Rao, H.-F. Yu, P. K. Ravikumar, and I. S. Dhillon, "Collaborative filtering with graph information: Consistency and scalable methods," *Advances in neural information processing systems*, vol. 28, 2015.

[42] L. Xiong, X. Chen, T.-K. Huang, J. Schneider, and J. G. Carbonell, "Temporal collaborative filtering with bayesian probabilistic tensor factorization," in *Proceedings of the 2010 SIAM international conference on data mining*. SIAM, 2010, pp. 211–222.

[43] T. Yokota, B. Erem, S. Guler, S. K. Warfield, and H. Hontani, "Missing slice recovery for tensors using a low-rank model in embedded space," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8251–8259.

[44] F. Sedighin, A. Cichocki, T. Yokota, and Q. Shi, "Matrix and tensor completion in multiway delay embedded space using tensor train, with application to signal reconstruction," *IEEE Signal Processing Letters*, vol. 27, pp. 810–814, 2020.

[45] R. Yamamoto, H. Hontani, A. Imakura, and T. Yokota, "Fast algorithm for low-rank tensor completion in delay-embedded space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 2058–2066.

[46] G. Liu and W. Zhang, "Recovery of future data via convolution nuclear norm minimization," *IEEE Transactions on Information Theory*, 2022.

[47] G. Liu, "Time series forecasting via learning convolutionally low-rank models," *IEEE Transactions on Information Theory*, vol. 68, no. 5, pp. 3362–3380, 2022.

[48] P. C. Hansen, J. G. Nagy, and D. P. O'leary, *Deblurring images: matrices, spectra, and filtering*. SIAM, 2006.

[49] J. Ying, H. Lu, Q. Wei, J.-F. Cai, D. Guo, J. Wu, Z. Chen, and X. Qu, "Hankel matrix nuclear norm regularized tensor completion for n-dimensional exponential signals," *IEEE Transactions on Signal Processing*, vol. 65, no. 14, pp. 3702–3717, 2017.

[50] J.-F. Cai, T. Wang, and K. Wei, "Fast and provable algorithms for spectrally sparse signal reconstruction via low-rank hankel matrix completion," *Applied and Computational Harmonic Analysis*, vol. 46, no. 1, pp. 94–121, 2019.

[51] Y. Chen and Y. Chi, "Spectral compressed sensing via structured matrix completion," in *International Conference on Machine Learning*. PMLR, 2013, pp. 414–422.

[52] T. Yokota and H. Hontani, "Tensor completion with shift-invariant cosine bases," in *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2018, pp. 1325–1333.

[53] F. Sedighin and A. Cichocki, "Image completion in embedded space using multistage tensor ring decomposition," *Frontiers in Artificial Intelligence*, vol. 4, 2021.

[54] Q. Shi, J. Yin, J. Cai, A. Cichocki, T. Yokota, L. Chen, M. Yuan, and J. Zeng, "Block hankel tensor arima for multiple short time series forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 5758–5766.

[55] X. Wang, Y. Wu, D. Zhuang, and L. Sun, "Low-rank hankel tensor completion for traffic speed estimation," *arXiv preprint arXiv:2105.11335*, 2021.

[56] S. Zhang and M. Wang, "Correction of corrupted columns through fast robust hankel matrix completion," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2580–2594, 2019.

[57] L. Wang, S. Wu, T. Wu, X. Tao, and J. Lu, "Hkmf-t: Recover from blackouts in tagged time series with hankel matrix factorization," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[58] R. P. Velu, G. C. Reinsel, and D. W. Wichern, "Reduced rank models for multiple time series," *Biometrika*, vol. 73, no. 1, pp. 105–118, 1986.

[59] R. Velu and G. C. Reinsel, *Multivariate reduced-rank regression: theory and applications*. Springer Science & Business Media, 1998, vol. 136.

[60] S. Basu, G. Michailidis *et al.*, "Regularized estimation in sparse high-dimensional time series models," *The Annals of Statistics*, vol. 43, no. 4, pp. 1535–1567, 2015.

[61] D. Wang, Y. Zheng, H. Lian, and G. Li, "High-dimensional vector autoregressive time series modeling via tensor decomposition," *Journal of the American Statistical Association*, pp. 1–19, 2021.

[62] A. Carriero, G. Kapetanios, and M. Marcellino, "Structural analysis with multivariate autoregressive index models," *Journal of Econometrics*, vol. 192, no. 2, pp. 332–348, 2016.

[63] G. Koop, D. Korobilis, and D. Pettenuzzo, "Bayesian compressed vector autoregressions," *Journal of Econometrics*, vol. 210, no. 1, pp. 135–154, 2019.

[64] F. Han, H. Lu, and H. Liu, "A direct estimation of high dimensional stationary vector autoregressions," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 3115–3150, 2015.

[65] S. Basu, X. Li, and G. Michailidis, "Low rank and structured modeling of high-dimensional vector autoregressions," *IEEE Transactions on Signal Processing*, vol. 67, no. 5, pp. 1207–1222, 2019.

[66] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Scientific Reports*, vol. 8, no. 1, p. 6085, 2018.

[67] K. Takeuchi, H. Kashima, and N. Ueda, "Autoregressive tensor factorization for spatio-temporal predictions," in *IEEE International Conference on Data Mining*, 2017, pp. 1105–1110.

[68] J. H. Stock and M. W. Watson, "Dynamic factor models, factor-augmented vector autoregressions, and structural vector autoregressions in macroeconomics," in *Handbook of macroeconomics*. Elsevier, 2016, vol. 2, pp. 415–525.

[69] S. Gultekin and J. Paisley, "Online forecasting matrix factorization," *IEEE Transactions on Signal Processing*, vol. 67, no. 5, pp. 1223–1236, 2018.

[70] D. M. Dunlavy, T. G. Kolda, and E. Acar, "Temporal link prediction using matrix and tensor factorizations," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 2, pp. 1–27, 2011.

[71] K. Kawabata, S. Bhatia, R. Liu, M. Wadhwa, and B. Hooi, "Ssmf: Shifting seasonal matrix factorization," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[72] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[73] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 2nd ed. OTexts, 2018.

[74] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed. The Johns Hopkins University Press, 2013.

[75] S. Ravishankar and Y. Bresler, "Sparsifying transform learning with efficient optimal updates and convergence guarantees," *IEEE Transactions on Signal Processing*, vol. 63, no. 9, pp. 2389–2404, 2015.

[76] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 95–104.

[77] Y. Zhang and Z. Lu, "Penalty decomposition methods for rank minimization," in *Advances in Neural Information Processing Systems*, 2011, pp. 46–54.

[78] K. Chen, H. Dong, and K.-S. Chan, "Reduced rank regression via adaptive nuclear norm penalization," *Biometrika*, vol. 100, no. 4, pp. 901–920, 2013.

[79] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs: Graph fourier transform," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 6167–6170.

[80] J. Wright and Y. Ma, *High-dimensional data analysis with low-dimensional models: Principles, computation, and applications*. Cambridge University Press, 2022.

[81] H. Bristow, A. Eriksson, and S. Lucey, "Fast convolutional sparse coding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 391–398.

[82] B. Wohlberg, "Efficient convolutional sparse coding," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 7173–7177.

[83] F. Heide, W. Heidrich, and G. Wetzstein, "Fast and flexible convolutional sparse coding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5135–5143.

[84] S. L. Brunton and J. N. Kutz, *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

[85] J. Yang, W. Yin, Y. Zhang, and Y. Wang, "A fast algorithm for edge-preserving variational multichannel image restoration," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 569–592, 2009.

[86] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 171–184, 2012.

[87] E. Candes and B. Recht, "Exact matrix completion via convex optimization," *Communications of the ACM*, vol. 55, no. 6, pp. 111–119, 2012.

[88] O. Semerci, N. Hao, M. E. Kilmer, and E. L. Miller, "Tensor-based formulation and nuclear norm regularization for multienergy computed tomography," *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 1678–1693, 2014.

[89] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, "Tensor robust principal component analysis with a new tensor nuclear norm," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 4, pp. 925–938, 2019.

[90] E. O. Brigham, *The fast Fourier transform and its applications.* Prentice-Hall, Inc., 1988.

[91] N. Golyandina, V. Nekrutkin, and A. A. Zhigljavsky, *Analysis of time series structure: SSA and related techniques.* CRC press, 2001.

[92] I. V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

[93] Y. Chi, Y. M. Lu, and Y. Chen, "Nonconvex optimization meets low-rank matrix factorization: An overview," *IEEE Transactions on Signal Processing*, vol. 67, no. 20, pp. 5239–5269, 2019.

[94] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning.* MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[95] T. Yokota, Q. Zhao, and A. Cichocki, "Smooth parafac decomposition for tensor completion," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5423–5436, 2016.

[96] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 208–220, 2012.

[97] C. De Fabritiis, R. Ragona, and G. Valenti, "Traffic estimation and prediction based on real time floating car data," in *2008 11th international IEEE conference on intelligent transportation systems.* IEEE, 2008, pp. 197–203.

[98] X. Chen, J. Yang, and L. Sun, "A nonconvex low-rank tensor completion model for spatiotemporal traffic data imputation," *arXiv preprint arXiv:2003.10271*, 2020.

[99] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, "Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[100] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, "Deep state space models for time series forecasting," in *Advances in Neural Information Processing Systems*, 2018, pp. 7785–7794.

[101] Y. Wang, A. Smola, D. Maddix, J. Gasthaus, D. Foster, and T. Januschowski, "Deep factors for forecasting," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 6607–6617.