

Tensor Factorization for Heterogeneous and Spatio-Temporal Data

異種・時空間データ解析のためのテンソル分解法

Koh Takeuchi

竹内 孝

Abstract

We observe various kinds of interesting events occurring in our daily lives, and try to represent these events in a simple manner. A relational model is one of the *de facto* systems used to digitally store these events due to its flexible representation abilities compared to other data models that have strict predefined structures. We try to discover interpretable patterns from the relational data, which is a traditional way to understand the latent characteristic or trend in subjective phenomena. These patterns are employed to recover missing records that have not observed in relational data. We can obtain better predictions of future events by leveraging the latent trends than by directly using the observed data. However, in practical data analysis, we often encounter situations where the relational data are sparse and noisy, which means that the data contain fewer messy records than the total number of combinations of the attributes. Pattern extraction methods may provide undesired results due to the lack of information sources.

In this dissertation, we attempt to solve the sparseness and noise problems by developing data analysis methods that adopt side information to discover interpretable and useful patterns. There are three essential challenges in this dissertation: (1) develop a method to merge heterogeneous information of associated relational data to extract patterns, (2) leverage spatio-temporal structures that can be represented as graphs or groups of data attributes to encourage latent patterns to be smooth, and (3) learn dependency for attributes in a data-driven manner and modify spatio-temporal structures to capture the patterns suitable for prediction problems.

To tackle these challenges, we propose tensor factorization methods that extract an interpretable pattern by decomposing a tensor into latent factors. For the first challenge, we develop Non-negative Multiple Matrix Factorization (NM²F), which decomposes a target matrix and auxiliary matrices simultaneously. As an exten-

sion of NM²F, we propose Non-negative Tensor Factorization (NMTF) to deal with higher-order arrays consisting of more than two types of attributes. By modeling the common latent factors shared between the target and the others, we can extract factors that are better than those extracted using the existing approaches. Moreover, our approach enables the extraction of shared patterns within and between tensors. For the second challenge, we propose Structurally-Regularized Non-negative Tensor Factorization, which leverages graph and group structures of data attributes to impose smoothness on the latent factors. We define a structured regularizer and propose an efficient minimization procedure to estimate the latent factors for this method. For the final challenge, we present Autoregressive Tensor Factorization for spatio-temporal relational data analysis. This factorization method simultaneously learns latent the factors and factor specific spatial and temporal autocorrelations. By leveraging extracted factors and autocorrelations, we are able to obtain accurate spatio-temporal predictions compared to the existing spatio-temporal data analysis methods.

Acknowledgments

First, I would like to express my great appreciation to all the people I met on my doctor's course. I would like to thank my doctor's course adviser, Prof. Hisashi Kashima. He kindly gave enormous supports through my doctor's course. His scintillating advice was invaluable for accomplishing my doctoral research. I would like to appreciate Prof. Akihiro Yamamoto and Prof. Hidetoshi Shimodaira who were the committees of this thesis. They reviewed this thesis in detail and gave insightful comments that contributed to improve this thesis.

I would like to thank collaborators in my earlier articles. I would like to express gratitude to Dr. Ryota Tomioka for introducing me to tensor factorization methods. I would like to greatly thank Prof. Yoshinobu Kawahara for instructing structured regularizers. I would like to give appreciation to Dr. Katsuhiko Ishiguro for guiding me on this research field. I would like to thank Dr. Tomoharu Iwata, Dr. Akisato Kimura, and Dr. Hiroshi Sawada for helpful comments. I would like to be grateful to Dr. Naonori Ueda for stimulus and encouraging discussions on my doctoral studies.

Finally, I would like to thank my family for their continuous encouragement and many years for support.

Contents

1	Introduction	17
1.1	Challenges	19
2	Preliminaries	23
2.1	Non-negative Tensor Factorization	23
2.2	Generalized Kullback-Leibler Divergence	25
2.3	Graph Laplacian Regularizer	25
3	Non-negative Multiple Matrix Factorization	27
3.1	Introduction	27
3.2	Related Work	28
3.3	Non-negative Multiple Matrix Factorization	29
3.3.1	Problem Formulation	29
3.3.2	Proposed Framework	30
3.3.3	NMF Framework	32
3.3.4	Parameter Estimation	32
3.3.5	NM ² F as probabilistic generative model	33
3.4	Experiments	34
3.4.1	Evaluation Measures	34
3.4.2	Synthetic data experiment	35
3.4.3	Real Data Experiment	36
3.5	Summary	40
4	Non-negative Multiple Tensor Factorization	41
4.1	Introduction	41

4.2	Related Work	43
4.3	Proposed Method	44
4.3.1	Non-negative Multiple Tensor Factorization (NMTF)	44
4.3.2	Multiplicative Update Rules	47
4.4	Parameter Estimation	48
4.4.1	NMTF as Probabilistic Generative Model	49
4.5	Experiments	50
4.5.1	Evaluation Measure	51
4.5.2	Synthetic Data Experiment	52
4.5.3	Real Data Experiments	52
4.5.4	Results	57
4.6	Summary	62
5	Structurally Regularized Non-negative Tensor Factorization for Spatio-temporal Pattern Discoveries	63
5.1	Introduction	63
5.2	Proposed model	66
5.3	Parameter estimation	68
5.4	Related Works	70
5.5	Experiments	71
5.6	Summary	74
5.7	Appendix	75
6	Autoregressive Tensor Factorization for Spatio-temporal Predictions	83
6.1	Introduction	83
6.2	Preliminaries	86
6.2.1	Tensor factorization methods	86
6.2.2	Autoregressive models	88
6.3	Proposed method	89
6.3.1	Autoregressive tensor factorization	89
6.3.2	Spatial autoregressive regularizer	90
6.3.3	Parameter estimation	92
6.4	Related works	94

6.5	Experiments	96
6.5.1	City traffic demand prediction experiments	96
6.5.2	Air quality prediction experiments	101
6.6	Summary	107
7	Conclusion	109
7.1	Future Work	111
	List of Publications	113

List of Figures

1.1	Tensor factorization methods extract pattern as latent factors from a relational database	18
1.2	Multiple tensor factorization for analyzing sparse tensor data.	20
1.3	Utilization of the spatio-temporal structures as a regularizer.	21
1.4	Learning spatio-temporal dependencies for predictions.	22
2.1	NTF factorizes a tensor $\mathcal{X}$ into factor matrices $\mathbf{U}^{(1)}$, $\mathbf{U}^{(2)}$, and $\mathbf{U}^{(3)}$.	24
3.1	The original matrix $\mathbf{X}$ is factorized to the bases $\mathbf{W}$ and the coefficients $\mathbf{H}$ with the row wise additional matrix $\mathbf{Y}$ and the column wise additional matrix $\mathbf{Z}$	30
3.2	Graphical model of Non-negative Multiple Matrix Factorization. $\mathbf{X}$ and $\mathbf{Z}$ are generated from the same bases $\mathbf{W}$ but different coefficients $\mathbf{H}$ and $\mathbf{B}$. $\mathbf{Y}$ is generated from bases $\mathbf{A}$ and coefficients $\mathbf{H}$ shared by $\mathbf{X}$	34
3.3	The test log likelihood versus the scaling parameters α (horizontal) and β (vertical).	36
3.4	Example of a Togetter story shortened by authors.	38
3.5	Artists and Tags with the highest values in bases.	39
3.6	Users and Words with the highest values in bases.k	39
4.1	$\mathcal{Y}$, $\mathcal{A}^{(1)}$, $\mathcal{A}^{(2)}$ and $\mathcal{A}^{(3)}$ are included in $\mathcal{X}$. Transparent elements of $\mathcal{X}$ are undefined regions. Factor matrices $\mathbf{U}_y$, $\mathbf{T}_y$, $\mathbf{V}_y$ are shared among the target tensor and additional tensors.	45

4.2	Graphical model of NMTF. Tensors $\mathcal{Y}, \mathcal{A}^{(1)}, \mathcal{A}^{(2)}$ and $\mathcal{A}^{(3)}$ share the factor matrices $\mathbf{U}, \mathbf{T}$ and $\mathbf{V}$. Shared factor matrices are employed to generate elements of both the target tensor and auxiliary tensors. . .	51
4.3	Extracted factors for three different bases learned from the Beeradvocate data set.	59
4.4	Factors corresponds to four bases extracted from the YELP data set.	61
5.1	Example for a non-negative tensor factorization method on analysis of a traffic-flow data set, where latent patterns for location, time, and day modes are extracted in the latent factor.	64
5.2	Time factors of Proposed 1 on the traffic flow data	77
5.3	Time factors of Multi+Lap on the traffic flow data	77
5.4	Day factors of Proposed 1 on the traffic flow data	77
5.5	Day factors of Multi+Lap on the traffic flow data	77
5.6	A spatial pattern of the blue factor of Proposed 1 on the traffic flow data	78
5.7	A spatial pattern of the blue factor of Multi+Lap on the traffic flow data	78
5.8	Time factors of Proposed 1 on the bike-sharing data of Washington D.C.	79
5.9	Time factors of Multi+Lap on the bike-sharing data of Washington D.C.	79
5.10	Day factors of Proposed 1 on the bike-sharing data of Washington D.C.	79
5.11	Day factors of Multi+Lap on the bike-sharing data of Washington D.C.	79
5.12	A spatial pattern of the yellow factor of Proposed 1 on the bike-sharing data of Washington D.C.	80
5.13	A spatial pattern of the yellow factor of Multi+Lap on the bike-sharing data of Washington D.C.	80

5.14	Minimum s/t -cut problem of Problem (5.12). Given graph $G = (V', E')$ for our proposed method, capacities of edges $c(v'_i, v'_j)$ are defined as: $c(s, u_k) = \theta_k$, $c(v_i, v_j) = w_{i,j}$, $c(u_k, v_i) = c_{k,i}$, $c(s, v_i) = z_i - (\gamma - \alpha)$ if $z_i > \gamma - \alpha$, and $c(v_i, t) = (\gamma - \alpha) - z_i$ if $z_i < \gamma - \alpha$. Nodes u_k $k = (1, \dots, K)$ are hyper nodes that correspond to the groups g_k . And s, t , and v_i are source, sink, and parameters nodes, respectively.	81
6.1	Tensor factorization for multiple types of spatio-temporal data. Blocks $\mathcal{X}_1$, $\mathcal{X}_2$, and $\mathcal{X}_3$ correspond to problems of interpolation of unknown locations, prediction of known locations, and prediction of unknown locations, respectively.	85
6.2	Two-dimensional autoregressive model and our directed autoregressive model. Our method splits the 2π angle into N discrete parts.	91
6.3	Examples of results of NY taxi data (backgrounds with green, blue, and red correspond to $\mathcal{X}_2$, $\mathcal{X}_1$, and $\mathcal{X}_3$, respectively)	102
6.4	Examples of results of bike-sharing system data in NY (backgrounds with green, blue, and red correspond to $\mathcal{X}_2$, $\mathcal{X}_1$, and $\mathcal{X}_3$, respectively)	103
6.5	Temporal and spatial factors extracted with our proposed method (DAR+NN) from NY taxi data	104
6.6	Temporal and spatial factors extracted with our proposed method (DAR+NN) from bike-sharing system data in NY	105
6.7	Examples of results for air quality data in Beijing (backgrounds with green, blue, and red correspond to $\mathcal{X}_2$, $\mathcal{X}_1$, and $\mathcal{X}_3$, respectively)	108

List of Tables

3.1	Comparison on the average test log likelihood of non-zero entries among different sparseness. Means and one standard deviations of four data sets are presented.	35
3.2	Comparison of average test log likelihood of non- zero entries among different data sets. Scores are divided by $\times 10^{-3}$	39
4.1	Synthetic data experiment : The average test log likelihood for different sparseness values of the target tensor $\mathcal{Y}$	53
4.2	Synthetic data experiment: The average test log likelihood of the target tensor $\mathcal{Y}$ for different numbers of auxiliary tensors. The sparseness of $\mathcal{Y}$ is set at 99%. The numbers of R is set to 5, 10, and 20.	54
4.3	Correspondence of target and auxiliary tensors for real-world data sets.	56
4.4	Real-world data sets: Mean and standard deviation of average test log likelihoods on the target tensor $\mathcal{Y}$. Large values mean a good estimation of the hidden structures of $\mathcal{Y}$	57
5.1	NRAME for the traffic flow data of our proposed method (Proposed 1), our proposed method with the graph Laplacian regularizer (Proposed 2), NTF estimated by ADMM (ADMM), NTF with the graph Laplacian regularizer estimated by the multiplicative update rule (Multi+Lap), and NTF (Multi)	72
5.2	NRAME for the bike-sharing record data of Washington D.C.	72
5.3	NRAME for the bike-sharing record data of New York	73

5.4	ND for the traffic flow data of our proposed method (Proposed 1), our proposed method with the graph Laplacian regularizer (Proposed 2), NTF estimated by ADMM (ADMM), NTF with the graph Laplacian regularizer estimated by the multiplicative update rule (Multi+Lap), and NTF (Multi)	73
5.5	ND for the bike-sharing record data of Washington D.C.	74
5.6	ND for the bike-sharing record data of New York	74
6.1	Results of NY taxi data	99
6.2	Results of bike-sharing system data in NY	99
6.3	Results of bike-sharing system data in Washington D.C.	100
6.4	Results of air quality data in Beijing	106
6.5	Results of air quality data in Shanghai	106

Chapter 1

Introduction

When we encounter an interesting event, we often attempt to describe or record it to share our observation with others. Some may utilize a natural language sentence to express the event or take a photo snap to save what has happened visually. Others may check their watch to confirm the time or measure the position to identify the location of the event. The relational model [34] is one of the general approaches used to record such events as a set of simple tuples. For example, suppose that we observe an event in which a person buys a book from a store on a particular day, we can uniquely record this event by using a tuple containing the values of attributes such as *who*, *what*, *where*, and *when*.

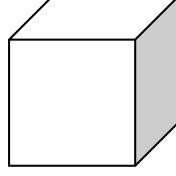
A relational database is a *de facto* database system for flexibly storing and managing the records of the relational model. After collecting a large amount of records, one might want to reveal a tendency or a trend in the database for understanding the objective phenomena. However, relational data consist of a messy and noisy collection of observed events, and it is hard to grasp such tendencies by checking the records or counting the appearances on a specific combination of the attribute values. To overcome this difficulty, pattern extraction procedures, such as clustering, have been widely utilized for revealing hidden patterns from an unstructured relational database.

A tensor [61], which is also known as a higher-order array is a common way to represent relational data numerically. For the aforementioned example where the number of attributes was four, we can construct a fourth-order tensor consisting of *who*, *what*, *where*, and *when* modes, and can represent the event by setting an

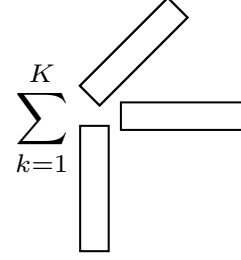
Relational database

Who	What	Where	When
Bob	Book	Shop	7 Dec. 2018
Alice	Pen	Store	9 Dec. 2018
Bob	Notebook	Store	7 Dec. 2018
...	...	...	...

Tensor/Matrix (Higher-Order Array)



Latent Factors (Patterns)



Tensor/Matrix Factorization

Figure 1.1: Tensor factorization methods extract pattern as latent factors from a relational database

element of the tensor corresponding with the record to a specific value. When the number of attributes is one or two, a tensor is equal to a vector or a matrix. Tensor factorization [51, 104] is an unsupervised machine learning method to automatically extract latent patterns from a tensor representation of relational data. This method has been successfully applied in various applicational domains to reveal reasonable patterns by decomposing a tensor into a few essential latent factors. We illustrate an example of a tensor factorization in Fig. 1.1.

We often need to analyze relational databases whose values consist of only non-negative values, such as values representing the number of occurrences of a specific type of events [48]. In such a case, we can naturally add an additional assumption for latent factors, that is, the latent factors consists of only non-negative values. To make such assumptions on factorization methods, Non-negative Matrix Factorization (NMF) [63, 64], which is a traditional unsupervised machine learning technique, has been proposed to decompose a non-negative matrix into a set of base and coefficient factors under a non-negative constraint. Non-negative Tensor Factorization (NTF) [33] is a generalization of NMF used to deal with a non-negative value tensor. Thanks to the non-negative constraints, the factors extracted by NMF and NTF reveal reasonably interpretable patterns, which help us understand the tendencies and trends hidden in a relational database easily. NMF and NTF with sparse constraints are also known for extracting reasonable components from noisy data.

1.1 Challenges

In practical data analysis, we often encounter situations where a tensor contains only a few observed elements. This problem often appears because the number of records in real-world relational databases is less of the combination of the number attributes. Despite the desired advantage for extracting interpretable patterns as latent factors from a non-negative relational database, NMF tends to give undesired results in the case of highly sparse data because the information included in the data is insufficient to decompose. NTF performs poorly when the tensor is exceptionally sparse, which is often the case with real-world data and higher-order tensors. In this dissertation, we propose several approaches to tackle this problem by raising the following fundamental questions.

1. How can we utilize auxiliary relational database as an auxiliary matrix/tensor to increase the available information for extracting latent factors?
2. How can we employ the structure information of attributes, such as graphs and groups of attributes for extracting smooth latent factors from a few records?
3. Is it possible to learn factor-specific dependencies and employ it for extracting reasonable factors?

A natural approach to deal with the problem of sparseness is to utilize other data that would record different events having a relationship with the data under consideration. Our key idea is that we can ease this problem if complementary data that can be utilized for the estimation of the latent factors are available.

In Chapter 3, we propose a novel matrix factorization method called Non-negative Multiple Matrix Factorization (NM²F), which utilizes complementary data as additional matrices that share the row or column indices of the original matrix. The data sparseness is improved by decomposing the original and additional matrices simultaneously, since additional matrices provide information about the bases and coefficients. We formulate NM²F as a generalization of NMF, and then present a parameter estimation procedure derived from the multiplicative update rule. We examined NM²F in both synthetic and real data experiments. The effect of the additional matrices appeared in the improved NM²F performance. We also confirmed

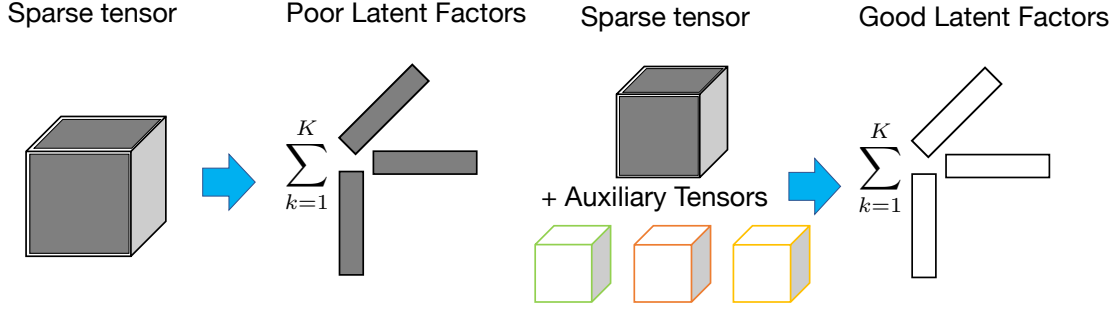


Figure 1.2: Multiple tensor factorization for analyzing sparse tensor data.

that the latent factors that NM²F obtained from the real data were intuitive and reasonable thanks to the non-negative constraint. Moreover, in Chapter 4, we propose Non-negative Multiple Tensor Factorization (NMTF), which decomposes the target tensor and auxiliary tensors simultaneously. Auxiliary data tensors will compensate for the sparseness of the target data tensor. The proposed methods allow us to examine the target data from several different aspects, because we can obtain the factors of the auxiliary tensors. We confirmed experimentally that NM²F and NMTF performs better than NMF/NTF in terms of reconstructing the given data. In Chapter 4, we demonstrate that NMTF can successfully extract the spatio-temporal patterns of people’s daily life such as leisure, drinking, and shopping activities by analyzing the latent factors extracted from online review datasets.

Auxiliary information takes another form to represent the relationship between tensor attributes. Suppose that we are dealing with a relational database whose attributes correspond to *when* and *where*. Some may naturally consider that a pair or group of events that occurred within a short duration would contain similar information based on common knowledge of the temporal domain. Others can naturally consider that a set of events would have underlying relationships if these events are observed within a small radius. We use this sort of common knowledge in our daily life; however, the existing NTF based methods are not aware of such relationships between attributes and treat parameters in latent factors independently. Due to this limitation of the existing NTF methods, they often fall short of learning the latent factors when a large portion of the missing observation exists in the relational data.

In Chapter 5, we present a novel NTF method for extracting smooth and flat latent factors by leveraging various kinds of spatial and temporal structures. Our method incorporates a unified structured regularizer into NTF that can represent

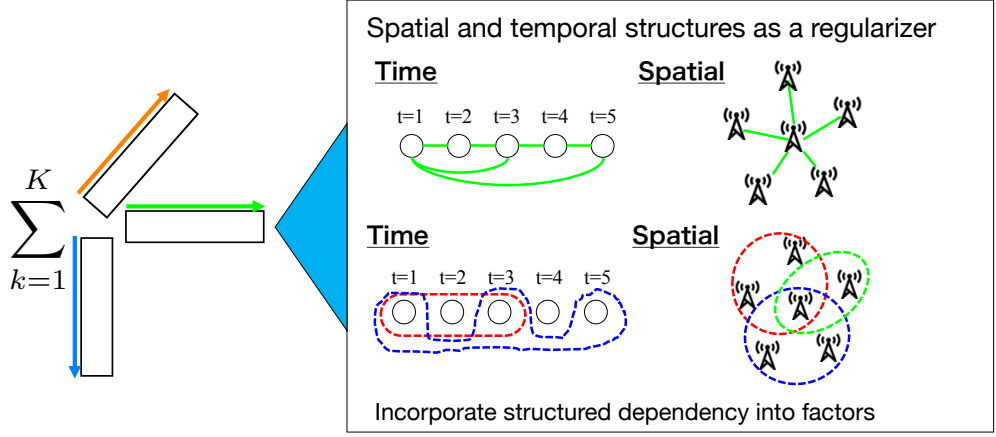


Figure 1.3: Utilization of the spatio-temporal structures as a regularizer.

various kinds of auxiliary information, such as the order of timestamps, daily and weekly periodicity, distances between sensor locations, and areas of the locations. For the estimation of the factors for our model, we present a simple and efficient optimization procedure based on the alternating direction method of multipliers. Through missing value interpolation experiments of traffic flow data and bike-sharing system data, we demonstrate that our proposed method improves the interpolation performances of the existing NTF methods, especially when there exists a large portion of missing values.

From the perspective of spatio-temporal relational databases, the analysis of spatio-temporal data is a common research topic that requires the interpolations of unknown locations and the predictions of feature observations by utilizing information about where and when the data were observed. One of the most difficult problems is making predictions of unknown locations. Tensor factorization methods are popular in this field because of their capabilities of handling multiple types of spatio-temporal data, which deal with missing values, and providing computationally efficient parameter estimation procedures. However, unlike traditional approaches such as spatial autoregressive models, the existing tensor factorization methods do not try to learn spatial autocorrelations. These methods employ previously inferred spatial dependencies, often resulting in poor performances with regard to the problem of making interpolations and predictions of unknown locations.

In Chapter 6, we propose a new tensor factorization method that estimates low-rank latent factors by simultaneously learning the spatial and temporal autocor-

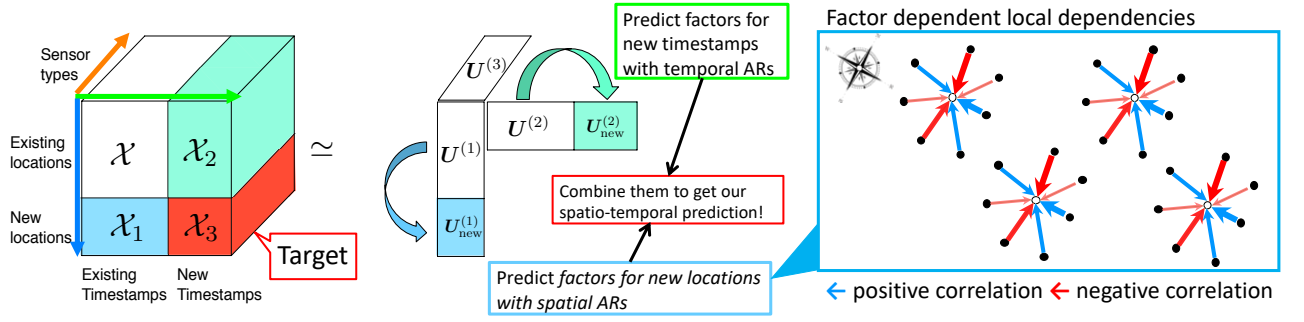


Figure 1.4: Learning spatio-temporal dependencies for predictions.

relations. We introduce new spatial autoregressive regularizers based on existing spatial autoregressive models and provide an efficient estimation procedure. By conducting experiments on publicly available traffic transporting data and air quality observing data, we demonstrate that our proposed method significantly improves the predictive performances in these problems compared to the existing state-of-the-art spatio-temporal analysis methods.

We conclude this dissertation in Chapter 7.

Chapter 2

Preliminaries

Notations: we use a lowercase x as a scalar, a bold lowercase $\mathbf{x}$ as a vector, and a bold uppercase $\mathbf{X}$ as a matrix, respectively. A calligraphic uppercase $\mathcal{X} \in \mathbb{R}^{I_1 \times \dots \times I_N}$ denotes an N -way tensor, where I_n is the number of features of the n -th mode. We use $i = (i_1, \dots, i_N)$ to represent an element in a tensor, and $\Omega = \{i \mid x_i \text{ is observed in } \mathcal{X}\}$ to denote a subset of observed elements. A transpose is denoted as $\mathbf{x}^\top$. We denote a n -th mode multiplication of a tensor $\mathcal{X}$ with a matrix $\mathbf{Y} \in \mathbb{R}^{I_n \times J}$ as $\mathcal{X} \times_n \mathbf{Y}$. A Kronecker product and a Khatri-Rao product [61] of two matrices $\mathbf{X}$ and $\mathbf{Y}$ are represented as $\mathbf{X} \otimes \mathbf{Y}$ and $\mathbf{X} \odot \mathbf{Y}$. The n -th mode matricization (unfolding) of $\mathcal{X}$ is represented as $\mathcal{X}_{(n)} \in \mathbb{R}^{I_n \times J_n}$, where $J_n = \prod_{n' \neq n} I_{n'}$.

2.1 Non-negative Tensor Factorization

We denote a number of latent factors as $K \ll \min(I_1, \dots, I_N)$, the k -th factor vector for the n -th mode as $\mathbf{u}_k^{(n)} \in \mathbb{R}^{I_n}$, and the n -th factor matrix as $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times K}$, respectively. We represent a set of factor vectors as $\Theta = \{\mathbf{u}_k^{(n)} \mid \forall(n, k)\}$ and Khatri-Rao products of factor matrices as $\odot_{n=1}^N \mathbf{U}^{(n)} = \mathbf{U}^{(1)} \odot \dots \odot \mathbf{U}^{(N)}$.

Non-negative Tensor Factorization (NTF) decomposes the observed values of tensor $\mathcal{X}$ into K latent non-negative factors. An estimation of an element $x_i \in \mathcal{X}$ is defined as the sum-product of the latent factor values as follows:

$$\hat{x}_i = \sum_{k=1}^K \prod_{n=1}^N u_{i_n k}^{(n)} \in \hat{\mathcal{X}}. \quad (2.1)$$

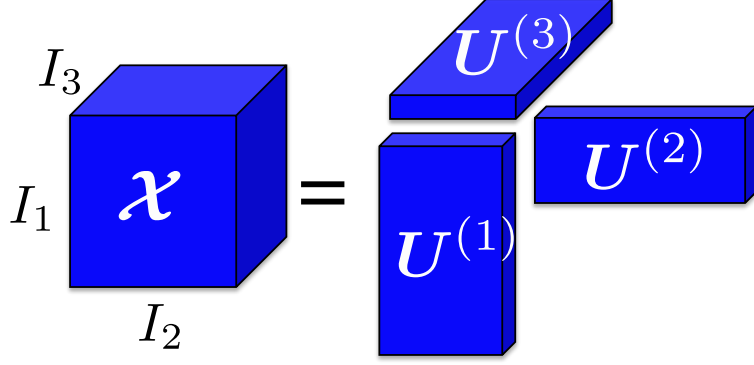


Figure 2.1: NTF factorizes a tensor $\mathcal{X}$ into factor matrices $\mathbf{U}^{(1)}$, $\mathbf{U}^{(2)}$, and $\mathbf{U}^{(3)}$

Let us denote the element-wise divergence used to measure the discrepancy between x_i and $\hat{x}_i$ as $d(x_i \| \hat{x}_i)$. We define the divergence between the observed tensor $\mathcal{X}$ and its estimation $\hat{\mathcal{X}}$ as a sum of the element-wise divergence on the observed elements as

$$D_{\Omega}(\mathcal{X} \| \hat{\mathcal{X}}) = \sum_{i \in \Omega} d(x_i \| \hat{x}_i). \quad (2.2)$$

When all the elements of a tensor $\mathcal{X}$ are observed, we simply denote the divergence as $D(\mathcal{X} \| \hat{\mathcal{X}})$. The divergence is non-convex function with respect to the latent factors Θ [33], thus the goal of NTF is to obtain a local minimizer Θ^* under a non-negative constraint:

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}} D_{\Omega}(\mathcal{X} \| \hat{\mathcal{X}}) \text{ subject to } \mathbf{U}^{(n)} \geq 0, \forall n. \quad (2.3)$$

An example of NTF with three-order tensor is shown in Figure 2.1. When $N = 2$, NTF is equal to the Non-negative Matrix Factorization (NMF).

$$\mathbf{U}^{(1)}, \mathbf{U}^{(2)} = \underset{\mathbf{U}^{(1)}, \mathbf{U}^{(2)}}{\operatorname{argmin}} D_{\Omega}(\mathbf{X} \| \hat{\mathbf{X}}) \text{ subject to } \mathbf{U}^{(1)}, \mathbf{U}^{(2)} \geq 0, \quad (2.4)$$

where $\hat{\mathbf{X}} = \mathbf{U}^{(1)} \mathbf{U}^{(2)\top}$.

In addition to the non-negative constraint, we often utilize additional regularizers for incorporating prior knowledge of data attributes into latent factors. We define a

loss function for NTF with additional regularizers as:

$$f(\Theta) = D_{\Omega}(\mathcal{X} \|\hat{\mathcal{X}}) + \sum_{n=1}^N \sum_{k=1}^K g^{(n)}(\mathbf{u}_k^{(n)}), \quad (2.5)$$

We use $g^{(n)}(\cdot)$ to denote a penalty function for the n -th mode factor vector. The goal of NTF with regularizers is to obtain a local minimizer Θ^* of the loss under a non-negative constraint:

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}} f(\Theta) \text{ subject to } \mathbf{u}_k^{(n)} \geq 0 \quad \forall (n, k). \quad (2.6)$$

2.2 Generalized Kullback-Leibler Divergence

The choice of the divergence $d(x_i \|\hat{x}_i)$ critically affects the final performance of NTF. There are a number of candidate divergences for NTF, including Euclidean divergence, Itakura-Saito divergence, and the generalized Kullback-Leibler divergence. However, there is no definitive way of choosing the divergence. When we are interested in discrete value observations such as stars in product reviews, we choose the generalized Kullback-Leibler (gKL) divergence, which is formulated as follows:

$$d(x_i \|\hat{x}_i) = -x_i \log \hat{x}_i + x_i + x_i \log \hat{x}_i - x_i. \quad (2.7)$$

Since it is known that minimizing the gKL divergence is equivalent to maximizing the log likelihood of the NTF model, if we assume that the observations are Poisson distributed, which is a natural choice for a discrete value observation [33].

2.3 Graph Laplacian Regularizer

The graph regularized Non-negative Matrix Factorization [19] employs a graph Laplacian regularizer [86, 96] to represent the smoothness of the latent factors. The adjacency matrix for the n -th mode features is denoted as $\mathbf{W}^{(n)} \in \mathbb{R}^{I_n \times I_n}$, which represents a graph whose nodes and capacities of edges correspond to the features of the n -th mode and the similarity measures between the two features, respectively. The Laplacian matrix can be denoted as $\mathbf{L}^{(n)} = \mathbf{D}^{(n)} - \mathbf{W}^{(n)}$, where $\mathbf{D}^{(n)}$ is a diagonal

matrix whose elements are the sums of each row of $\mathbf{W}^{(n)}$. Then, a graph Laplacian regularizer can be defined as follows:

$$g^{(n)}(\mathbf{u}_k^{(n)}) = \mathbf{u}_k^{(n)\top} \mathbf{L}^{(n)} \mathbf{u}_k^{(n)}. \quad (2.8)$$

This regularizer penalty function encourages smoothness because its formulation equals putting a weighted quadratic term on the difference between the adjacency elements.

Chapter 3

Non-negative Multiple Matrix Factorization

3.1 Introduction

Non-negative matrix factorization (NMF) [63] is a matrix factorization method that is widely used in various fields including audio processing [95], text mining [115], image analysis [56, 63] and brain signal analysis [30]. The non-negative constraint of NMF is known to reveal intuitive and reasonable factorization results in many applications. NMF with sparse constraint [24, 56] provides good performance for decomposing noisy data [1, 37]. In general, however, matrix factorization methods including NMF perform poorly when the data to be factorized are very sparse and not sufficiently informative [2].

Recently, the variety of sensed data and human-generated data is increasing greatly. An event can be observed as, for example, sounds, movies, geo-locations, text messages, annotations, and action histories. These features can often be high dimensional, and the problem of data sparseness is not negligible [13, 62, 69]. One convincing solution for this problem is to combine a different set of data and analyze them simultaneously. We expect these different data to be correlated with each other because they are different representations of real-world events. Thus the analysis of these data could be improved by utilizing another set of data as complementary information, and vice versa.

Our key idea is that the factorization of the original data could be improved if complementary data are available. Actually, the effectiveness of this approach has been confirmed in item recommendation tasks based on collaborative filtering. In that field, matrix factorization methods have been used in many studies such as [62]. To deal with sparsity, some researchers (e.g. [74, 87, 108]) increase the available information for factorization by adding additional data related to the original data matrix. These methods improved recommendation accuracy, however, these previous studies ignored the non-negative constraint. As described above, the non-negative constraint plays an essential role for obtaining intuitive and understandable data decomposition results.

In this chapter, we propose a novel matrix factorization method called Non-negative Multiple matrix factorization (NM^2F) that utilizes complementary data for non-negative matrix factorization. NM^2F integrates complementary data as additional matrices that share rows or columns with the original data matrix. NM^2F decomposes all matrices simultaneously, and this greatly improves the decomposition performance under sparse data scenarios. In contrast to previous work on collaborative filtering, NM^2F is formulated under the non-negative constraint. Therefore we can expect more intuitive and understandable decomposition results. In this chapter, we formulate NM^2F as a generalization of the original NMF and derive parameter estimation rules to obtain a local optimal solution. Experimental results revealed the good performance of the proposed NM^2F under sparse data situations, both in synthetic and real-world datasets.

3.2 Related Work

Non-negative Matrix Factorization (NMF) [63] is an unsupervised matrix factorization method. NMF is designed to minimize the loss (distance) between a non-negative observed data matrix and its low rank decomposition. In this decomposition, the observed data matrix is represented as the weighted linear sum of bases with a non-negative constraint. The non-negative constraint of decomposed bases and coefficients make the parameter estimation non-convex. Instead, we obtain sparse bases and coefficients that enable us to easily understandable their meaning in many applications.

There are many kinds of loss employed in NMF such as Euclid distance, I-divergence and Itakura-Saito distance [30] which belong to the α divergence family. NMF with I-divergence has been proved to be equal to Probabilistic Latent Semantic Analysis (pLSA) [38, 55]. There are also a number of model estimation methods yielding a local optima of NMF, such as the multiplicative update rule [64], Alternating Least Squares (ALS) [29], and Gibbs sampling [93]. Non-parametric Bayes extensions of NMF have also been proposed including Ga-P NMF [54] and iFiHMM [80]. In many applications, NMF exhibits good performance and extracts understandable patterns thanks to its non-negativity [18, 71]. Furthermore, the calculation could be safely accelerated with distributed processing [70] and online learning methods [22, 109].

Many matrix factorization techniques have been used in item recommendation tasks (e.g. [62]). Among them, the Probabilistic Matrix Factorization (PMF) method developed by [92] has become a seminal technique in the field. PMF has been extended in two ways. One incorporates topic information using LDA [108]. The other augments PMF using link information available from social networks [74, 78, 85]. In one recent paper [87] the authors combine both ingredients in their model. Also, related models have been used in the document analysis research [97]. Our model is closely related to these researches in a technical sense. However, the target domains and goals are very different. These recommendation studies often ignore the non-negative constraint. In this section, we propose a matrix factorization method with additional matrices under the non-negative constraint for intuitive and reasonable decomposition results.

3.3 Non-negative Multiple Matrix Factorization

3.3.1 Problem Formulation

Before describing the model, let us explain data structure, notations, and an overview the problem.

Our proposed method defines three matrices called the original matrix, row wise additional matrix and column wise additional matrix. Let us denote an index of features as $i \in \{1 \cdots I\}$ and an index of data as $j \in \{1, \cdots, J\}$. Let $x_{i,j}$ be the observed value of feature i in the j -th data. Then let $\mathbf{X} = \{x_{i,j}\} \in \mathbb{R}^{I \times J}$ be the

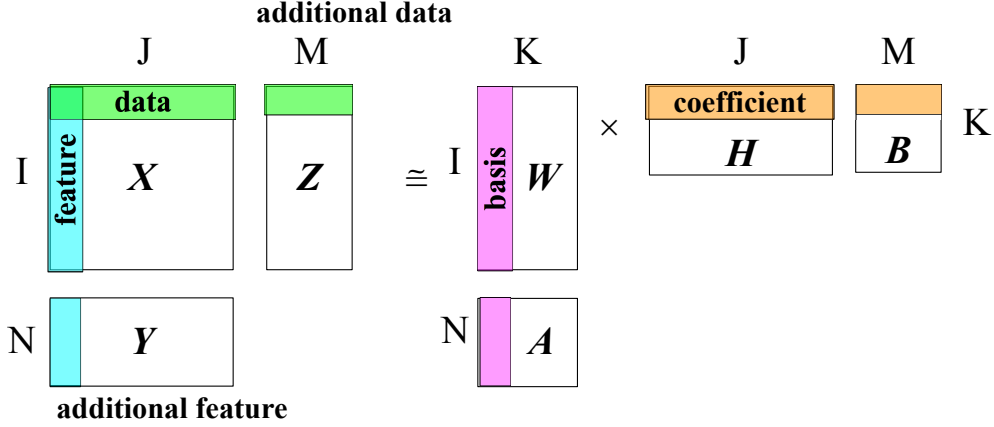


Figure 3.1: The original matrix $\mathbf{X}$ is factorized to the bases $\mathbf{W}$ and the coefficients $\mathbf{H}$ with the row wise additional matrix $\mathbf{Y}$ and the column wise additional matrix $\mathbf{Z}$.

original matrix. Let us denote an index of additional features as $n \in \{1, \dots, N\}$. Then $y_{n,j}$ denotes the additional observed value of feature n in the j -th data. Then we denote $\mathbf{Y} = \{y_{n,j}\} \in \mathbb{R}^{N \times J}$ as the row-wise additional matrix. Let us denote an index of additional data as $m \in \{1, \dots, M\}$. Let $z_{i,m}$ be the observed value of feature i in the m -th additional data. Then let $\mathbf{Z} = \{z_{i,m}\} \in \mathbb{R}^{I \times M}$ be the column-wise additional matrix. Figure 3.1 shows matrices $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$.

The aim of this chapter is to estimate appropriate bases and coefficients. Our method takes additional matrices $\mathbf{Y}$ and $\mathbf{Z}$ into account when seeking bases and coefficients. Furthermore, our method simultaneously estimates patterns of $\mathbf{Y}$ and $\mathbf{Z}$. For example, in Section 3.4, we undertake an experiment to factorize a matrix $\mathbf{X}$ consisting of the number of users listening artists with $\mathbf{Y}$ comprising tags annotated on the artists and $\mathbf{Z}$ consisting of information about a user's network of friends.

3.3.2 Proposed Framework

This subsection describes our proposal, Non-negative Multiple Matrix Factorization (NM²F). NM²F factorizes the observed matrix into a set of bases and coefficients. This method requires the data, the bases and the coefficient matrix to be non-negative. Non-negative constraints yield sparse solutions for the bases and the coefficient matrix, which are preferable for interpretation.

The key idea of our method is to augment the available information by employing additional matrices. NM²F employs two types of additional data in order to factorize

original data $\mathbf{X}$. The row-wise additional matrix $\mathbf{Y}$ is assumed to share coefficients matrix $\mathbf{H}$ with $\mathbf{X}$, and the column-wise additional matrix $\mathbf{Z}$ is assumed to share bases matrix $\mathbf{W}$ with $\mathbf{X}$.

Let K be the number of bases. Let $w_{i,k}$ be the value of the feature i in the k -th basis. We denote $\mathbf{W} = \{w_{i,k}\} \in \mathbb{R}^{I \times K}$ as the basis matrix. Let $h_{k,j}$ be the coefficient of the k -th basis in the j -th data. We denote the coefficient matrix as $\mathbf{H} = \{h_{k,j}\} \in \mathbb{R}^{K \times J}$. Let $a_{n,k}$ be the value of the k -th basis for additional feature n . We denote $\mathbf{A} = \{a_{n,k}\} \in \mathbb{R}^{N \times K}$ as the additional basis matrix. Let $b_{k,m}$ be the coefficient of the k -th basis in the m -th additional data. We denote additional coefficient matrix as $\mathbf{B} = \{b_{k,m}\} \in \mathbb{R}^{K \times M}$.

NM²F minimizes the divergence between the given matrices $\{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$ and their factored matrices $\{\mathbf{W}, \mathbf{H}, \mathbf{A}, \mathbf{B}\}$ with scaling parameters α, β :

$$\begin{aligned} & \underset{\mathbf{W}, \mathbf{H}, \mathbf{A}, \mathbf{B}}{\operatorname{argmin}} D(\mathbf{X}, \mathbf{Y}, \mathbf{Z} \| \mathbf{W}, \mathbf{H}, \mathbf{A}, \mathbf{B}; \alpha, \beta) \\ & s.t. \mathbf{W}, \mathbf{H}, \mathbf{A}, \mathbf{B} \geq 0, \alpha, \beta \geq 0, \end{aligned} \quad (3.1)$$

where D is a divergence. Let us denote $\hat{x}_{i,j}, \hat{y}_{n,j}$ and $\hat{z}_{i,m}$ as the approximated values of $x_{i,j}$, $y_{n,j}$ and $z_{i,m}$ obtained from sums of bases weighted by coefficients:

$$\hat{x}_{i,j} = \sum_{k=1}^K w_{i,k} h_{k,j}, \quad \hat{y}_{n,j} = \sum_{k=1}^K a_{n,k} h_{k,j}, \quad \hat{z}_{i,m} = \sum_{k=1}^K w_{i,k} b_{k,m}. \quad (3.2)$$

Then a divergence between the model and data is defined as

$$\begin{aligned} & D(\mathbf{X}, \mathbf{Y}, \mathbf{Z} \| \mathbf{W}, \mathbf{H}, \mathbf{A}, \mathbf{B}; \alpha, \beta) \\ & = D(\mathbf{X} \| \mathbf{W}, \mathbf{H}) + \alpha D(\mathbf{Y} \| \mathbf{A}, \mathbf{H}) + \beta D(\mathbf{Z} \| \mathbf{W}, \mathbf{B}) \\ & = \sum_{i,j} d(x_{i,j} \| \hat{x}_{i,j}) + \alpha \sum_{n,j} d(y_{n,j} \| \hat{y}_{n,j}) + \beta \sum_{i,m} d(z_{i,m} \| \hat{z}_{i,m}), \end{aligned} \quad (3.3)$$

where α and β satisfying $\alpha \geq 0$, $\beta \geq 0$ are scaling parameters to balance three divergences, and $d(\cdot)$ specifies an element-wise divergence. We employ the gKL-divergence in our model. The gKL-divergence is widely applied to NMF, but other distances such as the Euclid distance and Itakura-Saito distance could be used as alternatives.

3.3.3 NMF Framework

The goal of NMF is to seek a basis matrix $\mathbf{W}$ and a coefficient matrix $\mathbf{H}$ that minimize the distance between the estimated parameters $\mathbf{W}, \mathbf{H}$ and the original matrix $\mathbf{X}$, subject to $\mathbf{W}, \mathbf{H} \geq 0$.

$$\underset{\mathbf{W}, \mathbf{H}}{\operatorname{argmin}} D(\mathbf{X} \| \mathbf{W}, \mathbf{H}) \text{ s.t. } \mathbf{W}, \mathbf{H} \geq 0 \quad (3.4)$$

If we set $\alpha = 0, \beta = 0$ in Eq.(3.3), we have

$$D(\mathbf{X}, \mathbf{Y}, \mathbf{Z} \| \mathbf{W}, \mathbf{H}, \mathbf{A}, \mathbf{B}; \alpha, \beta) = D(\mathbf{X} \| \mathbf{W}, \mathbf{H}). \quad (3.5)$$

Therefore, NMF is regarded as a special case of NM²F.

3.3.4 Parameter Estimation

We derive multiplicative update rules of NM²F, similar to the case of NMF. The objective of the parameter estimation is to minimize the divergence. Let us denote auxiliary variables as $r_{i,j,k}, s_{n,j,k}, t_{i,m,k}$ ($\sum_k r_{i,j,k} = 1, \sum_k s_{n,j,k} = 1, \sum_k t_{i,m,k} = 1$). By Jensen's inequality, an upper bound F of D is derived as below.

$$\begin{aligned} & D(\mathbf{X}, \mathbf{Y}, \mathbf{Z} \| \mathbf{W}, \mathbf{H}, \mathbf{A}, \mathbf{B}; \alpha, \beta) \\ & \leq \sum_{i=1}^I \sum_{j=1}^J \left(\hat{x}_{i,j} - x_{i,j} \sum_{k=1}^K r_{i,j,k} \log \frac{w_{i,k} h_{k,j}}{r_{i,j,k}} \right) \\ & + \alpha \sum_{n=1}^N \sum_{j=1}^J \left(\hat{y}_{n,j} - y_{n,j} \sum_{k=1}^K s_{n,j,k} \log \frac{a_{n,k} h_{k,j}}{s_{n,j,k}} \right) \\ & + \beta \sum_{i=1}^I \sum_{m=1}^M \left(\hat{z}_{i,m} - z_{i,m} \sum_{k=1}^K t_{i,m,k} \log \frac{w_{i,k} b_{k,m}}{t_{i,m,k}} \right) \triangleq F. \end{aligned}$$

We have equality if and only if $r_{i,j,k} = \frac{w_{i,k} h_{k,j}}{\sum_{k=1}^K w_{i,k} h_{k,j}}, s_{n,j,k} = \frac{a_{n,k} h_{k,j}}{\sum_{k=1}^K a_{n,k} h_{k,j}}, t_{i,m,k} = \frac{w_{i,k} b_{k,m}}{\sum_{k=1}^K w_{i,k} b_{k,m}}$. A partial differentiation of F for w_{ik} is

$$\frac{\partial F}{\partial w_{ik}} = \sum_{j=1}^J \left(h_{k,j} - x_{i,j} \frac{r_{i,j,k}}{w_{i,k}} \right) + \beta \sum_{j=1}^J \left(b_{k,j} - z_{i,m} \frac{t_{i,m,k}}{w_{i,k}} \right). \quad (3.6)$$

We obtain the multiplicative update rule of $w_{i,k}$:

$$w_{i,k}^{new} = w_{i,k} \frac{\left(\sum_{j \in \mathcal{J}_i} \frac{x_{i,j}}{\hat{x}_{i,j}} h_{k,j} + \beta \sum_{m \in \mathcal{M}_i} \frac{z_{k,m}}{\hat{z}_{k,m}} b_{i,m} \right)}{\sum_j h_{k,j} + \beta \sum_m b_{k,m}}. \quad (3.7)$$

We can derive the multiplicative update rule for other parameter in a similar way.

In summary, we obtain the local minimum of NM²F by iterating update rules of $\mathbf{W}$, $\mathbf{H}$, $\mathbf{A}$ and $\mathbf{B}$ as shown below:

$$w_{i,k}^{new} = w_{i,k} \frac{\left(\sum_{j \in \mathcal{J}_i} \frac{x_{i,j}}{\hat{x}_{i,j}} h_{k,j} + \beta \sum_{m \in \mathcal{M}_i} \frac{z_{k,m}}{\hat{z}_{k,m}} b_{i,m} \right)}{\sum_j h_{k,j} + \beta \sum_m b_{k,m}}, \quad (3.8)$$

$$h_{k,j}^{new} = h_{k,j} \frac{\left(\sum_{i \in \mathcal{I}_j} \frac{x_{i,j}}{\hat{x}_{i,j}} w_{k,j} + \alpha \sum_{n \in \mathcal{N}_j} \frac{y_{n,j}}{\hat{y}_{n,j}} a_{n,k} \right)}{\sum_i w_{k,j} + \alpha \sum_n a_{n,k}}, \quad (3.9)$$

$$a_{n,k}^{new} = a_{n,k} \frac{\sum_{j \in \mathcal{N}_j} \frac{y_{n,j}}{\hat{y}_{n,j}} h_{k,j}}{\sum_j h_{k,j}}, \quad b_{k,m}^{new} = b_{k,m} \frac{\sum_{i \in \mathcal{M}_i} \frac{z_{i,m}}{\hat{z}_{i,m}} w_{i,k}}{\sum_i w_{i,k}}. \quad (3.10)$$

For speeding up the calculation, let us denote $\mathcal{I}_j$ as sets of indices of non-zero entries on the j -th column. Let $\mathcal{J}_i$, $\mathcal{M}_i$, $\mathcal{N}_j$ similarly as sets of indices of non-zero entries on the i -th row in $\mathbf{X}$, the i -th row in $\mathbf{Z}$ and the j -th row in $\mathbf{Y}$. We denote $\sum_{i \in \mathcal{I}_j}$ as to sum up only non-zero entries of the j -th column, and so on.

3.3.5 NM²F as probabilistic generative model

We can rewrite NM²F with gKL divergence as a probabilistic generative model with a Poisson distribution, as with NMF. The log likelihood of NM²F can be rewritten as

$$\begin{aligned} & \ln p(\mathbf{X}, \alpha \mathbf{Y}, \beta \mathbf{Z} \| \mathbf{W}, \mathbf{H}, \alpha \mathbf{A}, \beta \mathbf{B}) \\ &= \ln p(\mathbf{X}, \| \mathbf{W}, \mathbf{H}) + \ln p(\alpha \mathbf{Y} \| \mathbf{H}, \alpha \mathbf{A}) + \ln p(\beta \mathbf{Z} \| \mathbf{W}, \beta \mathbf{B}) \\ &= \sum_{ij} \ln p(x_{i,j} \| \hat{x}_{i,j}) + \alpha \sum_{n,j} \ln p(y_{n,j} \| \hat{y}_{n,j}) + \beta \sum_{i,m} \ln p(z_{i,j} \| \hat{z}_{i,j}) \\ &= D(\mathbf{X}, \mathbf{Y}, \mathbf{Z} \| \mathbf{W}, \mathbf{H}, \mathbf{A}, \mathbf{B}; \alpha, \beta) + \text{const}, \end{aligned} \quad (3.11)$$

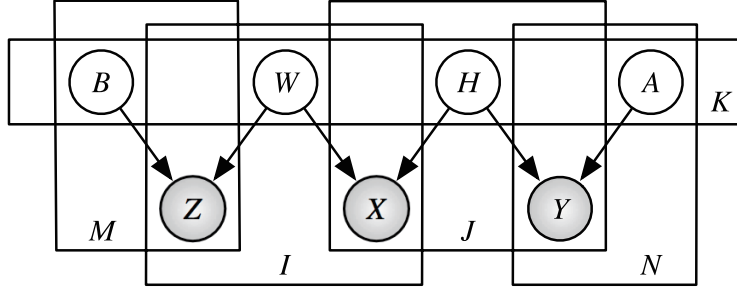


Figure 3.2: Graphical model of Non-negative Multiple Matrix Factorization. $\mathbf{X}$ and $\mathbf{Z}$ are generated from the same bases $\mathbf{W}$ but different coefficients $\mathbf{H}$ and $\mathbf{B}$. $\mathbf{Y}$ is generated from bases $\mathbf{A}$ and coefficients $\mathbf{H}$ shared by $\mathbf{X}$.

where p denotes the Poisson distribution. Minimizing the divergence of NM²F is equivalent to maximizing the log likelihood of probabilistic generative models. The graphical model of NM²F is shown in Figure 3.2.

3.4 Experiments

In this section, we present experimental validations of the proposed method on synthetic dataset and real world dataset.

3.4.1 Evaluation Measures

Though our primal objective is to obtain intuitive decomposition of the data matrix, a quantitative measure would help us understand and compare behaviors of several models. In the sparse data scenario, we are often only interested in non-zero entries of the data matrix, as well non-zero highest values in decomposed bases and coefficients. Therefore, modeling precision of non-zero entries is a reasonable measure for the factorization models.

Specifically, we employ the log likelihood for a test dataset, randomly picked up from non-zero entries of the original data matrix $\mathbf{X}$ as our quantitative measure. Model parameters were estimated by a 5-fold cross validation. A higher test log likelihood indicates better modeling of the data structure.

Table 3.1: Comparison on the average test log likelihood of non-zero entries among different sparseness. Means and one standard deviations of four data sets are presented.

Sparse	NMF	VBNMF	PMF	NM ² F
0 %	-1.24 ± 0.21	-2.72 ± 0.03	-2.47 ± 0.239	-1.03 ± 0.09
9 %	-19.39 ± 2.82	-8.49 ± 0.60	-13.00 ± 2.74	-0.99 ± 0.08
99 %	-42.45 ± 6.30	-14.55 ± 1.40	-16.25 ± 6.05	-1.07 ± 0.25
99.9 %	-43.25 ± 33.45	-15.30 ± 6.30	-12.85 ± 11.20	-0.86 ± 0.55
ine				

3.4.2 Synthetic data experiment

This experiment evaluates the performance of NMF, VBNMF (NMF with sparse constraint) [24], PMF [92] and NM²F. These models are examined for several sparseness datasets. We also conducted a grid search for α and β to assess the effect of the scaling parameters α and β ,

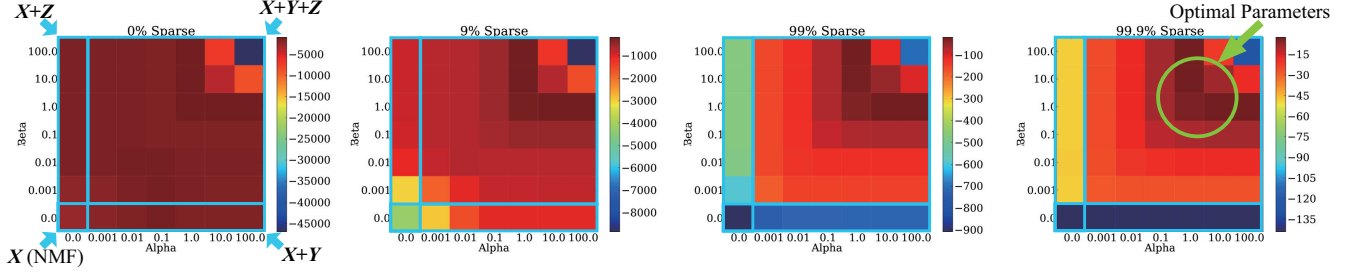
A synthetic dataset was obtained from the probabilistic model with Poisson distribution shown in Section 3.3.5. We set the size of matrices $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{A}$ and $\mathbf{B}$ as $I = J = N = M = 100$. We used values of 0%, 0.9%, 9% and 99% for the sparseness of the synthetic dataset. We compared the means of the test log likelihood for each model. The parameters α and β were set to at the values that present the highest log likelihood for the training dataset. Before the verification, we iterated the parameter estimation 50 times. Note that, if $\alpha = \beta = 0$ then NM²F is reduced to the original NMF.

Results

The test log likelihood values are presented in Table 3.1. As is evident from these results, the proposed model improved the test log likelihood for all datasets.

The results in Table 3.1 confirm that NM²F achieves good performance by making use of additional matrices. Figure 3.3 shows the result of grid searches over α and β . With a dense $\mathbf{X}$ (0% sparse), the scaling parameters α and β scarcely contributed to the improvement of the log likelihood. This result means that if $\mathbf{X}$ is dense then additional matrices are not so useful for improvement. On the other hand, the log likelihood is affected by the choice of scaling parameters as the original data become more sparse. Therefore we conclude that when the observed matrix is highly sparse, additional matrices and appropriate scaling parameters yield better factorization.

Figure 3.3: The test log likelihood versus the scaling parameters α (horizontal) and β (vertical).



3.4.3 Real Data Experiment

Next, we evaluate the proposed model on two real datasets. VBNMF and PMF are omitted from these experiments due to their high computational costs.

Last.fm dataset

The Last.fm dataset is provided by *Hetrec 2011*¹. This dataset consists of the user action history of the Last.fm service: it includes the user's listening history, tag information about artists and friend link information among users.

The dataset contains 1,892 unique users, 17,632 unique artists and 11,946 unique tags. We denote the indices of users, artists, tags and friend users as i , j , n and m , respectively. Let $x_{i,j}$ be a count of the listening of user i to the artist j . Let $\mathbf{X} \in \mathbb{R}_+^{1,892 \times 17,632}$ be the original matrix. Let $y_{n,j}$ denote the count of tags n appearing for the j -th artist. Let $\mathbf{Y} \in \mathbb{R}_+^{11,946 \times 17,632}$ be the column-wise additional matrix. Then $z_{i,m}$ denotes the link state between the i -th user and the m -th user, if users are friends then $z_{i,m} = 1$ else $z_{i,m} = 0$. Let $\mathbf{Z} \in \mathbb{R}_+^{1,892 \times 1,892}$ be the row-wise additional matrix. We evaluated the performance of NMF and NM²F by factorizing $\mathbf{X}$. Note that the dataset is highly sparse: only 0.25% (3,687 elements) of $x_{i,j}$ are non-zero. The number of bases was set to $K = 20$ based on preliminary experiments.

¹<http://ir.ii.uam.es/hetrec2011/>

Social Curation Dataset

In this experiment, we employ a dataset of the curating service Togetter² that was established for summarizing Twitter messages into a *story* [40, 58]. In Figure 3.4, we present an example of a curated story. This story is entitled “20110311 JAPAN MEGA QUAKE M8.8 -ENGLISH NEWS TL”, and includes multiple Twitter messages. Words related to the title such as “quake” and “tsunami” appear several times. Multiple Twitter users are mentioned in the story; some of them appear multiple times (multiple posts are included in this story). Our goal with this experiment was to detect structures of users and stories with additional information.

The dataset contained 1,823,184 unique users, 235,086 stories including 23,859,294 tweets and 165,046 unique words. Let us denote the indices of users, stories, words and additional features as i , j , n and m , respectively. Let $x_{i,j}$ be the count of the appearances (equivalently, a number of messages) of user i in a story j . Let $\mathbf{X} \in \mathbb{R}_+^{1,823,184 \times 235,086}$ be the core matrix of user story observation. Let $y_{n,j}$ denote the count of word n appearing in the j -th story. Let $\mathbf{Y} \in \mathbb{R}_+^{165,046 \times 1,823,184}$ be the row wise additional matrix. We additionally utilize the Twitter based features of each user. One is the number of followers and the other is the number of Lists. We scale these features $\tilde{z}$ by $z = \log(\tilde{z} + 1)$. Let $z_{i,m}$ be a score of the user i in a feature m . Let $\mathbf{Z} \in \mathbb{R}_+^{235,086 \times 2}$ be a column-wise additional matrix. Note that only 0.0018% (7,714,891 elements) of $x_{i,j}$ are non-zero in the dataset. Furthermore, the dataset is very large, but parameter estimations of NM²F can be processed in hours by utilizing dataset sparseness. K was fixed to 200 based on preliminary experiments.

Results

The computed test log likelihood values of non-zero entries are presented in Table 3.2. It is noteworthy that these two datasets are heavily long-tailed; dynamic ranges of observed data values are extremely wide. Combined with the severe sparseness, predicting non-zero entries is a difficult task. As evidenced from these results, however, we confirmed that NM²F achieved better performance than NMF for both datasets. Additional data and appropriate scaling parameters surely contributed to improving the validation scores. Though our primal objective is intuitive decomposition, these

²<http://togetter.com/>

Figure 3.4: Example of a Togetter story shortened by authors.



scores indicate the modeling capability of the proposed NM^2F .

Figures 3.5 and 3.6 present three bases estimated in each experiment. We show the top 10 highest valued artists $\mathbf{W}$ and tags $\mathbf{A}$, and users $\mathbf{W}$ and words $\mathbf{A}$ in the figure. The bars in the figure present the value of each basis.

In figure 3.5, we present bases extracted from the Last.fm dataset. We can confirm that reasonably related artist names and tags appear in the same basis. For example, basis #1 includes popular women singers such as *Britney Spears* and *Lady Gaga*. In basis #2, classic rock musicians are included. Rap and hip-hop singers are presented in basis #3.

Figure 3.6 shows examples of learned bases of Social Curation dataset. Basis #1 consists mass media accounts (*Asahi_Shakai*, *nhk_tokuho*), government official (*pref_iwate*), military officials (*US7thFlt*, *JGSDF_pr*) and a Twitter original account. In basis #2, professional programmer and open source contributors are extracted. Finally, basis #3 includes various accounts (from amateurs to professionals) who post feeds about *the Arab Spring* headline. This result indicates that NM^2F is able to extract meaningful bases from the highly sparse data.

Table 3.2: Comparison of average test log likelihood of non- zero entries among different data sets. Scores are divided by $\times 10^{-3}$

Dataset	NMF	VBNMF	PMF	NM ² F
Last.fm	-6.90 ± 0.03	N/A	N/A	-6.17 ± 0.03
Togetter	-27.27 ± 0.23	N/A	N/A	-12.97 ± 0.48

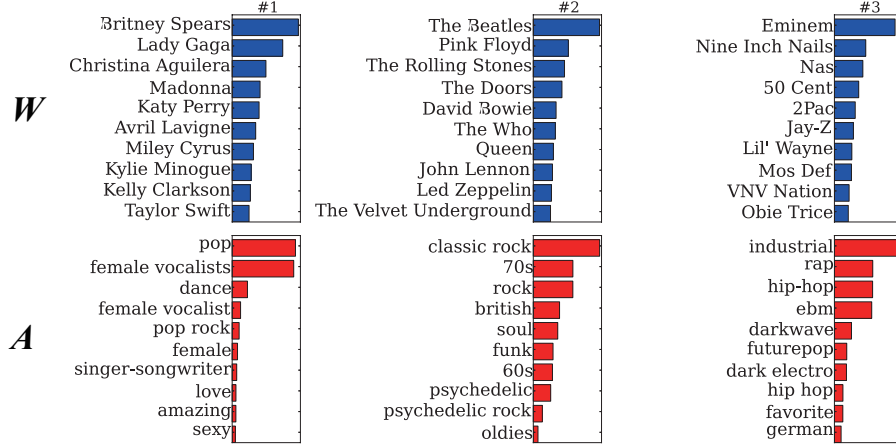


Figure 3.5: Artists and Tags with the highest values in bases.

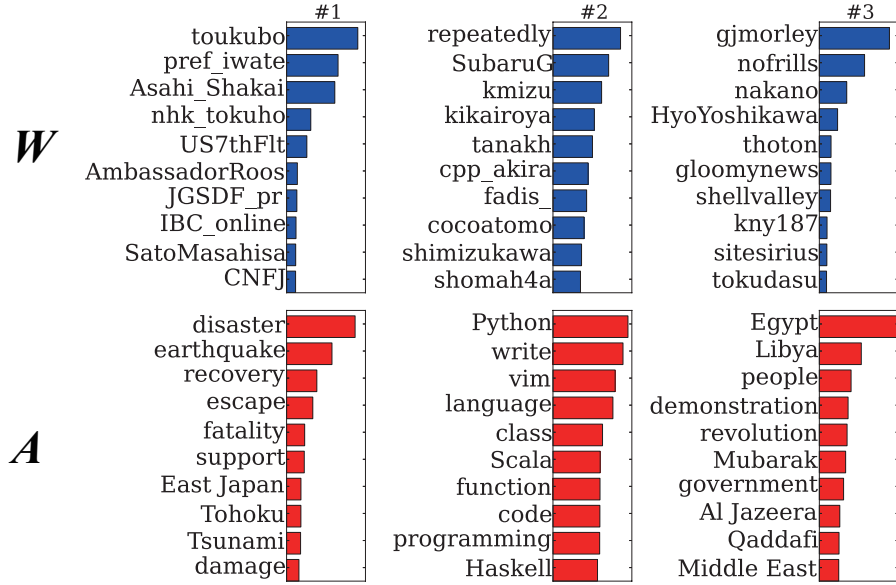


Figure 3.6: Users and Words with the highest values in bases.k

3.5 Summary

In this chapter, we proposed Non-negative Multiple Matrix Factorization (NM²F), which integrates additional matrices in estimating bases and coefficients. Our key idea is to overcome the sparseness of the original data with complementary data. We derived a parameter estimation procedure as a multiplicative update rule. We presented NM²F with the gKL divergence as a probabilistic generative model with Poisson distributions. We evaluated NM²F experimentally using a synthetic dataset and two real datasets, and confirmed that NM²F performs better than existing factorization methods in a quantitative measure. We also confirmed that the estimated bases of NM²F are reasonably intuitive and easy to understand. As a future work, we need to evaluate NM²F with other divergence choices. Also, finding a way to realize the automatic tuning of scaling parameters would be another interesting challenge.

Chapter 4

Non-negative Multiple Tensor Factorization

4.1 Introduction

As the amount and variety of available data has been grown rapidly in recent years, there are many tensors consisting of only non-negative data: e.g. ratings of users $\times$ shops $\times$ times, and social connections between users $\times$ friends $\times$ social networking systems. Non-negative Tensor Factorization (NTF) [94, 111] is a generalization of Non-negative Matrix Factorization (NMF)[63, 64]. We can also regard NTF as CANDECOMP/PARAFAC decomposition [23, 51] with non-negativity constraints. NTF factorizes a target tensor consisting of non-negative values into factor matrices under the non-negative constraints. The non-negativity constraints yield sparse and reasonably interpretable factorization results [33, 43]. NTF has been applied in various fields and performs well in such applications as signal processing [44, 119], image analysis [31, 52], text mining [26], brain computer interface [66] and climate exchange analysis [124]. An advantage of tensor data over conventionally studied matrix data is their ability to manipulate observations with various attributes. In fact, tensor factorization techniques have been extended to collaborative filtering [27, 90, 112], missing value estimation [72] and multi relational networks [59, 83, 84] and have been proven effective as regards these problems.

In spite of these advantages, data sparseness and computational costs have sig-

nificantly hampered the application of tensor factorization methods to real world problems. This problem becomes more serious as the order of the tensor becomes higher and also causes the predictive performance of NTF to degrade.

The problem of sparseness has also been a serious problem in matrix factorization [13, 62]. To deal with this problem, a few solutions have been proposed in the context of multiple data analysis [74, 78, 85, 87, 108], which have been developed based on probabilistic matrix factorization [92]. To compensate for the sparseness of the data matrix, these approaches augment the target matrix by incorporating auxiliary matrices in the analysis. Then the target matrix and the auxiliary matrices are simultaneously factorized. Factorizing multiple matrices simultaneously performs better than factorizing a single target matrix. Employing auxiliary data also seems a promising way to resolve the sparse issue for tensors. However, there have been almost no attempts to validate the idea of factorizing multiple tensors simultaneously. The only exception is described in [120] but this method assumes that we have multiple tensors of exactly the same size. We cannot apply this method to ordinary data sets including tensors of different sizes such as commercial purchasing recordings and social network data.

In this chapter, we propose a novel tensor factorization method called Non-negative Multiple Tensor Factorization (NMTF), which naturally incorporates auxiliary data tensors into standard tensor factorization. The auxiliary tensor shares indices (axes) with the target tensors, thus NMTF is able to merge information from multiple tensors and mitigates the problem of sparseness. Furthermore, factorizing auxiliary tensors simultaneously allows us to examine given data from several different aspects, because we obtain factors from auxiliary tensors as well. The influence of auxiliary tensors during factorization can be tuned by employing scaling parameters. We show that the simultaneous tensor decomposition approach can be reformulated into the decomposition of a larger (partially observed) tensor. NMTF is a generalization of NTF and is reduced to NTF in the special case where all scaling parameters are set at zero (no influence). In this chapter, we employ the generalized Kullback-Leibler (gKL) divergence as the metric of NMTF [119, 122], which is widely used in the context of NMF/NTF. Minimizing gKL divergence is known to be equivalent to maximizing the Poisson distribution likelihood, which fits particularly well if the data are discrete values. We employ a three-way tensor in this chapter

but the extension of NMTF to a N -way tensor is straightforward.

Empirical results showed that NMTF achieved better performance than NTF in a quantitative way. The synthetic data experiments revealed that the performance improvement of NMTF compared with NTF become larger as the target tensor become sparser. We also found that NMTF factorization results are highly interpretable and suggestive for the analysis of real-world complex review data. NMTF reveals the detailed user preferences of beers and spatio-temporal patterns of people’s daily life such as leisure, drinking, and shopping activity from online review data sets.

4.2 Related Work

Non-negative Tensor Factorization (NTF) was first proposed in [94]. NTF is regarded as a generalization of Non-negative Matrix factorization (NMF) [63, 64]. NTF is based on a CONDECAMP/PARAFAC (CP) decomposition [61] and imposes non-negative constraints on tensor and factor matrices. There has been a lot of NTF research concerning sparse factors [32, 53], optimization methods [26, 31, 33] and acceleration techniques [8, 124]. As explained above, data sparsity becomes a serious problem for large or high order tensors. However, there have been no study concerning the data sparsity in NTF.

As we explained in the introduction, combining multiple data matrices has proved effective in sparse matrix analysis [15, 49, 74, 78, 85, 87]. Therefore combining multiple tensors is a reasonable idea for solving the sparsity problem. Unfortunately, to the best of our knowledge, this approach has only been reported in the literature except for [120], which assumes that we have a collection of tensors of the same size, for example, EEG data from different subjects. The authors proposed decomposing this collection of tensors simultaneously to obtain subject-dependent and subject-independent factors. This appears to be a solution to our sparse problem but actually this is not applicable for the following two reasons. First, this method assumes that we have multiple tensors of exactly the same size. Therefore, we cannot apply this method to data sets consisting of different sizes such as commercial purchasing data or social network data. This severely limits the areas for which the method can be applied. Another reason is that the method assumes that data tensors are dense. In fact, the data sets used in the experiments are small images that are densely filled

with pixel values. This is inconsistent with our assumption that the data tensor is extremely sparse.

As we have seen, there are currently no solutions to our sparse data tensor problem. In this chapter, we propose a novel solution for this problem that uses multiple additional tensors.

4.3 Proposed Method

Matrix factorization techniques have been extended to make use of multiple auxiliary matrices. These techniques handle multiple data as matrices and factorize them simultaneously to ease the problem of sparsity. However, none of these studies have tried factoring multiple tensors simultaneously. In this chapter, we propose Non-negative Multiple Tensor Factorization (NMTF), which effectively combines multiple data tensors under a non-negative constraint. NMTF eases the problem of tensor sparsity and further allows us to examine given data from several different aspects.

4.3.1 Non-negative Multiple Tensor Factorization (NMTF)

In NMTF, we have a target tensor and a few auxiliary tensors that share one or two indices (axes) with the target tensor. We want to factorize these tensors simultaneously in order to make use of available auxiliary information.

Let us denote the target tensor as $\mathcal{Y} \in \mathbb{R}_{\geq 0}^{I_y \times J_y \times K_y}$. We denote the n -th auxiliary tensors as $\mathcal{A}^{(n)} \in \mathbb{R}_{\geq 0}^{I_n \times J_n \times K_n}$ ($n = 1, \dots, N$). We set $N = 3$ in this chapter. We define the three factor matrices of $\mathcal{Y}$ as $\Theta_y = \{\mathbf{T}_y, \mathbf{U}_y, \mathbf{V}_y\}$ where $\mathbf{T}_y \in \mathbb{R}^{I_y \times R}$, $\mathbf{U}_y \in \mathbb{R}^{J_y \times R}$, and $\mathbf{V}_y \in \mathbb{R}^{K_y \times R}$. Let us define the factor matrices of the n -th auxiliary tensor as $\Theta^{(n)} = \{\mathbf{T}^{(n)}, \mathbf{U}^{(n)}, \mathbf{V}^{(n)}\}$ where $\mathbf{T}^{(n)} \in \mathbb{R}^{I_n \times R}$, $\mathbf{U}^{(n)} \in \mathbb{R}^{J_n \times R}$, and $\mathbf{V}^{(n)} \in \mathbb{R}^{K_n \times R}$. We define estimated (reconstructed) tensor values based on factor matrix elements as follows:

$$\begin{aligned}\hat{y}_{ijk} &= \sum_{r=1}^R t_{ir}^y u_{jr}^y v_{kr}^y, \\ \hat{a}_{ijk}^{(n)} &= \sum_{r=1}^R t_{ir}^{(n)} u_{jr}^{(n)} v_{kr}^{(n)} \quad (n = 1, \dots, N).\end{aligned}\tag{4.1}$$

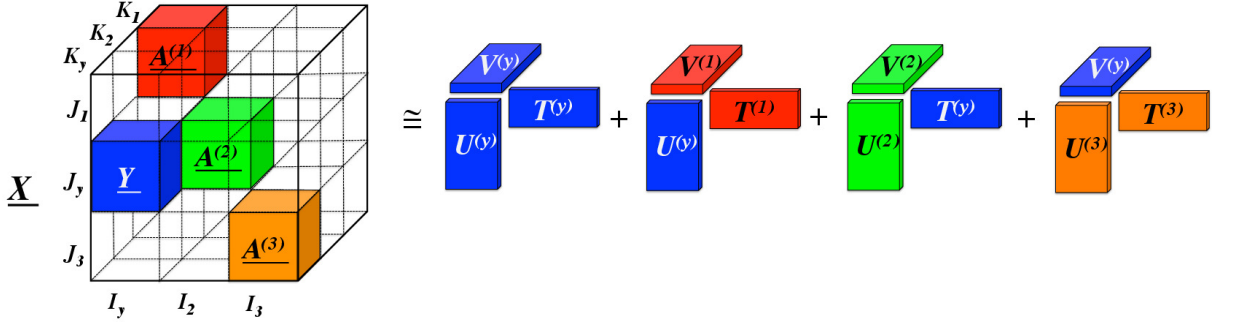


Figure 4.1: $\mathcal{Y}$, $\mathcal{A}^{(1)}$, $\mathcal{A}^{(2)}$ and $\mathcal{A}^{(3)}$ are included in $\mathcal{X}$. Transparent elements of $\mathcal{X}$ are undefined regions. Factor matrices $\mathbf{U}_y, \mathbf{T}_y, \mathbf{V}_y$ are shared among the target tensor and additional tensors.

For simplicity, we assume that $\mathcal{Y}$ shares first indices with $\mathcal{A}^{(1)}$, second indices with $\mathcal{A}^{(2)}$, and third indices with $\mathcal{A}^{(3)}$. Note that it is straightforward to extend our results to cases where auxiliary tensors share two indices with the target tensor. According to this assumption, we set $I_1 = I_y$, $J_2 = J_y$ and $I_3 = K_y$. We set $\mathbf{T}^{(1)} = \mathbf{T}_y$, $\mathbf{U}^{(2)} = \mathbf{U}_y$ and $\mathbf{V}^{(3)} = \mathbf{V}_y$.

The goal of our NMTF is to factorize the target tensor $\mathcal{Y}$ with the auxiliary tensors $\mathcal{A}^{(n)}$ ($n = 1, \dots, N$) simultaneously. To achieve this goal, we regard tensors $\mathcal{Y}$ and $\mathcal{A}^{(n)}$ ($n = 1, \dots, N$) as parts of a larger tensor $\mathcal{X} \in \mathbb{R}_{\geq 0}^{(I_y+I_1+I_2) \times (J_1+J_y+J_3) \times (K_y+K_2+K_1)}$, and perform NTF on this larger tensor (see Fig. 4.1).

We augment the target tensor $\mathcal{Y}$ into $\mathcal{X}$ as follows. First, three axes are augmented by simply concatenating the indices of the target tensor and auxiliary tensors. For example, the size of the first axis is enlarged from I to $I_y + I_2 + I_3$. Given a large empty tensor $\mathcal{X}$, we place data tensors so as to match all indices as in Fig. 4.1. Please recall that $I_y = I_1$, $J_y = J_2$, and $K_y = K_3$. As evident from Fig. 4.1, there are several blocks with no observation, called undefined regions. To encompass these regions, we introduce the following binary variable m_{ijk} :

$$m_{ijk} = \begin{cases} 1 & x_{ijk} \text{ is defined,} \\ 0 & x_{ijk} \text{ is undefined.} \end{cases} \quad (4.2)$$

Let $\mathcal{M} \in \mathbb{R}_{\geq 0}^{I \times J \times K}$ be a tensor consisting of m_{ijk} . We set undefined regions in $\mathcal{X}$ using $\mathcal{M}$. Let us denote sets M_y and $M^{(n)}$ ($n = 1, \dots, N$) consisting of indices of $\mathcal{Y}$ and

$\mathcal{A}^{(n)}$ ($n = 1, \dots, N$).

$$m_{ijk} = \begin{cases} 1 & \text{if } \{i, j, k\} \in M_y \\ \hat{\eta}^{(n)} & \text{if } \{i, j, k\} \in M^{(n)} \text{ } (n = 1, \dots, N) \\ 0 & \text{otherwise.} \end{cases} \quad (4.3)$$

where $\eta^{(n)} \geq 0$ ($n = 1, \dots, N$) are scaling parameters of $\mathcal{A}^{(n)}$ ($n = 1, \dots, N$). The elements m_{ijk} are set at 0 if the indices do not indicate the observed values in $\mathcal{Y}$ or $\mathcal{A}^{(n)}$ ($n = 1, \dots, N$). Let us denote $\hat{D}$ as the divergence of Non-negative Tensor Factorization with an undefined region.

$$\hat{D}(\mathcal{X} \parallel \hat{\mathcal{X}}; \mathcal{M}) = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K m_{ijk} d(x_{ijk} \parallel \hat{x}_{ijk}) \quad (4.4)$$

Then we set $\mathbf{T}$, $\mathbf{U}$, and $\mathbf{V}$ as,

$$\mathbf{T} = \begin{pmatrix} \mathbf{T}_y \\ \mathbf{T}^{(2)} \\ \mathbf{T}^{(3)} \end{pmatrix}, \quad \mathbf{U} = \begin{pmatrix} \mathbf{U}^{(1)} \\ \mathbf{U}_y \\ \mathbf{U}^{(3)} \end{pmatrix}, \quad \mathbf{V} = \begin{pmatrix} \mathbf{V}_y \\ \mathbf{V}^{(2)} \\ \mathbf{V}^{(1)} \end{pmatrix}. \quad (4.5)$$

Therefore $\hat{D}$ is rewritten as,

$$\begin{aligned} \hat{D}(\mathcal{X} \parallel \hat{\mathcal{X}}; \mathcal{M}) &= \sum_{i=1}^{I_y} \sum_{j=1}^{J_y} \sum_{k=1}^{K_y} d(y_{ijk} \parallel \hat{y}_{ijk}) + \sum_{n=1}^N \left[\sum_{i=1}^{I_n} \sum_{j=1}^{J_n} \sum_{k=1}^{K_n} \hat{\eta}^{(n)} d(a_{ijk}^{(n)} \parallel \hat{a}_{ijk}^{(n)}) \right] \\ &= D(\mathcal{Y} \parallel \hat{\mathcal{Y}}; \Theta_y) + \sum_{n=1}^N D(\hat{\eta}^{(n)} \mathcal{A}^{(n)} \parallel \hat{\eta}^{(n)} \hat{\mathcal{A}}^{(n)}; \Theta^{(n)}) \\ &= D(\mathcal{Y}, \hat{\eta}^{(n)} \mathcal{A}^{(n)} \parallel \hat{\mathcal{Y}}, \hat{\eta}^{(n)} \hat{\mathcal{A}}^{(n)}; \hat{\Theta}, n = 1, \dots, N), \end{aligned} \quad (4.6)$$

where we set factorized matrices as $\hat{\Theta} \triangleq \{\Theta_y, \Theta^{(n)}, n = 1, \dots, N\}$ and $\hat{\eta}^{(n)} = \frac{\eta^{(n)}}{I_n J_n K_n}$. NMTF minimizes the divergence between the given non-negative tensors

$\{\mathcal{Y}, \mathcal{A}^{(n)}; n = 1, \dots, N\}$ and those estimated under the non-negative constraints.

$$\begin{aligned} \min_{\hat{\Theta}} D(\mathcal{Y}, \hat{\eta}^{(n)} \mathcal{A}^{(n)} \| \hat{\mathcal{Y}}, \hat{\eta}^{(n)} \hat{\mathcal{A}}^{(n)}; \hat{\Theta}, n = 1, \dots, N) \\ \text{subject to } \mathbf{T}, \mathbf{U}, \mathbf{V} \geq 0. \end{aligned} \quad (4.7)$$

Note that if the scaling parameters $\eta^{(1)} = \eta^{(2)} = \eta^{(3)} = 0$, NMTF is reduced to the original NTF. And if the non-negative constraints are removed, this model could be regarded as a multiple tensor version of CP factorization.

4.3.2 Multiplicative Update Rules

In this section, we derive multiplicative update rules for NMTF, similar to those of NTF. Remember that we set $t_{ijk}^{(1)} = t_{ijk}^y$, $u_{ijk}^{(2)} = u_{ijk}^y$ and $v_{ijk}^{(3)} = v_{ijk}^y$. We derive a multiplicative update rule for t_{ijk} as below:

$$\begin{aligned} t_{ir}^{y(\text{new})} &= t_{ir}^y \frac{\sum_{j=1}^{J_y} \sum_{k=1}^{K_y} \left[\frac{y_{ijk}}{\hat{y}_{ijk}} u_{jr}^y v_{kr}^y \right] + \eta^{(1)} \sum_{j=1}^{J_1} \sum_{k=1}^{K_1} \left[\frac{a_{ijk}^{(1)}}{\hat{a}_{ijk}^{(1)}} u_{jr}^y v_{kr}^{(1)} \right]}{\sum_{j=1}^{J_y} \sum_{k=1}^{K_y} u_{jr}^y v_{kr}^y + \eta^{(1)} \sum_{j=1}^{J_1} \sum_{k=1}^{K_1} u_{jr}^y v_{kr}^{(1)}}, \\ u_{jr}^{y(\text{new})} &= u_{jr}^y \frac{\sum_{i=1}^{I_y} \sum_{k=1}^{K_y} \left[\frac{y_{ijk}}{\hat{y}_{ijk}} t_{ir}^y v_{kr}^y \right] + \eta^{(2)} \sum_{i=1}^{I_2} \sum_{k=1}^{K_2} \left[\frac{a_{ijk}^{(2)}}{\hat{a}_{ijk}^{(2)}} t_{ir}^y v_{kr}^{(2)} \right]}{\sum_{i=1}^{I_y} \sum_{k=1}^{K_y} t_{ir}^y v_{kr}^y + \eta^{(2)} \sum_{i=1}^{I_2} \sum_{k=1}^{K_2} t_{ir}^y v_{kr}^{(2)}}, \\ v_{kr}^{y(\text{new})} &= v_{kr}^y \frac{\sum_{i=1}^{I_y} \sum_{j=1}^{J_y} \left[\frac{y_{ijk}}{\hat{y}_{ijk}} t_{ir}^y u_{jr}^y \right] + \eta^{(3)} \sum_{j=1}^{J_3} \sum_{k=1}^{K_3} \left[\frac{a_{ijk}^{(3)}}{\hat{a}_{ijk}^{(3)}} t_{ir}^{(3)} u_{jr}^y \right]}{\sum_{i=1}^{I_y} \sum_{j=1}^{J_y} t_{ir}^y u_{jr}^y + \eta^{(3)} \sum_{j=1}^{J_3} \sum_{k=1}^{K_3} t_{ir}^{(3)} u_{jr}^y}. \end{aligned} \quad (4.8)$$

Note that this algorithm is stable. Let us focus on the update of t_{ir}^y . The updated t_{ir}^y is always non-negative because all elements in the r.h.s. of the update equations are non-negative.

The only concern regarding computational stability is the case where some u and v take the value 0, then the r.s.h. of the equation becomes $\frac{0}{0}$. But this will never happens if we handle the data and the model initialization appropriately. First, ensure that $\sum_{j=1}^{J_y} \sum_{k=1}^{K_y} y_{i,j,k} > 0$ and $\sum_{j=1}^{J_1} \sum_{k=1}^{K_1} a_{i,j,k} > 0$ for all i . This is easily accomplished by excluding the empty index i' that violates these conditions. Second, initialize the model so that $\forall \hat{y} > 0$ and $\forall \hat{a} > 0$ hold.

If the above conditions are satisfied, we can also enjoy fast computation simply

by skipping summations part in the numerator. If $y_{i,j,k} = 0$ or $a_{i,j,k} = 0$, then we can simply skip the summation for the corresponding elements. This does not affect the computation of the numerator, and greatly reduces the computational loads since we assume severely sparse tensors.

4.4 Parameter Estimation

Multiplicative update rules for other parameters can be derived in the same way. It should be noted that we set $t_{ijk}^{(1)} = t_{ijk}^y$, $u_{ijk}^{(2)} = u_{ijk}^y$ and $v_{ijk}^{(3)} = v_{ijk}^y$. Plugging the definition of generalized Kullback-Leibler divergence into $D(\mathcal{X} \parallel \hat{\mathcal{X}})$, we obtain the following:

$$\begin{aligned}
& D(\mathcal{X} \parallel \hat{\mathcal{X}}; \eta_n, n = 1, \dots, N) \\
&= \sum_{i=1}^{I_y} \sum_{j=1}^{J_y} \sum_{k=1}^{K_y} \left[-y_{ijk} \log \hat{y}_{ijk} + \hat{y}_{ijk} \right] \\
&+ \sum_{n=1}^N \eta^{(n)} \left\{ \sum_{i=1}^{I^{(n)}} \sum_{j=1}^{J^{(n)}} \sum_{k=1}^{K^{(n)}} \left[-a_{ijk}^{(n)} \log \hat{a}_{ijk}^{(n)} + \hat{a}_{ijk}^{(n)} \right] \right\} + \text{const.} \\
&\simeq \sum_{i=1}^{I_y} \sum_{j=1}^{J_y} \sum_{k=1}^{K_y} \left[-y_{ijk} \log \sum_{r=1}^R t_{ir}^y u_{jr}^y v_{kr}^y + \sum_{r=1}^R t_{ir}^y u_{jr}^y v_{kr}^y \right] \\
&+ \sum_{n=1}^N \eta^{(n)} \left\{ \sum_{i=1}^{I^{(n)}} \sum_{j=1}^{J^{(n)}} \sum_{k=1}^{K^{(n)}} \left[-a_{ijk}^{(n)} \log \sum_{r=1}^R t_{ir}^{(n)} u_{jr}^{(n)} v_{kr}^{(n)} + \sum_{r=1}^R t_{ir}^{(n)} u_{jr}^{(n)} v_{kr}^{(n)} \right] \right\}. \quad (4.9)
\end{aligned}$$

Let us denote auxiliary variables as $\phi_{ijk}, \psi_{ijk}^{(n)}$ ($n = 1, \dots, N$), which satisfy $\sum_{r=1}^R \phi_{ijk} = 1$, $\sum_{r=1}^R \psi_{ijk}^{(n)} = 1$ ($n = 1, \dots, N$). To derive an upper bound F , we adopt Jensen's inequality in the logarithm terms of D .

$$\begin{aligned}
& D(\mathcal{X} \parallel \hat{\mathcal{X}}) \leq F \\
&\triangleq \sum_{i=1}^{I_y} \sum_{j=1}^{J_y} \sum_{k=1}^{K_y} \left[-y_{ijk} \sum_{r=1}^R \phi_{ijk} \log \frac{t_{ir}^y u_{jr}^y v_{kr}^y}{\phi_{ijk}} + \sum_{r=1}^R t_{ir}^y u_{jr}^y v_{kr}^y \right] \\
&+ \sum_{n=1}^N \eta^{(n)} \left\{ \sum_{i=1}^{I^{(n)}} \sum_{j=1}^{J^{(n)}} \sum_{k=1}^{K^{(n)}} \left[-a_{ijk}^{(n)} \sum_{r=1}^R \psi_{ijk}^{(n)} \log \frac{t_{ir}^{(n)} u_{jr}^{(n)} v_{kr}^{(n)}}{\psi_{ijk}^{(n)}} + \sum_{r=1}^R t_{ir}^{(n)} u_{jr}^{(n)} v_{kr}^{(n)} \right] \right\}. \quad (4.10)
\end{aligned}$$

We have equality in the above inequality if and only if

$$\begin{aligned}\phi_{ijk r} &= \frac{t_{ir}^y u_{jr}^y v_{kr}^y}{\sum_{r=1}^R t_{ir}^y u_{jr}^y v_{kr}^y}, \\ \psi_{ijk r}^{(n)} &= \frac{t_{ir}^{(n)} u_{jr}^{(n)} v_{kr}^{(n)}}{\sum_{r=1}^R t_{ir}^{(n)} u_{jr}^{(n)} v_{kr}^{(n)}} \quad (n = 1, \dots, N).\end{aligned}\tag{4.11}$$

A partial differentiation of F with respect to t_{ir}^y is

$$\frac{\partial F}{\partial t_{ir}^y} = \sum_{j=1}^{J_y} \left[-y_{ijk} \frac{\phi_{ijk r}}{t_{ir}^y} + u_{jr}^y v_{kr}^y \right] + \eta^{(1)} \sum_{j=1}^{J^{(1)}} \sum_{k=1}^{K^{(1)}} \left[-a_{ijk}^{(1)} \frac{\psi_{ijk r}^{(1)}}{t_{ir}^y} + u_{jr}^y v_{kr}^{(1)} \right]. \tag{4.12}$$

We set $\frac{\partial F}{\partial t_{ir}^y} = 0$ then Eq. (4.12) can be rewritten as:

$$\begin{aligned}t_{ir}^{y \text{ (numerator)}} &= \sum_{j=1}^{J_y} \sum_{k=1}^{K_y} \left[y_{ijk} \sum_{r=1}^R \phi_{ijk r} \right] + \eta^{(1)} \sum_{j=1}^{J^{(1)}} \sum_{k=1}^{K^{(1)}} \left[a_{ijk}^{(1)} \sum_{r=1}^R \psi_{ijk r} \right], \\ t_{ir}^{y \text{ (denominator)}} &= \sum_{j=1}^{J_y} \sum_{k=1}^{K_y} u_{jr}^y v_{kr}^y + \sum_{j=1}^{J^{(1)}} \sum_{k=1}^{K^{(1)}} u_{jr}^y v_{kr}^{(1)}, \\ t_{ir}^y &= \frac{t_{ir}^{y \text{ (numerator)}}}{t_{ir}^{y \text{ (denominator)}}}.\end{aligned}\tag{4.13}$$

Finally, we derive a multiplicative update rule for t_{ijk} as Eq. (4.8),

$$t_{ir}^{y \text{ (new)}} = t_{ir}^y \frac{\sum_{j=1}^{J_y} \sum_{k=1}^{K_y} \left[\frac{y_{ijk}}{\hat{y}_{ijk}} u_{jr}^y v_{kr}^y \right] + \eta^{(1)} \sum_{j=1}^{J_1} \sum_{k=1}^{K_1} \left[\frac{a_{ijk}^{(1)}}{\hat{a}_{ijk}^{(1)}} u_{jr}^y v_{kr}^{(1)} \right]}{\sum_{j=1}^{J_y} \sum_{k=1}^{K_y} u_{jr}^y v_{kr}^y + \eta^{(1)} \sum_{j=1}^{J_1} \sum_{k=1}^{K_1} u_{jr}^y v_{kr}^{(1)}}.$$

Multiplicative update rules for other parameters u_{ir}^y , v_{ir}^y , and t_{ir}^y are derived in the same manner. And we derive multiplicative update rules for $u_{ir}^{(1)}$, $v_{ir}^{(1)}$, $t_{ir}^{(2)}$, and $v_{ir}^{(2)}$ in the same way as with NTF.

4.4.1 NMTF as Probabilistic Generative Model

We mentioned that a generalized Kullback-Leibler divergence is equal to a negative log likelihood of Poisson distribution. NMTF could be interpreted as a probabilistic generative model. Fig. 4.2 shows graphical models of NMTF. Let us denote a Poisson

distribution with a parameter ζ as

$$p(\xi|\zeta) = \text{Poisson}(\xi|\zeta). \quad (4.14)$$

From the definition, a negative log likelihood of the Poisson distribution is approximately equal to a generalized Kullback-Leibler divergence as below:

$$-\ln p(\xi|\zeta) = d_{gKL}(\xi||\zeta) + \text{const.} \quad (4.15)$$

We denote a log likelihood of a tensor $\mathcal{X}$ as:

$$\ln p(\mathcal{X}|\hat{\mathcal{X}}) = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K \ln p(x_{ijk}|\hat{x}_{ijk}). \quad (4.16)$$

Finally, the probabilistic generative model of NMTF can be written as:

$$\begin{aligned} \log p(\mathcal{X}|\hat{\mathcal{X}}) &= \log p(\mathcal{Y}|\mathbf{T}_y, \mathbf{U}_y, \mathbf{V}_y) + \log p(\hat{\eta}^{(1)} \mathcal{A}^{(1)}|\mathbf{T}_y, \hat{\eta}^{(1)} \mathbf{U}^{(1)}, \mathbf{V}^{(1)}) \\ &+ \log p(\hat{\eta}^{(2)} \mathcal{A}^{(2)}|\mathbf{T}^{(2)}, \mathbf{U}_y, \hat{\eta}^{(2)} \mathbf{V}^{(2)}) \\ &+ \log p(\hat{\eta}^{(3)} \mathcal{A}^{(3)}|\hat{\eta}^{(3)} \mathbf{T}^{(3)}, \mathbf{U}^{(3)}, \mathbf{V}_y) \\ &= \log p(\mathcal{Y}|\hat{\mathcal{Y}}) + \sum_{n=1}^N \log p(\hat{\eta}^{(n)} \mathcal{A}^{(n)}|\hat{\eta}^{(n)} \hat{\mathcal{A}}^{(n)}) \\ &\simeq -D(\mathcal{Y}, \hat{\eta}^{(n)} \mathcal{A}^{(n)}|\hat{\mathcal{Y}}, \hat{\eta}^{(n)} \hat{\mathcal{A}}^{(n)}; \hat{\Theta}, n = 1, \dots, N). \end{aligned} \quad (4.17)$$

Eq. (4.17) indicates that minimizing the divergence in NMTF (Eq. (4.7)) is approximately equivalent to maximizing the Poisson log likelihood of the probabilistic model in Fig. 4.2.

4.5 Experiments

We experimentally validate NMTF and the original NTF on synthetic data and real-world data. We compare NMTF with the original NTF. The real-world data are public data sets including two or three 3-way tensors.

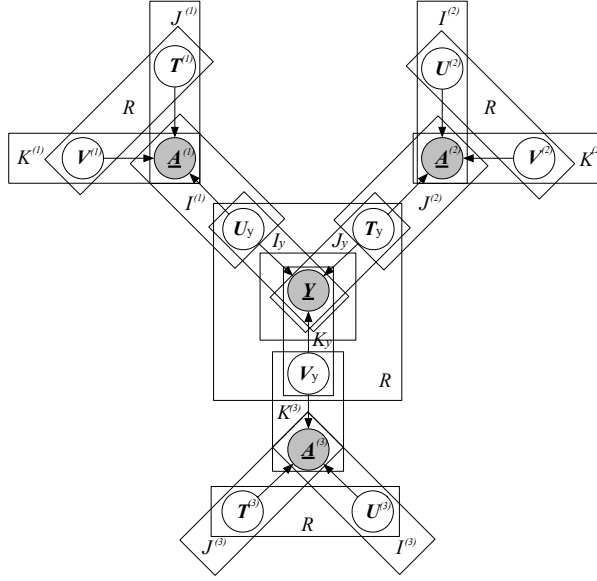


Figure 4.2: Graphical model of NMTF. Tensors $\mathcal{Y}$, $\mathcal{A}^{(1)}$, $\mathcal{A}^{(2)}$ and $\mathcal{A}^{(3)}$ share the factor matrices $\mathbf{U}$, $\mathbf{T}$ and $\mathbf{V}$. Shared factor matrices are employed to generate elements of both the target tensor and auxiliary tensors.

4.5.1 Evaluation Measure

In our experiments, almost all the tensor elements are equal to zero. This means that our target tensor is highly sparse. To evaluate the performance of the factorization results, we focus on the predictive log likelihood for the non-zero elements. We split the elements of the target tensor for 5-fold cross validation. We employ the average log likelihood of non-zero elements in the test sets. After optimizing the parameters of the models for the training sets, we measure the average log likelihood of the test sets evaluated on the learned model. A higher average log likelihood results in the better modeling performance. We define the average log likelihood as:

$$\frac{1}{M} \sum_{m=1}^M \log p(x_m | \theta), \quad (4.18)$$

where M is the number of non-zero elements in the test sets and θ is the estimated parameter of a model.

4.5.2 Synthetic Data Experiment

In this experiment we evaluate the performance of NTF and NMTF in terms of the average test log likelihoods on a synthetic data set. Data sets are stochastically sampled from the probabilistic model described in Fig. 4.2. We set the sizes of the tensors at $I_y = J_y = K_y = 100$ and $I^{(n)} = J^{(n)} = K^{(n)} = 100$ ($n = 1, \dots, N$), respectively. The number of factors is set at $R = 5, 10$, and 20 . We set the sparseness of the auxiliary tensors at 90% (i.e., 90% of all the tensor elements have zero values). We set the scaling parameters at $\eta^{(1)} = \eta^{(2)} = \eta^{(3)} = 1$.

In the first experiment, we evaluated the model performance on the sparseness of the target tensor. We examined cases where the sparseness of the target tensor was 90%, 99%, 99.9% and 99.99%. In the second experiment, we also evaluated the effect of using auxiliary tensors. We examined NMTF with different numbers of auxiliary tensors, N . The scaling parameters were determined by cross-validation. Note that NMTF reduces to NTF when no auxiliary tensors are available ($N = 0$).

Table 4.1a, 4.1b, and 4.1c also show the results of the first experiment evaluating the effect of different data sparseness. The numbers of bases R was set at 5, 10, and 20, respectively. The average log likelihood for test sets worsens as the data sparseness increases. We confirmed that NMTF significantly outperforms the original NTF especially when the target tensor is extremely sparse.

Table 4.2a, 4.2b and 4.2c show the results of the second experiment evaluating the effects of the number of available auxiliary tensors. It is evident that the average log likelihoods improve as the number of auxiliary tensors increase. This improvement indicates that NMTF successfully integrates the auxiliary tensor and target tensor information into factors.

4.5.3 Real Data Experiments

In this section, we evaluate NMTF and NTF with three public data sets provided by Beeradvocate¹ [76, 77], Yelp² [106], and MovieLens³. These data sets include the user’s reviews of, for example, places, beers, and movies with various auxiliary data such as time-stamp, geolocation, and check-in counts.

¹<http://beeradvocate.com>

²<http://www.yelp.com>

³<http://www.movielens.org>

Table 4.1: Synthetic data experiment : The average test log likelihood for different sparseness values of the target tensor $\mathcal{Y}$.

(a) $R = 5$		
Sparseness	NTF	NMTF
90% sparse	-1.72 ± 0.01	-1.97 ± 0.02
99% sparse	-4.16 ± 0.28	-4.44 ± 0.19
99.9% sparse	-56.65 ± 49.61	-6.12 ± 0.59
99.99% sparse	-273.65 ± 237.86	-8.18 ± 3.80
(b) $R = 10$		
Sparseness	NTF	NMTF
90% sparse	-3.23 ± 0.05	-3.48 ± 0.06
99% sparse	-8.74 ± 0.38	-8.22 ± 0.22
99.9% sparse	-92.34 ± 23.32	-12.53 ± 0.75
99.99% sparse	-628.25 ± 514.51	-13.61 ± 5.24
(c) $R = 20$		
Sparseness	NTF	NMTF
90% sparse	-6.20 ± 0.05	-6.33 ± 0.06
99% sparse	-18.82 ± 0.48	-14.65 ± 0.12
99.9% sparse	-250.02 ± 43.43	-25.44 ± 1.25
99.99% sparse	-628.25 ± 514.51	-13.00 ± 4.92

The Beeradvocate data set is a collection of consumer beer reviews. We construct a target tensor $\mathcal{Y}$ containing $I_y = 66,051$ beers and $J_y = 104$ styles, and $K_y = 465,654$ words. An element y_{ijk} represents the term-frequency with which the word k_y appears in reviews of the beer i_y that is in the style j_y . An auxiliary tensor $\mathcal{A}^{(1)}$ consists of $I_y = 66,051$ beers, $J^{(1)} = 33,387$ users, and $K^{(1)} = 5$ rating points. An element $x_{ijk}^{(1)}$ represents the user rating score of the beer i_y by user $j^{(1)}$ in the $k^{(1)}$ -th rating point. $\mathcal{Y}$ is 99.9993% sparse and $\mathcal{A}^{(1)}$ is 99.9287% sparse. We eliminate 30 stop words from the word vocabulary. The numbers of bases are set as $R = 5, 10$, and 20 based on preliminary experiments.

The Yelp data set is a collection of real-world reviews about numerous business places (e.g. restaurants, department stores, etc) in the greater Phoenix area, Arizona. We construct a target tensor $\mathcal{Y}$ containing $I_y = 1,228$ users, $J_y = 1,860$ business

Table 4.2: Synthetic data experiment: The average test log likelihood of the target tensor $\mathcal{Y}$ for different numbers of auxiliary tensors. The sparseness of $\mathcal{Y}$ is set at 99%. The numbers of R is set to 5, 10, and 20.

(a) $R = 5$	
Training set	NMTF
$\mathcal{Y}(= \text{NTF})$	-56.65 ± 49.61
$\mathcal{Y} + \mathcal{A}^{(1)}$	-44.47 ± 2.91
$\mathcal{Y} + \mathcal{A}^{(1)} + \mathcal{A}^{(2)}$	-6.49 ± 0.52
$\mathcal{Y} + \mathcal{A}^{(1)} + \mathcal{A}^{(2)} + \mathcal{A}^{(3)}$	-6.12 ± 0.59
(b) $R = 10$	
Training set	NMTF
$\mathcal{Y}(= \text{NTF})$	-92.34 ± 23.32
$\mathcal{Y} + \mathcal{A}^{(1)}$	-77.10 ± 16.64
$\mathcal{Y} + \mathcal{A}^{(1)} + \mathcal{A}^{(2)}$	-13.17 ± 0.98
$\mathcal{Y} + \mathcal{A}^{(1)} + \mathcal{A}^{(2)} + \mathcal{A}^{(3)}$	-12.53 ± 0.75
(c) $R = 20$	
Training set	NMTF
$\mathcal{Y}(= \text{NTF})$	-250.02 ± 43.43
$\mathcal{Y} + \mathcal{A}^{(1)}$	-168.08 ± 7.16
$\mathcal{Y} + \mathcal{A}^{(1)} + \mathcal{A}^{(2)}$	-26.08 ± 1.79
$\mathcal{Y} + \mathcal{A}^{(1)} + \mathcal{A}^{(2)} + \mathcal{A}^{(3)}$	-25.44 ± 1.25

places, and $K_y = 7$ days of weeks. An element y_{ijk} represents the user i_j who reviewed the business places j_y on the day k_y . We prepared two auxiliary tensors for the Yelp data set. The first auxiliary tensor $\mathcal{A}^{(1)}$ consists of users $I_y = 1,228$ user, $J^{(1)} = 235$ business categories, and $K^{(1)} = 92,052$ words. An element $x_{ijk}^{(1)}$ denotes the term-frequency of the word $k^{(1)}$ about the business category j^1 in reviews by user i_y . The second auxiliary tensor $\mathcal{A}^{(2)}$ contains $I^{(2)} = 63$ geolocation grids and $J_y = 1,860$ business places, and $K^{(2)} = (24 * 7) = 186$ hours of weeks. We assume an approximately $10\text{km} \times 10\text{km}$ geolocation grid. An element $x_{ijk}^{(2)}$ represents the check-ins count of at the business place j_y within the geolocation grid $i^{(2)}$ on the hour-time $k^{(2)}$. We eliminate business places and users whose total numbers of reviews with fewer than 30. $\mathcal{Y}$ is 99.9963% sparse and $\mathcal{A}^{(1)}$ is 99.9729% sparse, and $\mathcal{A}^{(2)}$ is 99.4265% sparse. The numbers of bases are determined at $R = 10, 20$, and 50 in preliminary experiments.

The MovieLens data set is a famous collection of commercial movie reviews. We construct a target tensor $\mathcal{Y}$ containing $I_y = 101,970$ movies, $J_y = 2,113$ users, and $K_y = 590$ weeks. The element y_{ijk} represents the rating user j_y rating the movie i_y posted in the week k_y . As with of Beeradvocate data, we construct one auxiliary tensor from the MovieLens data set. An auxiliary tensor $\mathcal{A}^{(1)}$ consists of $I_y = 101,970$ movies, $J^{(1)} = 186$ locations of movies, and $K^{(1)} = 20$ genres. An element $x_{ijk}^{(1)}$ is set at 1 if a location $j^{(1)}$ is included in meta-data about the shooting location of a movie i_y , and also the genre $k^{(1)}$ is tagged to the movie. $\mathcal{Y}$ is 99.9933% sparse and $\mathcal{A}^{(1)}$ is 99.9320% sparse. The numbers of bases are set at $R = 10, 20$, and 50 by preliminary experiments.

Our primary goal is to extract an interesting and interpretable hidden factor structure from the target tensor $\mathcal{Y}$. All evaluation measures are computed by 5-fold cross validations. Table 4.3 provides the correspondence of target and auxiliary tensors between these experiments.

Table 4.3: Correspondence of target and auxiliary tensors for real-world data sets.

	$\mathcal{Y} (I_y \times J_y \times K_y)$	$\mathcal{A}^{(1)} (I_y \times J^{(1)} \times K^{(1)})$	$\mathcal{A}^{(2)} (I^{(2)} \times J_y \times K^{(2)})$
Beeradvocate	beer $\times$ style $\times$ word	beer $\times$ user $\times$ rating points	-
Yelp	user $\times$ shop $\times$ weeks	user $\times$ category $\times$ word	geolocation $\times$ shop $\times$ hour
MovieLens	movie $\times$ user $\times$ week	movie $\times$ location $\times$ genre	-

Table 4.4: Real-world data sets: Mean and standard deviation of average test log likelihoods on the target tensor $\mathcal{Y}$. Large values mean a good estimation of the hidden structures of $\mathcal{Y}$.

(a) The Beeradvocate data set.		
	NTF	NMTF
$R = 5$	-24.02 ± 1.05	-11.91 ± 0.03
$R = 10$	-28.77 ± 2.11	-9.13 ± 0.04
$R = 20$	-30.31 ± 2.16	-6.72 ± 0.07
(b) The Yelp data set.		
	NTF	NMTF
$R = 10$	-22.30 ± 0.39	-17.72 ± 0.32
$R = 20$	-30.69 ± 0.69	-18.38 ± 0.30
$R = 50$	-62.97 ± 1.94	-21.00 ± 0.37
(c) The MovieLens data set.		
	NTF	NMTF
$R = 10$	-23.96 ± 0.09	-23.74 ± 0.12
$R = 20$	-27.16 ± 0.30	-24.31 ± 0.12
$R = 50$	-39.54 ± 0.44	-26.68 ± 0.37

4.5.4 Results

Quantitative analysis

Before closely examining the learned bases from the real-world data, we conduct numerical evaluations. The average test log likelihoods for the Beeradvocate data set, the Yelp data set, and the MovieLens data set are shown in Table 4.4b and 4.4a, and 4.4c, respectively. The scaling parameters are set at $\eta_1 = 0.001$ on the Beeradvocate data set, $\eta_1 = 0.1, \eta_2 = 0.001$ on the YELP data set, and $\eta_1 = 0.01$ on the MovieLens data set by 5-fold cross-validation. As with the synthetic data experiments, NMTF performed better than the original NTF with real-world sparse tensor data.

Qualitative analysis on Beeradvocate data set

Our main reason for using NMTF is to extract interesting patterns of reviews from several aspects, by combining auxiliary tensors. Now let us examine and evaluate

the learned basis in the Beeradvocate data set qualitatively.

Fig. 4.3 presents the extracted factors $\mathbf{T}_y$, $\mathbf{U}_y$, $\mathbf{V}_y$, and $\mathbf{V}^{(1)}$ for a few bases from the Beeradvocate data set. Colors used in the figures correspond to the colors of the factors in Fig. 4.1. In the top left we present a list of the beers with 5 highest values in $\mathbf{T}_y$ as blue bars. In the top right we present a list of words with the 5 highest values in $\mathbf{U}_y$ as blue bars. In the bottom left we present a list of the beer styles with the 5 highest values in $\mathbf{V}_y$ as blue bars. The red line in the bottom right shows the estimated volumes of rating points $\mathbf{T}^{(1)}$.

Fig. 4.3a shows decomposed factors for a specific basis ($r = 1$). The top 5 highest valued beers include Ales and Hefevizen. We found that “lager”, “amber”, and “smooth” were the highest valued words. The style factor consists of three major styles, and of these “Vienna Lager” is chosen as the most highest valued style. Taste and appearance attain higher ratings than other rating points.

Fig. 4.3b shows decomposed factors for another basis ($r = 3$). The highest valued beers of this factor include Stouts and dark colored beers. The top 5 words consist of “coffee”, “roasted”, “stout”, “black”, and “smooth”. These selected words coincide with the two main beer styles fairly well. The estimated ratings showed a similar pattern to the first basis.

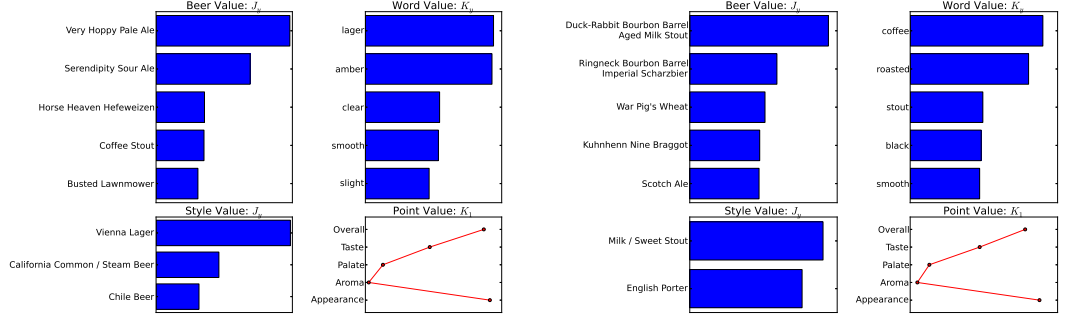
Fig. 4.3c shows decomposed factors for yet another basis ($r = 2$). This factor contains unique beers with names including the words “cherry”, “pumpkin”, and “oatmeal”. The top 5 word also includes interesting terms such as “fruit”, “wheat”, and “pretty”. Beer styles “Fruit / Vegetable Beer” and “Herbed / Spiced Beer” also indicate the types of beers. The rating pattern is different from that of the previous factors.

From the above analysis, we can safely say that NMTE successfully merged the target tensor and an auxiliary tensor, and factorized them into interpretable patterns of beers. As a result, we obtained interpretable factors extracted from the Beeradvocated data set and the improvement in the test average log likelihoods.

Qualitative analysis of YELP data set

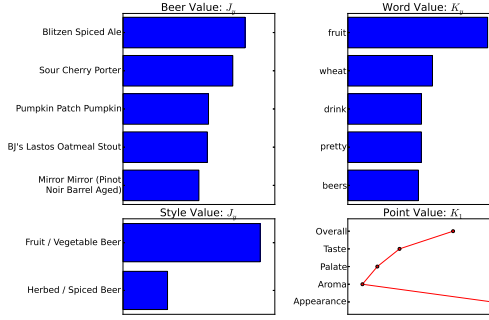
We constructed three tensors from the YELP data set including an auxiliary tensor of geolocations and check-in counts.

Fig. 4.4 show the extracted factors $\mathbf{U}_y$, $\mathbf{V}_y$, $\mathbf{T}^{(1)}$, $\mathbf{V}^{(1)}$, $\mathbf{T}^{(2)}$, and $\mathbf{V}^{(2)}$ of bases



(a) Factors for the “lager-taste” beer basis ($r = 1$).

(b) Factors for the “black and roasted” beer basis ($r = 2$).



(c) Factors for “spiced and fruity” beer basis ($r = 3$).

Figure 4.3: Extracted factors for three different bases learned from the Beeradvocate data set.

learned from the YELP data set. The colors in the figures corresponds to the colors of the factors in Fig. 4.1. The blue bars in the top left show the 10 highest values in $\mathbf{U}_y$ with corresponding the names of business places. The green and pink lines in the top right show the estimated check-in counts per hour on each day of the week, namely $\mathbf{V}^{(2)}$. The green lines indicate weekday responses while the pink lines indicate those of weekends. The red bars placed in the middle left present the 10 highest values in $\mathbf{T}^{(1)}$ with corresponding business categories. The red bars in the middle right show the 10 highest values in $\mathbf{V}^{(1)}$ with corresponding words. The blue circles in the bottom left show the locations in Phoenix city, Arizona in I_y , and the circle size indicates the estimated values in $\mathbf{U}_y$.

Fig. 4.4a shows decomposed factors for a specific learned base ($r = 1$). Zoo, museum, and a few parks are selected as representative business places. The esti-

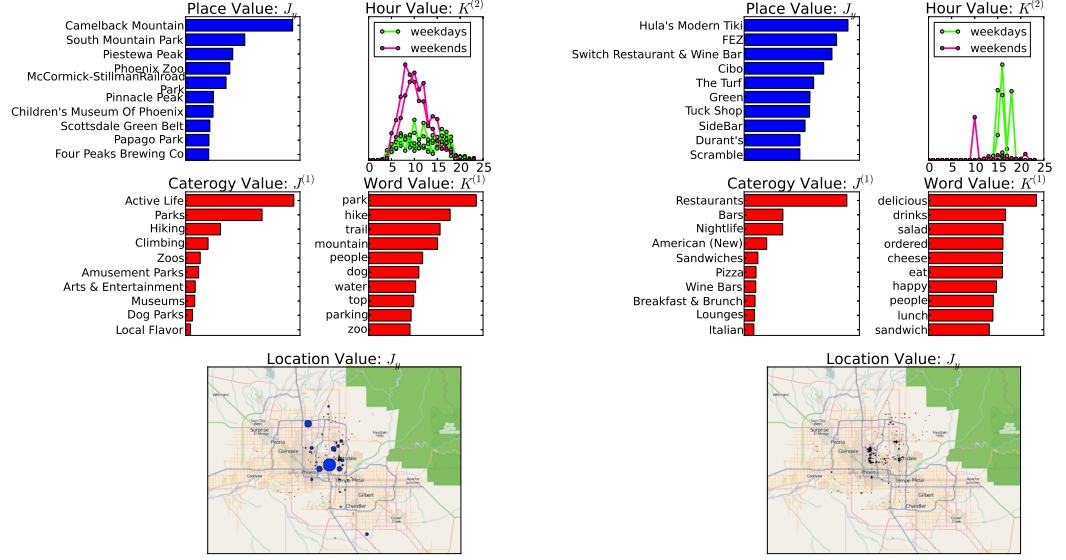
mated check-in counts hit peaks in the mornings on weekends, which is a reasonable pattern for these business places. Top categories and words are also easily interpretable in this base. Moreover, it is easy to see that the selected business places are located across the map. From this evidence, we can imagine that dining and drinking behavior on weekdays can be extracted in this basis.

Fig. 4.4b shows another extracted basis ($r = 2$). Restaurants, bars and pubs are the top-valued business places. The check-in pattern is very different from the previous basis: there are many check-in counts on weekday evenings and weekend mornings. The selected categories and words match these business places. The business places are densely located in the central area of Phoenix. We see that the going out and drinking patterns of users on weekdays are mainly extracted in this factor.

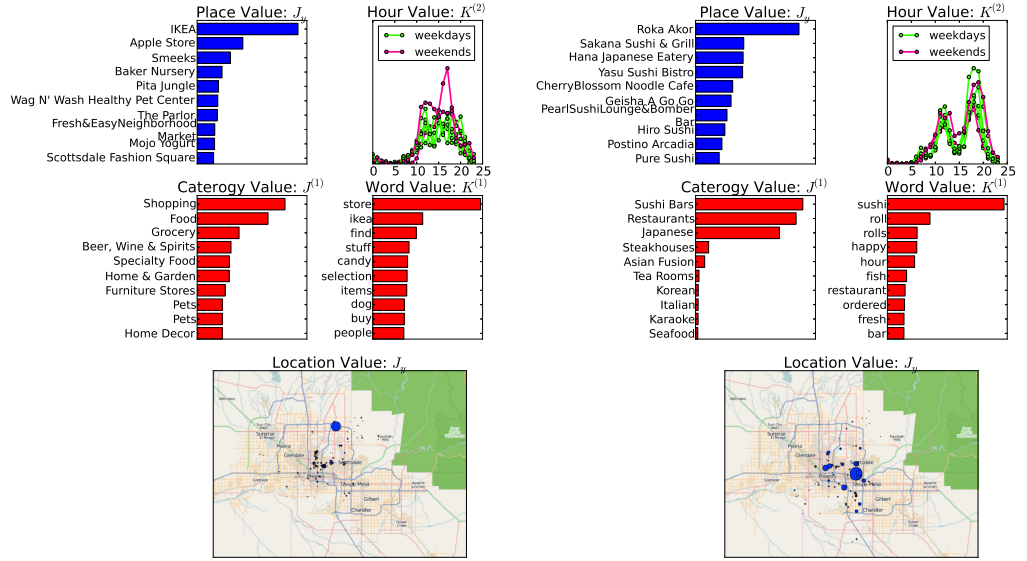
Fig. 4.4c shows yet another basis ($r = 3$). IKEA, Apple store, a toy store, a gardening store, and a pet shop achieve higher values for the place factor. Users checked into these business places most frequently at 5 p.m. on Sundays. The 10 highest categories and words include “shopping”, “food”, “store”, and “IKEA”. A place with a very large location weight is situated in the outer area of the city, but many other business places are located in the central downtown area. Our understanding of this basis is that it shows daily purchasing behavior downtown and occasional shopping on the periphery.

Finally, we present an interesting and completely different pattern in Fig. 4.4d ($r = 4$). The listed business places serve typical Asian foods such as sushi and noodles. The check-in counts do not vary greatly between weekdays and weekends. The categories and words are also related to Asian foods such as “Sushi Bar”, “Japanese”, “roll”, and “fish”. The patterns of users enjoying these typical foods are extracted in this basis.

The addition of these auxiliary features should yield more interesting and interpretable decomposed factors than the previous experiment. NMTF revealed expressive patterns with geolocation and check-in counts in this experiment.



(a) Amusement and park activity factor on weekend morning (r=1). (b) Bar and restaurant factor on weekday night and weekend morning (r=2).



(c) Shopping at various stores factor on weekend evening (r=3). (d) Asian foods consumption factor on both weekdays and weekends (r=4).

Figure 4.4: Factors corresponds to four bases extracted from the YELP data set.

4.6 Summary

In this chapter, we proposed a novel tensor factorization technique called Non-negative Multiple Tensor Factorization (NMTF). We formulated NMTF as a generalization of NTF and NMTF includes NTF as a special choice of scaling parameters. We adopted the generalized Kullback-Leibler divergence as the distance metric between tensors and we derived a method for parameter estimation based on the multiplicative update rule. We evaluated NMTF and the original NTF on both synthetic and real-world data sets. The performance of NMTF was quantitatively better than that of NTF. We also confirmed that NMTF successfully extracted informative and understandable factors from multiple tensors.

We considered a case where each auxiliary tensor shares only one axis with the target tensor. One natural extension of this work is to allow the tensor to share multiple axes. We also assumed that only three-way tensors are available in this chapter. Decomposing different orders of tensors should allow us to model more complex and rich data set, for example, factorizing matrices and three-way tensors or factorize higher 4 or 5-way tensors simultaneously. Other choices of divergence such as Euclidean distance are also important topics to investigate. Finally, we mention about the choice of the number of factor bases R . Determining appropriate R is an unsolved issue in NTF including the proposed NMTF. Though there are no definitive ways, this is also an important task to investigate.

Chapter 5

Structurally Regularized Non-negative Tensor Factorization for Spatio-temporal Pattern Discoveries

5.1 Introduction

Spatio-temporal data covering a wide area of a city have become available due to the commoditization of sensor-monitoring systems and mobile-phone networks. These monitoring systems observe various types of data, such as vehicle transportation counts on a road network, bike-renting counts of a bike-sharing system, and the purchasing records of shops around a city, where missing values often appear due to the failure of sensor nodes, data transmission errors, and trouble with data recording systems. We can find rich and bounteous information in such spatio-temporal data. However, it becomes difficult to grasp what spatio-temporal activities appeared in the data at a glance. Therefore, understanding of such activities via pattern extractions is a typical problem of spatio-temporal data analysis, in which the interpretability of the extracted patterns is regarded as one of the most important property for analysis methods.

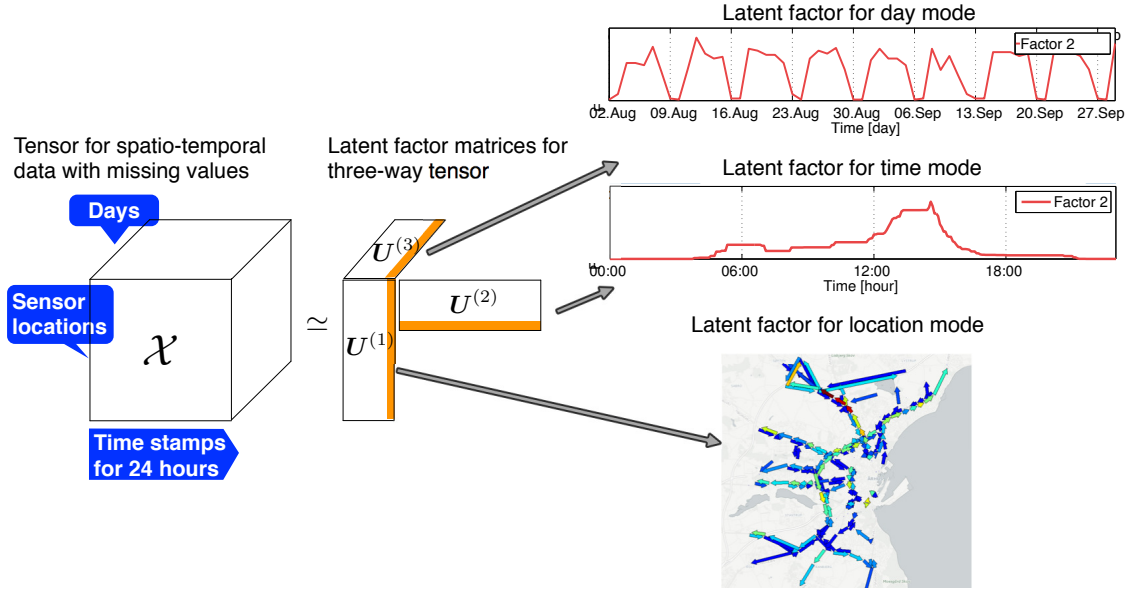


Figure 5.1: Example for a non-negative tensor factorization method on analysis of a traffic-flow data set, where latent patterns for location, time, and day modes are extracted in the latent factor.

Tensor factorization methods have been widely applied to discover spatial and temporal patterns from various kinds of spatio-temporal data [61]. These methods represent spatio-temporal data as a higher-order dimensional array, called a tensor that is a generalization of a matrix. For example, we can represent spatio-temporal data as a three-way tensor whose first, second, and third modes correspond to sensor locations, timestamps for 24 hours, and the observed days. We illustrated an example of a tensor for spatio-temporal data analysis in Fig. 6.1. With this formulation, we can naturally incorporate an assumption that daily or weekly periodicity can be found in data and similar spatial patterns appear on different days. We can extract a few numbers of spatial, temporal, and daily patterns into latent factors by decomposing the tensor. However, since most existing tensor factorization methods do not consider the non-negativity of data where observations only contain non-negative values, they often result in messy and hard to interpret factors.

Unlike those tensor factorization methods, Non-negative Tensor Factorization (NTF) [33], which leverages non-negativity, is effective for extracting interpretable patterns from the non-negative data [48, 65]. This method has successively yielded interpretable factors from various kinds of spatio-temporal data, such as location-based social network services [50, 100], mobile phone GPS logs [42], log messages of

network equipment [60], and traffic records of road networks [118]. However, NTF was not applicable to the existence of missing values. To deal with missing values, NTF was recently extended to learn the latent factors from a subset of elements in a tensor, called the non-negative tensor completion [57, 116]. With this NTF, we can interpolate missing values in data by learned latent factors. However, NTF methods for the missing value completion problem suffer from overfitting when just a few observations are available. Because they ignore spatial and temporal contextual information such as the order of time stamps, weekly periodicity, the distances between sensor locations and treats each feature of the tensor independently.

To incorporate such contextual information, most matrix/tensor factorization methods have employed a graph Laplacian based regularizer for encouraging the latent factors to be smooth with spatio-temporal dependencies [88]. The graph regularized non-negative matrix factorization [19] is a variant of such schemes and has been widely utilized in many applications, however, it does not consider scenarios where missing values exist and analyzing higher-order dimensional arrays.

Another choice for representing such auxiliary information is structured regularizers [9] that have become popular in the fields of machine learning, signal processing, and data mining [25, 110]. For example, the fused lasso [103], which is also known as the total variation, approximates parameters by piecewise-constant values with the order of parameters. Since its estimated parameters have the same estimated value, this is beneficial for finding segments of parameters. In a pioneering work [113], the penalized matrix decomposition was proposed to utilize the fused lasso as a regularizer on latent factors and was applied to a gene data analysis problem. They presented latent factors easy to find gene segments rather than existing matrix factorization methods incorporated the lasso regularizer. However, this method and its subsequent works have only considered the fused lasso without incorporating more general structured regularizer such as spatial dependencies of sensors, and also ignored the non-negative properties and the existence of missing values.

In this chapter, we attempt to solve a problem of extracting latent factors from spatio-temporal data where a lot of missing values exists. To tackle this problem, we propose a novel NTF that learns factors by employing spatial and temporal auxiliary information as regularizers. We utilize this information to represent phenomena often appear in spatio-temporal data, such as counts of vehicles passed roads smoothly

grow or decrease or take the same value along with space and time. To exploit such information, we introduce a regularizer that consists of both a graph-based Laplacian regularizer and structured regularizers that incorporate not only the order of features but also more general graph and group based structures [12, 101]. With our regularizer, we can utilize various kinds of auxiliary information into NTF including a daily and weekly periodicity, distances between sensor locations, and areas of locations. Our proposed method is highly robust to the presence of a large portion of missing values because it encourages latent factors to be smooth and flat with spatial and temporal structures, where we regard segments of parameters that take the same value as flat. To estimate the latent factors for our proposed method, we present an efficient optimization procedure of the alternating direction method of multipliers [16] that utilizes simple proximity operators of the conjugate gradient method [88] and a parametric network flow algorithm [46].

We conducted missing value interpolation experiments with real-world traffic flow data and compared the performance of our proposed method with existing NTF methods. We demonstrate that our proposed method improved the interpolation performances from existing NTF methods. We also show that our extracted factors were interpretable to detect change points. Because our factors have segments, we can easily find a boundary of segments as a change point.

5.2 Proposed model

We introduce a unified structured regularizer to employ both smooth and piecewise-constant properties with auxiliary structures:

$$g^{(n)}(\mathbf{u}_k^{(n)}) = \sum_{m=1}^3 \lambda_m g_m^{(n)}(\mathbf{u}_k^{(n)}) + g_{\geq 0}^{(n)}(\mathbf{u}_k^{(n)}), \quad (5.1)$$

where λ_1, λ_2 and λ_3 are the hyperparameters for each regularizer. We employ a Generalized Fused Lasso (GFL) [17, 114] and a Higher-Order Fused Lasso (HOFL) [101] as $g_1^{(n)}$ and $g_2^{(n)}$, respectively. $g_3^{(n)}$ corresponds to the Laplacian regularizer for ex-

tracting smooth patterns. We use an indicator function for the non-negative region:

$$g_{\geq 0}^{(n)}(\mathbf{u}_k^{(n)}) = \begin{cases} 0 & (\text{if } u_{i,k} \geq 0, \forall i) \\ +\infty & (\text{otherwise}) \end{cases}. \quad (5.2)$$

The GFL penalty is defined:

$$g_1(\mathbf{u}_k^{(n)}) = \sum_{j=1}^{I_n} \sum_{j'=1}^{I_n} w_{j,j'}^{(n)} |u_{j,k}^{(n)} - u_{j',k}^{(n)}|. \quad (5.3)$$

The GFL prefers parameters with the same value if they are adjacent on the given graph, such as distances between sensor locations and temporal lags between time stamps. The HOFL encourages parameters in a given group to take identical values [101]. With this regularizer, we can utilize auxiliary information, such as sensors placed in a specific area that may output similar values and a group of time stamps when a specific train leaves from a station. We denote the r -th group of features in the n -th mode as $g_r^{(n)} \subseteq D_n$ and a set of groups by $\mathcal{G}^{(n)} = \{g_1^{(n)}, \dots, g_{R_n}^{(n)}\}$, where D_n and R_n are a set of elements in the n -th mode and the number of groups, respectively. The weights of each element for the r -th group on the n -th mode are denoted by $c_{r,m}^{(n)} = \bar{c}_{r,m}^{(n)}$ if $m \in g_r^{(n)}$, and 0 otherwise, where $\bar{c}_{r,m}^{(n)} > 0$. Then a simplified HOFL penalty $g_2(\mathbf{u}_k^{(n)})$ is given:

$$\sum_{r=1}^R \sum_{m=1}^{I_n} c_{r,j_m}^{(n)} |u_{j_m,k}^{(n)} - \bar{u}_{r,j_m,k}^{(n)}| + \theta_r^{(n)} (u_{s_r,k}^{(n)} - u_{t_r,k}^{(n)}), \quad (5.4)$$

where $\theta_r^{(n)} > 0$ is a hyperparameter that controls the consistency of the parameters in a group. $\bar{u}_{r,k}^{(n)}$ is defined as $\bar{u}_{r,m,k}^{(n)} = u_{s_k,k}^{(n)}$ (if $m \geq s_k$), $u_{t_k,k}^{(n)}$ (if $m \leq t_k$) and $u_{j_m,k}^{(n)}$ (otherwise) for distinct indices $j_1, j_2, \dots, j_{I_n} \in D_n$ that correspond to a permutation that arranges the entries of $\mathbf{u}_k^{(n)}$ in a non-increasing order. Thresholding indices s_r and t_r are given as $s_k = \min \{m' \mid \sum_{m=1}^{m'} c_{r,j_m}^{(n)} \geq \theta_r^{(n)}\}$ and $t_k = \min \{m' \mid \sum_{m=m'}^{I_n} c_{r,j_m}^{(n)} < \theta_r^{(n)}\}$.

For convenience, we denote $\bar{g}^{(n)}(\mathbf{u}_k^{(n)}) = \sum_{m=1}^2 \lambda_m g_m^{(n)}(\mathbf{u}_k^{(n)}) + g_{\geq 0}^{(n)}(\mathbf{u}_k^{(n)})$. By adopting our structured regularizers to the loss of NTF, we define the following

minimization problem for our purpose:

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}} D_{\Omega}(\mathcal{X} \|\hat{\mathcal{X}}) + \sum_{n=1}^N \sum_{k=1}^K \bar{g}^{(n)}(\mathbf{u}_k^{(n)}) + \lambda_3 g_3^{(n)}(\mathbf{u}_k^{(n)}). \quad (5.5)$$

Note that when $\lambda_1 = \lambda_2 = \lambda_3 = 0$, our method is reduced to an original NTF. When $\lambda_2 = \lambda_3 = 0$, our method can be regarded as a tensor extension of the graph-regularized non-negative matrix factorization. Our method includes those methods as special cases.

5.3 Parameter estimation

We present an efficient parameter estimation procedure for obtaining a local minimizer of our proposed method. We employ a scaled formulation of the Alternating Direction Method of Multipliers (ADMM) for NTF [57]. The minimization problem for our proposed method can be rewritten:

$$\begin{aligned} \min_{\Theta, \mathcal{Z}} D_{\Omega}(\mathcal{X} \|\mathcal{Z}) + \sum_{n=1}^N \sum_{k=1}^K \bar{g}^{(n)}(\mathbf{u}_k^{(n)}) + g_3^{(n)}(\mathbf{u}_k^{(n)}) \\ \text{subject to } \mathcal{Z} = \hat{\mathcal{X}}, \mathbf{u}_k^{(n)} = \mathbf{a}_k^{(n)} \ (\forall n, k), \end{aligned} \quad (5.6)$$

where $\mathcal{Z}$ and $\mathbf{a}_k^{(n)}$ are auxiliary variables, and P_{Ω} is a projection function that only retains the divergence of the observed elements. To solve our problem efficiently with keeping both constraints and separability, we define an augmented Lagrangian for our problem:

$$\begin{aligned} L_{\rho}(A, B, \mathcal{Z}) = D_{\Omega}(\mathcal{X} \|\mathcal{Z}) + \frac{\rho}{2} \|\mathcal{Z} - \hat{\mathcal{X}} + \mathcal{P}\|_{\mathcal{F}}^2 \\ \sum_{n=1}^N \sum_{k=1}^K \bar{g}^{(n)}(\mathbf{a}_k^{(n)}) + g_3^{(n)}(\mathbf{u}_k^{(n)}) + \frac{\rho}{2} \|\mathbf{u}_k^{(n)} - \mathbf{a}_k^{(n)} + \mathbf{p}_k^{(n)}\|_2^2, \end{aligned} \quad (5.7)$$

where $\mathcal{U}$ and $\mathbf{p}_k^{(n)}$ are Lagrangian multipliers, and ρ is a step-size parameter, respectively. We summarize the minimization procedure for our proposed method in Algorithm 1. The minimization for ADMM can be efficiently calculated if a simple minimization operator for each of each $\mathbf{u}_k^{(n)}$ and $\mathbf{a}_k^{(n)}$ exists.

Algorithm 1: Alternative direction method of multiplier for our proposed non-negative tensor factorization

Input : $\mathcal{X}, \Omega, \lambda_1, \lambda_2, \lambda_3, K, \mathbf{U}^{(W)}$
Output: set of factor matrices A

- 1 Initialize parameters
- 2 Sample A, B , and $\mathcal{Z}$ from random distributions
- 3 **repeat**
- 4 Alternatively update parameters;
- 5 $\mathcal{Z} \leftarrow \operatorname{argmin}_{\mathcal{Z}} D_{\Omega}(\mathcal{X} \parallel \mathcal{Z}) + (\rho/2) \|\mathcal{Z} - \hat{\mathcal{X}} + \mathcal{P}\|_F^2$
- 6 **for** $n = 1$ **to** N **do**
- 7 Update $\mathbf{U}^{(n)}$ by solving Eq. (5.8)
- 8 **for** $k = 1$ **to** K **do**
- 9 Update $\mathbf{a}_k^{(n)}$ by solving Eq. (5.9)
- 10 **end**
- 11 $\mathbf{P}^{(n)} \leftarrow \mathbf{P}^{(n)} + (\mathbf{P}^{(n)} - \mathbf{B}^{(n)})$
- 12 **end**
- 13 $\mathcal{P} \leftarrow \mathcal{P} + (\mathcal{X} - \hat{\mathcal{X}})$
- 14 **until** *convergence*;

The loss function with respect to $\mathbf{U}^{(n)}$ and $\mathbf{a}_k^{(n)}$ contains the graph Laplacian regularizer and the non-separable graph-based and group-based penalties, respectively. Thus the main difficulty with our proposed method lies in the minimization of $\mathbf{U}^{(n)}$ and $\mathbf{a}_k^{(n)}$, whose minimization problems can be rewritten:

$$\mathbf{U}^{(n)} = \operatorname{argmin}_{\mathbf{U}^{(n)}} \frac{\rho}{2} \|\bar{\mathcal{Z}}_n - \mathbf{U}^{(n)} \mathbf{V}_n^\top\|_2^2 + \frac{\rho}{2} \|\mathbf{U}^{(n)} - \bar{\mathbf{V}}_n\|_2^2 + \lambda_3 \sum_{k=1}^K g_3^{(n)}(\mathbf{u}_k^{(n)}) \quad (5.8)$$

$$\mathbf{a}_k^{(n)} = \operatorname{argmin}_{\mathbf{a}_k^{(n)}} \bar{g}^{(n)}(\mathbf{a}_k^{(n)}) + \frac{\rho}{2} \|\bar{\mathbf{v}}_k^{(n)} - \mathbf{a}_k^{(n)}\|_2^2, \quad (5.9)$$

where $\bar{\mathcal{Z}} = \mathcal{Z} + \mathcal{U}$, $\mathbf{V}_n = \odot_{n=n'}^N \mathbf{U}^{(n)}$, $\bar{\mathbf{V}}_n = \mathbf{B}^{(n)} - \mathbf{U}^{(n)}$, and $\bar{\mathbf{v}}_k^{(n)} = \mathbf{u}_k^{(n)} + \mathbf{p}_k^{(n)}$. We efficiently solve the minimization of Eq. (5.8) by using the fact that it corresponds to the loss function of the graph regularized alternating least squares [88], which approximately runs in $\mathcal{O}(\operatorname{nnz}(\mathbf{L}^{(n)})K)$ ($\operatorname{nnz}(\cdot)$ is the number of non-zero elements).

The minimization problem in Eq. (5.9) corresponds to the calculation of the proximity operator, which is defined as: $\operatorname{prox}_{\gamma h}(\theta) = \operatorname{argmin}_{\theta} h(\theta) + \frac{1}{2\gamma} \|\hat{\theta} - \theta\|_2^2$. We present a minimization procedure for Eq. (5.9) by leveraging the properties of the proximity operator, and obtaining a minimizer for the sum of the non-negative indi-

cation function and other convex functions by the following property [102]: $\text{prox}_{\bar{g}^{(n)}} = \text{prox}_{g_{\geq 0}^{(n)}} \circ \text{prox}_{\lambda_1 g_1^{(n)} + \lambda_2 g_2^{(n)}}$. Thus, if we have a minimizer for $\lambda_1 g_1^{(n)} + \lambda_2 g_2^{(n)}$, we can attain the exact minimizer for $\bar{g}^{(n)}$ by setting negative parameters to zeros. A minimizer for $\lambda_1 g_1^{(n)} + \lambda_2 g_2^{(n)}$ can be simply calculated by employing a submodular function minimization procedure. Because the penalty functions of GFL and HOFL are the Lovász extensions [73] of the graph-representable submodular functions [45], we can attain a minimizer for the sum of functions $\lambda_1 g_1^{(n)} + \lambda_2 g_2^{(n)}$ by an efficient parametric network flow algorithm [25, 101, 114]. We show the details of our minimization procedure for this function in the appendix.

5.4 Related Works

There has been a lot of articles in which NTF was applied to analyze spatio-temporal data. Kimura et. al proposed a special NTF that decomposes a three-way tensor into two-factor matrices and a three-mode tensor for extracting log messages related to network failures [60]. Yang et. al proposed a combination of NTF without regularizers and post-processing for modeling user activities [118]. Koh et. al proposed an NTF that simultaneously decomposes multiple tensors to extract patterns appeared among different tensors[100]. NTF was used to extract spatio-temporal patterns from human-flow data [42]. However, all of those methods did not employ regularizers into NTF and their methods were not applicable to missing values. One exception is a paper of Sun et. al [99], in which they proposed a probabilistic non-negative Tucker decomposition for discovering interactions among factors. However, they did not incorporate the spatial and temporal structures into regularizers. Han et. al proposed an extension of NTF for predicting future observations [50]. However, they did not consider spatial structures. Our method can be applied to their framework to utilizing spatial and temporal regularizers. The estimation procedures of them were based on the multiplicative update rule and EM algorithm. Our proposed method can utilize graphs and groups of spatial and temporal features to regularize parameters and also employ ADMM as an estimation procedure.

5.5 Experiments

We conducted missing value completion problems with a traffic flow data set provided by City Pulse [3] and two bike-sharing system data sets recorded in Washington D. C.¹ and New York² [3].

The traffic flow data consist of the numbers of cars that passed at 419 locations every thirty minutes in Arhus City, Denmark. We picked 30 days from August 2nd to 31st 2014, and constructed three-way tensor $\mathcal{X} \in \mathbb{R}^{48 \times 30 \times 441}$ whose modes corresponded to 48 daily time points, 30 days, and 441 observation locations, respectively. From the bike-sharing system data in Washington D.C. and New York, we employed 15 days from April 1st to the 15th with 351 and 344 bike stations. We constructed three-way tensors $\mathcal{X} \in \mathbb{R}^{24 \times 15 \times 351}$ and $\mathcal{X} \in \mathbb{R}^{24 \times 15 \times 44}$ whose values were the numbers of bikes returned to the station in an hour. For the time mode, we utilized the adjacency of the time points as a graph. For the day mode, we employed the adjacency of days and the days of the week as a graph and groups. For the location mode, we used the inverse of the Euclid distance of GPS locations and clusters attained by k -nearest neighbors ($k = 5, 10$) for a graph and groups.

We exploited the Euclid distance as the divergence in experiments. We compared our proposed method (Proposed 1) and our proposed method with only the graph Laplacian regularizer (Proposed 2, $\lambda_1 = \lambda_2 = 0$) with NTF estimated by ADMM [57] (ADMM), NTF with the graph Laplacian regularizer [19] estimated by a multiplicative update rule considering missing values (Multi+Lap) [36], and NTF estimated by the multiplicative update rule (Multi). We set the proportion of observations to $p = \{0.1, 0.01, 0.005, 0.001\}$. By five-fold cross validation, we selected K and other hyperparameters from $K = \{3, 5, 10\}$ and $\{0.1, 1, 10\}$. We utilized the normalized RMSE (NRMSE) and the normalized deviation (ND) as error measurements:

$$\text{NRMSE} = \sqrt{(1/|\Omega|) \sum_{(p,t) \in \Omega} (x_{p,t} - \hat{x}_{p,t})^2 / Q}, \quad (5.10)$$

$$\text{ND} = (1/|\Omega|) \sum_{(p,t) \in \Omega} |x_{p,t} - \hat{x}_{p,t}| / Q, \quad (5.11)$$

¹<https://www.capitalbikeshare.com>

²<http://www.citibikenyc.com/>

Table 5.1: NRAMe for the traffic flow data of our proposed method (Proposed 1), our proposed method with the graph Laplacian regularizer (Proposed 2), NTF estimated by ADMM (ADMM), NTF with the graph Laplacian regularizer estimated by the multiplicative update rule (Multi+Lap), and NTF (Multi)

	$p = 0.1$	$p = 0.01$	$p = 0.005$	$p = 0.001$
Proposed 1	0.50 (0.00)	0.99 (0.03)	1.49 (0.03)	1.87 (0.01)
Proposed 2	0.51 (0.00)	1.12 (0.05)	1.49 (0.03)	1.89 (0.01)
ADMM	0.51 (0.00)	1.15 (0.03)	1.50 (0.02)	1.91 (0.00)
Multi+Lap	0.52 (0.00)	2.98 (1.86)	2.92 (1.00)	11.9 (11.7)
Multi	0.52 (0.00)	2.89 (2.22)	3.27 (1.80)	5.98 (6.24)

Table 5.2: NRAMe for the bike-sharing record data of Washington D.C.

Method	$p = 0.1$	$p = 0.01$	$p = 0.005$	$p = 0.001$
Proposed 1	1.67 (0.02)	2.14 (0.02)	2.21 (0.04)	2.43 (0.02)
Proposed 2	1.68 (0.01)	2.14 (0.05)	2.22 (0.05)	2.62 (0.08)
ADMM	1.68 (0.02)	2.21 (0.05)	2.32 (0.03)	2.47 (0.01)
Multi+Lap.	1.69 (0.01)	2.72 (0.22)	2.76 (0.24)	11.2 (4.62)
Multi	1.70 (0.01)	299.1 (405.7)	8.25 (4.87)	16.3 (3.13)

here $Q = (1/|\Omega|) \sum_{(p,t) \in \Omega} |x_{p,t}|$. We ran our experiments five times with randomly selected different missing values.

The results are shown in Tables 5.1, 5.2, 5.3, 5.4, 5.5, and 5.6, where the left and right values in a cell correspond to the average and the standard deviation of those values. We confirmed that our proposed methods showed the best performance in every setting. Our proposed method was robust to the appearance of a large portion of missing values for every data set $p = \{0.01, 0.005, 0.001\}$. Our proposed method with both the graph-based Laplacian and structured regularizer (Proposed 1) showed better or competitive performance with our proposed method with the graph-based Laplacian regularizer (Proposed 2). Furthermore, our proposed method with the graph-based Laplacian regularizer (Proposed 2) always outperformed the same model estimated by the multiplicative update rule (Multi+Lap). This result was caused by the benefits of simultaneously combining graph-based and structured regularizers with graph and group structures. Thus our proposed model and parameter estimation procedure both contributed to the improvements on missing value interpolations. The existing methods resulted in poor performances with settings where a large portions of tensor elements were missing.

To check the qualitative performances of the interpretability, we showed the ex-

Table 5.3: NRAMe for the bike-sharing record data of New York

Method	$p = 0.1$	$p = 0.01$	$p = 0.005$	$p = 0.001$
Proposed 1	0.98 (0.00)	1.28 (0.02)	1.42 (0.01)	1.62 (0.01)
Proposed 2	0.98 (0.00)	1.30 (0.03)	1.44 (0.01)	1.63 (0.01)
ADMM	0.98 (0.00)	1.34 (0.02)	1.49 (0.01)	1.65 (0.01)
Multi+Lap	1.00 (0.02)	27.2 (41.3)	5.68 (3.22)	1.86 (0.17)
Multi	1.00 (0.02)	53.7 (22.0)	26.6 (29.4)	3.04 (1.03)

Table 5.4: ND for the traffic flow data of our proposed method (Proposed 1), our proposed method with the graph Laplacian regularizer (Proposed 2), NTF estimated by ADMM (ADMM), NTF with the graph Laplacian regularizer estimated by the multiplicative update rule (Multi+Lap), and NTF (Multi)

Method	$p = 0.1$	$p = 0.01$	$p = 0.005$	$p = 0.001$
Proposed 1	0.27 (0.00)	0.46 (0.01)	0.70 (0.01)	0.92 (0.01)
Proposed 2	0.28 (0.00)	0.51 (0.02)	0.71 (0.02)	0.94 (0.00)
ADMM	0.28 (0.00)	0.53 (0.01)	0.73 (0.01)	0.94 (0.00)
Multi+Lap	0.28 (0.00)	0.61 (0.08)	0.78 (0.02)	1.19 (0.16)
Multi	0.28 (0.00)	0.60 (0.08)	0.81 (0.04)	1.18 (0.24)

tracted factors of proposed method (Proposed 1) and existing NTF with the Laplacian regularizer (Multi+Lap) from traffic flow data in Figs. 5.2, 5.3, 5.4, 5.5, 5.6, and 5.7, where $p = 0.1$. The degree of freedom (DoF) in Figures corresponded the number of segments in a factor matrix. Thanks to the Laplacian and structured regularizers, proposed method extracted the interpretable latent factors in which both smooth and flat properties appeared, whose DoF of parameters in factor matrices were extremely less than that of NTF with the Laplacian regularizer. Our factors with low DoF were easy to find change points. For example, the blue factor had a change at 3 am and gradually grew until 6 am. Then it took the constant values until 3 pm in Fig. 5.2. This factor also has the same value from day 2 to day 6 and from day 8 to day 13. Thus, we can easily understand that the blue factor in Figs. 5.2 and 5.4 corresponded to activity that occurred in weekday during daylight with a spatial pattern in Fig. 5.6. However, NMF with the Laplacian regularizer resulted in messy factors. We also showed that of bike-sharing data in Washington D.C. in Figs. 5.8, 5.9, 5.10, 5.11, 5.12, and 5.13. Our proposed method also extracted more interpretable patterns than existing NTF. For example, the yellow factor of ours in Fig. 5.8 had a change point at 8 am. After it had taken a peak at 12 am, it kept the same value from 1 pm to 5 pm. Then its value gradually decreased to zero. The

Table 5.5: ND for the bike-sharing record data of Washington D.C.

Method	$p = 0.1$	$p = 0.01$	$p = 0.005$	$p = 0.001$
Proposed 1	0.81 (0.00)	0.91 (0.01)	1.04 (0.02)	1.04 (0.08)
Proposed 2	0.81 (0.01)	0.91 (0.00)	1.04 (0.02)	1.05 (0.09)
ADMM	0.81 (0.01)	0.91 (0.01)	1.10 (0.01)	1.20 (0.01)
Multi+Lap	0.81 (0.00)	1.19 (0.05)	1.51 (0.15)	1.99 (0.54)
Multi	0.81 (0.00)	9.70 (6.91)	1.44 (0.10)	2.56 (0.31)

Table 5.6: ND for the bike-sharing record data of New York

Method	$p = 0.1$	$p = 0.01$	$p = 0.005$	$p = 0.001$
Proposed 1	0.60 (0.00)	0.72 (0.01)	0.81 (0.01)	0.93 (0.01)
Proposed 2	0.60 (0.00)	0.73 (0.01)	0.82 (0.01)	0.93 (0.01)
ADMM	0.60 (0.00)	0.74 (0.01)	0.84 (0.01)	0.94 (0.01)
Multi+Lap	0.60 (0.00)	2.13 (1.48)	1.53 (0.21)	1.05 (0.05)
Multi	0.60 (0.00)	4.48 (0.79)	2.51 (0.95)	1.21 (0.07)

yellow factor in Fig. 5.10 had the same high value on day 2, 3, 9, and 10. Thus, we confirmed that this factor indicated a weekend afternoon activity with a spatial pattern in Fig. 5.12. Similar interpretations can be obtained from other factors of ours.

5.6 Summary

In this chapter, we proposed a structurally regularized non-negative tensor factorization that incorporated both the graph Laplacian and the structured regularizers on latent factors. For the structured regularizer, we employed the generalized fused lasso and the higher-order fused lasso to represent both graph-based and group-based information in time and space. We introduced a flexible and efficient parameter estimation method based on the alternating direction method of multipliers and showed a proximity operator for our unified structured regularizer. With experiments on a missing value imputation problem of three data sets, we confirmed that our proposed method showed the best quantitative performance and successfully extracted more interpretable latent factors than the existing non-negative tensor factorization methods.

5.7 Appendix

Although the issue in Eq. (5.9) is a general problem containing the previous problems [17, 101, 114] as special cases, we can solve it in a similar manner as these works. We first briefly introduce the parametric optimization method for a non-decreasing set function. Let $\alpha \in \mathbb{R}_{\geq 0}$, and define set function $l_\alpha(S) = l(S) - \alpha \mathbf{1}(S)$ ($\forall S \subset V$), where $\mathbf{1}(S) = \sum_{i \in S} 1$. Then if l is a non-decreasing submodular function, then there exists a set of $r + 1$ ($\leq |V|$) subsets: $S^* = \{S_0 \subset S_1 \subset \dots \subset S_r\}$, where $S_j \subset V$, $S_0 = \emptyset$, and $S_r = V$, and $r + 1$ subintervals Q_r of α : $Q_0 = [0, \alpha_0)$, $Q_1 = [\alpha_1, \alpha_2)$, $\dots$, $Q_r = [\alpha_r, \infty)$, such that, for each $j \in \{0, 1, \dots, r\}$, S_j is the unique maximal minimizer of $h_\alpha(S)$, $\forall \alpha \in Q_j$. A minimizer of Eq. (5.9) $\mathbf{t}^* = (t_1^*, t_2^*, \dots, t_{|V|}^*)$ is then determined: $t_i^* = \frac{f(S_{j+1}) - f(S_j)}{\mathbf{1}(S_{j+1} \setminus S_j)}$, $\forall i \in (S_{j+1} \setminus S_j)$, $j = (1, \dots, r)$. We introduce two lemmas [79] to see that l is a non-decreasing submodular function.

Lemma 1 (Lemma) *For any $\eta \in \mathbb{R}$ and submodular function h , $\mathbf{t}^*$ is an optimal solution to $\min_{\mathbf{t} \in \mathcal{B}(l)} \|\mathbf{t}\|_2^2$ if and only if $\mathbf{t}^* - \eta \mathbf{1}$ is an optimal solution to $\min_{\mathbf{t} \in \mathcal{B}(l) + \eta \mathbf{1}} \|\mathbf{t}\|_2^2$.*

Lemma 2 (Lemma) *Set $\eta = \max_{i=1, \dots, |V|} \{0, l(V \setminus \{i\}) - l(V)\}$, and then $l + \eta \mathbf{1}$ is a non-decreasing submodular function.*

With Lemma 2, we solve

$$\min_{S \subset V} f(S) - \hat{z}(S) + (\eta - \alpha) \mathbf{1}(S), \quad (5.12)$$

and apply Lemma 1 to obtain a solution to the original problem. With fixed α , we can efficiently attain the optimal of Eq. (5.12) because this is a minimum cut problem.

Proposition 1 *The problem in Eq. (5.12) is equivalent to a minimum s/t-cut problem.*

Proof 1 *Each component in f is graph-representable. The graph is obtained due to the additive property of the graph-representative submodular functions, where the groups of parameters are represented with hyper nodes u_1^k, u_0^k that corresponds to each group, and the capacities of the edges between hyper and ordinal nodes $v_i \in V$.*

The attained graph includes both of the GFL and HOFL graphs as spacial cases. As a consequence, we can attain a sequence of solutions for all α of the parametric s/t minimum-cut problem (5.12) using an efficient parametric-flow algorithm, such as [46], that runs in $O(|V'||E'|\log(|V'|^2/|E'|))$ as the worst case and $|V'|$ and $|E'|$ are the number of nodes and edges of the graph.

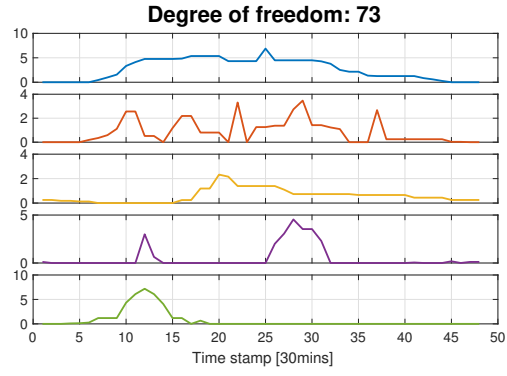


Figure 5.2: Time factors of Proposed 1 on the traffic flow data

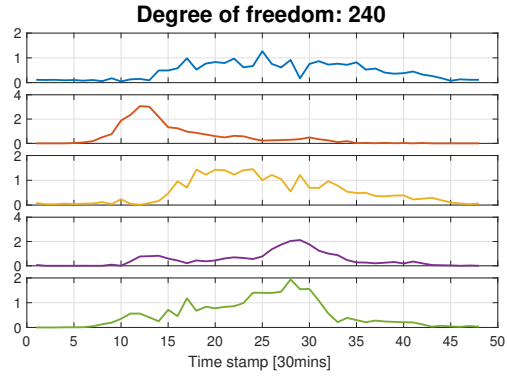


Figure 5.3: Time factors of Multi+Lap on the traffic flow data

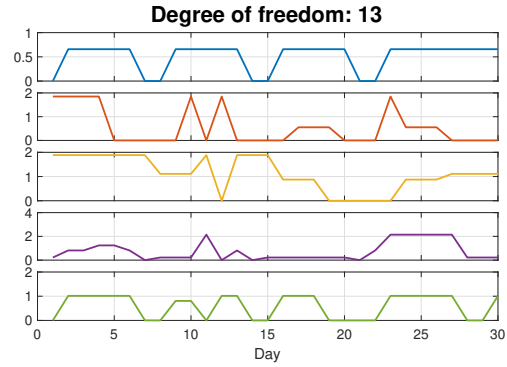


Figure 5.4: Day factors of Proposed 1 on the traffic flow data

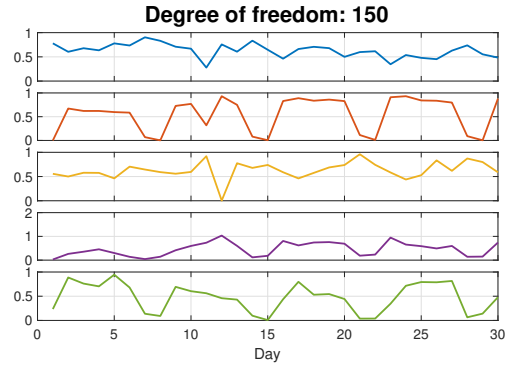


Figure 5.5: Day factors of Multi+Lap on the traffic flow data

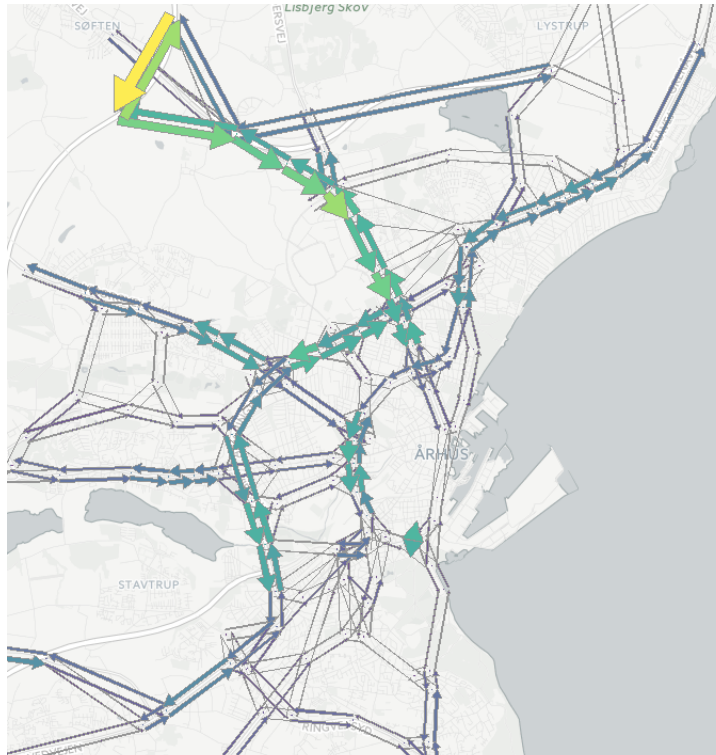


Figure 5.6: A spatial pattern of the blue factor of Proposed 1 on the traffic flow data



Figure 5.7: A spatial pattern of the blue factor of Multi+Lap on the traffic flow data

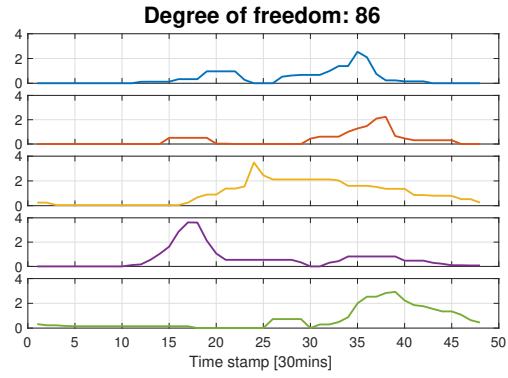


Figure 5.8: Time factors of Proposed 1 on the bike-sharing data of Washington D.C.

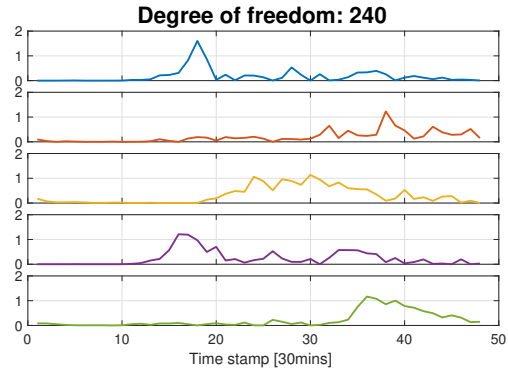


Figure 5.9: Time factors of Multi+Lap on the bike-sharing data of Washington D.C.

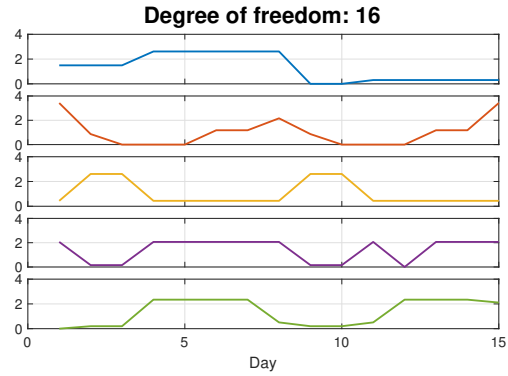


Figure 5.10: Day factors of Proposed 1 on the bike-sharing data of Washington D.C.

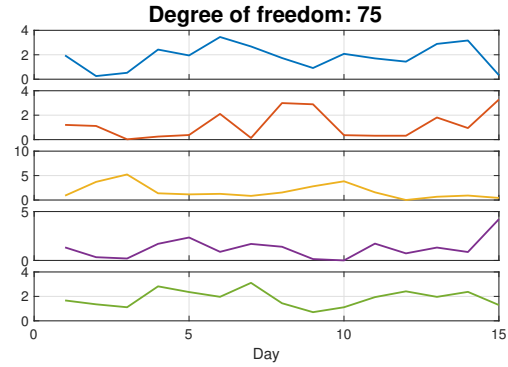


Figure 5.11: Day factors of Multi+Lap on the bike-sharing data of Washington D.C.

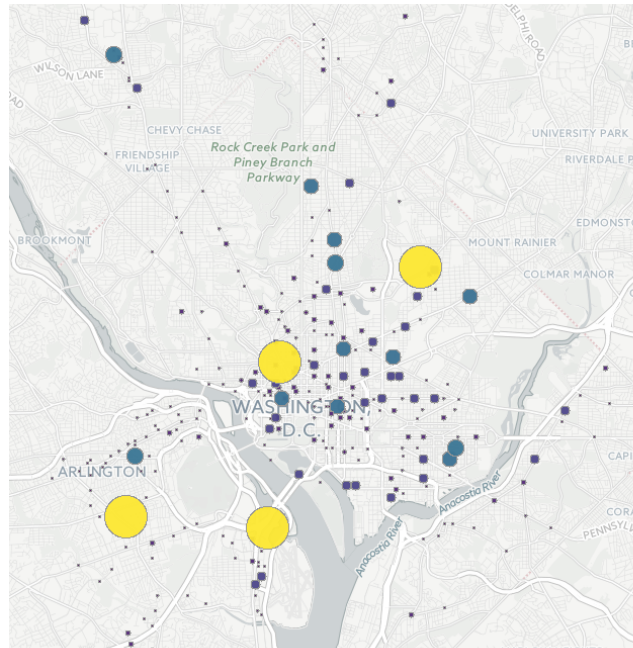


Figure 5.12: A spatial pattern of the yellow factor of Proposed 1 on the bike-sharing data of Washington D.C.

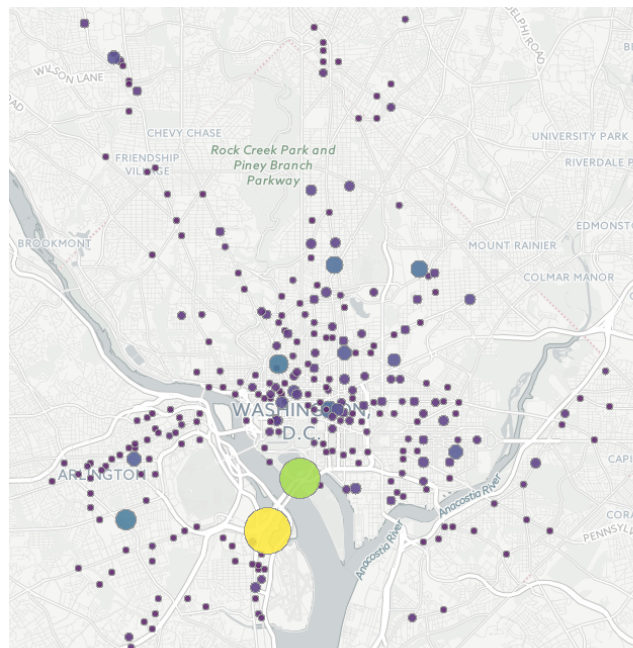


Figure 5.13: A spatial pattern of the yellow factor of Multi+Lap on the bike-sharing data of Washington D.C.

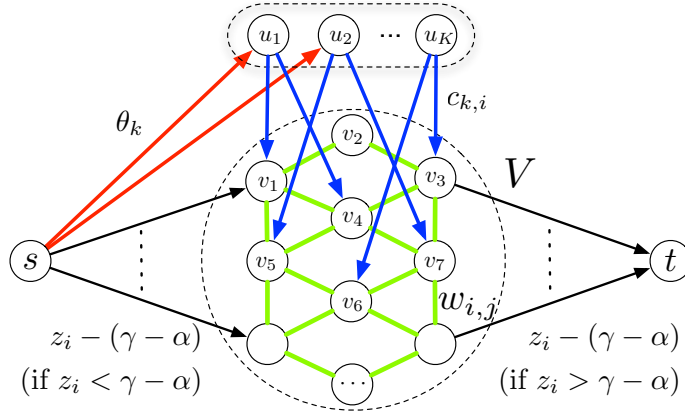


Figure 5.14: Minimum s/t -cut problem of Problem (5.12). Given graph $G = (V', E')$ for our proposed method, capacities of edges $c(v'_i, v'_j)$ are defined as: $c(s, u_k) = \theta_k$, $c(v_i, v_j) = w_{i,j}$, $c(u_k, v_i) = c_{k,i}$, $c(s, v_i) = z_i - (\gamma - \alpha)$ if $z_i > \gamma - \alpha$, and $c(v_i, t) = (\gamma - \alpha) - z_i$ if $z_i < \gamma - \alpha$. Nodes u_k $k = (1, \dots, K)$ are hyper nodes that correspond to the groups g_k . And s, t , and v_i are source, sink, and parameters nodes, respectively.

Chapter 6

Autoregressive Tensor

Factorization for Spatio-temporal Predictions

6.1 Introduction

High-resolution spatial-temporal data covering a wide area have become available due to highly effective and inexpensive sensor monitoring systems, which provide rich and complex information about our cities and its civic activities. Spatio-temporal data analysis, which is a common research topic for understanding specific phenomena by inspecting such data, has been considered in various fields, including spatial statistics [75], spatial economics [5], climatology [41, 126], remote sensing [35], spatial ecology [47], and public health [105, 107]. A unique property of spatio-temporal data is providing where and when the data were observed. By utilizing this property, researchers have tried to interpolate observations of unknown locations from known locations and predict future observations from past time points. The most challenging problem in such analysis is to predict observations of unknown locations because that requires methods that simultaneously capture spatial and temporal dependencies and leverage them. Such predictions must be beneficial for detecting spatial and temporal demands, for example, helping taxi allocation services dispatch vehicles. Sharing economy services starts forecasting consumer demands for sharing

bikes, rides, or the accommodations of new service locations.

In modern situations, the number of observed locations in an area covered by a sensor monitoring system exceeds hundreds or thousands, and each sensor is equipped with multiple types of data sources. New observations are sent every hour or less often over several weeks or months. For example, a bike-sharing service in New York City consists of three hundred bike stations and observes the number of bikes at the station every 30 minutes. An air quality monitoring system contains dozens of monitoring posts covering Beijing City and every hour provides multiple types of observations, such as, PM25, PM10, NO2, temperature, and humidity [125]. Unfortunately, it remains difficult for a monitoring system to cover an entire region of a city with moderate resolution, because a trade-off exists between the number or type of sensors and the size of the covered area. On the other hand, because inexpensive monitoring systems are not robust, these data almost always contain observation failures due to errors of sensor equipment, data transmissions, or storage malfunctions [67]. Therefore, not only the size and multiple types of data but also dealing with missing values are problematic for analyzing the current spatio-temporal data.

A vector autoregressive model (VAR) and Gaussian processes (GP) [89] are common choices to make a prediction from a multivariate time-series. These methods incorporate linear or non-linear spatial and temporal correlations into predictions. However, when we attempt to analyze a multivariate time-series, VAR and GP need exponential computational costs [11, 14]. Thus, these models are infeasible to analyze large-size spatio-temporal data.

Alternatives for large-scale spatio-temporal data analysis are matrix/tensor factorization techniques [11, 21, 67, 88, 121]. These methods employ tensor formulation $\mathcal{X}$ for multiple types of a multivariate time-series, whose first, second, and third modes correspond to the number of P univariate time-series, time-stamps T , and M observation type, and decompose the tensor into a few latent factors. Tensor factorization methods scale linearly as with P , T , and M and easily handle the missing values in the data. We illustrate a tensor formulation for spatio-temporal data on the left of Fig. 6.1. When we attempt to predict the future taxi demands of new business locations by analyzing past trip records, the problems can be regarded as estimating three-colored blocks from existing business district data (white block $\mathcal{X}$). The interpolation problem is to estimate the past demands of the new locations (blue block

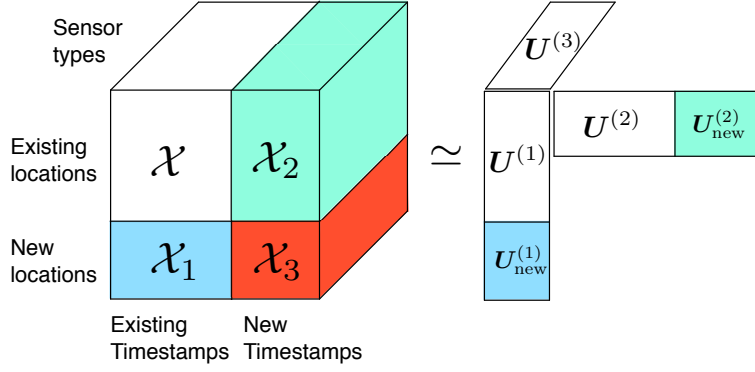


Figure 6.1: Tensor factorization for multiple types of spatio-temporal data. Blocks $\mathcal{X}_1$, $\mathcal{X}_2$, and $\mathcal{X}_3$ correspond to problems of interpolation of unknown locations, prediction of known locations, and prediction of unknown locations, respectively.

$\mathcal{X}_1$). Similarly, the task of predicting the existing business districts corresponds to estimates of the green block $\mathcal{X}_2$. The outputs are predictions for the future demands of the new business locations (red block $\mathcal{X}_3$).

To incorporate spatial and temporal dependencies into decomposition, most existing factorization methods have adopted graph-based regularizers [88] that encourage latent factors to be smooth with auxiliary graph representing spatial and temporal structures. These methods demand a previous inference of weighted spatial and temporal graph matrices, however, there is almost no unified guideline for making such matrices [4, 121]. Thus, the performance of those methods heavily relies on the choices of weights and graph structures. Furthermore, these methods fall short in cases where negative correlations appear. Yu et al. [121] recently proposed a novel method called temporal regularized matrix factorization (TRMF), which introduced an autoregressive regularizer into factorization frameworks. By simultaneously learning latent factors and temporal dependencies, TRMF showed considerable improvement on predicting the future observations of known locations. However, they did not consider the problem of learning spatial dependencies. Thus, the question remains open whether we can make accurate interpolations and predictions for unknown locations.

In this chapter, we tackle the problem of interpolating and predicting the future observations of unknown locations. To solve it, we propose a new tensor factorization method that decomposes a tensor into latent factors by simultaneously learning the spatial and temporal autocorrelations in each latent factor. We introduce spa-

tial autoregressive regularizers based on traditional autoregressive models, known as the two-dimensional autoregressive model [39] and the simultaneous autoregressive model [7]. By combining our spatial autoregressive regularizer and temporal regularizers [121], we can predict the feature observations of unknown locations. In addition, by utilizing a property that all observations in data take only non-negative values, we can extract more robust and interpretable latent factors from spatial-temporal data. To estimate factor matrices and coefficients for spatial and temporal autoregressive models, we show a computationally efficient procedure. We experimentally assess the predictive performances of our proposed method on five publicly available spatio-temporal data sets and demonstrate how it significantly improved the predictive accuracy on interpolating and predicting the observations of unknown locations.

6.2 Preliminaries

6.2.1 Tensor factorization methods

We represent spatio-temporal data observed from P locations with T time-stamps and M types of data sources as a three-way tensor, $\mathcal{X} \in \mathbb{R}^{P \times T \times M}$. A set of observed elements in $\mathcal{X}$ is represented as $\Omega = \{(p, t, m) \mid x_{p,t,m} \text{ is observed.}\}$. We denote a rank- K factor matrix for each mode as $\mathbf{U}^{(1)} \in \mathbb{R}^{P \times K}$, $\mathbf{U}^{(2)} \in \mathbb{R}^{T \times K}$, and $\mathbf{U}^{(3)} \in \mathbb{R}^{M \times K}$ where $K \ll \min(P, T, M)$. We employ CP decomposition as a tensor factorization model in this chapter, because the computational costs of the Tucker decomposition model are much higher than CP decomposition. An empirical loss function of a tensor factorization (or completion) method with regularizers is defined as:

$$\begin{aligned}
& L(\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}) \\
&= \sum_{(p,t,m) \in \Omega} \left(x_{p,t,m} - \sum_{k=1}^K u_{p,k}^{(1)} u_{t,k}^{(2)} u_{m,k}^{(3)} \right)^2 + \sum_{n=1}^2 \lambda_n g_n(\mathbf{U}^{(n)}) \\
&= \left\| P_{\Omega}(\mathcal{X} - \mathcal{C} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}) \right\|_2^2 + \sum_{n=1}^2 \lambda_n g_n(\mathbf{U}^{(n)}), \tag{6.1}
\end{aligned}$$

where we denote $g_n, \lambda_n, \mathcal{C} \in \mathbb{R}^{K \times K \times K}$ and $P_\Omega(\cdot)$ as a penalty function, a hyperparameter for the n -th mode, a unit diagonal core tensor, and a projection function that only retains the entries of observed values, respectively. The first term of the above equation represents the discrepancy between actual observed value $x_{p,t,m}$ and its estimation $\hat{x}_{p,t,m} = \sum_{k=1}^K u_{p,k}^{(1)} u_{t,k}^{(2)} u_{m,k}^{(3)} \in \hat{\mathcal{X}}$. The second term is a sum of the regularizer functions that encourage factor matrices to represent specific structures. The goal of the tensor factorization (completion) method is to learn optimal factor matrices Θ that minimize the empirical loss function:

$$\underset{\Theta}{\operatorname{argmin}} L(\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}). \quad (6.2)$$

We can attempt to solve the predictive problems in Fig. 6.1 by utilizing extracted factor matrices. The estimation for interpolation (blue block $\mathcal{X}_1$) can be obtained by an inner product of $\mathcal{C} \times_1 \mathbf{U}^{(1)}_{\text{new}} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}$. Similarly, the predictions of known locations (green block $\mathcal{X}_2$) correspond to $\mathcal{C} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)}_{\text{new}} \times_3 \mathbf{U}^{(3)}$. To predict the future observations of unknown locations, we estimate red block $\mathcal{X}_3$ by $\mathcal{C} \times_1 \mathbf{U}^{(1)}_{\text{new}} \times_2 \mathbf{U}^{(2)}_{\text{new}} \times_3 \mathbf{U}^{(3)}$.

A common choice for penalty function g_n is absolute function $\|\mathbf{U}^{(n)}\|_1$, also known as the lasso, that takes sparsity assumptions into the factor matrices. However, this regularizer is not appropriate for spatio-temporal data analysis because it independently imposes absolute loss on each parameters without considering the spatial and temporal dependencies among the features. It is invariant to permutations of the features on each mode.

Another popular choice of penalty function is a indicator function onto the non-negative region $\mathcal{P}_{\geq 0}(\mathbf{U}^{(n)})$, which can be defined as:

$$\mathcal{P}_{\geq 0}(\mathbf{U}^{(n)}) = \begin{cases} 0 & \text{if } (u_{i,k}^{(n)} \geq 0, \forall(i, k)), \\ +\infty & \text{otherwise} \end{cases}. \quad (6.3)$$

Non-negative tensor factorization methods [33, 117] employ this function to put an assumption that latent factors of a tensor whose elements have only non-negative values should be also non-negative. With this regularization, non-negative tensor factorization prevents overfitting and results in more interpretable factor matrices than tensor factorization methods without this regularizer [48].

A graph Laplacian regularizer has been widely applied to tensor factorization methods to adopt spatial and temporal dependencies on its decomposition. Denote a weighted adjacency matrix for the n -th mode factor matrix as $\mathbf{W}^{(n)} \in \mathbb{R}^{I_n \times I_n}$. Laplacian matrix $\mathbf{L}^{(n)}$ can be defined as: $\mathbf{L}^{(n)} = \mathbf{D}^{(n)} - \mathbf{W}^{(n)}$, where $\mathbf{D}^{(n)}$ is a diagonal matrix whose elements are the sums of each row of $\mathbf{W}^{(n)}$. Then the graph Laplacian regularizer can be represented as:

$$\begin{aligned} g_n(\mathbf{U}^{(n)}) &= \sum_{k=1}^K \mathbf{u}_k^{(n)\top} \mathbf{L}^{(n)} \mathbf{u}_k^{(n)} \\ &= \sum_{k=1}^K \sum_{i=1}^{I_n} \sum_{j=1}^{I_n} w_{i,j}^{(n)} (u_{i,k}^{(n)} - u_{j,k}^{(n)})^2. \end{aligned} \quad (6.4)$$

When we set a large weight value in the adjacency matrix, the penalty function prefers that the adjacent parameters in a factor matrix take similar values. Thus, we can induce smoothness with the auxiliary structures by this regularizer. Although it has been shown useful in spatio-temporal analysis, however, the construction for the weights of the adjacency matrix depends on the user's choice. A common problem is that explicit dependency information, like spatial and temporal features, is not available, even if we have such information, these pre-constructed weights are not suitable for the latent factors. Moreover, this formulation cannot capture the negative correlation among locations or time-stamps.

6.2.2 Autoregressive models

The vector autoregressive model (VAR) employs the autocorrelations of a multivariate time-series to predict the future observations. Denote a multivariate time-series $\mathbf{y}_t \in \mathbb{R}^K$, lag set L , and weights $A = \{\mathbf{A}_l \in \mathbb{R}^{K \times K} \mid l \in L\}$. A VAR model approximates vector $\mathbf{y}_t$ by a weighted sum of previous ones with Gaussian noises ϵ_t : $\mathbf{y}_t = \sum_{l \in L} \mathbf{A}_l \mathbf{y}_{t-l} + \epsilon_t$. Yu et al. [121] proposed a temporal regularizer that corresponds to this AR model:

$$\frac{1}{2} \sum_{t=\bar{L}}^T \left\| \mathbf{y}_t - \sum_{l \in L} \mathbf{A}_l \mathbf{y}_{t-l} \right\|_2^2 + \frac{\eta}{2} \sum_{t=1}^T \|\mathbf{y}_t\|, \quad (6.5)$$

where $\bar{L} = 1 + \max(L)$ and η is a weight parameter for ℓ_2 norm term. To prevent overfitting, they considered diagonal $\mathbf{A}_l$ that reduced the VAR model to a univariate AR model.

In the field of social economics, the simultaneous autoregressive (SAR) model is a popular method to analyze spatial data [7] by incorporating spatial autocorrelations into regression frameworks in a different way from the 2D-AR model. We denote weighted adjacency matrix $\mathbf{W} \in \mathbb{R}^{P \times P}$ whose row vector is normalized to $\sum_{p'=1}^P w_{p,p'} = 1$. Thus $\mathbf{W}$ is a single stochastic matrix. SAR can be defined as:

$$\mathbf{y} = \rho \mathbf{W} \mathbf{y} + \boldsymbol{\epsilon}, \quad (6.6)$$

where $\rho \in \mathbb{R}$ is a parameter for modeling a spatial gap among sensor locations.

The two-dimensional autoregressive (2D-AR) model [39] was proposed in the field of computer vision to take into account the spatial autocorrelations in image analysis. Denote vector $\mathbf{y} \in \mathbb{R}^P$ whose element y_p corresponds to an observed value of the (i, j) -th pixel in an image. We represent a spatial lag set with L -th order as $E = \{(r, s) \mid 1 \leq r < L \cup 1 \leq s < L\}$ and weights as $\Theta = \{b_{r,s} \mid (r, s) \in E\}$. The 2D-AR model assumes that an observed value of $\mathbf{y}$ is a weighted sum of its neighbors with Gaussian noise:

$$y_{i,j} = \sum_{(r,s) \in E} b_{r,s} y_{i+r,j+s} + \epsilon. \quad (6.7)$$

The 2D-AR model has been successful in various tasks, such as estimating the observations of an image. However, because this method was designed for computer vision, it is only applicable for a set of locations having a grid shape.

6.3 Proposed method

6.3.1 Autoregressive tensor factorization

The tensor factorization method decomposed a tensor into K latent factors in which different spatial and temporal patterns were extracted. Thus, it is natural to suppose different spatial and temporal dependencies or autocorrelations also exist in each factor. To incorporate this idea, we propose a tensor factorization method

that incorporates autoregressive regularizers as penalty functions on the spatial and temporal modes. We define our proposed method as:

$$L(\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}) = \|P_{\Omega}(\mathcal{X} - \mathcal{C} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)})\|_2^2 + \lambda_1 g_1(\mathbf{U}^{(1)}, A) + \lambda_2 g_2(\mathbf{U}^{(2)}, B), \quad (6.8)$$

where g_1 and g_2 are our spatial autoregressive regularizer and the temporal autoregressive regularizer [121]. In addition, when all of the observed elements of tensor $\mathcal{X}$ are non-negative, we can add the non-negative constraints on factor matrices by putting the indicator function onto the non-negative region as: $\sum_{n=1}^3 \mathcal{P}_{\geq 0}(\mathbf{U}^{(n)})$. We denote the parameters of the spatial and temporal regularizers for each factor as A and B . Note that this formulation can be generalized to more higher-order tensors.

For learning the spatial dependencies of latent factors, we propose two variants of spatial regularizers based on the spatial autoregressive models that were introduced in Section 6.2.2. The main advantage of this formulation is that our proposed method does not demand previous inference of the spatial and temporal dependencies. This method can learn the individual dependencies suitable for each latent factor.

6.3.2 Spatial autoregressive regularizer

Unlike the existing assumption of the 2D-AR model, the locations of sensors are not always placed in a grid shape due to the topological condition of the road networks or the purpose of the sensors. Thus, this method is inapplicable to spatio-temporal data analysis. The SAR model is a common choice for dealing with such non-grid locations of sensors. However, since its parameter consists of a single scalar ρ , its capability to detect the spatial dependency is limited.

To solve these problem, we propose a directed autoregressive (DAR) regularizer that is a generalization of both the 2D-AR model and the SAR model. Our key assumption is that there is a direction based dependency among sensor observations and thus an observation of a sensor can be approximated by a sum of these coefficients, distance based weights, and observations of its neighbors. We illustrate the main idea of our DAR in Fig. 6.2. We split the 2π angle into N discrete directions and define the coefficient parameters for each direction.

Denote the coefficients for our DAR model as $B = \{b_{k,n} \mid \forall(k, n)\}$, the k -nearest

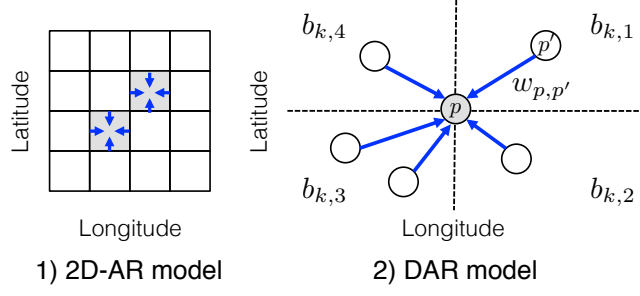


Figure 6.2: Two-dimensional autoregressive model and our directed autoregressive model. Our method splits the 2π angle into N discrete parts.

neighbors of the p -th location as E_p , and the normalized weights between the p -th location and its neighbors as $w_{p,p'} \in \mathbb{R}_{\geq 0}$ where $\sum_{p' \in E_p} w_{p,p'} = 1$. We define a penalty function of our DAR regularizer as:

$$g^{\text{DAR}}(\mathbf{U}^{(1)}, B) = \sum_{k=1}^K \sum_{p=1}^P \left(u_{p,k}^{(1)} - \sum_{p' \in E_p} b_{k,n_{p'}} w_{p,p'} u_{p',k}^{(1)} \right)^2 + \frac{\eta}{2} \|\mathbf{u}_k^{(1)}\|_2^2, \quad (6.9)$$

where $n_{p'}$ indicates the direction from the p -th location to the p' -th location. The normalized weights can represent the inverse distance between locations and, in addition, avoid the problem when all of its neighbors are in the same direction as SAR model.

We add the ℓ_2 loss term with parameter η to assure strong convexity for our penalty function, which can be rewritten in simple formulation:

$$g^{\text{DAR}}(\mathbf{U}^{(1)}, B) = \sum_{k=1}^K \left\| \mathbf{u}_k^{(1)} - \bar{\mathbf{B}}_k \mathbf{u}_k^{(1)} \right\|_2^2 + \frac{\eta}{2} \|\mathbf{u}_k^{(1)}\|_2^2. \quad (6.10)$$

We denote matrix $\bar{\mathbf{B}}_k \in \mathbb{R}^{P \times P}$ whose elements are set to:

$$\bar{b}_{k,p,p'} = \begin{cases} b_{k,n_{p'}} w_{p,p'} & (\text{if } p' \in E_p), \\ 0 & (\text{otherwise}) \end{cases}. \quad (6.11)$$

In a similar way, we introduce a spatial autoregressive regularizer (SAR) employing

the simultaneous autoregressive model:

$$g^{\text{SAR}}(\mathbf{U}^{(1)}, \rho) = \sum_{k=1}^K \left\| \mathbf{u}_k^{(1)} - \rho_k \mathbf{W}^{(1)} \mathbf{u}_k^{(1)} \right\|_2^2 + \frac{\eta}{2} \|\mathbf{u}_k^{(1)}\|_2^2. \quad (6.12)$$

6.3.3 Parameter estimation

We propose an alternative minimizing procedure to attain a minimizer for our proposed method by rewriting our loss function as a constrained minimization problem [20]:

$$\begin{aligned} \min_{\Theta_1, \Theta_2} \quad & \|\mathcal{Z} - \mathcal{C} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}\|_2^2 + \lambda_1 g_1(\mathbf{U}^{(1)}, \Theta_1) + \lambda_2 g_2(\mathbf{U}^{(2)}, \Theta_2) \\ \text{subject to} \quad & P_\Omega(\mathcal{X} - \mathcal{Z}). \end{aligned} \quad (6.13)$$

Since our loss function is multi-convex, which means that the loss function is non-convex but convex as with each block parameter, we can easily alternatively update each block parameter in an efficient manner.

Updating Θ_1 : From the definition, penalty function g^{DAR} for $\mathbf{b}_k = (b_{k,1}, \dots, b_{k,N})^\top$ takes the following form:

$$g^{\text{DAR}}(\mathbf{B}_k) = \sum_{p=1}^P \left(u_{p,k}^{(1)} - \sum_{p' \in E_p} b_{k,n_{p'}} w_{p,p'} u_{p',k}^{(1)} \right)^2 + \frac{\eta}{2} \|\mathbf{u}_k^{(1)}\|_2^2. \quad (6.14)$$

This is a convex quadratic function, and thus we can obtain a minimizer by solving an ordinal least square problem:

$$\mathbf{b}_k^* = (\mathbf{S}\mathbf{S}^\top + \eta \mathbf{I}_N)^{-1} \mathbf{S} \mathbf{u}_k^{(1)}, \quad (6.15)$$

where we denote a diagonal matrix with the N dimension as $\mathbf{I}_N$. We use matrix $\mathbf{S} \in \mathbb{R}^{N \times P}$ to denote $s_{n,p} = \sum_{p' \in E_{n,p}} w_{p,p'} u_{p',k}^{(1)}$, where $E_{n,p} = \{p' \mid p' \in E_p \cup n_{p'} = n\}$.

Penalty function g^{SAR} for ρ_k equals a loss function of the SAR model:

$$g^{\text{SAR}}(\rho_k) = \sum_{k=1}^K \left\| \mathbf{u}_k^{(1)} - \rho_k \mathbf{W}^{(1)} \mathbf{u}_k^{(1)} \right\|_2^2 + \frac{\eta}{2} \|\mathbf{u}_k^{(1)}\|_2^2. \quad (6.16)$$

There are many parameter estimation methods for the SAR model. The cheapest

solution is an ordinal least square solution; however, since this solution is biased [6], we use a maximum likelihood solution as an optimal parameter for ρ_k .

Updating Θ_2 : A parameter estimation problem for w_l equals solving the ridge regression problem [121], which we can solve by Cholesky factorization whose computational cost is $\mathcal{O}(|L|^3 + T|L|^2)$.

Updating $\mathbf{U}^{(1)}$: The minimization problem for $\mathbf{U}^{(1)}$ can be defined as:

$$\min_{\mathbf{U}^{(1)}} \|\mathcal{Z}_1 - \mathbf{U}^{(1)} \mathbf{V}^\top\|_F^2 + \lambda_1 \sum_{k=1}^K \|\mathbf{u}_k^{(1)} - \mathbf{G}_k \mathbf{u}_k^{(1)}\|_2^2 + \frac{\eta}{2} \|\mathbf{u}_k^{(n)}\|_2^2, \quad (6.17)$$

where $\mathbf{V} = \mathbf{U}^{(2)} \otimes \mathbf{U}^{(3)}$ and $\mathbf{G}_k$ is set to $\mathbf{B}_k$ or $\rho_k \mathbf{W}$. Because this is a convex minimization problem, we alternatively update each factor vector $\mathbf{u}_k^{(1)}$ by solving a least square problem:

$$\min_{\mathbf{u}_k^{(1)}} \|\hat{\mathcal{Z}}_1 - \mathbf{u}_k^{(1)} \mathbf{v}_k^\top\|_F^2 + \lambda_1 \|\mathbf{u}_k^{(1)} - \mathbf{G}_k \mathbf{u}_k^{(1)}\|_2^2 + \frac{\eta}{2} \|\mathbf{u}_k^{(n)}\|_2^2, \quad (6.18)$$

where we use $\hat{\mathcal{Z}}_1$ to denote $\mathcal{Z}_1 - \sum_{k' \neq k} \mathbf{u}_{k'}^{(1)} \mathbf{v}_{k'}^\top$. By setting the gradient of the loss function in Eq. (6.18) to zero, we get a linear equation:

$$\mathbf{H}_k \mathbf{u}_k^{(1)} - \hat{\mathcal{Z}}_1 \mathbf{v}_k = 0, \quad (6.19)$$

where $\mathbf{H}_k = (\mathbf{v}_k \mathbf{v}_k^\top + \lambda_1) \mathbf{I}_P - \lambda \mathbf{G}_k$. In this chapter, we employ a conjugate gradient method [88]. For the above equation, the computational complexity of each step of it is $\mathcal{O}(\text{nnz}(\mathbf{H}_k) + P)$, where nnz is the number of non-zero elements in $\mathbf{H}_k$. In our method, since $P \ll \text{nnz}(\mathbf{H}_k)$, the complexity for all of the latent factors approximately equals $\mathcal{O}(\sum_{k=1}^K \text{nnz}(\mathbf{H}_k))$. When we put the non-negative constraint on factor matrices, we obtain a minimizer by applying a projection function to the outputs of the conjugate gradient method:

$$u_{p,k}^{(n)} = \begin{cases} u_{p,k}^{(n)} & (\text{if } u_{p,k}^{(n)} \geq 0), \\ 0 & \text{otherwise} \end{cases}. \quad (6.20)$$

Updating $\mathbf{U}^{(2)}$ and $\mathbf{U}^{(3)}$: The minimization problem for $\mathbf{U}^{(2)}$ can be defined as:

$$\min_{\mathbf{U}^{(2)}} \|\mathbf{Z}_2 - \mathbf{U}^{(2)} \mathbf{V}_2^\top\|_F^2 + \lambda_2 \sum_{k=1}^K \mathbf{u}_k^{(2)\top} \mathbf{L}_k^{(2)} \mathbf{u}_k^{(2)} + \frac{\eta}{2} \|\mathbf{u}_k^{(n)}\|_2^2, \quad (6.21)$$

where $\mathbf{V}_2 = (\mathbf{U}^{(1)} \otimes \mathbf{U}^{(3)})$. Since this problem belongs to a class of graph regularized alternating least squares, we can obtain a minimizer by applying a conjugate gradient descent algorithm [88] that runs in $\mathcal{O}(\text{nnz}(\mathbf{L}_k^{(2)})K)$. Then, we utilize the projection function if we employ the non-negative constraint. We update factor matrix $\mathbf{U}^{(3)}$ in a similar manner.

Algorithm 2: Alternative minimization algorithm for our proposed method

Input : $\mathcal{X}, \Omega, \lambda_1, \lambda_2, K, N, E_p, \mathbf{W}^{(1)}, L, \eta$
Output: $\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}$

- 1 Initialize parameters
- 2 **for** $n = 1$ **to** 3 **do**
- 3 Matricization $\mathcal{X}$ to obtain $\mathcal{X}_n$
- 4 Initialize $\mathbf{U}^{(n)}$ and Θ_n with random values
- 5 **end**
- 6 **repeat**
- 7 Alternatively update parameters;
- 8 Update $\mathcal{Z}$ by $\hat{\mathcal{X}} + P_\Omega(\mathcal{Z} - \hat{\mathcal{X}})$
- 9 Update $\mathbf{U}^{(1)}$ by Eq. (6.19)
- 10 Update $\mathbf{U}^{(2)}$ and $\mathbf{U}^{(3)}$ by Eq. (6.21)
- 11 **for** $n = 1$ **to** 2 **do**
- 12 **if** $\lambda_n > 0$ **then**
- 13 Update Θ_n by Eq. (6.15), (6.16), or [121]
- 14 **end**
- 15 **end**
- 16 **until** *convergence*;;

6.4 Related works

A vector autoregressive model (VAR) is a common choice to make a prediction from a multivariate time-series. When we attempt to analyze a multivariate time-series that consists of P locations and T time stamps, a VAR model with L time order contains $\mathcal{O}(LP^2)$ parameters and its estimation needs $\mathcal{O}(TL^2P^4)$ computational costs. Thus,

VAR models are infeasible to analyze large-size spatio-temporal data [121]. Another popular method for spatio-temporal data analysis is Gaussian processes (GP) [89], which incorporate non-linear spatial and temporal correlations into predictions. For the interpolation problem, the computational complexity of the exact GP amounts to $\mathcal{O}(P^3T^3)$ [14]. An inexact GP using a subset of data points can handle spatio-temporal data with moderate sizes of T and P . However, this method often falls short because of the difficulty of capturing spatio-temporal dependency from a small subset. This method needs an additional procedure so that practitioners carefully choose the kernel functions.

Graph-based regularizers, such as a Laplacian regularizer [96], have been successfully utilized in tensor factorization methods [19, 68, 81, 88]. For example, Cai et al. proposed a tensor factorization method that employed such a regularizer to encourage smooth latent factors with spatial dependencies [21]. To represent the temporal smoothness of factors, this method employed the multi-linear dynamical systems [28, 91, 98]. However, the orders of the temporal dependencies for these methods were restricted to very small values, for example, $L = 1$ [21], thus these methods result in poor performance for long-term prediction problems.

Yu et al. [121] recently proposed a state-of-the-art method called temporal regularized matrix factorization (TRMF), which introduced an autoregressive regularizer into matrix factorization frameworks. By simultaneously learning latent factors and temporal dependencies, TRMF showed considerable improvement on predicting the future observations of known locations. However, all of aforementioned methods including TRMF only utilized a previously inferred spatial weighted matrix to encourage spatial smoothness of the spatial factor matrices.

Bahadori et al. proposed a fast multivariate spatio-temporal analysis method based on tensor regression frameworks [11], in which a VAR model and a graph-based regularizer were combined with tensor factorization methods. By employing higher-order VAR models, this method showed considerable improvement on the predictions of known places and interpolations from existing analysis methods, including the multi-task Gaussian process [14]; however, it only employed a prefixed spatial matrix to represent spatial dependencies.

6.5 Experiments

In this section, we evaluate the performance of our proposed method by predicting and interpolating experiments with five publicly available data sets.

6.5.1 City traffic demand prediction experiments

We employed a taxi transportation record data set provided by the NYC Taxi and Limousine Commission¹ and two bike-sharing system data sets [123] recorded in New York² and Washington D. C.³. With the taxi data set, we conducted a task that predicted the number of customers dropped off by taxis in NY. Since, the records included in this data set consists of GPS locations, we constructed grids whose areas correspond to 500m². The problem for the bike-sharing system data sets were to predict the number of customers who returned bicycles to bike stations. The number of observing locations for the taxi data, bike-sharing system data in D.C. and NY were 640, 351, and 344, respectively. Observations were sent every 30 minutes.

We used the records for the two weeks in each data set. We picked each of past seven days as a test day and the seven days before the test day as training data. Then we randomly selected 10% of the locations as unknown test locations and used the remaining locations as known training locations. We used our proposed method with the DAR regularizer (DAR), the DAR regularizer with the non-negative constraints (DAR+NN), the SAR regularizer (SAR), and the SAR regularizer with the non-negative constraints (SAR+NN). We compared our proposed method with Gaussian processes (GP) [89], the fast multivariate spatio-temporal analysis method (FAST) [11], and temporal regularized matrix factorization (TRMF) [121]. Because FAST is not able to make predictions of unknown locations, we first interpolated observations of unknown locations and then made predictions.

For the spatial regularizer $\mathbf{W}^{(1)}$ of our proposed method, we employed the k -nearest neighbor graph whose weight was set to a Gaussian similarity measure $w_{p,p'} = \exp(-\|\mathbf{r}_p - \mathbf{r}_{p'}\|_2^2 / \sigma^2)$ [82], where $\mathbf{r}_p$ is a GPS location for the p -th sensor and σ is a scaling parameter, and then normalized each row. Since the difference of GPS locations are relatively small, we set $\sigma = 100$. For FAST and TRMF, we utilized

¹http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

²<http://www.citibikenyc.com/>

³<https://www.capitalbikeshare.com>

the fully connected graph laplacian matrix whose elements were set to a Gaussian similarity measure. For our proposed method with the DAR regularizer, we set the number of angle divisions N to 4 or 8 and the number of nearest neighbors K of locations with the Euclid distance to 4 or 8. HYper parameters λ_1 and λ_2 were set to $\{0, 0.1, 1.0, 10.0\}$. We tuned the hyperparameters by five-fold cross validation. For GP, we used 5,000 randomly sampled data points for training and optimized the hyperparameters by the maximum likelihood estimation procedure. We used publicly available matlab codes for the GP, FAST, TRMF, and tensor calculus [10].

The performances of each method were assessed on three different types of predictive tasks. The first task predicted *future* observations on *unknown* locations ($\mathcal{X}_3$). The second predicted *future* observations on *already known* locations ($\mathcal{X}_2$). The last interpolated observations on *unknown* locations ($\mathcal{X}_1$). Based on previous work [121], we utilized the normalized deviation (ND) and the normalized RMSE (NRMSE) as error measurements: $ND = (1/|\Omega|) \sum_{(p,t,m) \in \Omega} |x_{p,t,m} - \hat{x}_{p,t,m}|/Q$, $NRMSE = ((1/|\Omega|) \sum_{(p,t,m) \in \Omega} (x_{p,t,m} - \hat{x}_{p,t,m})^2)^{1/2}/Q$, where $Q = (1/|\Omega|) \sum_{(p,t,m) \in \Omega} |x_{p,t,m}|$. The results are shown in Tables 6.1, 6.2, and 6.3. The left and right side values in each cell correspond to the average and standard deviation of the ND and NRMSE obtained from seven runs on different test days. The boldface numbers indicates the minimum average errors among all the methods.

From these results, we confirmed that our proposed method with the DAR regularizers outperformed other methods on the task that predicted the feature observations and the interpolations of unknown locations for the whole data sets. Our proposed method with both the DAR regularizer and the non-negative constraints showed better performances without the non-negative constraints because this constrain can avoid overfitting. As with ND for the first task, the improvements of our DAR regularizers from the comparisons were 42.9%, 9.8%, and 18.3% for the NY taxi data, the bike-sharing system on NY data, and that on Washington D.C. data, respectively. Similarly, the improvements as with NRMSE were 26.9%, 3.1%, and 13.4%. The improvements of our proposed method on the first task were caused by reducing the predictive errors on the task of interpolating the observations of unknown locations, because our proposed method can learn different spatial dependencies for each latent factor by utilizing our spatial autoregressive regularizers. However, the existing methods cannot simultaneously learn spatial and temporal dependencies.

FAST showed the best performances on the task of predicting the future observations of known locations with the experiments of the bike-sharing system data sets. Our proposed method with the SAR regularizer showed superior performances to the existing methods, however its performance was poor than that with the DAR regularizer. The reason is that the SAR model only had a scalar parameter and cannot capture complex dependencies among latent factors.

Table 6.1: Results of NY taxi data

Method	Pred. on unknown		Interpolation		Pred. on known	
	ND	NRMSE	ND	NRMSE	ND	NRMSE
Prop. (DAR)	0.569 (0.079)	1.399 (0.244)	0.509 (0.002)	1.197 (0.005)	0.317 (0.107)	0.960 (0.384)
Prop. (DAR+NN)	0.568 (0.091)	1.375 (0.224)	0.499 (0.001)	1.194 (0.024)	0.312 (0.108)	0.971 (0.390)
Prop. (SAR)	0.746 (0.063)	1.874 (0.221)	0.740 (0.005)	1.801 (0.012)	0.340 (0.135)	0.979 (0.410)
Prop. (SAR+NN)	0.778 (0.019)	1.938 (0.098)	0.774 (0.004)	1.935 (0.022)	0.316 (0.102)	0.983 (0.393)
TRMF	0.996 (0.041)	1.859 (0.148)	0.997 (0.001)	1.929 (0.017)	0.350 (0.128)	1.129 (0.423)
FAST	1.095 (0.010)	2.131 (0.127)	1.056 (0.002)	2.059 (0.014)	0.653 (0.005)	1.359 (0.043)
GP	1.256 (0.056)	2.257 (0.104)	1.095 (0.060)	2.153 (0.096)	1.683 (0.188)	2.812 (0.114)

Table 6.2: Results of bike-sharing system data in NY

Method	Pred. on unknown		Interpolation		Pred. on known	
	ND	NRMSE	ND	NRMSE	ND	NRMSE
Prop. (DAR)	0.843 (0.116)	1.514 (0.346)	0.677 (0.021)	1.289 (0.087)	0.760 (0.127)	1.244 (0.211)
Prop. (DAR+NN)	0.821 (0.089)	1.510 (0.333)	0.678 (0.018)	1.294 (0.082)	0.818 (0.295)	1.231 (0.215)
Prop. (SAR)	0.863 (0.082)	1.607 (0.271)	0.766 (0.036)	1.447 (0.076)	0.721 (0.098)	1.215 (0.217)
Prop. (SAR+NN)	0.902 (0.133)	1.646 (0.272)	0.757 (0.018)	1.468 (0.086)	0.790 (0.240)	1.224 (0.206)
TRMF	1.065 (0.210)	1.794 (0.268)	0.845 (0.032)	1.525 (0.060)	0.744 (0.101)	1.225 (0.180)
FAST	0.910 (0.055)	1.571 (0.281)	0.818 (0.015)	1.460 (0.066)	0.710 (0.058)	1.115 (0.101)
GP	3.589 (5.801)	5.599 (9.229)	0.743 (0.065)	1.355 (0.117)	3.701 (6.139)	5.705 (9.835)

Table 6.3: Results of bike-sharing system data in Washington D.C.

Method	Pred. on unknown		Interpolation		Pred. on known	
	ND	NRMSE	ND	NRMSE	ND	NRMSE
Prop. (DAR)	1.159 (0.231)	2.510 (0.341)	1.014 (0.021)	2.251 (0.059)	1.018 (0.278)	1.972 (0.444)
Prop. (DAR+NN)	1.176 (0.241)	2.502 (0.366)	1.025 (0.020)	2.231 (0.045)	0.969 (0.204)	1.916 (0.357)
Prop. (SAR)	1.882 (0.461)	2.905 (0.443)	1.780 (0.060)	2.743 (0.085)	1.018 (0.276)	1.978 (0.435)
Prop. (SAR+NN)	2.202 (0.742)	3.210 (0.792)	2.056 (0.105)	3.017 (0.135)	0.969 (0.205)	1.917 (0.356)
TRMF	1.419 (0.287)	3.045 (0.335)	1.395 (0.057)	2.709 (0.059)	0.977 (0.164)	1.999 (0.473)
FAST	1.640 (0.139)	2.891 (0.223)	1.557 (0.026)	2.804 (0.047)	0.942 (0.054)	1.811 (0.201)
GP	1.959 (0.394)	3.401 (0.463)	1.589 (0.032)	2.755 (0.047)	1.599 (0.253)	2.807 (0.299)

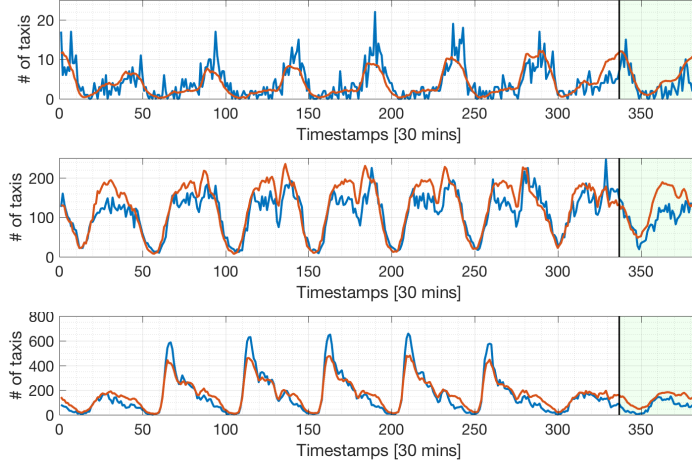
To check the qualitative performances of our proposed method, we illustrate the estimated predictions of the DAR regularizer with the NY taxi data set in Fig. 6.3 and the bike-sharing system in NY in Fig. 6.4. The red and blue lines are the predicted values and the ground truth, respectively. The solid black line indicates the end of the training period. From these figures, we confirmed that our proposed method accurately predicted observations of known locations by capturing the temporal dynamics. It also performed well on the predictions for unknown locations. These results support the effectiveness of learning both spatial and temporal dependencies by our proposed DAR regularizers.

Furthermore, we illustrated the extracted temporal and spatial factors with the NY taxi data and the bike-sharing system data in NY in Figs. 6.5 and 6.6. We also showed temporal factors extracted from without the non-negative constraint by gray lines in these figures. From Figs. 6.5a and 6.6a, we confirmed that different dynamics were extracted in each temporal factor and that the spatial factors with the non-negative constraints obtained smooth and easy to interpretable shapes than without that constraints. From the blue temporal factor and its corresponding spatial factor 6.5b, we can easily understand that this factor appears in every afternoon and near to Empire State Building. The red temporal factor and its corresponding spatial factor 6.5c indicate a daily transportation occurring in the night on both Manhattan and Brooklyn. From Fig. 6.6, we also confirmed that our proposed method successfully extracted smooth spatial patterns from non-grid locations.

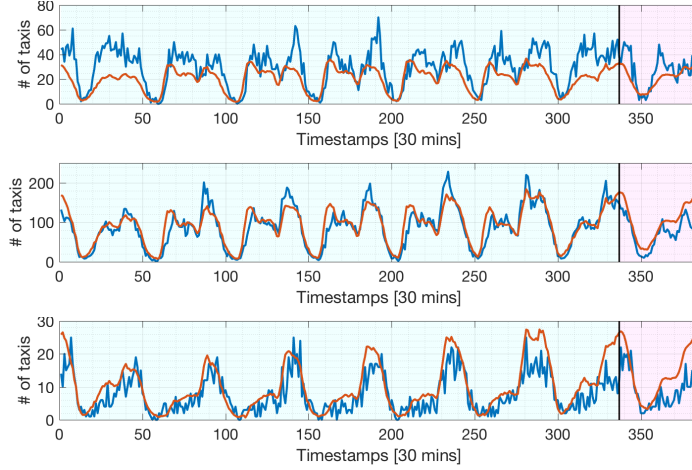
6.5.2 Air quality prediction experiments

We conducted air quality predicting experiments with air quality and meteorology data set provided by Zheng et al. [125]. We employed five types of air quality observations from this data set including PM25, PM10, NO2, temperature, and humidity, all of which were observed in Beijing and Shanghai, China. The number of sensor locations of Beijing and Shanghai were 32 and 11, respectively. Each observation were sent in every hour.

We constructed three-way tensors whose modes corresponded to locations, time stamps, and types of sensors. Seven consecutive days were used for the training data, and the following day was used for the test data. Because the numbers of locations in these data sets were not abundant, we randomly picked two locations for the test



(a) Predictions of known locations

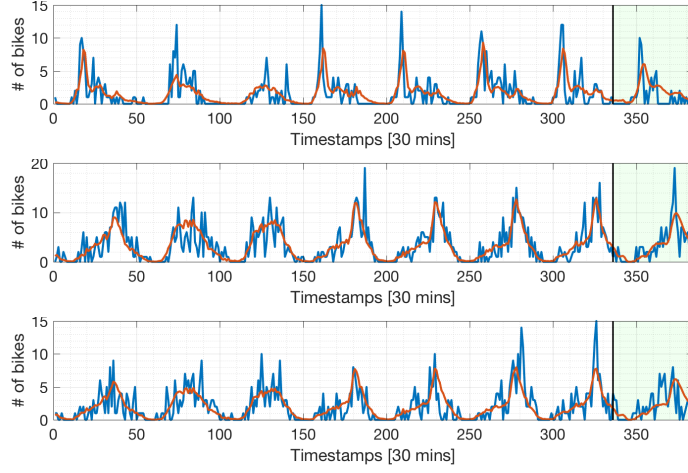


(b) Predictions and interpolations of unknown locations

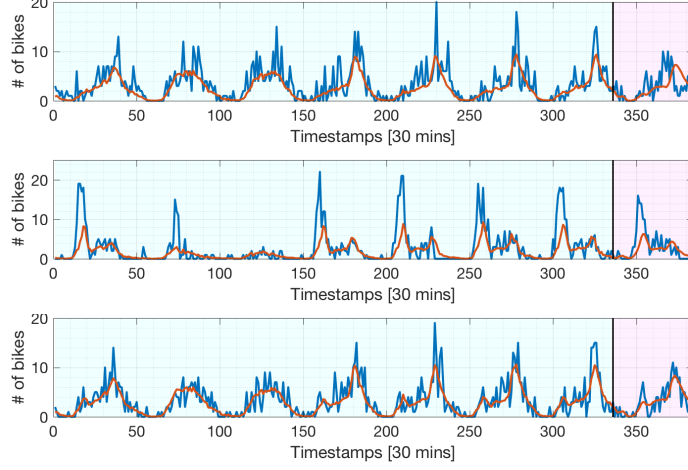
Figure 6.3: Examples of results of NY taxi data (backgrounds with green, blue, and red correspond to $\mathcal{X}_2$, $\mathcal{X}_1$, and $\mathcal{X}_3$, respectively)

data for each period. Unlike the previous three data sets, these data sets contain observation failures due to sensor troubles and network corruptions. Thus, we needed to predict observations by treating the missing values. We compared our proposed method with the multi-task Gaussian process [14] and the tensor extended TRMF. FAST was omitted from the result because it was not applicable to deal with missing values. Hyperparameters were selected by the same way of the previous experiments.

The results are shown in Tables 6.4 and 6.5. We confirmed that our proposed method showed the best performance Prediction problems in these experiments. Since the number of locations were smaller than the previous experiment, our pro-



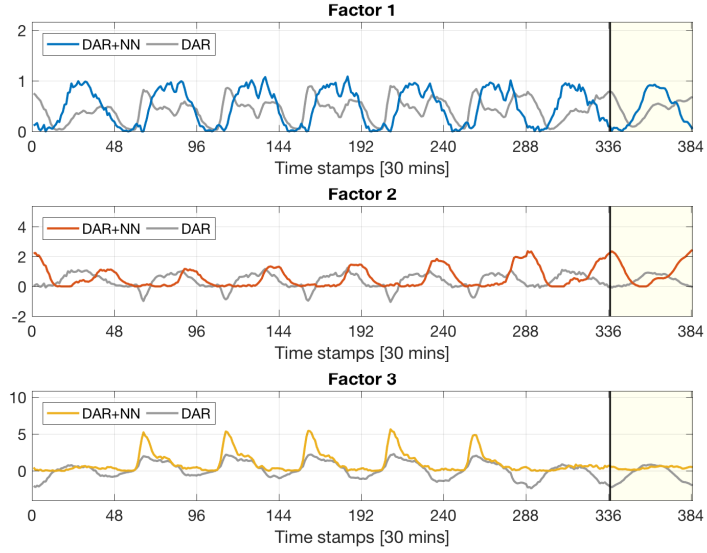
(a) Predictions of known locations



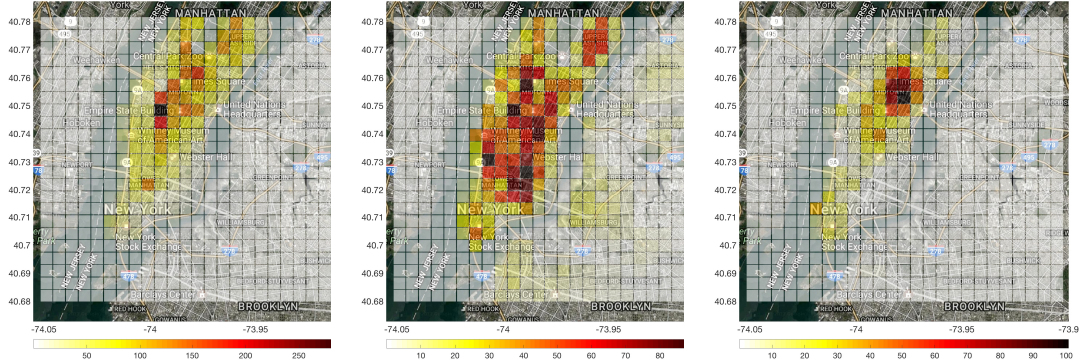
(b) Predictions and interpolations of unknown locations

Figure 6.4: Examples of results of bike-sharing system data in NY (backgrounds with green, blue, and red correspond to $\mathcal{X}_2$, $\mathcal{X}_1$, and $\mathcal{X}_3$, respectively)

posed method with the SAR regularizer and GP showed the best performance on predictions on known location for Beijing data and interpolations for Shanghai data, respectively. Examples of predictions and interpolations are shown in Fig. 6.7. On both the tasks of prediction on a known location and an unknown location, our method performed well, but its predictions on temperature were worse than the other data sources, because the dynamics of temperature were different from the others. These results indicated that our proposed method was effective for analyzing tensor data including multiple types of sensors.

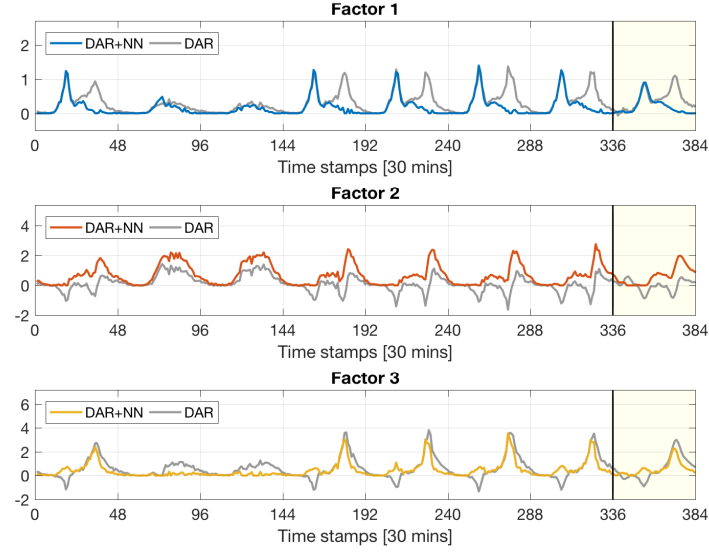


(a) Temporal factors (the yellow background corresponds to the test period)

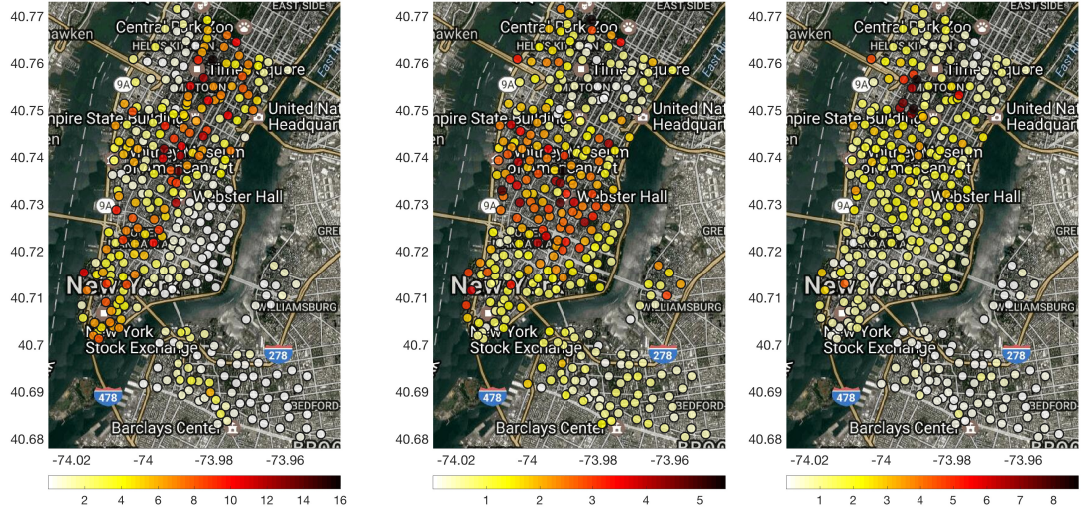


(b) Spatial factor 1 (blue) (c) Spatial factor 2 (red) (d) Spatial factor 3 (yellow)

Figure 6.5: Temporal and spatial factors extracted with our proposed method (DAR+NN) from NY taxi data



(a) Temporal factors (the yellow background corresponds to the test period)



(b) Spatial factor 1 (blue) (c) Spatial factor 2 (red) (d) Spatial factor 3 (yellow)

Figure 6.6: Temporal and spatial factors extracted with our proposed method (DAR+NN) from bike-sharing system data in NY

Table 6.4: Results of air quality data in Beijing

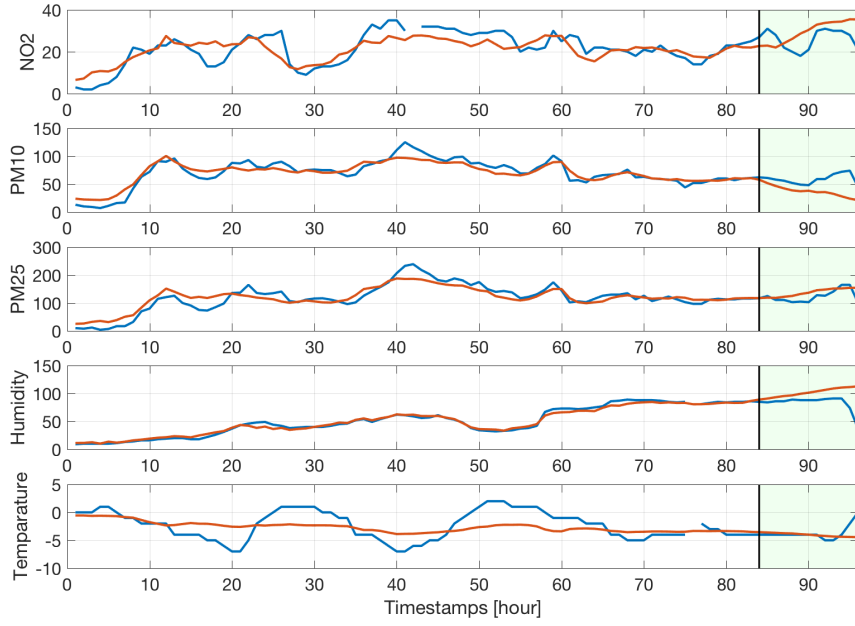
Method	Pred. on unknown		Interpolation		Pred. on known	
	ND	NRMSE	ND	NRMSE	ND	NRMSE
Prop. (DAR+sim)	0.246 (0.084)	0.386 (0.156)	0.229 (0.048)	0.385 (0.094)	0.382 (0.417)	0.563 (0.585)
Prop. (DAR+NN)	0.297 (0.201)	0.489 (0.429)	0.232 (0.067)	0.382 (0.131)	0.276 (0.192)	0.451 (0.410)
Prop. (SAR)	0.616 (0.046)	0.960 (0.083)	0.634 (0.036)	1.162 (0.146)	0.259 (0.104)	0.401 (0.167)
Prop. (SAR+NN)	0.501 (0.238)	0.807 (0.414)	0.519 (0.238)	0.910 (0.443)	0.169 (0.089)	0.249 (0.134)
TRTF	0.909 (0.790)	1.255 (1.025)	1.057 (0.875)	1.732 (1.379)	0.292 (0.184)	0.450 (0.299)
GP	13.280 (4.570)	16.926 (5.774)	0.672 (0.051)	0.991 (0.217)	4.408 (1.034)	5.732 (1.248)

Table 6.5: Results of air quality data in Shanghai

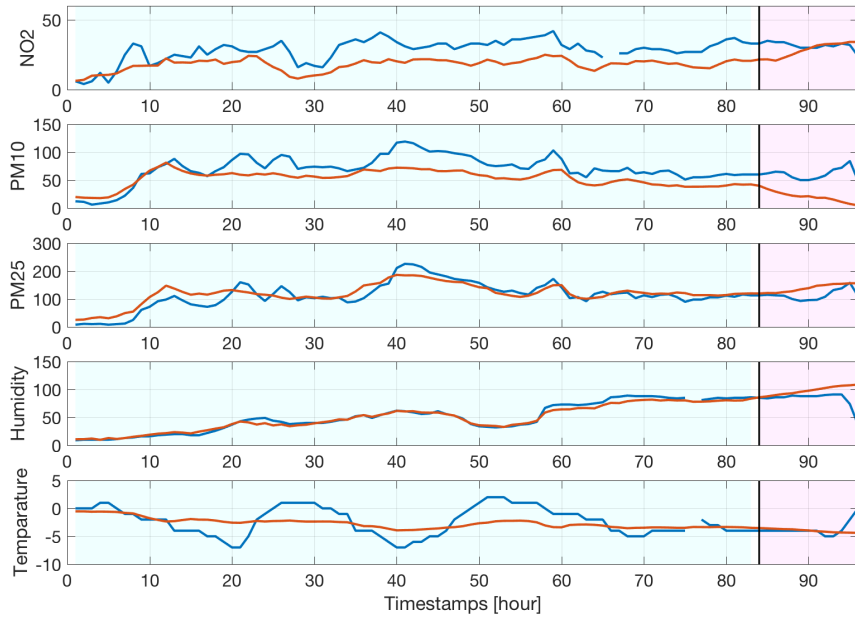
Method	Pred. on unknown		Interpolation		Pred. on known	
	ND	NRMSE	ND	NRMSE	ND	NRMSE
Prop. (DAR)	0.957 (0.133)	1.417 (0.212)	1.063 (0.075)	1.951 (0.253)	0.281 (0.108)	0.434 (0.161)
Prop. (DAR+NN)	0.682 (0.316)	1.094 (0.563)	0.789 (0.370)	1.405 (0.692)	0.227 (0.210)	0.370 (0.421)
Prop. (SAR)	0.861 (0.187)	1.474 (0.204)	0.880 (0.002)	2.070 (0.197)	0.257 (0.106)	0.395 (0.169)
Prop. (SAR+NN)	0.761 (0.087)	1.397 (0.356)	0.834 (0.074)	1.879 (0.355)	0.254 (0.147)	0.378 (0.229)
TRTF	1.560 (2.249)	2.096 (2.902)	2.110 (2.122)	3.459 (3.608)	0.311 (0.239)	0.480 (0.383)
GP	19.433 (10.973)	25.135 (14.503)	0.711 (0.157)	1.007 (0.381)	6.075 (2.669)	7.967 (3.550)

6.6 Summary

We proposed a tensor factorization method that simultaneously learns data-driven spatial and temporal autocorrelations from spatio-temporal data and incorporates these dependencies as regularizers for estimating latent factors. With experiments of traffic transportation data and air quality monitoring data, we demonstrated that our proposed methods significantly improved the performance of predicting feature observations of unknown locations and interpolating the observations. We believe that our result is beneficial for many applications that demand previous predictions of spatial and temporal observations with large-scale data, such as demand forecasting for sharing economy services.



(a) Predictions of a known location



(b) Predictions and interpolations of an unknown location

Figure 6.7: Examples of results for air quality data in Beijing (backgrounds with green, blue, and red correspond to $\mathcal{X}_2$, $\mathcal{X}_1$, and $\mathcal{X}_3$, respectively)

Chapter 7

Conclusion

In this dissertation, we considered a problem of extracting interpretable patterns from highly sparse and noisy relational data with rising three fundamental questions:

1. How can we utilize auxiliary relational database as an auxiliary matrix/tensor to increase the available information for extracting latent factors?
2. How can we employ the structure information of attributes, such as graph and groups of attributes for extracting smooth latent factors from a few records?
3. Is it possible to learn factor-specific dependencies and employ it for extracting reasonable factors?

We developed three approaches based on Non-negative Matrix and Tensor Factorization methods. Our findings from these approaches can be summarized as follows:

1. To overcome the sparseness of the original data, complementary data can be regarded as useful information sources by employing multiple tensor factorization procedures. The extracted factors resulted in interpretable within and between data patterns.
2. Incorporating both the graph Laplacian and structured regularizers helps us extract smooth and flat latent factors. This approach showed better performances for interpolation problems with spatio-temporal data.

3. We can learn data-driven spatial and temporal autocorrelations from spatio-temporal data and incorporate these dependencies as regularizers. This approach is advantageous for spatio-temporal prediction problems.

For the first approach, we proposed a matrix factorization technique called Non-negative Multiple Matrix Factorization (NM²F) and its tensor extension Non-negative Multiple Tensor Factorization (NMTF). Our key idea was to integrate additional matrices and tensors for estimating the latent factors. We adopted the generalized Kullback-Leibler divergence as the distance metric between tensors and latent factors. We derived a method for parameter estimation based on the multiplicative update rule. We showed a connection between NMF by providing that NM²F is a special case of NMF with undefined regions. We evaluated NM²F experimentally using a synthetic dataset and two real datasets, and confirmed that NM²F performs better than the existing factorization methods in terms of a quantitative measure. We also confirmed that the estimated bases of NM²F are reasonably intuitive and easy to understand. To enable NM²F to deal with higher-order arrays, we formulated NMTF as a generalization of NTF, and showed that NMTF includes NTF as a special choice of scaling parameters. The performance of NMTF was quantitatively better than that for NTF on experiments with both synthetic and real-world data sets. We also confirmed that NMTF successfully extracted informative and understandable factors from multiple tensors, which cannot be done by NM²F.

For the second approach, we proposed Structurally Regularized Non-negative Tensor Factorization, which incorporated both the graph Laplacian and structured regularizers for the latent factors. For the structured regularizer, we employed the generalized fused lasso and the higher-order fused lasso to represent both graph-based and group-based information in time and space. With this regularizer, we can encourage latent factors to be temporary and spatially dependent. This constraint helps in learning smooth and flat patterns from highly sparse spatio-temporal data. By performing missing value imputation experiments with three real-world data sets, we confirmed that our proposed method showed the best quantitative performance and successfully extracted more interpretable latent factors than the existing methods. We also checked the interpretability of the extracted patterns qualitatively.

Finally, we proposed Autoregressive Tensor Factorization, which simultaneously learns data-driven spatial and temporal autocorrelations from spatio-temporal data

and incorporates these dependencies as regularizers for estimating the latent factors. By learning factor independent autocorrelations, our method can flexibly extract spatial patterns and their underlying dynamics. Experiments with traffic transportation data and air quality monitoring data showed that our proposed methods significantly improved the performance of predicting feature observations of unknown locations and interpolating the observations.

7.1 Future Work

As a future work, we would like to provide some directions for pattern extraction methods based on tensor factorization methods.

First, all of the methods considered in this dissertation utilized the sum-product of latent factors as an estimated value of tensors. Our methods showed better performances in the experiments by utilizing simultaneous factorization procedure and various regularizers into the linear formation of estimated values. However, in real-world data sets, there exists a possibility that latent patterns have more complex interactions that a linear model cannot represent. To deal with such complex interactions, the extensions of our methods with a non-linear function should be deal candidates for increasing the representativity of the tensor factorization methods and improving its performance for the real-world data applications. However, once we employ a non-linear function in tensor factorizations, it is not clear whether simple non-negative constraints remain the interpretability of factors as the existing methods. Therefore, obtaining a balance between the non-linearity and interpretability of factorization methods would become another problem.

With the rise of big data and deep learning, relational data including spatio-temporal domains have become denser than during the last ten years, and non-linear low-rank approximation methods, including autoencoders, have received considerable attention. However, similar to our autoregressive tensor factorization, data-driven approaches that simultaneously learns latent representations and their dependencies have not been established. There would be a significant potential to improve the performance of spatio-temporal data analysis by leveraging both the power of data sizes and model flexibility if we can smoothly incorporate our knowledge on spatio-temporal domains into such models.

List of Publications

1. Koh Takeuchi, Katsuhiko Ishiguro, Akisato Kimura, and Hiroshi Sawada. Non-negative Multiple Matrix Factorization. In *Proceedings of IJCAI*, 1713–1720, 2013.
2. Koh Takeuchi, Ryota Tomioka, Katsuhiko Ishiguro, Akisato Kimura, and Hiroshi Sawada. Non-negative Multiple Tensor Factorization. In *Proceedings of ICDM*, 1199–1204, 2013.
3. Koh Takeuchi, Yoshinobu Kawahara, and Tomoharu Iwata. Higher order fused regularization for supervised learning with grouped parameters. *Machine Learning and Knowledge Discovery in Databases, Lecture Notes in Computer Science series (LNCS)*, 577–593, 2015.
4. Koh Takeuchi, Yoshinobu Kawahara, and Tomoharu Iwata. Structurally Regularized Non-negative Tensor Factorization for Spatio-temporal Pattern Discovery. *Machine Learning and Knowledge Discovery in Databases, Lecture Notes in Computer Science series (LNCS)*, 582–598, 2017.
5. Koh Takeuchi, Hisashi Kashima, and Naonori Ueda. Autoregressive Tensor Factorization for Spatio-temporal Predictions. In *Proceedings of ICDM*, 1105–1110, 2017.

6. 竹内 孝, 石黒勝彦, 木村昭悟, 澤田宏. “非負制約下における複合行列分解とそのソーシャルメディア解析への応用”. 情報処理学会論文誌:数理モデル化と応用(*TOM*), 7(1), 71-83, 2014.

Bibliography

- [1] S. Abdallah and M. Plumbley. Polyphonic music transcription by non-negative sparse coding of power spectra. In *Proceedings of ISMIR*, 2004.
- [2] M. Aharon, M. Elad, and A. Bruckstein. On the uniqueness of overcomplete dictionaries, and a practical way to retrieve them. *Linear algebra and its applications*, 416(1):48–67, 2006.
- [3] M. I. Ali, F. Gao, and A. Mileo. Citybench: A configurable benchmark to evaluate rsp engines using smart city datasets. In *Proceedings of ISWC*, 2015.
- [4] L. Anselin. Under the hood issues in the specification and interpretation of spatial regression models. *Agricultural economics*, 27(3):247–267, 2002.
- [5] L. Anselin. Thirty years of spatial econometrics. *Papers in regional science*, 89(1):3–25, 2010.
- [6] L. Anselin. *Spatial econometrics: methods and models*, volume 4. Springer Science & Business Media, 2013.
- [7] L. Anselin and A. K. Bera. Spatial dependence in linear regression models with an introduction to spatial econometrics. *Statistics Textbooks and Monographs*, 155:237–290, 1998.
- [8] J. Antikainen, J. Havel, R. Josth, A. Herout, P. Zemcik, and M. Hauta-Kasari. Nonnegative Tensor Factorization Accelerated Using GPGPU. *IEEE Trans.*, 22(7):1135–1141, 2011.
- [9] F. R. Bach. Structured sparsity-inducing norms through submodular functions. In *Proceedings of NIPS*, 2010.

- [10] B. W. Bader, T. G. Kolda, et al. MATLAB Tensor Toolbox Version 2.6. Available online, February 2015.
- [11] M. T. Bahadori, Q. R. Yu, and Y. Liu. Fast Multivariate Spatio-temporal Analysis via Low Rank Tensor Learning. In *Proceedings of NIPS*. 2014.
- [12] A. Barbero and S. Sra. Fast Newton-type methods for total variation regularization. In *Proceedings of ICML*, 2011.
- [13] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [14] E. V. Bonilla, K. M. Chai, and C. Williams. Multi-task Gaussian Process Prediction. In *Proceedings of NIPS*, 2008.
- [15] G. Bouchard, S. Guo, and D. Yin. Convex Collective Matrix Factorization. In *Proceedings of AISTATS*, 2013.
- [16] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.
- [17] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.
- [18] D. Cai, X. He, J. Han, and T. S. Huang. Graph Regularized Nonnegative Matrix Factorization for Data Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1548–1560, 2011.
- [19] D. Cai, X. He, J. Han, and T. S. Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1548–1560, 2011.
- [20] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.
- [21] Y. Cai, H. Tong, W. Fan, P. Ji, and Q. He. Facets: Fast comprehensive mining of coevolving high-order time series. In *Proceedings of SIGKDD*, 2015.

- [22] B. Cao, D. Shen, J. Sun, X. Wang, Q. Yang, and Z. Chen. Detect and track latent factors with online nonnegative matrix factorization. In *Proceedings of IJCAI*, 2007.
- [23] J. D. 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.
- [24] A. Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009.
- [25] A. Chambolle and J. Darbon. On total variation minimization and surface evolution using parametric maximum flows. *International journal of computer vision*, 84(3):288, 2009.
- [26] E. C. Chi and T. G. Kolda. On tensors, sparsity, and nonnegative factorizations. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1272–1299, 2012.
- [27] W. Chu and Z. Ghahramani. Probabilistic models for incomplete multidimensional arrays. In *Proceedings of AISTATS*, 2009.
- [28] F. C. T. Chua, R. J. Oentaryo, and E. P. Lim. Modeling Temporal Adoptions Using Dynamic Matrix Factorization. In *Proceedings of ICDM*, 2013.
- [29] A. Cichocki, R. Zdunek, and S. Amari. Hierarchical ALS Algorithms for Non-negative Matrix and 3D Tensor Factorization. In *Proceedings of ICA*, 2007.
- [30] A. Cichocki, A. H. P. R. Zdunek, and S. Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis*, John Wiley. Wiley, 2009.
- [31] A. Cichocki, R. Zdunek, and S. Amari. Hierarchical ALS algorithms for non-negative matrix and 3D tensor factorization. In *Proceedings of ICA*, 2007.
- [32] A. Cichocki, R. Zdunek, S. Choi, R. Plemmons, and S. Amari. Novel multi-layer non-negative tensor factorization with sparsity constraints. In *Proceedings of ICANNGA*, 2007.

- [33] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.
- [34] E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.
- [35] P. J. Curran and P. M. Atkinson. Geostatistics and remote sensing. *Progress in Physical Geography*, 22(1):61–78, 1998.
- [36] I. S. Dhillon and S. Sra. Generalized Nonnegative Matrix Approximations with Bregman Divergences. In *Proceedings of NIPS*, 2005.
- [37] O. Dikmen and C. Fevotte. Maximum Marginal Likelihood Estimation for Nonnegative Dictionary Learning in the Gamma-Poisson Model. *IEEE Transactions on Signal Processing*, 60(10):5163–5175, 2012.
- [38] C. Ding, T. Li, and W. Peng. NMF and PLSI: Equivalence and A Hybrid Algorithm. In *Proceedings of SIGIR*, 2006.
- [39] S. R. Dubois and F. H. Glanz. An autoregressive model approach to two-dimensional shape classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1):55–66, 1986.
- [40] K. Duh, T. Hirao, A. Kimura, K. Ishiguro, T. Iwata, and C. A. Yeung. Creating Stories: Social Curation of Twitter Messages. In *Proceedings of ICWSM*, 2012.
- [41] J. B. Elsner, R. E. Hodges, and T. H. Jagger. Spatial grids for hurricane climate research. *Climate dynamics*, 39(1-2):21–36, 2012.
- [42] Z. Fan, X. Song, and R. Shibasaki. CitySpectrum: a non-negative tensor factorization approach. In *Proceedings of UbiComp*, 2014.
- [43] C. Févotte and A. Ozerov. Notes on nonnegative tensor factorization of the spectrogram for audio source separation: statistical insights and towards self-clustering of the spatial cues. In *Proceedings of CMMR*, 2010.
- [44] D. FitzGerald, M. Cranitch, and E. Coyle. Sound source separation using shifted non-negative tensor factorisation. In *Proceedings of ICASSP*, 2006.

- [45] S. Fujishige. *Submodular functions and optimization*, volume 58. Elsevier, 2005.
- [46] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1):30–55, 1989.
- [47] A. Getis. A history of the concept of spatial autocorrelation: A geographer’s perspective. *Geographical Analysis*, 40(3):297–309, 2008.
- [48] N. Gillis. The why and how of nonnegative matrix factorization. In *Regularization, Optimization, Kernels, and Support Vector Machines*, pages 257–291. Chapman and Hall/CRC, 2014.
- [49] S. K. Gupta, D. Phung, B. Adams, T. Tran, and S. Venkatesh. Nonnegative shared subspace learning and its application to social media retrieval. In *Proceedings of SIGKDD*, 2010.
- [50] Y. Han and F. Moutarde. Analysis of Large-Scale Traffic Dynamics in an Urban Transportation Network Using Non-Negative Tensor Factorization. *International Journal of Intelligent Transportation Systems Research*, 14(1):36–49, 2016.
- [51] R. A. Harshman. Foundations of the PARAFAC procedure: models and conditions for an ”explanatory” multimodal factor analysis. *UCLA Working Papers in Phonetics*, 1970.
- [52] T. Hazan, S. Polak, and A. Shashua. Sparse image coding using a 3D non-negative tensor factorization. In *Proceedings of ICCV*, 2005.
- [53] M. Heiler and C. Schnörr. Controlling sparseness in non-negative tensor factorization. In *Proceedings of ECCV*, 2006.
- [54] M. D. Hoffman, D. M. Blei, and P. R. Cook. Bayesian Nonparametric Matrix Factorization for Recorded Music. In *Proceedings of ICML*, 2011.
- [55] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of SIGIR*, 1999.

- [56] P. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.
- [57] K. Huang, N. D. Sidiropoulos, and A. P. Liavas. A flexible and efficient algorithmic framework for constrained matrix and tensor factorization. *IEEE Transactions on Signal Processing*, 64(19):5052–5065, 2016.
- [58] K. Ishiguro, A. Kimura, and K. Takeuchi. Towards Automatic Image Understanding and Mining via Social Curation. In *Proceedings of ICDM*, 2012.
- [59] R. Jenatton, N. Le Roux, A. Bordes, G. Obozinski, et al. A latent factor model for highly multi-relational data. In *Proceedings of NIPS*, 2012.
- [60] T. Kimura, K. Ishibashi, T. Mori, H. Sawada, T. Toyono, K. Nishimatsu, A. Watanabe, A. Shimoda, and K. Shiimoto. Spatio-temporal factorization of log data for understanding network events. In *Proceedings of INFOCOM*, 2014.
- [61] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.
- [62] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE*, 2009.
- [63] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, October 1999.
- [64] D. D. Lee, D. D., and S. H. Sebastian. Algorithms for Non-negative Matrix Factorization. In *Proceedings of NIPS*, 2000.
- [65] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.
- [66] H. Lee, Y.-D. Kim, A. Cichocki, and S. Choi. Nonnegative tensor factorization for continuous EEG classification. *International Journal of Neural Systems*, 17(04):305–317, 2007.

- [67] L. Li, J. McCann, N. S. Pollard, and C. Faloutsos. Dynammo: Mining and summarization of coevolving sequences with missing values. In *Proceedings of SIGKDD*, 2009.
- [68] W.-J. Li and D.-Y. Yeung. Relation Regularized Matrix Factorization. In *Proceedings of IJCAI*, 2009.
- [69] J. Lin, R. Snow, and W. Morgan. Smoothing techniques for adaptive online language models: topic tracking in tweet streams. In *Proceedings of SIGKDD*, 2011.
- [70] C. Liu, H. Yang, J. Fan, L. He, and Y. Wang. Distributed Nonnegative Matrix Factorization for Web-Scale Dyadic Data Analysis on MapReduce. In *Proceedings of WWW*, 2010.
- [71] H. Liu, Z. Yang, Z. Wu, and X. Li. A-Optimal Non-negative Projection for Image Representation. In *Proceedings of CVPR*, 2012.
- [72] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. In *Proceedings of ICCV*, 2009.
- [73] L. Lovász. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pages 235–257. Springer, 1983.
- [74] H. Ma, H. Yang, M. R. Lyu, and I. King. SoRec: Social Recommendation Using Probabilistic Matrix Factorization. In *Proceedings of CIKM*, 2008.
- [75] G. Matheron. Principles of geostatistics. *Economic geology*, 58(8):1246–1266, 1963.
- [76] J. McAuley and J. Leskovec. From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise through Online Reviews. In *Proceedings of WWW*, 2013.
- [77] J. McAuley, J. Leskovec, and D. Jurafsky. Learning attitudes and attributes from multi-aspect reviews. In *Proceedings of ICDM*, 2012.
- [78] Q. Mei, D. Cai, D. Zhang, and C. Zhai. Topic Modeling with Network Regularization. In *Proceedings of WWW*, 2008.

- [79] K. Nagano, Y. Kawahara, and K. Aihara. Size-constrained submodular minimization through minimum norm base. In *Proceedings of ICML*, 2011.
- [80] M. Nakano, J. L. Roux, H. Kameoka, T. Nakamura, N. Ono, and S. Sagayama. Bayesian nonparametric spectrogram modeling based on infinite factorial infinite hidden Markov model. In *Proceedings of WASPAA*, 2011.
- [81] A. Narita, K. Hayashi, R. Tomioka, and H. Kashima. Tensor factorization using auxiliary information. In *Proceedings of ECMLPKDD*. Springer, 2011.
- [82] A. Y. Ng, M. I. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an algorithm. In *Proceedings of NIPS*. 2002.
- [83] M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of ICML*, 2011.
- [84] M. Nickel, V. Tresp, and H.-P. Kriegel. Factorizing YAGO: scalable machine learning for linked data. In *Proceedings of WWW*, 2012.
- [85] J. Noel, S. Sanner, K. Tran, P. Christen, L. Xie, E. V. Bonilla, E. Abbasnejad, and N. D. Penna. New Objective Functions for Social Collaborative Filtering. In *Proceedings of WWW*, 2012.
- [86] J. Pang and G. Cheung. Graph Laplacian Regularization for Image Denoising: Analysis in the Continuous Domain. *IEEE Transactions on Image Processing*, 26(4):1770–1785, 2017.
- [87] S. Purushotham, Y. Liu, and C. C. J. Kuo. Collaborative Topic Regression with Social Matrix Factorization for Recommendation Systems. In *Proceedings of ICML*, 2012.
- [88] N. Rao, H.-F. Yu, P. K. Ravikumar, and I. S. Dhillon. Collaborative Filtering with Graph Information: Consistency and Scalable Methods. In *Proceedings of NIPS*. 2015.
- [89] C. E. Rasmussen. Gaussian processes for machine learning. MIT Press, 2006.
- [90] S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of WSDM*, 2010.

- [91] M. Rogers, L. Li, and S. J. Russell. Multilinear Dynamical Systems for Tensor Time Series. In *Proceedings of NIPS*, 2013.
- [92] R. Salakhutdinov and A. Mnih. Probabilistic Matrix Factorization. In *Proceedings of NIPS*, 2008.
- [93] M. N. Schmidt, O. Winther, and L. K. Hansen. Bayesian non-negative matrix factorization. In *Proceedings of ICASSP*, 2009.
- [94] A. Shashua and T. Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of ICML*, 2005.
- [95] P. Smaragdis and J. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Proceedings of WASPA*, 2003.
- [96] A. J. Smola and R. Kondor. Kernels and regularization on graphs. In *Learning theory and kernel machines*, pages 144–158. Springer, 2003.
- [97] M. Steyvers, P. Smyth, and T. Griffiths. Probabilistic Author-Topic Models for Information Discovery. In *Proceedings of SIGKDD*, 2004.
- [98] J. Z. Sun, K. R. Varshney, and K. Subbian. Dynamic matrix factorization: A state space approach. In *Proceedings of ICASSP*, 2012.
- [99] L. Sun and K. W. Axhausen. Understanding urban mobility patterns with a probabilistic tensor factorization framework. *Transportation Research Part B: Methodological*, 91:511–524, 2016.
- [100] K. Takeuchi, R. Tomioka, K. Ishiguro, A. Kimura, and H. Sawada. Non-negative Multiple Tensor Factorization. In *Proceedings of ICDM*, 2013.
- [101] K. Takeuchi, Y. Kawahara, and T. Iwata. Higher order fused regularization for supervised learning with grouped parameters. *Machine Learning and Knowledge Discovery in Databases, Lecture Notes in Computer Science series (LNCS)*, pages 577–593, 2015.
- [102] R. Tandon and S. Sra. Sparse nonnegative matrix approximation: new formulations and algorithms. *Rapport technique*, 193:38–42, 2010.

- [103] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.
- [104] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.
- [105] J. M. Ver Hoef, E. Peterson, and D. Theobald. Spatial statistical models that use flow and stream distance. *Environmental and Ecological statistics*, 13(4):449–464, 2006.
- [106] P. Wais, S. Lingamneni, D. Cook, J. Fennell, B. Goldenberg, D. Lubarov, D. Marin, and H. Simons. Towards building a high-quality workforce with Mechanical Turk. *Proceedings of Computational Social Science and the Wisdom of Crowds (NIPS)*, 2010.
- [107] L. A. Waller and C. A. Gotway. *Applied spatial statistics for public health data*, volume 368. John Wiley & Sons, 2004.
- [108] C. Wang and D. M. Blei. Collaborative Topic Modeling for Recommending Scientific Articles. In *Proceedings of SIGKDD*, 2011.
- [109] F. Wang, C. Tan, A. König, and P. Li. Efficient document clustering via online nonnegative matrix factorizations. In *Proceedings of SIAM*, 2011.
- [110] Y.-X. Wang, J. Sharpnack, A. Smola, and R. J. Tibshirani. Trend filtering on graphs. *Journal of Machine Learning Research*, 17(105):1–41, 2016.
- [111] M. Welling and M. Weber. Positive tensor factorization. *Pattern Recognition Letters*, 22(12):1255–1261, 2001.
- [112] J. Weston, C. Wang, R. Weiss, and A. Berenzweig. Latent Collaborative Retrieval. In *Proceedings of ICML*, 2012.
- [113] D. M. Witten, R. Tibshirani, and T. Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534, 2009.

- [114] B. Xin, Y. Kawahara, Y. Wang, and W. Gao. Efficient Generalized Fused Lasso with its Application to the Diagnosis of Alzheimer’s Disease. In *Proceedings of AAAI*, 2014.
- [115] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of SIGIR*, 2003.
- [116] Y. Xu and W. Yin. A block coordinate descent method for regularized multi-convex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on imaging sciences*, 6(3):1758–1789, 2013.
- [117] Y. Xu, W. Yin, Z. Wen, and Y. Zhang. An alternating direction algorithm for matrix completion with nonnegative factors. *Frontiers of Mathematics in China*, 7(2):365–384, 2012.
- [118] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu. Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):129–142, 2015.
- [119] Y. K. Yilmaz and A. T. Cemgil. Probabilistic latent tensor factorization. In *Proceedings of LVA/ICA*, 2010.
- [120] T. Yokota, A. Cichocki, and Y. Yamashita. Linked PARAFAC/CP tensor decomposition and its fast implementation for multi-block tensor analysis. In *Proceedings of ICONIP*, 2012.
- [121] H.-F. Yu, N. Rao, and I. S. Dhillon. Temporal Regularized Matrix Factorization for High-dimensional Time Series Prediction. In *Proceedings of NIPS*. 2016.
- [122] S. Zafeiriou and M. Petrou. Nonnegative tensor factorization as an alternative Csiszar–Tusnady procedure: algorithms, convergence, probabilistic interpretations and novel probabilistic tensor latent variable analysis algorithms. *Data Mining and Knowledge Discovery*, 22(3):419–466, 2011.
- [123] J. Zhang, X. Pan, M. Li, and S. Y. Philip. Bicycle-sharing system analysis and trip prediction. In *Proceedings of MDM*, volume 1, 2016.

- [124] Q. Zhang, M. W. Berry, B. T. Lamb, and T. Samuel. A parallel nonnegative tensor factorization algorithm for mining global climate data. In *Proceedings of ICCS*, 2009.
- [125] Y. Zheng, F. Liu, and H.-P. Hsieh. U-air: When urban air quality inference meets big data. In *Proceedings of SIGKDD*, 2013.
- [126] X. Zhou, S. Shekhar, and R. Y. Ali. Spatiotemporal change footprint pattern discovery: an inter-disciplinary survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(1):1–23, 2014.