Facets: Fast Comprehensive Mining of Coevolving High-order Time Series

Yongjie Cai
The Graduate Center, CUNY
ycai@gradcenter.cuny.edu

Hanghang Tong
Arizona State University
hanghang.tong@asu.edu

Wei Fan
Big Data Labs - Baidu USA
fanwei03@baidu.com

Ping Ji
The Graduate Center, CUNY
pji@jjay.cuny.edu

Qing He
University at Buffalo, SUNY
qinghe@buffalo.edu

ABSTRACT

Mining time series data has been a very active research area in the past decade, exactly because of its prevalence in many high-impact applications, ranging from environmental monitoring, intelligent transportation systems, computer network forensics, to smart buildings and many more. It has posed many fascinating research questions. Among others, three prominent challenges shared by a variety of real applications are (a) high-order; (b) contextual constraints and (c) temporal smoothness. The state-of-the-art mining algorithms are rich in addressing each of these challenges, but relatively short of comprehensiveness in attacking the coexistence of multiple or even all of these three challenges.

In this paper, we propose a comprehensive method, FACETS, to simultaneously model *all* these three challenges. We formulate it as an optimization problem from a dynamic graphical model perspective. The key idea is to use tensor factorization to address multi-aspect challenges, and perform careful regularizations to attack both contextual and temporal challenges. Based on that, we propose an effective and scalable algorithm to solve the problem. Our experimental evaluations on three real datasets demonstrate that our method (1) outperforms its competitors in two common data mining tasks (imputation and prediction); and (2) enjoys a linear scalability w.r.t. the length of time series.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data mining*

Keywords

A network of time series; tensor factorization

1. INTRODUCTION

Mining time series data has been a very active research area in the past decade, exactly because of its prevalence in many high-impact applications, ranging from environmental monitoring, intelligent transportation systems, computer network forensics, to smart buildings and many more. It has posed many fascinating research questions. We identify and summarize three prominent challenges as follows:

C1. *High-order*[1]. Multiple time series arising from real applications are often collected from multiple locations with multiple types (See Fig. 1(a) for an example). Yet, many classic time series analysis tools, such as Kalman filtering, often fall short in modeling such multi-aspect, high-order time series data.

C2. *Contextual constraints.* Many real time series data is accompanied by contextual information (e.g., the sensor network in Fig. 1(b)). How to effectively leverage such contextual information remains an open question for time series data mining.

C3. *Temporal smoothness.* This refers to the correlation among the adjacent observations along the temporal dimension (e.g., the smooth curves in Fig. 1(b)(c)). Despite its key importance for some data mining tasks (e.g., imputation and prediction), temporal smoothness is often ignored in certain time series mining algorithms (e.g., the standard matrix/tensor decomposition), which have been increasingly attracting attention in the recent years.

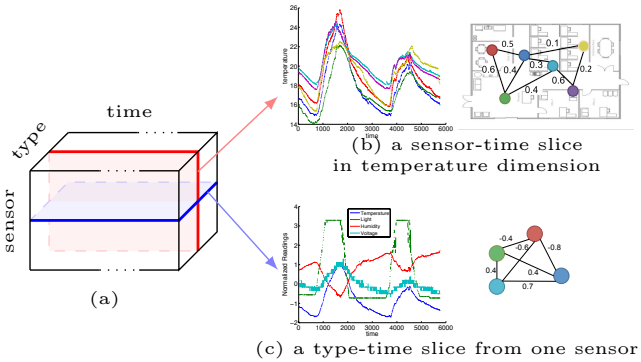


Figure 1: An illustrative example of high-order time series with rich contextual networks. (Best viewed in color).

[1]a.k.a multivariate in statistics.

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.

DOI: http://dx.doi.org/10.1145/2783258.2783348.

An example of time series that exposes these three challenges is illustrated in Fig. 1. The time series data are generated from a set of sensors deployed in a smart building. Each sensor generates multiple time series by measuring certain types of room condition (e.g., temperature, humidity, etc.) over time. As shown in Fig. 1(a), the time series data can be viewed as a cube, in which sensor, measure type, and time step are represented in each dimension, respectively (*high-order*). As shown in Fig. 1(b), by selecting one measurement type (e.g., temperature in the figure), we obtain a sensor-time slice that consists of multiple temperature time series from the sensor network. The weight labeled on the network edge indicates the similarity between connected nodes. We use the same color to represent time series and its corresponding network node. It is clear that the time series in the sensor-time slice are connected with each other by the underlying sensor network (*contextual constraints*). Similarly, if we extract a type-time slice from one sensor, we can also find the time series of multiple types are essentially connected by the type network shown in Fig. 1(c). Specifically, temperature, light, voltage have similar daily patterns, while humidity shows a trend inverse to those of others. Finally, we can easily observe *temporal smoothness* from time series figures.

The state-of-the-art mining algorithms are rich in dealing with each of the aforementioned challenges, but relatively short of comprehensiveness in overcoming the coexistence of multiple or all of the challenges. For example, dynaMMo [17] applies linear dynamic systems after interpolation to learn the temporal dynamics and improve the accuracy of estimating the missing values. Dynamic Tensor Analysis (DTA) provides a compact summary for high-order and high-dimensional data [25]. Note that neither dynaMMo (as well as its high-order generalization [23]) nor DTA encodes the contextual constraints (e.g., the sensor network in a smart building). On the other hand, the dynamic contextual matrix factorization (DCMF) algorithm [5] encodes both contextual information and temporal dynamics, but falls short in modeling the high order of time series data.

In this paper, we propose a method of fast comprehensive mining of coevolving high-order time series (FACETS). It formulates high-order time series as *tensors* and adopts the tensor decomposition model to find the latent factors of time series data. By encoding the contextual constraints, FACETS finds similar latent factors from similar time series. It further encodes temporal smoothness with multilinear dynamic systems.

The main contributions of this paper can be summarized as follows.

- **Problem Definition**. To the best of our knowledge, we are the first to collectively address C1-C3 on mining high-order time series data.
- **Algorithms and Analysis**. We propose a comprehensive model to solve all of the three challenges, and propose an effective and scalable algorithm that naturally fits in the tasks of imputation and prediction; and analyze its effectiveness and complexity.
- **Empirical Evaluations**. The experimental evaluations on real datasets demonstrate that our method (1) outperforms its competitors in two common data mining tasks (i.e., imputation and prediction); and (2) enjoys a linear scalability w.r.t. the length of time series.

Table 1: Symbols and Definitions.

Symbol	Definition and Description
$\mathcal{A}$,...	tensors (calligraphic style)
$\mathbf{A}$,...	matrices (bold upper case)
$\mathbf{A}_{ij}$	the element at the i^{th} row and the j^{th} column of matrix $\mathbf{A}$
$\mathbf{A}_j, \mathbf{A}_{.j}$	the j^{th} column of matrix $\mathbf{A}$
$\mathbf{A}_{i.}$	the i^{th} row of matrix $\mathbf{A}$
$\mathbf{A}'$	transpose of matrix $\mathbf{A}$
$\mathbf{a}, \mathbf{b}, ...$	column vectors (bold lower case)
$\odot$	element-wise multiplication
$\otimes$	kronecker product
$\circledast$	tensor product
$\mathcal{X}$	time series tensor with all time steps
$\mathcal{X}_t$	time series tensor at t^{th} time step
$\mathcal{W}$	indicator tensor
$\mathcal{S}$	contextual matrix set $\{\mathbf{S}^{(1)}, ..., \mathbf{S}^{(M)}\}$
$\mathcal{Z}$	time series latent variable
$\mathcal{B}$	transition variable
$\mathcal{U}$	coefficient latent variable
$\mathcal{V}$	contextual latent matrix set $\{\mathbf{V}^{(1)}, ..., \mathbf{V}^{(M)}\}$
M	order of $\mathcal{X}_t$
T	length of time series
N_m	dimensions of $\mathcal{X}$ or $\mathcal{W}$ on mode m
L_m	dimensions of latent variables on mode m
$\text{vec}(\mathcal{X})$	vectorization of tensor $\mathcal{X}$
$\text{mat}(\mathcal{X})$	matricization of tensor $\mathcal{X}$
$\mathbf{X}_{(n)}$	mode-n matricization of tensor $\mathcal{X}$

The rest of this paper is organized as follows: In Section 2, we introduce the notations and formally define the problem. Then we briefly introduce the background knowledge in Section 3. We present the proposed solution and its analysis in Section 4, and provide the experimental results in Section 5. The related work is reviewed in Section 6, followed by the conclusions in Section 7.

2. PROBLEM DEFINITION

Table 1 lists the main symbols we use throughout this paper. Besides the standard notations, we use an $(M+1)$-order[2] tensor $\mathcal{X} \in \mathbb{R}^{N_1 \times ... \times N_M \times T}$ to denote time series, where $N_m (1 \leq m \leq M)$ is the dimensionality of the m^{th} mode and the last mode of the time series tensor represents the temporal mode with T dimensions (i.e., the time series has T time steps). We can also rewrite the time series tensor as a sequence of M-order tensors $\mathcal{X}_1, \mathcal{X}_2 ..., \mathcal{X}_T$, where each $\mathcal{X}_t \in \mathbb{R}^{N_1 \times ... \times N_M} (1 \leq t \leq T)$ denotes the observed data at t^{th} time step. We use an indicator tensor $\mathcal{W} \in \mathbb{R}^{N_1 \times ... \times N_M \times T}$ to indicate whether a single entry in $\mathcal{X}$ is observed or missing. Specifically, $\mathcal{W}_{n_1 ... n_M t} = 0$ if $\mathcal{X}_{n_1 ... n_M t}$ is missing, otherwise $\mathcal{W}_{n_1 ... n_M t} = 1$. Besides, we have a set of contextual matrices $\mathcal{S} = \{\mathbf{S}^{(1)}, \mathbf{S}^{(2)}, ..., \mathbf{S}^{(M)}\}$, where each $\mathbf{S}^{(m)} \in \mathbb{R}^{N_m \times N_m} (1 \leq m \leq M)$ represents the contextual network of $\mathcal{X}$'s m^{th} mode and each entry of $\mathbf{S}^{(m)}$

[2]In this paper, order and mode are interchangeable.

indicates the correlations between the corresponding two dimensions in the m^{th} mode.

With the above notations, we generalize the concept of *a Network of Time Series*, which was first introduced in [5], and formally define **A Network of High-order Time Series (Net-HiTs)** as follows:

Definition 2.1. *A Network of High-order Time Series (Net-HiTs).*
A Network of High-order Time Series is defined as a quadruplet $\mathcal{R} = \langle \mathcal{X}, \mathcal{W}, \mathcal{S}, \zeta \rangle$*, where* $\mathcal{X} \in \mathbb{R}^{N_1 \times \ldots \times N_M \times T}$ *is a partially observed* $(M+1)$*-order time series tensor;* $\mathcal{W}$ *is an indicator tensor in the same size with* $\mathcal{X}$ ($\mathcal{W} \in \mathbb{R}^{N_1 \times \ldots \times N_M \times T}$)*,* T *in both tensors represents the dimensionality of the time mode;* $\mathcal{S}$ *contains a set of contextual matrices, which represent the correlations between any two dimensions in each mode of* $\mathcal{X}$*; and* ζ *is a one-to-one mapping function, which maps each dimension of the time series* $\mathcal{X}$ *to a node in* $\mathcal{S}$*.*

Accordingly, the problem of time series missing value recovery and prediction can be defined as follows:

Problem 2.1. *Net-HiTs Missing Value Recovery.*
Given*: a network of high-order time series* $\mathcal{R} = \langle \mathcal{X}, \mathcal{W}, \mathcal{S}, \zeta \rangle$*;*
Recover*: its missing parts indicated by the indicator tensor* $\mathcal{W}$*.*

Problem 2.2. *Net-HiTs Prediction.*
Given*: a network of high-order time series* $\mathcal{R} = \langle \mathcal{X}, \mathcal{W}, \mathcal{S}, \zeta \rangle$*, and the time step* t *to predict ;*
Predict*:* t *time steps after* $\mathcal{X}$*.*

3. PRELIMINARIES

In this section, we briefly introduce some definitions and lemmas in multilinear algebra (a.k.a tensor algebra or multilinear analysis) from tensor related literatures [13, 23].

Definition 3.1. *Vectorization.*
The vectorization of an M*-order tensor* $\mathcal{X} \in \mathbb{R}^{N_1 \times \ldots \times N_M}$ $\text{vec}(\mathcal{X})$ *is obtained by iterating elements of* $\mathcal{X}$*.* $\text{vec}(\mathcal{X}) \in \mathbb{R}^{N_1 \ldots N_M}$*.*

The ordering of the elements does not matter as long as it is consistent. In this paper, we follow Matlab linear indexing with multidimensional arrays to index the elements. Specifically, the j^{th} element of $\text{vec}(\mathcal{X})$ is given by $\text{vec}(\mathcal{X})_j = \mathcal{X}_{n_1,\ldots,n_M}$, where $j = 1 + \sum_{k=1}^{M}(n_k - 1)\prod_{m=1}^{k-1} N_m$.

Definition 3.2. *Matricization.*
Let the ordered sets $\mathcal{R} = \{r_1, \ldots, r_L\}$ *and* $\mathcal{C} = \{c_1, \ldots, c_{M-L}\}$ *be a partitioning of the modes* $\{1,2,\ldots,M\}$*, the matricization of a tensor* $\mathcal{X} \in \mathbb{R}^{N_1 \times \ldots \times N_M}$ *can be specified by*

$$\mathbf{X}_{(\mathcal{R} \times \mathcal{C}: N_1 \times \ldots \times N_M)} \in \mathbb{R}^{J \times K} \text{ with } J = \prod_{n \in \mathcal{R}} N_n \text{ and } K = \prod_{n \in \mathcal{C}} N_n.$$

The indices in $\mathcal{R}$ *and* $\mathcal{C}$ *are mapped to the rows and the columns, respectively. Specifically,* $\left(\mathbf{X}_{(\mathcal{R} \times \mathcal{C}: N_1 \times \ldots \times N_M)}\right)_{jk} = x_{n_1 n_2 \ldots n_M}$*, where* $j = 1 + \sum_{l=1}^{L}(n_{r_l} - 1)\prod_{i=1}^{l-1} N_{r_i}$*, and* $k = 1 + \sum_{m=1}^{M-L}(n_{c_m} - 1)\prod_{i=1}^{m-1} N_{c_i}$

Given a tensor $\mathcal{X} \in \mathbb{R}^{N_1 \times \ldots \times N_M \times L_1 \times \ldots \times L_M}$, we partition the first half of modes $\{N_1, \ldots, N_M\}$ as rows and the second half $\{L_1, \ldots, L_M\}$ as columns. The element of the matricization $\text{mat}(\mathcal{X})$ is given by $\text{mat}(\mathcal{X})_{ij} = \mathcal{X}_{n_1,\ldots,n_M,l_1,\ldots,l_M}$, where $i = 1 + \sum_{k=1}^{M}(n_k - 1)\prod_{m=1}^{k-1} N_m$, and $j = 1 + \sum_{k=1}^{M}(l_k - 1)\prod_{m=1}^{k-1} L_m$.

A special case is *mode-n matricizing*, which happens when $\mathcal{R}$ is a singleton. For example, given an M-order tensor $\mathcal{X} \in \mathbb{R}^{N_1 \times \ldots \times N_M}$, the mode-n matricizing $\mathbf{X}_{(n)} \in \mathbb{R}^{N_n \times \prod_{i \neq n} N_i}$, i.e., $\mathcal{R} = \{n\}, \mathcal{C} = \{1, 2..., n-1, n+1, ..., M\}$.

Definition 3.3. *Product or Contracted Product.*
Given two tensors $\mathcal{U} \in \mathbb{R}^{N_1 \times \ldots \times N_M \times L_1 \times \ldots \times L_M}$ *and* $\mathcal{Z} \in \mathbb{R}^{L_1 \times \ldots \times L_M}$*, the product or the contracted product* $\mathcal{X} = \mathcal{U} \circledast \mathcal{Z}$ *is given by* $\mathcal{X}_{n_1,\ldots,n_M} = \sum_{l_1,\ldots,l_M} \mathcal{U}_{n_1,\ldots,n_M,l_1,\ldots,l_M} \mathcal{Z}_{l_1,\ldots,l_M}$*,* $\mathcal{X} \in \mathbb{R}^{N_1 \times \ldots \times N_M}$*.*

Definition 3.4. *Tensor Factorization.*
Given a tensor $\mathcal{U} \in \mathbb{R}^{N_1 \times \ldots \times N_M \times L_1 \times \ldots \times L_M}$*, the factorization of* $\mathcal{U}$ *is to decompose it into* M *factor matrices* $\{\mathbf{U}^{(m)} \in \mathbb{R}^{N_m \times L_m}\}_{m=1}^{M}$*, so that* $\mathcal{U}_{n_1,\ldots,n_M,l_1,\ldots,l_M} = \prod_{m=1}^{M} \mathbf{U}^{(m)}_{n_m l_m}$*. It can also be written as* $\text{mat}(\mathcal{U}) = \mathbf{U}^{(M)} \otimes \mathbf{U}^{(M-1)} \otimes \ldots \otimes \mathbf{U}^{(1)}$*, where* $\otimes$ *denotes the Kronecker product.*

With the above definitions, we can easily prove the following two lemmas.

Lemma 3.1. *Given two tensors* $\mathcal{U} \in \mathbb{R}^{N_1 \times \ldots \times N_M \times L_1 \times \ldots \times L_M}$ *and* $\mathcal{Z} \in \mathbb{R}^{L_1 \times \ldots \times L_M}$*, let* $\mathcal{X} = \mathcal{U} \circledast \mathcal{Z}$*, then* $\text{vec}(\mathcal{X}) = \text{mat}(\mathcal{U})\,\text{vec}(\mathcal{Z})$*. If* $\mathcal{U}$ *is factorizable with matrices* $\{\mathbf{U}^{(m)}\}_{m=1}^{M}$*, then* $\text{vec}(\mathcal{X}) = [\mathbf{U}^{(M)} \otimes \mathbf{U}^{(M-1)} \otimes \ldots \otimes \mathbf{U}^{(1)}]\,\text{vec}(\mathcal{Z})$*.*

Lemma 3.2. *Given two tensors* $\mathcal{U} \in \mathbb{R}^{N_1 \times \ldots \times N_M \times L_1 \times \ldots \times L_M}$*,* $\mathcal{Z} \in \mathbb{R}^{L_1 \times \ldots \times L_M}$*, and* $\text{mat}(\mathcal{U}) = \mathbf{U}^{(M)} \otimes \mathbf{U}^{(M-1)} \otimes \ldots \otimes \mathbf{U}^{(1)}$

$$\mathcal{X} = \mathcal{U} \circledast \mathcal{Z} \Leftrightarrow$$
$$\mathbf{X}_{(n)} = \mathbf{U}^{(n)} \mathbf{Z}_{(n)} (\mathbf{U}^{(M)} \otimes \ldots \otimes \mathbf{U}^{(n+1)} \otimes \mathbf{U}^{(n-1)} \otimes \ldots \otimes \mathbf{U}^{(1)})'$$

In addition, we introduce a lemma of matrix normal distribution [9] and the definition of tensor normal distribution [23]:

Lemma 3.3. *Matrix Normal Distribution.*
Given a matrix $\mathbf{X} \in \mathbb{R}^{N \times P}$*,* $\mathbf{X}$ *follows the matrix normal distribution* $\mathcal{MN}(\mathbf{M}, \mathbf{U}, \mathbf{V})$ *if and only if* $\text{vec}(\mathbf{X}) \sim \mathcal{N}(\text{vec}(\mathbf{M}), \mathbf{V} \otimes \mathbf{U})$*, where* $\mathbf{M} \in \mathbb{R}^{N \times P}, \mathbf{U} \in \mathbb{R}^{N \times N}, \mathbf{V} \in \mathbb{R}^{P \times P}$*.*

Definition 3.5. *Tensor Normal Distribution.*
Given a tensor $\mathcal{X} \in \mathbb{R}^{N_1 \times \ldots \times N_M}$*,* $\mathcal{X}$ *follows tensor normal distribution* $\mathcal{N}(\mathcal{U}, \mathcal{D})$ *if* $\text{vec}(\mathcal{X}) \sim \mathcal{N}(\text{vec}(\mathcal{U}), \text{mat}(\mathcal{D}))$*, where* $\mathcal{U} \in \mathbb{R}^{N_1 \times \ldots \times N_M}$ *and* $\mathcal{D} \in \mathbb{R}^{N_1 \times \ldots \times N_M \times N_1 \times \ldots \times N_M}$*.*

4. PROPOSED APPROACH: FACETS

In this section, we present our proposed FACETS for fast comprehensive mining of coevolving high-order time series. We give the formal formulation of the model and then provide the detailed algorithm to learn the model.

4.1 Our Optimization Formulation

In order to collectively address all the three challenges outlined in the introduction, we present a regularization optimization formulation from a dynamic graphical model perspective.

Step 1 - Addressing both C1 and C2. FACETS adopts the tensor decomposition model to find the latent factors for the input high-order (C1) time series data. It further encodes the contextual information (C2) to encourage the similar time series to share similar latent factors.

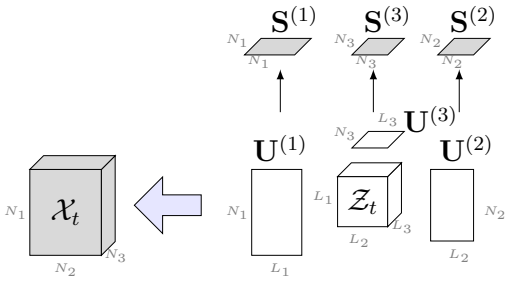


Figure 2: Step 1 - Addressing both C1 and C2

Formally, let $\mathcal{X}_1$, $\mathcal{X}_2$ ..., $\mathcal{X}_T$ be an M-order time series, and $\mathcal{S} = \{\mathbf{S}^{(1)}, \mathbf{S}^{(2)}, ..., \mathbf{S}^{(M)}\}$ be a set of contextual matrices, where each $\mathbf{S}^{(m)}$ represents the contextual network of $\mathcal{X}$'s m^{th} mode, we define the conditional distribution of $\mathcal{X}_t$ to be a multilinear Gaussian distribution with the covariance $\mathcal{D}$ and the mean as the product of two latent factors $\mathcal{U}$ and $\mathcal{Z}_t$, shown in Eq. (1). $\mathcal{U}$ is further factorized into M factor matrices $\mathbf{U}^{(1)}, ..., \mathbf{U}^{(M-1)}, \mathbf{U}^{(M)}$, shown in Eq. (2). Each $\mathbf{U}^{(m)}$ indicates the time series similarity among the dimensions in the m^{th} mode. Then, we define the conditional distribution of the j^{th} column in the m^{th} contextual matrix $\mathbf{S}_j^{(m)}$ as a linear Gaussian distribution with the covariance $\Xi^{(m)}$ and the mean as the product of the latent factors $\mathbf{U}^{(m)}$ and $\mathbf{V}_j^{(m)}$, presented in Eq. (3). In Eq. (4), we define zero-mean spherical Gaussian priors on $\mathbf{V}^{(m)}$, with entries in $\mathbf{S}^{(m)}$ scaled to $[-1, 1]$.

$$\mathcal{X}_t | \mathcal{Z}_t, \mathcal{U} \sim \mathcal{N}\left(\mathcal{U} \circledast \mathcal{Z}_t, \mathcal{D}\right), \tag{1}$$

$$\text{mat}(\mathcal{U}) = \mathbf{U}^{(M)} \otimes \mathbf{U}^{(M-1)} \otimes ... \otimes \mathbf{U}^{(1)}, \tag{2}$$

$$\mathbf{S}_j^{(m)} | \mathbf{V}_j^{(m)}, \mathbf{U}^{(m)} \sim \mathcal{N}\left(\mathbf{U}^{(m)}\mathbf{V}_j^{(m)}, \Xi^{(m)}\right), \tag{3}$$

$$\mathbf{V}_j^{(m)} \sim \mathcal{N}(0, \Gamma^{(m)}), \tag{4}$$

$$j = 1, ..., N_m; m = 1, ..., M.$$

Fig. 2 illustrates this step in the case of $M = 3$, where we omit $\mathbf{V}^{(m)}$ for clarity.

Step 2 - Addressing C3. FACETS encodes temporal smoothness (C3) with multilinear dynamical systems [23]. Specifically, we define the conditional distribution of the latent factor $\mathcal{Z}_t$ as a multilinear Gaussian distribution with a multilinear transition tensor $\mathcal{B}$ and the covariance $\mathcal{O}$, shown in Eq. (6). $\mathcal{B}$ is also defined as factorizable, formulated in Eq. (7). As defined in Eq. (5), the initial state of $\mathcal{Z}_1$ is generated based on the tensor normal distribution with the mean $\mathcal{Z}_0$ and the covariance $\mathcal{O}_0$.

$$\mathcal{Z}_1 \sim \mathcal{N}\left(\mathcal{Z}_0, \mathcal{O}_0\right), \tag{5}$$

$$\mathcal{Z}_t | \mathcal{Z}_{t-1} \sim \mathcal{N}\left(\mathcal{B} \circledast \mathcal{Z}_{t-1}, \mathcal{O}\right), \tag{6}$$

$$\text{mat}(\mathcal{B}) = \mathbf{B}^{(M)} \otimes \mathbf{B}^{(M-1)} \otimes ... \otimes \mathbf{B}^{(1)}. \tag{7}$$

In addition, since missing values in time series exist in many applications, we modify Eq. (1) as follows:

$$\mathcal{X}_t^* | \mathcal{Z}_t, \mathcal{U} \sim \mathcal{N}\left(\mathcal{U}^* \circledast \mathcal{Z}_t, \mathcal{D}^*\right), \tag{8}$$

where $\mathcal{X}_t^*$ represents observed entries of $\mathcal{X}_t$ (i.e., the corresponding entries in $\mathcal{W}_t$ equal 1.), which is a subset of $\mathcal{X}_t$. $\mathcal{U}^*, \mathcal{D}^*$ are the subsets of $\mathcal{U}$ and $\mathcal{D}$, corresponding to the entries of $\mathcal{X}_t^*$.

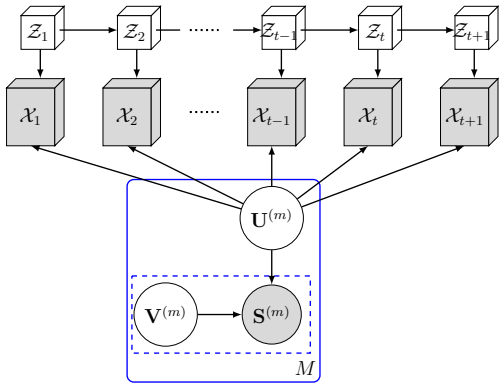


Figure 3: Graphical Model of FACETS. The blue solid box is a plate representation of M instances, of which only a single instance is shown explicitly. The dashed rectangle indicates that $\mathbf{V}^{(m)}$ and $\mathbf{S}^{(m)}$ can be ignored in the model.

Our goal is to estimate the model parameters $\theta = \{\mathcal{B}, \mathcal{Z}_0, \mathcal{O}_0, \mathcal{O}, \mathcal{D}, \{\Xi^{(m)}\}, \{\Gamma^{(m)}\}\}$ and find the latent factors $\mathcal{U}, \mathcal{Z}$, $\{\mathbf{V}^{(m)}\}_{m=1}^M$ that maximize the following joint distribution:

$$\underset{\mathcal{U},\mathcal{Z},\{\mathbf{V}^{(m)}\},\theta}{\text{argmax}} p(\mathcal{X}, \mathcal{S}, \mathcal{U}, \mathcal{Z}, \{\mathbf{V}^{(m)}\}) = \underset{\mathcal{U},\mathcal{Z},\{\mathbf{V}^{(m)}\},\theta}{\text{argmax}} p(\mathcal{Z}_1) \underbrace{\prod_{t=2}^{T} p(\mathcal{Z}_t | \mathcal{Z}_{t-1})}_{\text{C3. temporal smoothness}}$$

$$\underbrace{\prod_{t=1}^{T} p(\mathcal{X}_t | \mathcal{Z}_t, \mathcal{U})}_{\text{C1. tensor time series}} \times \prod_{m=1}^{M} \prod_{j=1}^{N_m} p(\mathbf{V}_j^{(m)}) \underbrace{\prod_{m=1}^{M} \prod_{j=1}^{N_m} p(\mathbf{S}_j^{(m)} | \mathbf{V}_j^{(m)}, \mathbf{U}^{(m)})}_{\text{C2. contextual information}}, \tag{9}$$

where we omit the model parameters in the equation.

The complete graphical model of FACETS is shown in Fig. 3. The dashed rectangle means that $\mathbf{V}^{(m)}$ and $\mathbf{S}^{(m)}$ can be ignored if the contextual matrix $\mathbf{S}^{(m)}$ is unavailable or inapplicable. In this model, we include a contextual weight vector $\lambda \in \mathbb{R}^M$ to control the contributions of contextual matrices. The contextual matrix of the m^{th} mode is ignored if λ_m is set to zero.

In order to accommodate sparse time series datasets with our model, we simplify the covariances by assuming that the transition noise and the observation noise are independent and identically distributed (*i.i.d.*), and the covariances of latent variable $\{\mathbf{V}_j^{(m)}\}$ are *i.i.d.* Thus, the covariances $\Xi^{(m)}, \mathcal{D}, \mathcal{O}, \mathcal{O}_0$ and $\Gamma^{(m)}$ are reduced to $(\xi^{(m)})^2, \sigma_R^2, \sigma_O^2, \sigma_0^2$ and σ_{Vm}^2, respectively.

4.2 Proposed Optimization Algorithm

It is difficult to find the global optimal solution of Eq. (9) due to three couplings in the model: a) the latent factors $\mathcal{U}$ and $\mathcal{Z}$ jointly determine $\mathcal{X}$, and $\mathbf{U}^{(m)}$ and $\mathbf{V}^{(m)}$ jointly determine $\mathbf{S}^{(m)}$; b) both the parameters and the latent factors are unknown; c) $\mathcal{U}$ is determined by $\mathbf{U}^{(1)}$, $\mathbf{U}^{(2)}$,..., and $\mathbf{U}^{(M)}$ jointly, and so is $\mathcal{B}$. Hence, we aim to find its local optimum instead following the expectation-maximization (E-M) algorithm. Specifically, FACETS employs the following steps to address the aforementioned difficulties: a) it regards $\mathcal{U}$ as a model parameter; b) FACETS searches a local optimal solution by updating the parameters θ and the expectations of $\mathcal{Z}$ and $\mathbf{V}^{(m)}$ alternatively; c) FACETS iteratively updates $\mathbf{U}^{(m)}$ and $\mathbf{B}^{(m)}$ when keeping other factors of $\mathcal{U}$ and $\mathcal{B}$ fixed.

Inferring latent factors $\mathcal{Z}$ and $\{\mathbf{V}^{(m)}\}$. With the fixed parameters, we first perform the vectorizations and matricizations in Eq. (10). With $\mathcal{X}_t$ and $\mathcal{Z}_t$ reshaped into vectors, we apply the forward and backward algorithms to obtain the expectations of the latent factors as in the DCMF algorithm [5]. The details are presented in Appendix A.

$$\begin{aligned}\operatorname{vec}(\mathcal{Z}_1) &\sim \mathcal{N}\left(\operatorname{vec}(\mathcal{Z}_0), \operatorname{mat}(\mathcal{O}_0)\right),\\ \operatorname{vec}(\mathcal{Z}_t)|\operatorname{vec}(\mathcal{Z}_{t-1}) &\sim \mathcal{N}\left(\operatorname{mat}(\mathcal{B})\operatorname{vec}(\mathcal{Z}_{t-1}), \operatorname{mat}(\mathcal{O})\right),\\ \operatorname{vec}(\mathcal{X}_t)|\operatorname{vec}(\mathcal{Z}_t) &\sim \mathcal{N}\left(\operatorname{mat}(\mathcal{U})\operatorname{vec}(\mathcal{Z}_t), \operatorname{mat}(\mathcal{D})\right). \quad (10)\end{aligned}$$

Updating non-multilinear parameters. In this step, we obtain new estimations of the non-multilinear parameters to maximize the expectation of the log likelihood defined in Eq. (11).

$$\begin{aligned}Q(\theta,\theta^{old}) &= \mathbb{E}_{\mathcal{Z},\{\mathbf{V}^{(m)}\}|\theta^{old}}\left[\ln p(\mathcal{X},\mathcal{S},\mathcal{Z},\{\mathbf{V}^{(m)}\}|\theta)\right],\\ \theta^{new} &= \operatorname*{argmax}_{\theta} Q(\theta,\theta^{old}), \quad (11)\end{aligned}$$

where $\theta = \{\mathcal{U},\mathcal{B},\mathcal{Z}_0,\sigma_O^2,\sigma_0^2,\sigma_R^2,\{(\xi^{(m)})^2\},\{\sigma_{V_m}^2\}\}$, and $p(\mathcal{X},\mathcal{S},\mathcal{Z},\{\mathbf{V}^{(m)}\}|\theta)$ is defined in Eq. (9).

With the expectations obtained in the latent factor inferring step, each non-multilinear model parameter can be obtained by taking the derivative of $Q(\theta,\theta^{old})$ w.r.t. that parameter and setting the derivative to zero. The parameters are updated as follows:

$$\begin{aligned}(\sigma_{V_m}^2)^{new} &= \frac{1}{N_m L_m}\sum_{j=1}^{N_m}\operatorname{tr}(\mathbb{E}[\mathbf{V}_j^{(m)}(\mathbf{V}_j^{(m)})']),\\ \operatorname{vec}(\mathcal{Z}_0^{new}) &= \mathbb{E}[\operatorname{vec}(\mathcal{Z}_1)],\\ (\sigma_0^2)^{new} &= \frac{1}{\prod_{m=1}^M L_m}\operatorname{tr}\left[\mathbb{E}[\operatorname{vec}(\mathcal{Z}_1)\operatorname{vec}(\mathcal{Z}_1)'] - \mathbb{E}[\operatorname{vec}(\mathcal{Z}_1)]\mathbb{E}[\operatorname{vec}(\mathcal{Z}_1)']\right],\\ (\sigma_O^2)^{new} &= \frac{1}{(T-1)\prod_{m=1}^M L_m}\operatorname{tr}\Bigg(\sum_{t=2}^{T}\mathbb{E}[\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_t)']\\ &- \operatorname{mat}(\mathcal{B})\sum_{t=2}^{T}\mathbb{E}[\operatorname{vec}(\mathcal{Z}_{t-1})\operatorname{vec}(\mathcal{Z}_t)'] - \sum_{t=2}^{T}\mathbb{E}[\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_{t-1})']\operatorname{mat}(\mathcal{B})'\\ &+ \operatorname{mat}(\mathcal{B})\sum_{t=2}^{T}\mathbb{E}[\operatorname{vec}(\mathcal{Z}_{t-1})\operatorname{vec}(\mathcal{Z}_{t-1})']\operatorname{mat}(\mathcal{B})'\Bigg),\end{aligned}$$

$$\begin{aligned}((\xi^{(m)})^2)^{new} &= \frac{1}{N_m^2}\sum_{j=1}^{N_m}\Big[(\mathbf{S}_j^{(m)})'\mathbf{S}_j^{(m)} - 2(\mathbf{S}^{(m)})_j'\mathbf{U}^{(m)}\mathbb{E}[\mathbf{V}_j^{(m)}]\\ &+ \operatorname{tr}\left(\mathbf{U}^{(m)}\mathbb{E}\left[\mathbf{V}_j^{(m)}(\mathbf{V}_j^{(m)})'\right](\mathbf{U}^{(m)})'\right)\Big],\\ (\sigma_R^2)^{new} &= \frac{1}{\sum_{t=1}^T\sum_i \operatorname{vec}(\mathcal{W}_t)_i}\sum_{t=1}^{T}\big[\operatorname{vec}(\mathcal{X}_t^*)'\operatorname{vec}(\mathcal{X}_t^*)\\ &+ \operatorname{tr}\left(\operatorname{mat}(\mathcal{U})^*\mathbb{E}\left[\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_t)'\right](\operatorname{mat}(\mathcal{U})^*)'\right)\\ &-2\operatorname{vec}(\mathcal{X}_t^*)'\operatorname{mat}(\mathcal{U})^*\mathbb{E}[\operatorname{vec}(\mathcal{Z}_t)]\big]. \quad (12)\end{aligned}$$

Updating multilinear operators. Though the new estimations of $\operatorname{mat}(\mathcal{U})$ and $\operatorname{mat}(\mathcal{B})$ can be derived in the same way with updating non-multilinear parameters, it cannot determine the factors of $\mathcal{U}$ and $\mathcal{B}$. We apply the following steps to obtain a closed-form solution for $\mathbf{B}^{(m)}$ and $\mathbf{U}^{(m)}$, by keeping other factors fixed.

Based on Lemma 3.1-3.3, we can get

$$\begin{aligned}\mathcal{Z}_t|\mathcal{Z}_{t-1} &\sim \mathcal{N}\left(\mathcal{B}\circledast\mathcal{Z}_{t-1}, \sigma_O^2\mathcal{I}\right) \quad (13)\\ \Leftrightarrow \operatorname{vec}(\mathcal{Z}_t)|\operatorname{vec}(\mathcal{Z}_{t-1}) &\sim \mathcal{N}\left(\operatorname{mat}(\mathcal{B})\operatorname{vec}(\mathcal{Z}_{t-1}), \sigma_O^2\mathbf{I}\right),\\ \Leftrightarrow \mathbf{Z}_{t,(m)}|\mathbf{Z}_{t-1,(m)} &\sim \mathcal{MN}\left(\mathbf{B}^{(m)}\mathbf{Z}_{t-1,(m)})\mathbf{F}, \mathbf{I}, \sigma_O^2\mathbf{I}\right),\end{aligned}$$

where $\mathbf{F} = (\mathbf{B}^{(M)}\otimes\ldots\otimes\mathbf{B}^{(m+1)}\otimes\mathbf{B}^{(m-1)}\ldots\otimes\mathbf{B}^{(1)})'$, $\mathbf{Z}_{t,(m)}$ denotes mode-m matricizing of $\mathcal{Z}_t$ and $\mathcal{I}/\mathbf{I}$ denotes the identity tensor/matrix. Then, we can replace $p(\mathcal{Z}_t|\mathcal{Z}_{t-1})$ with $p(\mathbf{Z}_{t,(m)}|\mathbf{Z}_{t-1,(m)})$ in Eq. (11) and obtain the derivative w.r.t. $\mathbf{B}^{(m)}$. By setting the derivative to zero, we get a new estimation of $\mathbf{B}^{(m)}$:

$$\begin{aligned}(\mathbf{B}^{(m)})^{new} &= C_2/C_1,\\ C_1 &= \sum_{t=2}^{T}\sum_{j=1}^{\prod_{n\neq m}L_n}\mathbb{E}[(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)'],\\ C_2 &= \sum_{t=2}^{T}\sum_{j=1}^{\prod_{n\neq m}L_n}\mathbb{E}[(\mathbf{Z}_{t,(m)})_j(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)'], \quad (14)\end{aligned}$$

where $\mathbb{E}[(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)']$ and $\mathbb{E}[(\mathbf{Z}_{t,(m)})_j(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)']$ are calculated from the expectations and the covariances of the latent factors. Specifically,

$$\begin{aligned}\mathbb{E}[(\mathbf{Z}_{t,(m)})_j(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)'] &= \operatorname{cov}((\mathbf{Z}_{t,(m)})_j, (\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)')\\ &+\mathbb{E}[(\mathbf{Z}_{t,(m)})_j]\mathbf{F}_j'\mathbb{E}[\mathbf{Z}_{t-1,(m)}]', \quad (15)\\ \mathbb{E}[(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)'] &= \operatorname{cov}(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j, (\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)')\\ &+\mathbb{E}[(\mathbf{Z}_{t-1,(m)})]\mathbf{F}_j\mathbf{F}_j'\mathbb{E}[\mathbf{Z}_{t-1,(m)}]'. \quad (16)\end{aligned}$$

Each entry a_{pq} of $\operatorname{cov}((\mathbf{Z}_{t,(m)})_j, (\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)')$ and each entry b_{pq} of $\operatorname{cov}(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j, (\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)')$ are obtained by:

$$\begin{aligned}a_{pq} &= \sum_k \mathbf{F}_{kj}\operatorname{cov}(\mathbf{Z}_{t,(m)}, \mathbf{Z}_{t-1,(m)})_{pjqk}, \quad (17)\\ b_{pq} &= \sum_i\sum_k \mathbf{F}_{kj}\mathbf{F}_{ij}\operatorname{cov}(\mathbf{Z}_{t-1,(m)}, \mathbf{Z}_{t-1,(m)})_{piqk}, \quad (18)\end{aligned}$$

where $\operatorname{cov}(\mathbf{Z}_{t,(m)}, \mathbf{Z}_{t-1,(m)})$ is calculated as follows: 1) revert the inferred $\operatorname{cov}(\operatorname{vec}(\mathcal{Z}_t), \operatorname{vec}(\mathcal{Z}_{t-1}))$ to the tensor form; 2) permute the order of the mode from $[1,2,...,2M]$ to $[m,1,...,m-1,m+1,...,M,m+M,1+M,...,m-1+M,m+1+M,...,2M]$; 3) reshape the reordered covariance tensor by keeping the first and $(M+1)^{th}$ mode fixed and concatenating data from the 2nd mode to the M^{th} mode into one mode, and data from the $(M+2)^{th}$ mode to the $(2M)^{th}$ mode into another mode, and get the four-order $\operatorname{cov}(\mathbf{Z}_{t,(m)}), \mathbf{Z}_{t-1,(m)})$. We perform similar operations to construct $\mathbb{E}[(\mathbf{Z}_{t,(m)})]$ from $\mathbb{E}[\operatorname{vec}(\mathcal{Z}_t)]$, $\operatorname{cov}(\mathbf{Z}_{t,(m)}, \mathbf{Z}_{t,(m)})$ from $\operatorname{cov}(\operatorname{vec}(\mathcal{Z}_t), \operatorname{vec}(\mathcal{Z}_t))$.

We can also perform similar transformations for $\mathbf{U}^{(m)}$ shown in Eq. (19). We only include the observed data in the learning process and ignore the missing entries of time series data indicated by $(\mathbf{W}_{t,(m)})$. Consequently, we update each row $\mathbf{U}_{i.}^{(m)}$ with Eq. (20).

$$\begin{aligned}\mathcal{X}_t|\mathcal{Z}_t, \mathcal{U} &\sim \mathcal{N}\left(\mathcal{U}\circledast\mathcal{Z}_t, \sigma_R^2\mathcal{I}\right) \quad (19)\\ \Leftrightarrow \mathbf{X}_{t,(m)}|\mathbf{Z}_{t,(m)} &\sim \mathcal{MN}\left(\mathbf{U}^{(m)}\mathbf{Z}_{t,(m)})\mathbf{G}, \mathbf{I}, \sigma_R^2\mathbf{I}\right),\end{aligned}$$

where $\mathbf{G} = (\mathbf{U}^{(M)}\otimes\ldots\otimes\mathbf{U}^{(m+1)}\otimes\mathbf{U}^{(m-1)}\ldots\otimes\mathbf{U}^{(1)})'$, and $\mathbf{X}_{t,(m)}$ denotes mode-m matricizing of $\mathcal{X}_t$.

$$(\mathbf{U}_{i.}^{(m)})^{new} = \frac{\lambda_m A_{11}/(\xi^{(m)})^2 + (1-\lambda_m)A_{12}/\sigma_R^2}{\lambda_m A_{21}/(\xi^{(m)})^2 + (1-\lambda_m)A_{22}/\sigma_R^2}, \quad (20)$$

$$A_{11} = \sum_{j=1}^{N_m} \mathbf{S}_{ij}^{(m)} \mathbb{E}[\mathbf{V}_j^{(m)}]',$$

$$A_{21} = \sum_{j=1}^{N_m} \mathbb{E}[\mathbf{V}_j^{(m)} (\mathbf{V}_j^{(m)})'],$$

$$A_{12} = \sum_{t=1}^{T} \sum_{j=1}^{\prod_{n\neq m} N_n} (\mathbf{W}_{t,(m)})_{ij} (\mathbf{X}_{t,(m)})_{ij} \mathbb{E}[(\mathbf{Z}_{t,(m)} \mathbf{G}_j)'],$$

$$A_{22} = \sum_{t=1}^{T} \sum_{j=1}^{\prod_{n\neq m} L_n} (\mathbf{W}_{t,(m)})_{ij} \mathbb{E}[\mathbf{Z}_{t,(m)} \mathbf{G}_j (\mathbf{Z}_{t,(m)} \mathbf{G}_j)'],$$

where λ_m represents the contextual weight of mode m and $\mathbb{E}[\mathbf{Z}_{t,(m)}\mathbf{G}_j(\mathbf{Z}_{t,(m)}\mathbf{G}_j)']$ can be obtained similarly to $\mathbb{E}[(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)(\mathbf{Z}_{t-1,(m)}\mathbf{F}_j)']$.

The overall algorithm. Putting everything together, we have the overall algorithm (FACETS) summarized in Algorithm 1 to get a local optimal solution of Eq. (9). Given a network of high-order time series (NetHiTs) $\mathcal{R}$, the dimensions of latent factors $L \in \mathbb{R}^M$, and the contextual weight vector $\lambda \in \mathbb{R}^M$, our algorithm aims to find the latent factors $\mathcal{U}, \mathcal{Z}, \mathcal{V}$ and other model parameters in θ.

The FACETS algorithm first randomly initializes the model parameters θ (step 1) and obtains the mode number and the dimensionality of $\mathcal{X}_t$'s each mode (step 2). Then it performs matricizations of $\mathcal{X}, \mathcal{W}$ that are repeatedly used in the E-M steps. It also calculates the expectations of $\mathbf{V}^{(m)}$ before the first iteration (step 3-10). Then the algorithm alternatively updates the parameters and the latent factors until the convergence. In each repetition, it conducts M iterations to update each $\mathbf{B}^{(m)}$ and $\mathbf{U}^{(m)}$, while keeping other $\mathbf{B}^{(n)}$ and $\mathbf{U}^{(n)} (n \neq m)$ fixed (step 13-21). Specifically, it iteratively infers the expectations and the covariances of vectorized $\mathcal{Z}$, including $\mathbb{E}[\text{vec}(\mathcal{Z}_t)], \mathbb{E}[(\text{vec}(\mathcal{Z}_t)\,\text{vec}(\mathcal{Z}_t)']$, $\mathbb{E}[(\text{vec}(\mathcal{Z}_t)\,\text{vec}(\mathcal{Z}_{t-1})']$, $\text{cov}(\text{vec}(\mathcal{Z}_t), \text{vec}(\mathcal{Z}_t))$, and $\text{cov}(\text{vec}(\mathcal{Z}_t), \text{vec}(\mathcal{Z}_{t-1}))$ (step 14). Afterwards, it updates the model parameters (step 15-17). Before the next iteration, it updates the expectations related to $\mathbf{V}^{(m)}$ if $\lambda_m > 0$ (step 18-20).

Our FACETS algorithm converges to a local optima of Eq. (9), as it strictly follows the EM algorithm procedure.

Time complexity. The time complexity of our proposed algorithm is summarized in Lemma 4.1.

Lemma 4.1. *Given a network of high-order time series $\mathcal{R}$ that consists of $\mathcal{X} \in \mathbb{R}^{N_1 \times,...,\times N_M \times T}$,$\mathcal{W} \in \mathbb{R}^{N_1 \times,...,\times N_M \times T}$, and $\mathcal{S} = \{\mathbf{S}^{(1)}, \mathbf{S}^{(2)}, ..., \mathbf{S}^{(M)}\}$, where each $\mathbf{S}^{(m)} \in \mathbb{R}^{N_m \times N_m}$ $(1 \leq m \leq M)$, the contextual weight $\lambda \in \mathbb{R}^M$ and the hidden latent dimensions $L \in \mathbb{R}^M$, the time complexity of the* FACETS *algorithm is upper bounded by*

$$O(\#iterations \cdot (l^2 nT + \sum_m L_m^2 N_m^2))$$

where $l = \prod_{m=1}^M L_m$, $n = \prod_{m=1}^M N_m$.

Proof. Omitted for Brevity. □

In practice, the length (T) of the time series is often orders of magnitude larger than the number of the time series (n). Hence, the actual running time of FACETS is dominated by the term related to the length of the time series T, which is linear in T.

Algorithm 1: FACETS

Input : Net-HiTs $\mathcal{R} = \langle \mathcal{X}, \mathcal{W}, \mathcal{S}, \zeta \rangle$,
dimension of latent factors L
weight of contextual information λ
Output: $\theta = \{\mathcal{U}, \mathcal{B}, \mathcal{Z}_0, \sigma_O^2, \sigma_0^2, \sigma_R^2, \{(\xi^{(m)})^2\}, \{\sigma_{V_m}^2\}\}$,
$\mathbb{E}[\mathcal{Z}], \mathbb{E}[\mathcal{V}]$

1 Initialize θ;
2 $M \leftarrow \mathcal{X}_t$'s mode; $N_1, N_2, ..., N_M \leftarrow$ dimensions of $\mathcal{X}_t$;
3 Matricize $\mathcal{X}, \mathcal{W}$ along time mode to obtain $\mathbf{X}, \mathbf{W}$;
4 **for** *m =1:M* **do**
5 Matricize each $\mathcal{X}_t, \mathcal{W}_t$ to obtain $\mathbf{X}_{t,(m)}, \mathbf{W}_{t,(m)}$;
6 **if** $\lambda_m > 0$ **then**
7 **for** *j=1:N_m* **do**
8 Infer $\mathbb{E}[\mathbf{v}_j^{(m)}], \mathbb{E}[\mathbf{v}_j^{(m)}(\mathbf{v}_j^{(m)})']$
9 **end**
10 **end**
11 **end**
12 **repeat**
13 **for** *m = 1:M* **do**
14 Infer the expectations and covariances of vectorized latent factors;
15 With Eq. (12), update $\mathcal{Z}_0, \sigma_O^2, \sigma_0^2, \sigma_R^2$ and if $\lambda_m > 0$, update $(\xi^{(m)})^2, \sigma_{V_m}^2$;
16 Reshape $\text{cov}(\text{vec}(\mathcal{Z}_t), \text{vec}(\mathcal{Z}_{t-1}))$, $\text{cov}(\text{vec}(\mathcal{Z}_t), \text{vec}(\mathcal{Z}_t))$ and $\mathbb{E}[\text{vec}(\mathcal{Z}_t)]$ to $\text{cov}(\mathbf{Z}_{t,(m)}, \mathbf{Z}'_{t-1,(m)})$, $\text{cov}(\mathbf{Z}_{t,(m)}, \mathbf{Z}'_{t,(m)})$ and $\mathbb{E}[\mathbf{Z}_{t,(m)}]$, respectively;
17 Update $\mathbf{B}^{(m)}$ and $\mathbf{U}^{(m)}$ by Eq. (14) - (20);
18 **if** $\lambda_m > 0$ **then**
19 update the expectations related to $\mathbf{V}^{(m)}$;
20 **end**
21 **end**
22 **until** *convergence*;

4.3 Data Mining Applications

Our proposed FACETS algorithm captures temporal smoothness and contextual correlations with observed time series data and input contextual constraints. It naturally fits in the tasks of imputation and forecasting. Other data mining applications where our algorithm can be conveniently applied include denoising, anomaly detection and time series clustering. We omit the details for the limited space.

A1 - Imputation. With the output of the FACETS algorithm, the vectorized version of the reconstructed time series $\hat{\mathcal{X}}$ can be obtained by $\text{vec}(\hat{\mathcal{X}}_t) = \text{mat}(\mathcal{U})\mathbb{E}[\text{vec}(\mathcal{Z}_t)]$. The missing values of $\mathcal{X}_t$ can be inferred from the corresponding entries of $\text{vec}(\hat{\mathcal{X}}_t)$.

A2 - Prediction. Given the output parameter θ and the inferred $\hat{\mathcal{Z}}_t$, we can predict $\hat{\mathcal{Z}}_{t+1}$ as $\mathcal{B} \circledast \hat{\mathcal{Z}}_t$, and $\hat{\mathcal{X}}_{t+1}$ as $\mathcal{U} \circledast \hat{\mathcal{Z}}_{t+1}$. In addition, once the real data $\mathcal{X}_{t+1}$ is arrived at $(t+1)^{th}$ time step, FACETS will update $\hat{\mathcal{Z}}_{t+1}$ according to the observed $\mathcal{X}_{t+1}$ to improve the accuracy of predicting the latter time series. Note that FACETS can naturally deal with the missing values in the newly arriving time series data.

5. EXPERIMENTAL RESULTS

In this section, we present the empirical evaluations on three real datasets, which are designed to answer the following two questions:

- *Effectiveness*: how accurate is the proposed FACETS algorithm in terms of imputation and prediction of time series?
- *Efficiency*: how does the proposed FACETS algorithm scale w.r.t. the size of the input time series?

5.1 Experimental Setup

A. Baseline Methods. We compare our method with the following state-of-the-art algorithms: Probabilistic Matrix Factorization (PMF) [21], Social Recommendation (SoRec) [19], SmoothPMF and SmoothSoRec. SmoothPMF and SmoothSoRec are improved algorithms of PMF and SoRec, which encode temporal smoothness by adding regularizations on consecutive time series latent factors [5]. We also compare with DCMF, which is a special case of our FACETS algorithm where $M = 1$.

B. Datasets. We use the following three real datasets in our experiments.

SST Dataset The Sea-Surface Temperature (SST) dataset consists of hourly temperature measurements from a 5-by-6 grid of sea-surface located from $5°N$, $180°W$ to $5°S$, $110°W$ [3]. The measurement started from April 26, 1994 at 7:00 p.m. to July 19, 1994 at 3:00 a.m., a total of 2000 time steps. Since no contextual network data is available in this dataset, we perform matriciztions along each mode and construct a contextual matrix based on the cosine similarity of each pair of time series in that mode. In the experimental evaluations, we set the dimensions of latent factors as $L = [3, 3]$ for this dataset.

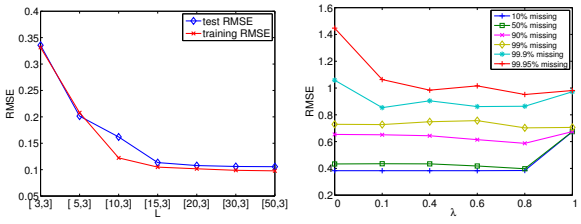
Motes Dataset The Motes dataset consists of 4 data types collected from 54 sensors deployed at the Intel Berkeley Research Lab over a month [1]. The data types include three room conditions (i.e., temperature, humidity, light) and the sensor voltage level. Considering that the running time of some baselines is quite slow, we evaluate all the algorithms only on the first-day measurement data with a total of 2880 time steps. We construct the contextual sensor network in the same way as in [5] and ignore the contextual information in the type mode for simplicity. By default, we set the dimensions of latent factors as $L = [15, 3]$ in the evaluations.

SPMD Dataset The Safety Pilot Model Deployment (SPMD) dataset was collected to understand the potential safety benefits of connected vehicle safety technologies [2]. It consists of multimodal and multidimensional traffic data mostly within the test site of Ann Arbor, Michigan. The available one-day sample dataset contains trajectory traces of monitored vehicles collected on April 11, 2013, including a total of 369 trips and 124 vehicles ranging from passenger cars, trucks to buses. The trip duration ranges from seconds to hours with an average around 20 mins. We select the trips lasting over 20 mins, which result in a total of 52 trips. Then, we set the departure time of each trip as the first time step and sample two-dimensional location coordinates every one second with a duration of 20 minutes, resulting in a total of 1200 time steps. Theoretically, if we have sufficient trips, we can learn the contextual matrix for trips based on drivers' behaviors. With only one-day data available in our evaluations, we define the trip contextual matrix based on the cosine similarities among trip trajectories. Since the longitude and the latitude are not strongly correlated practically, we ignore the contextual constraint in the location mode. In the evaluations, we set the dimensions of latent factors as $L = [30, 2]$ by default.

C. Evaluation Metrics. To evaluate the effectiveness, we perform the vectorization of the observed time series tensors and the estimated time series tensors. Then, we calculate the root mean squared error (RMSE) [5] between them.

5.2 Sensitivity Results

We perform the parametric studies w.r.t. the two hyperparameters in our algorithm - the dimensions of latent factors L and the contextual weight λ. Our results shown in Fig. 4 indicate that the performance of our algorithm is robust in a large range of both parameters.



(a) Impact of L. (b) Impact of λ.

Figure 4: Parameter sensitivity results.

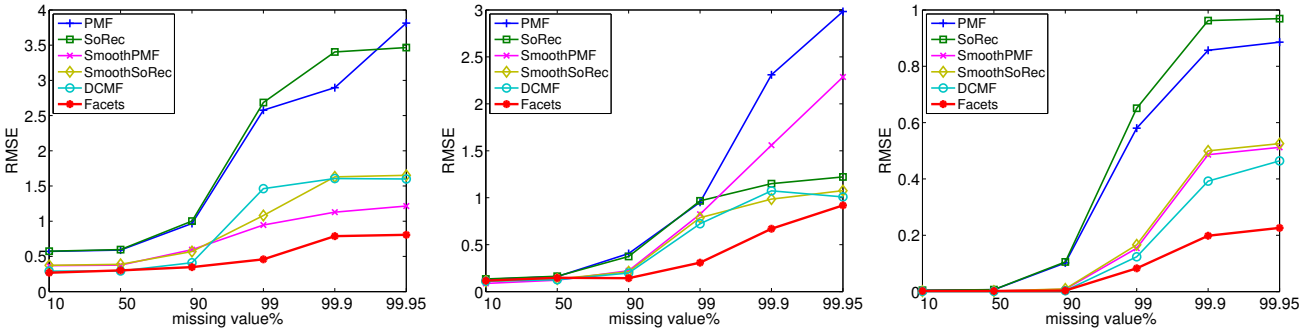
Fig. 4(a) shows the impact of L on the training parts and the test parts of the Motes dataset. We keep the latent dimension of the second mode fixed in 3. As we can see, the RMSEs of our method fluctuate within a range of less than 0.01 between [15, 3] and [50, 3]. Fig. 4(b) shows the impact of λ on the Motes dataset. We vary the the contextual weight from zero to one. If λ equals zero, the contextual information is ignored. If λ equals one, only the contextual information is included to learn $\mathcal{U}$. As we can see, the RMSEs of our method fluctuate within a range of less than 0.05 in general cases (i.e., $\lambda > 0$); and in the meanwhile, the RMSEs significantly increase if we ignore the contextual information (i.e., $\lambda = 0$), especially when time series data is very sparse, which indicates the importance of modeling the contextual network information.

5.3 Effectiveness Results

To evaluate the effectiveness of the algorithms in the task of missing value recovery, we incrementally generate training sets and test sets with an increasing amount of test data $(10\%, 50\%, ...)$ within time series. As a result, the subsequent test set always contains the previous test data.

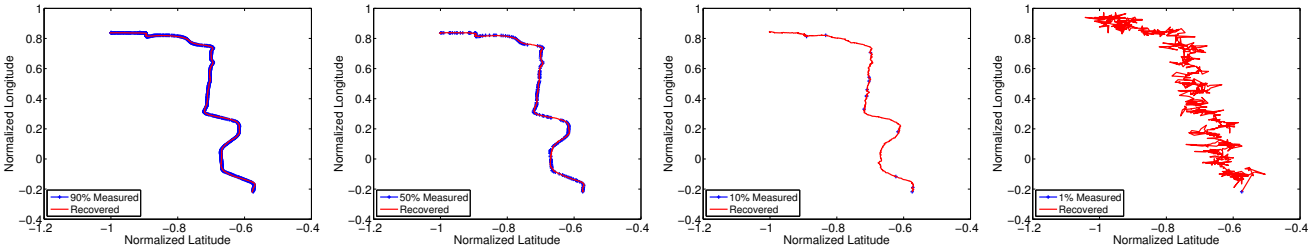
Fig. 5 shows the imputation results on the three datasets. We can clearly see that our FACETS significantly outperforms the others, especially when the percentage of the missing values is large. Fig. 6 presents our FACETS's imputation results of a trip instance in the SPMD dataset. The x-axis and the y-axis denote the normalized latitude and longitude, respectively. In each subfigure, the blue curve with crosses represents the training data with missing values and the red curve denotes the recovered trace by the FACETS algorithm. As we can see, FACETS achieves good approximations with a few training data (90% - 10%). Even with only 1% training data (i.e., 6 pairs of x-y coordinates), our algorithm also achieves good performance shown in Fig. 6(d).

Fig. 7 shows the prediction results. In the evaluations of the SST dataset and the Motes dataset, we use the first k time steps of time series as training data, where $k = \lfloor T * (1 - prediction\%) \rfloor$. For the SPMD dataset, the most common scenario is to predict the trace of a target trip based



(a) SST (b) Motes (c) SPMD

Figure 5: Effectiveness of imputation. The lower the better.



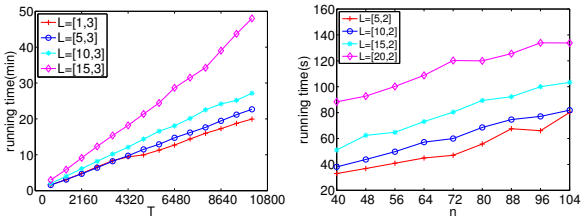
(a) 90% training data (b) 50% training data (c) 10% training data (d) 1% training data

Figure 6: FACETS's imputation results of one trip instance in SPMD Dataset.

on some completed trips in history. Therefore, we randomly select prediction% of the trips and hide their last prediction% of time steps as test data. As we can see from Fig. 7, our FACETS algorithm is quite robust for different prediction ratios, ranging from 10% to 40%. Our FACETS algorithm and its special case, DCMF, achieve much higher accuracy than others. Fig. 8 demonstrates FACETS's prediction results on two sensor instances from the Motes dataset.

5.4 Efficiency Results

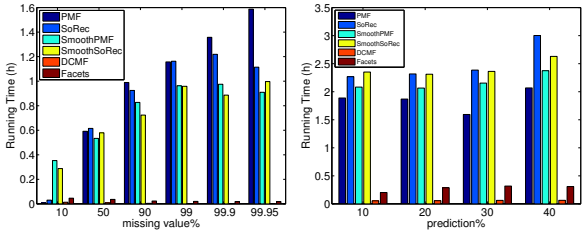
We test the scalability of the FACETS algorithm on a number of subsets of the Motes dataset and the SPMD dataset. As Fig. 9(a) shows, our proposed FACETS algorithm scales linearly w.r.t. the sequence length T, which is consistent with our complexity analysis in section 4. Fig. 9(b) demonstrates that the running time of FACETS is close to linear w.r.t. the aggregated dimensions of time series n.



(a) Motes Dataset (b) SPMD Dataset

Figure 9: Scalability.

We also compare the running time of our FACETS algorithm with other baselines. Fig. 10(a) presents the running time of the imputation experiments on the SST dataset. Combining with the results of Fig. 5(a), we can see that FACETS and DCMF yield superior effectiveness while spend much less time in comparison with PMF, SoRec, SmoothPMF and SmoothSoRec. We can get similar observations from Fig. 10(b), which presents the running time of the prediction experiments on the SPMD dataset.



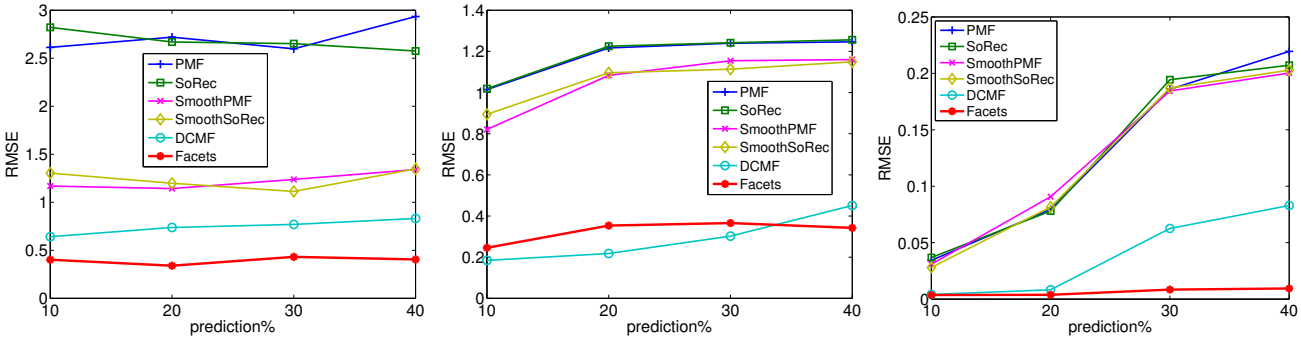
(a) Imputation on SST (b) Prediction on SPMD

Figure 10: Efficiency

6. RELATED WORK

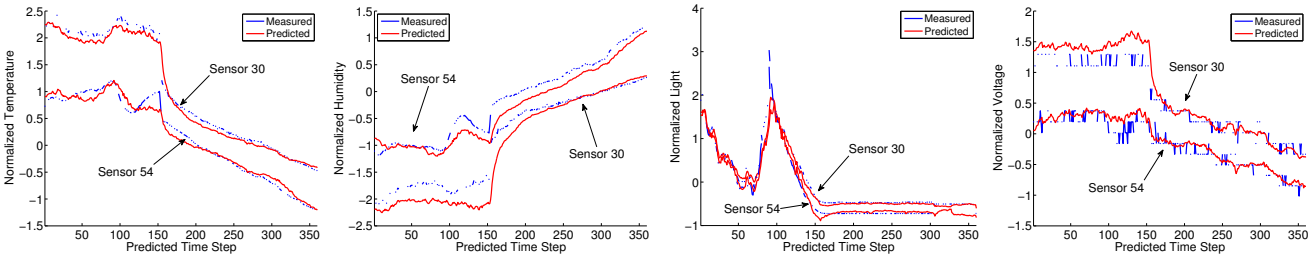
There are plenty of work in mining time series data [12], including representation [24, 22], classification [30, 8, 11], outlier detection [16], etc. Li et al. proposed DynaMMo for mining coevolving time series with missing values based on linear dynamic systems [17]. Matsubara et al. developed AutoPlait, which combined a multi-level chain model and a cost model to find typical patterns and meaningful segments in multiple time series [20]. Shieh et al. proposed a multi-resolution symbolic representation to index time series datasets for fast exact search and approximate search [24].

Tensor decompositions have been successfully applied in many domains including signal processing, computer vision, neuroscience and data mining [14]. Most of tensor compositions are based on tucker decomposition [27], canonical/parrallel-factors (CP) decomposition [10, 6], and multilinear PCA [18].



(a) SST (b) Motes (c) SPMD

Figure 7: Effectiveness of prediction. The lower the better.



(a) Temperature (b) Humidity (c) Light (d) Voltage

Figure 8: FACETS's prediction for Sensor 30 and Sensor 54 in Motes Dataset.

Sun et al. formally introduced tensors in time-evolving settings and proposed dynamic and streaming tensor analysis, which learned a latent, low-dimensional core tensor and a set of projection matrices to summarize large tensor sequences and detect patterns [25]. Xiong et al. extended two-dimensional collaborative filter problems into three-order tensor space after introducing the time factor [28]. They proposed a bayesian probabilistic tensor factorization method to compute CP decompositions to get the latent factors in each mode. Rogers et al. presented multilinear dynamical systems to model tensor time series based on linear dynamic systems [23]. All of these models ignore contextual information embedded in time series data. Bahadori et al. proposed a neat and flexible framework, under which either spatial or temporal information can be realized/modeled, respectively [4]. Yet, how to simultaneously model these two aspects was not answered - which is exactly one major advantage of our work.

If time series tensor is degraded to time series matrix, our work is also related to matrix factorization, which is widely applied in the recommendation systems. In the recommendation problems, the matrix factorization methods find low-rank latent factors to represent users and items, which can be fit into the user-item rating matrix and make rating prediction [21, 19, 15, 26, 29]. To improve the prediction accuracy, some side information, such as the user social network and/or the item-item similarity are included in the probabilistic factor analysis [19, 29]. Recently, dynamic matrix factorization methods have been proposed to capture the evolving user preferences [7, 26]. For example, Sun et al. applied a dynamic state space model on probabilistic matrix factorization to track the temporal dynamics of the user latent factor[26] or model the changes of user preferences on the user-item adoption problem [7].

7. CONCLUSION

In this paper, we have proposed a comprehensive method, FACETS, to address three prominent challenges in mining a network of high-order time series data: (a) high-order; (b) contextual constraints; and (c) temporal smoothness. Based on tensor factorization and multilinear dynamic systems, for the first time, our algorithm effectively and efficiently solve all of the three challenges at the same time. The experimental evaluations on three real datasets demonstrated the effectiveness and the scalability of our algorithm. For future work, we are interested in applying our FACETS method to other time series mining tasks, such as anomaly detection; and finding alternative to the EM algorithm, such as approximate inference and sampling.

8. ACKNOWLEDGMENT

This material is supported by the National Science Foundation under the grant number IIS1017415 and CNS-0904901, by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053, by National Institutes of Health under the grant number R01LM011986, Region II University Transportation Center under the project number 49997-32-25 and 49997-33-25.

The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

9. REFERENCES

[1] Motes dataset. http://db.csail.mit.edu/labdata/labdata.html.

[2] Safty pilot model deployment. https://www.its-rde.net/showds?dataEnvironmentNumber=10014.

[3] Tropical atmosphere ocean project. http://www.pmel.noaa.gov/tao/data_deliv/deliv.html.

[4] M. T. Bahadori, Q. R. Yu, and Y. Liu. Fast multivariate spatio-temporal analysis via low rank tensor learning. In *NIPS*, 2014.

[5] Y. Cai, H. Tong, W. Fan, and P. Ji. Fast mining of a network of coevolving time series. In *SDM*, 2015.

[6] J. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of 'eckart-young' decomposition. *Psychometrika*, 35(3):283–319, 1970.

[7] F. C. T. Chua, R. J. Oentaryo, and E.-P. Lim. Modeling temporal adoptions using dynamic matrix factorization. In *ICDM*, 2013.

[8] J. Gao, B. Ding, W. Fan, J. Han, and P. S. Yu. Classifying data streams with skewed class distributions and concept drifts. *IEEE Internet Computing*, 12(6):37–49, 2008.

[9] A. K. Gupta and D. K. Nagar. *Matrix Variate Distributions*. CRC Press, 1999.

[10] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an 'explanatory' multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16(1):84, 1970.

[11] B. Hu, Y. Chen, J. Zakaria, L. Ulanova, and E. Keogh. Classification of multi-dimensional streaming time series by weighting each classifier's track record. In *ICDM*, 2013.

[12] E. Keogh. Tutorial: Machine learning in time series databases (and everything is a time series!). In *AAAI*, 2011.

[13] T. G. Kolda. *Multilinear operators for higher-order decompositions*. 2006.

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

[15] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[16] J.-G. Lee, J. Han, and X. Li. Trajectory outlier detection: A partition-and-detect framework. In *ICDE*, 2008.

[17] L. Li, J. McCann, N. S. Pollard, and C. Faloutsos. Dynammo: Mining and summarization of coevolving sequences with missing values. In *KDD*, 2009.

[18] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. Multilinear principal component analysis of tensor objects for recognition. In *ICPR*, 2006.

[19] H. Ma, H. Yang, M. R. Lyu, and I. King. Sorec: social recommendation using probabilistic matrix factorization. In *CIKM*, 2008.

[20] Y. Matsubara, Y. Sakurai, and C. Faloutsos. Autoplait: Automatic mining of co-evolving time sequences. In *SIGMOD*, 2014.

[21] A. Mnih and R. Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, 2007.

[22] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming pattern discovery in multiple time-series. In *VLDB*, 2005.

[23] M. Rogers, L. Li, and S. J. Russell. Multilinear dynamical systems for tensor time series. In *NIPS*, 2013.

[24] J. Shieh and E. Keogh. isax: indexing and mining terabyte sized time series. In *KDD*, 2008.

[25] J. Sun, D. Tao, and C. Faloutsos. Beyond streams and graphs: Dynamic tensor analysis. In *KDD*, 2006.

[26] J. Z. Sun, K. R. Varshney, and K. Subbian. Dynamic matrix factorization: A state space approach. In *ICASSP*, 2012.

[27] L. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[28] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, 2010.

[29] Y. Yao, H. Tong, G. Yan, F. Xu, X. Zhang, B. Szymanski, and J. Lu. Dual-regularized one-class collaborative filtering. In *CIKM*, 2014.

[30] L. Ye and E. Keogh. Time series shapelets: a new primitive for data mining. In *KDD*, 2009.

APPENDIX

A. Details of inferring latent factors

In Step 6-9 and Step 18-20 of Algorithm 1, the expectations of $\mathbf{V}_j^{(m)}$ are calculated with the follows equations:

$$\boldsymbol{\nu}_j^{(m)} = \Upsilon(\mathbf{U}^{(m)})'\mathbf{S}_j^{(m)},\ \Upsilon = \left[(\mathbf{U}^{(m)})'\mathbf{U}^{(m)} + (\xi^{(m)})^2\sigma_{V_m}^{-2}\right]^{-1},$$
$$\mathbb{E}[\mathbf{V}_j^{(m)}] = \boldsymbol{\nu}_j^{(m)},\quad \mathbb{E}[\mathbf{V}_j^{(m)}(\mathbf{V}^{(m)})_j'] = \Upsilon + \boldsymbol{\nu}_j^{(m)}(\boldsymbol{\nu}_j^{(m)})'. \quad (21)$$

In Step 14, Algorithm 1 infers the expectations and the covariances of latent factors $\mathcal{Z}_t$, which is difficult in the tensor spaces. Instead, FACETS performs the vectorizations and matricizations with Eq. (10), which reduces to find the expectations and covariances of $\text{vec}(\mathcal{Z}_t)$. For clarity, in the following equations, we write $\text{vec}(\mathcal{X}_t)$ as $\mathbf{x}_t$, $\text{vec}(\mathcal{Z}_t)$ as $\mathbf{z}_t$ and formulate $\text{vec}(\mathcal{W}_t)$ as $\mathbf{W}_t, (t = 1, .., T)$. We use $\mathbf{U}$, $\mathbf{B}$ to denote matricization results of $\text{mat}(\mathcal{U})$, $\text{mat}(\mathcal{B})$, respectively.

First we only count on the observed time series data. We use $\mathbf{o}_t$ to denote the indices of the observed entries of $\mathbf{x}_t$ and $\mathbf{x}_t^*$ and $\mathbf{H}_t$ are defined as follows:

$$\mathbf{o}_t = \{i|\mathbf{W}_{it} > 0, i = 1, ..., n\},\ \mathbf{x}_t^* = \mathbf{x}_t(\mathbf{o}_t, :),\ \mathbf{H}_t = \mathbf{U}(\mathbf{o}_t, :). \quad (22)$$

Let $p(\mathbf{z}_t|\mathbf{x}_1, ..., \mathbf{x}_t) = \mathcal{N}(\mathbf{z}_t|\boldsymbol{\mu}_t, \boldsymbol{\Psi}_t)$ and $p(\mathbf{z}_t|\mathbf{x}_1, ..., \mathbf{x}_T) = \mathcal{N}(\mathbf{z}_t|\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\Psi}}_t)$, Step 14 of Algorithm 1 is calculated with the following equations:

$$\mathbf{K}_1 = \sigma_0^2\mathbf{H}_1'(\sigma_0^2\mathbf{H}_1\mathbf{H}_1' + \sigma_R^2\mathbf{I})^{-1},$$
$$\boldsymbol{\mu}_1 = \mathbf{z}_0 + \mathbf{K}_1(\mathbf{x}_1^* - \mathbf{H}_1\mathbf{z}_0),\ \boldsymbol{\Psi}_1 = \sigma_0^2\mathbf{I} - \mathbf{K}_1\mathbf{H}_1,$$
$$\mathbf{P}_{t-1} = \mathbf{B}\boldsymbol{\Psi}_{t-1}\mathbf{B}' + \sigma_O^2\mathbf{I},\ \mathbf{K}_t = \mathbf{P}_{t-1}\mathbf{H}_t'(\mathbf{H}_t\mathbf{P}_{t-1}\mathbf{H}_t' + \sigma_R^2\mathbf{I})^{-1},$$
$$\boldsymbol{\mu}_t = \mathbf{B}\boldsymbol{\mu}_{t-1} + \mathbf{K}_t(\mathbf{x}_t^* - \mathbf{H}_t\mathbf{B}\boldsymbol{\mu}_{t-1}),\ \boldsymbol{\Psi}_t = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\mathbf{P}_{t-1},$$
$$\hat{\boldsymbol{\mu}}_t = \boldsymbol{\mu}_t + \mathbf{J}_t(\hat{\boldsymbol{\mu}}_{t+1} - \mathbf{B}\boldsymbol{\mu}_t),$$
$$\hat{\boldsymbol{\Psi}}_t = \boldsymbol{\Psi}_t + \mathbf{J}_t(\hat{\boldsymbol{\Psi}}_{t+1} - \mathbf{P}_t)\mathbf{J}_t',\ \mathbf{J}_t = \boldsymbol{\Psi}_t\mathbf{B}'(\mathbf{P}_t)^{-1},$$
$$\mathbb{E}[\mathbf{z}_t] = \hat{\boldsymbol{\mu}}_t,\ \text{cov}(\mathbf{z}_t, \mathbf{z}_{t-1}) = \hat{\boldsymbol{\Psi}}_t\mathbf{J}_{t-1}',\ \text{cov}(\mathbf{z}_t, \mathbf{z}_t) = \hat{\boldsymbol{\Psi}}_t,$$
$$\mathbb{E}[\mathbf{z}_t\mathbf{z}_{t-1}'] = \hat{\boldsymbol{\Psi}}_t\mathbf{J}_{t-1}' + \hat{\boldsymbol{\mu}}_t\hat{\boldsymbol{\mu}}_{t-1}',\ \mathbb{E}[\mathbf{z}_t\mathbf{z}_t'] = \hat{\boldsymbol{\Psi}}_t + \hat{\boldsymbol{\mu}}_t\hat{\boldsymbol{\mu}}_t'. \quad (23)$$