Latent Factorization for Hierarchical and Explainable Embeddings and Data Disaggregation

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Faisal M. Almutairi

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Prof. Nicholas D. Sidiropoulos, Advisor

August, 2021

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my advisor, Professor Nikolaos Sidiropoulos, for believing in me and helping me navigate the amazing and rewarding journey as a PhD student at the University of Minnesota (UMN). His work integrity, breadth of knowledge, and perpetual encouragement have inspired me and made me the scientist I am today. He was not only a research advisor, but also a life mentor supporting me through good and rough times.

I am very thankful to Professor Mehmet Akcakaya, Professor Christos Faloutsos, Professor Jarvis Haupt, and Professor George Karypis for serving on my thesis committee and providing valuable feedback on my dissertation.

I am very grateful for the inspiring discussions and collaborations I have had throughout my PhD with my former and current labmates: Xiao Fu, Kejun Huang, Charilaos Kanatsoulis, Aritra Konar, Bo Yang, Ahmed Zamzam, Nikos Kargas, Cheng Qian, Mohamed Salah, Paris Karakasis, Magda Amiridi, Mikael Sorensen, Panagiotis Alevizos, Yunmei Shi, and John Tranter.

During my PhD, I was fortunate to be given a great opportunity to collaborate with bright professors and colleagues within and outside the UMN. I would like to thank Professor Christos Faloutsos, Professor Vladimir Zadorozhny, Fan Yang, and Hyun Ah Song for their guidance and ideas throughout our project on time series disaggregation. I also had the pleasure to intern at IQVIA Inc. and collaborate with smart researchers and engineers. I would like to thank Yunlong Wang, Dong Wang, and Emily Zhao for the work we produced together during my internship, which is an important part of this dissertation.

I also would like to thank all the amazing friends I made here is Minnesota for the great memories we have had together and for their support.

Last but not least, I'd like to thank my parents, siblings, nephews and nieces for their unconditional love and unlimited support.

Dedication

In loving memory of my brother Muhammad.

Abstract

A tremendous growth in data collection has been an important enabler of the recent upsurge in Machine Learning (ML) models. ML techniques involve processing, analyzing, and discovering patterns from real user generated data. These data are usually high-dimensional, sparse, incomplete, and, in many applications, are only available at coarse granularity. For instance, a location mode can be at a state-level rather than county, or a time mode can be on a monthly basis instead of weekly. These (dis)aggregation challenges in real world data raise some intriguing questions and bring some challenging tasks. Given coarse-granular/aggregated data (e.g., monthly summaries), can we recover the fine-granular data (e.g., the daily counts)? Aggregated data enjoy concise representations and thus can be stored and transferred efficiently, which is critical in the era of data deluge. On the other hand, recent ML models are data hungry and benefit from detailed data for personalized analysis and prediction. Thus, data disaggregation algorithms are becoming increasingly important in various domains. In this thesis, we provide data disaggregation frameworks for one-dimensional time series data and multidimensional (tensor) data. The developed models recognize the structure of the data and exploit it to reduce the number of unknown parameters.

In a related setting, multidimensional data are often partially observed, e.g., recommender systems data are usually extremely sparse as a user interacts with only a small subset of the available items. Can we reconstruct/complete the missing data? This question is central in many recommendation and more general prediction tasks in various applications such as healthcare, learning and business analytics. A major challenge stems from the fact that the number of unknown parameters is usually much larger than the number of observed samples, which has motivated using prior information. Imposing the appropriate regularization prior limits the solution search to the ‘right’ space. In addition to sparsity, high-dimensionality also creates the challenge of ‘hiding’ the underlying structures and causes that can explain the data. In order to tackle this ‘dimensionality curse’, many dimensionality reduction (DR) methods such as principal component analysis (PCA) have been proposed. The reduced-dimension data usually yields better performance in downstream tasks, such as clustering. This suggests that the underlying structure (e.g., clustering) is more pronounced in some low-dimensional space compared to the original data domain. In this thesis, we present principled approaches that bridge incorporating prior information and DR techniques. We rely on low-rank (nonnegative) matrix factorization for DR and incorporate two different types of priors: i) hierarchical tree clustering, and ii) user-item embedding relationships. Imposing these regularization priors not only improves the quality of latent representations, but it also helps reveal more of the underlying structure in latent space. The tree prior provides a meaningful hierarchical clustering in an unsupervised data-driven fashion, while the user-item relationships underpin the latent factors and explain how the resulting recommendations are formed.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and Background

The impressive recent advancements in technology are associated with two major developments: i) an upsurge in Machine Learning (ML)/Artificial Intelligence (AI) models, and ii) a tremendous growth in data collection and analysis. These two developments are interrelated in many ways; for instance, recent ML models are more complex than the classical ones and they require large amount of data for training. Researchers from academia and industry are working hard to draw useful insights from data and to advance the role of ML in our lives in many diverse applications. In retail sales, for example, proper analysis of historical data helps in forecasting the future demand, and thus planing for economically efficient commerce. In healthcare, the accurate prediction of patients' prognosis and treatment helps pharmaceutical companies in identifying 'suitable' patients for clinical trials. Such prediction can also assist medical providers in early detection and diagnosis of disease. Another exciting application domain is in Learning Analytics – employing ML techniques for student performance prediction and course recommendation. These techniques enable early warning and degree planning "expert systems" to provide disciplined decision support to counselors, advisors, and educators. Recommendation engines are making an impact on our daily lives in a wide range of applications, pertaining to news, movies, music and podcasts, video platforms, and e-commerce.

ML techniques for the above applications (and numerous others) involve processing, analyzing, and discovering patterns from real user generated data. These data are usually high-dimensional, sparse, and incomplete. In many applications, data are available at a low-level (coarse) granularity, e.g., a location mode can be at a state level rather than county level, or a time mode can be on a monthly basis instead of weekly. In recommender systems, data are extremely sparse as a user interacts with only a small subset of the available items. Healthcare data are also high-dimensional, sparse (patients interact with small subsets of medical items), and aggregated in many dimensions (e.g., temporally, geographically, or by groups of hospitals), often to preserve privacy. The above challenges in real world data raise some intriguing questions and bring some challenging tasks. Given coarse-granular/aggregated data (e.g., monthly summaries), can we recover the fine-granular data (e.g., the daily counts)? Recent ML models are data hungry and benefit from detailed data for personalized analysis and prediction. Thus, data disaggregation algorithms are becoming increasingly important in various domains. In this thesis, we provide data disaggregation frameworks for one-dimensional time series data (chapter 2) and multidimensional data (chapter 3).

In a related setting, assume that we are given a small subset of high-dimensional data, such as users' purchase history or their ratings of items. Can we reconstruct/complete the data? This question is central in many tasks in various applications such as recommender systems, healthcare, and learning analytics. A major challenge stems from the fact that the number of unknown parameters is usually much larger than the number of observed samples (i.e., data is sparse). The sparsity challenge has motivated using prior information. Imposing the appropriate regularization prior limits the solution search to the 'right' space. In addition to sparsity, high-dimensionality also creates the challenge of 'hiding' the underlying structures and causes that can explain the data. Consider, for example, performing a k-nearest neighbors (kNN) algorithm to classify users based on their historical ratings of items. Even if in reality users belong to a small number of types (i.e., customer segments), the distance measure between users *in the original data domain* will not reveal those segments. In order to tackle this challenge, many dimensionality reduction (DR) methods such as principal component analysis (PCA) have been proposed. The dimension-reduced data usually yields better performance in downstream tasks, such as clustering. This suggests that the underlying structure (e.g., clustering structure) is more pronounced in some latent domain compared to the original data domain. Incorporating prior information into DR techniques has been shown to be effective in various applications such as clustering [113]. In this thesis, we present principled (nonnegative) matrix factorization-based approaches that incorporate two different types of priors: i) hierarchical tree clustering (chapter 4), and ii) user-item linear relationships (chapter 5). Imposing these regularization priors not

Index	Event	Begin Time	End Time	Count
1	Measles	1st Week	4th Week	300
2	Measles	3th Week	7th Week	400

Index	Event	Begin Time	End Time	Count
1	Measles	1st Week	4th Week	300
2	Measles	7th Week	11th Week	400

Index	Event	Begin Time	End Time	Count
1	Measles	1st Week	4th Week	300
2	Measles	1st Week	4th Week	400

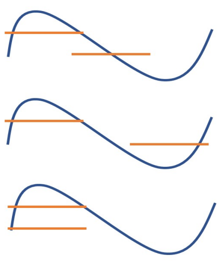


Figure 1.1: Example of data aggregated over weeks with overlap, gap, and conflict (from top to bottom).

only improves the latent representation quality, but also learn/reveal some underlying structure as we will explain in the following chapters.

1.2 Contributions and Thesis Outline

This thesis addresses the aforementioned problems for challenging cases that arise in ML and databases (DB). We provide intuitive, elegant, and effective solutions to these problems. The proposed framework for each task include concise and intuitive modeling, analytical analysis, efficient algorithms, and scalable implementation. Our contributions are summarized in the following subsections.

1.2.1 Chapter 2: Time Series Disaggregation

Data disaggregation considered in this chapter is a special case of data fusion as we aim to reconstruct an unknown time series from multiple aggregated observations with possible overlaps. For example, consider reconstructing the weekly counts of infection incidents (e.g., measles infections) in the United States from summaries aggregated over multiple weeks. In general, these aggregated summaries could have overlaps, gaps or conflicts between them; Figure 1.1 shows an illustrative example of each case, respectively. Formally, we want to reconstruct a fine-granularity time series $\mathbf{x} = \{x_n\}_{n=1}^{N}$, given the aggregated observations $\mathbf{y} = \{y_m\}_{m=1}^{M}$, where y_m corresponds to the sum of multiple elements in $\mathbf{x}$. Thus, we can specify a linear system $\mathbf{y} = \mathbf{O}\mathbf{x}$, where the m^{th} row in the observation matrix $\mathbf{O} \in \mathbb{R}^{M \times N}$ is a binary vector that has ones at the indices of $\mathbf{x}$ that contribute in y_m. This problem is usually under-determined in practice as the number of available aggregated measurements is much smaller than the length of the target series; thus, the Least Squares (LS) solution is meaningless – because the implicit minimum norm assumption used in standard underdetermined LS problems is simply not appropriate for real-life time-series. There have been previous works for solving the disaggregation problem that add different regularizers to the LS formulation, such as smoothness and periodicity [34, 69]. Enforcing smoothness and periodicity in the time domain is a reasonable approach since many of the time series that we observe are smooth and/or *quasi-periodic* in nature. The main issue with smoothness and periodicity regularized LS is the lack of identifiability, especially when the time series is not exactly smooth, nor exactly periodic.

In this chapter, we propose an efficient algorithm for solving the disaggregation problem in which we exploit an alternative representation of the target time series. More specifically, we search for the coefficients that best represent the series in a fixed dictionary of cosine basis, i.e., we solve for the coefficients of the Discrete Cosine Transform (DCT) of the series we are seeking. DCT with few non-zeros represents a sum of few cosines, i.e., few dominant periodicities. Therefore, DCT transformation is a good basis for quasi-periodic data. Moreover, expressing the time series using the DCT basis functions provides a *sparse* representation as most of the energy is compacted in the coefficients of low frequencies. One significant advantage of our approach is that it automatically detects the prominent periodicities in the data, as opposed to the methods in related works, which assume that there is only one or few known periodocities, typically corresponding to low frequencies only. We derive the steps of the *Alternating Direction Method of Multipliers* (ADMM) algorithm that can solve the optimization problem after adding non-negativity and smoothness constraints. Finally, we derive a scalable and memory efficient implementation of of the proposed algorithm. We demonstrate that the proposed framework helps to recover the time series much better than the competing baseline methods using real epidemiological datasets. The results of this chapter are reported in [10].

1.2.2 Chapter 3: Tensor Data Recovery from Multiple Aggregated Views

Multidimensional data are indexed by multiple indices, e.g., (i, j, k); thus, they can naturally be represented as a tensor (tensors are multi-way arrays). Tensor data have become ubiquitous and are frequently encountered in situations where the information is aggregated over multiple data atoms. The aggregation can be over time or other features, such as geographical location. We often have access to multiple aggregated views of the same data, each aggregated in one or more dimensions, especially when data are collected or measured by different agencies. For instance, item sales can be aggregated temporally, and over groups of stores based on their location or affiliation. Assume that we are given two coarse granularity tensors: i) a tensor indexed by (*store*, *item*, *week*), where the tensor elements are the sales count, and ii) another tensor index by (*groups of stores*, *item*, *day*). Note that the first tensor has a coarse granularity in the third mode, while the second tensor is aggregated over the store mode. Can we recover the fine-granular tensor indexed by (*store*, *item*, *day*)? The general disaggregation problem is ill-posed, which is clearly undesirable, even with multiple aggregates.

In this chapter, we propose a framework for fusing the multiple aggregates of multidimensional data. The proposed approach represents the target high resolution data as a tensor, and models that tensor using the canonical polyadic decomposition(CPD) to reduce the number of unknowns, while capturing correlations and higher-order statistical dependencies across dimensions. The proposed model employs a coupled CPD approach and estimates the low-rank factors of the target data, to perform the disaggregation task. This way, the originally ill-posed disaggregation problem is transformed to an overdetermined one, by leveraging the uniqueness properties of the CPD. Our approach is flexible in the sense that it can perform the disaggregation task on partially observed data, or data with missing entries. This is practically important as partially observed data appear often in real-world applications. Moreover, our framework handles the disaggregation task in cases where the aggregation pattern is unknown. To showcase the effectiveness of our methods, the chapter includes extensive experiments using real data from different domains: retail sales, crime counts, and weather observations. The results of this chapter are reported in [3, 4].

1.2.3 Chapter 4: Learning Tree-structured Embeddings

Matrix factorization (MF) plays an important role in a wide range of machine learning models, for various applications such as DR and embedding. A popular task is matrix completion, where the goal is to infer the unknown/missing matrix entries from the observed ones. A common approach is to employ MF to find a reduced-dimension representation (embedding) of each element corresponding to the matrix dimensions (e.g., users and items) These embeddings capture the essential information due to the ability of MF to capture correlations and higher-order statistical dependencies across dimensions. The entry corresponding to the i^{th} user and j^{th} item can be inferred by the inner product of their embeddings. Matrix completion finds a wide range of applications including collaborative filtering (CF) in recommender systems [58], disease and

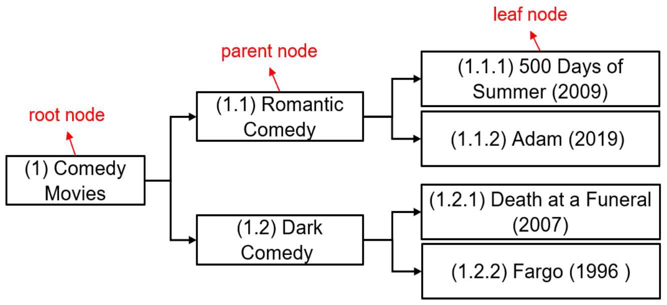


Figure 1.2: An example of hierarchical movie categories.

treatment prediction and patient subtyping [110] in healthcare analytics, student performance prediction and course recommendation in learning analytics [7], and image processing [68].

Incorporating side contextual information or priors, e.g., sparsity [45], smoothness, and latent clustering [113], is well-motivated in matrix factorization of sparse data. This is because a major challenge stems from the fact that we aim to find latent representations from very few samples. In this chapter, we present a principled approach that incorporates the *unknown* implicit tree structure prior. In many applications, categories of items display a hierarchical tree structure. In higher education, for instance, courses form multiple trees via their prerequisite hierarchy. Movie genres, e.g., comedy, action, and fantasy, comprise different fine subcategories as illustrated in the example in Fig. 1.2. Another example appears in Electronics Health Records (EHR) in healthcare analytics, where medical service (diagnoses, procedures, and prescription) can be clustered into subcategories, and these subcategories can also be grouped into coarse categories (examples are provided in the experimental results in Fig. 4.2). Individuals, e.g., users, students, and patients, also exhibit hierarchical clusters where the common traits between people increase as we move down from the root nodes to the leaf nodes in a tree [71, 109]. In many applications, the categorical hierarchy is either unknown, or requires manual labeling of massive amounts of data.

In this chapter, we propose eTREE (Learning Tree-structured Embeddings), a framework that integrates the unknown implicit tree structure into a low-rank nonnegative factorization model to improve the quality of embeddings. eTREE does not require any extra information and jointly learns: i) the embeddings of all the tree nodes (items, subcategories, and main categories), and ii) the tree clustering in an unsupervised fashion. The obtained tree provides clear hierarchical clusters as each node belongs to exactly one parent node, e.g., an item belongs to one subcategory, and a subcategory belongs to one main category. The formulation of eTREE handles partially observed data matrices, which appear often in real-world applications. By leveraging the special uniqueness properties of nonnegative matrix factorization (NMF), we prove identifiability of eTREE's model. We derive an efficient algorithm to compute eTREE with a scalable implementation that leverages parallel computing, computation caching, and warm-start strategies. We showcase the effectiveness of eTREE on real data from various application domains: healthcare, recommender systems, and education. We also demonstrate the meaningfulness of the tree obtained from eTREE by means of domain experts interpretation. The results of this chapter are reported in [9].

1.2.4 Chapter 5: Explainable Embeddings for Feature-based Collaborative Filtering

CF methods are making an impact on our daily lives in a wide range of applications, including recommender systems and personalization. CF involves processing big but sparse data to extract (filter) what is relevant to a user of interest. Latent factor methods, e.g., matrix factorization (MF), have been the state-of-the-art in CF, however they lack interpretability and do not provide a straightforward explanation for their predictions. Explainability is gaining momentum in recommender systems for accountability, and because a good explanation can swing an undecided user. Most recent explainable recommendation methods require auxiliary data such as review text or item content on top of item ratings.

In this chapter, we address the case where no additional data are available and propose augmenting the classical MF framework for CF with a prior that encodes each user's embedding as a sparse linear combination of item embeddings, and vice versa for each item embedding. Our approach automatically reveals these user-item relationships, which underpin the latent factors and explain how the resulting recommendations are formed. We showcase the effectiveness of our approach on real data from various application domains. We also evaluate the explainability of the user-item relationship obtained from our algorithm through numeric evaluation and case study examples. The results of this chapter are reported in [8].

1.3 Notational Conventions

The notation used in this thesis is summarized in Table 1.1.

Table 1.1: Symbols and Definitions

Symbol	Definition
$x, \mathbf{x}, \mathbf{X}, \underline{\mathbf{X}}$	Scalar, vector, matrix, tensor
$\mathcal{I}, \mathcal{J}$	sets
$\mathbf{X}_n$	Mode-n matricization (unfolding)
$\|.\|_F$	Frobenius norm of a matrix/tensor
$\mathbf{X}^T$	Transpose of matrix $\mathbf{X}$
$\mathbf{X}^\dagger$	Moore-Penrose pseudo-inverse of a matrix
x_i	The i^{th} element in vector $\mathbf{x}$
$\mathbf{X}(i,:)$	the i^{th} row of $\mathbf{X}$
$\mathbf{X}(:,j)$	The j^{th} column of $\mathbf{X}$
$\mathbf{X}(\mathcal{J},:)$	The rows of $\mathbf{X}$ in the set $\mathcal{J}$
vec(.)	Vectorization operator of a matrix/tensor
$(\mathbf{x})_+$	The non-negative projection of a vector $\mathbf{x}$ (zeroing out the negative elements in $\mathbf{a}$)
$[\![.]\!]$	Kruskal operator, e.g., $\underline{\mathbf{X}} \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$
$\circ$	Outer product
$\otimes$	Kronecker product
$\odot$	Khatri-Rao product (column-wise Kronecker)
$\circledast$	Hadamard (element-wise) product

Chapter 2

Time Series Disaggregation

2.1 Introduction

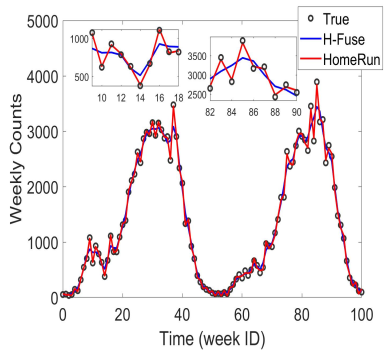
Gathering and analyzing information from multiple historical data sources requires reconstructing the time sequences in finer scale. For example, given multiple *monthly* sums of patient counts, how can we recover the *weekly* patient counts? This is so-called data disaggregation problem [92].

Notable challenges of historical data disaggregation are: 1) each data source may report the aggregated sums on *different scales* (e.g., one data source may report the weekly number of patients while another source reports on monthly scale), 2) the *time periods* covered by different data sources *may or may not overlap* (e.g., one source may report the number of patients for years 1920-1930, and another for 1940-1950, resulting in missing information for years 1930-1940), and 3) the reports may have conflicts (e.g., one data source may report 100 patients while another may report 80 patients for the same time period). Our informal problem definition is given as follows:

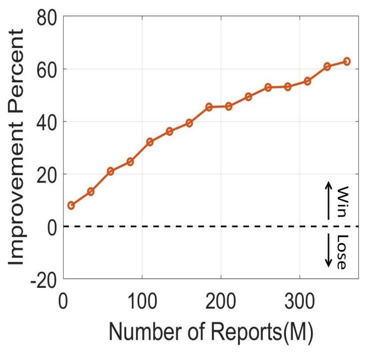
Informal Problem 2.1 (Disaggregation)**.**

1. *Given: the multiple reports of the aggregated sums of the time sequence (e.g., monthly sums)*
2. *Recover: the time sequence in finer scale (e.g., weekly sums)*

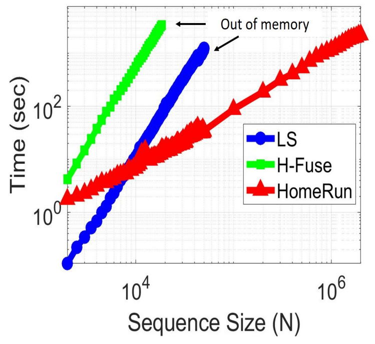
The prevailing approach is to formulate the problem as linear Least Squares (LS), however, as we will explain in more details later, this problem is usually under-determined in practice. In cases



(a) More accurate reconstruction



(b) HomeRun wins



(c) Linear scalability

Figure 2.1: HomeRun is effective and scalable: (a) visible improvement of HomeRun over the baseline method H-Fuse; (b) performance of HomeRun versus H-Fuse across different number of reports; (c) HomeRun is memory efficient and scales linearly with the length of the target sequence.

where the number of available reports is much smaller than the length of the target sequence, the LS approach becomes inefficient. There have been previous works for solving the disaggregation problem that add different regularizers to the LS, such as smoothness and periodicity in the data. Enforcing smoothness and periodicity in the time domain is a reasonable approach since many of the time sequences that we observe are smooth and *quasi-periodic* in nature. The main issue with smoothness and periodicity regularized LS is the lack of identifiability, especially when the time series is not exactly smooth, nor exactly periodic.

In this chapter, we propose HomeRun– an efficient algorithm for solving the disaggregation problem in which we exploit an alternative representation of the target time sequence. More specifically, we search for the coefficients that best represent the sequence in a fixed dictionary of cosine basis, i.e., we solve for the coefficients of the Discrete Cosine Transform (DCT) of the sequence we are seeking. As we will explain in the following section, DCT with few non-zeros represents a sum of few cosines, i.e., few dominant periodicities. Therefore, DCT transformation is a good basis for quasi-periodic historical data. Moreover, expressing the time sequence using the DCT basis functions provides a *sparse* representation as most of the energy is compacted in the coefficients of low frequencies.

We formulate the data disaggregation in the form of the so-called *Basis Pursuit* (BP) where we enforce sparsity in the DCT coefficients of the target sequence. We call the resulting BP formulation HOMERUN-0, which is the basic version of our proposed method. One significant advantage of our approach is that it automatically detects the prominent periodicities in the data, as opposed to the methods in related works, which assume that there is only one or few known periodicities. In addition to the periodicity, other common domain knowledge properties of the time sequences are non-negativity and smoothness over timestamps. We also propose HOMERUN-N method that improves HOMERUN-0 by enforcing non-negativity constraint on the time domain sequence. We further extend our method by imposing smoothness in the time sequence in addition to non-negativity, resulting in the final version of the proposed method: HOMERUN. We derive the steps of the *Alternating Direction Method of Multipliers* (ADMM) algorithm that can solve the optimization problem after adding non-negativity and smoothness constraints. Finally, we derive a scalable and memory efficient implementation of HOMERUN.

We apply HOMERUN to the epidemiological data from the Tycho project [105]. Our dataset contains the number of cases for major epidemic diseases (hepatitis A, measles, mumps, pertussis, polio, rubella, and smallpox) in the US over 100 years. We demonstrate that HOMERUN helps to recover the time sequences much better than the competing baseline methods, H-FUSE [69] and LS.

Figure 2.1 shows an example of the results of HOMERUN when reconstructing the weekly counts of measles, given multiple aggregated reports. We observe in Fig. 2.1 (a) that HOMERUN is closer to the true sequence compared to the baseline H-FUSE, HOMERUN estimates the number of patients with Root Mean Square Error (RMSE $= 20.30$), while the RMSE of H-FUSE is 104.23. In data analysis, e.g., studying the impact of vaccination, not only the average error matters, but also the weekly single error. We can see that for several weeks, H-FUSE underestimates (or overestimates) the counts by the order of hundreds, while HOMERUN is very close to the true value. Fig. 2.1 (b) shows the percentage of improvement/diminishment in the RMSE between HOMERUN and the baseline H-FUSE with various numbers of given aggregated reports – HOMERUN always improves the RMSE, '70' means HOMERUN reduces the RMSE of H-FUSE by 70%, and so on. Fig. 2.1 (c) compares the running time of HOMERUN with the baselines. It shows how HOMERUN is memory efficient and scales linearly in time with the sequence length (up to 2 million) – note the log scales. In summary, the contributions of our work are as follows:

- **Formulation and Algorithm**: we propose to formulate the data disaggregation problem in the form of so-called Basis Pursuit (BP), add domain knowledge constraints, and derive the iterative updates of the ADMM algorithm to solve the resulting optimization problem.

- **Effectiveness**: our HOMERUN method recovers the time sequences with up to 94% improvement in the accuracy of the best of baseline methods.

- **Scalability**: we derive an efficient accelerated implementation of HOMERUN that scales linearly with the length of the target sequence.

- **Adaptability**: HOMERUN is parameter-free and it adapts to the input signal and automatically detects the prominent periodicities in the data.

Reproducibility: The Tycho dataset is publicly available [105]; the code is available at `https://github.com/FaisalAlmutairi/HomeRun_time_series_disaggregation`.

The chapter structure is as follows. We explain the necessary background and the related work in section 2.2, and introduce our proposed method in section 2.3. Then, we explain our experimental setup in section 2.4 and show the experimental results in section 3.5. We conclude in section 2.6.

2.2 Background

In this section, we provide background on both the problem of historical data disaggregation *and* the techniques we employ in the proposed approach to solve this problem. We also review the related work relevant to both the problem and the proposed method.

Notation: Table 2.1 summarizes the symbols we use frequently in this chapter.

Table 2.1: Symbols and Definitions

Symbol	Definition
$\mathbf{y}$	$\in \mathbb{R}^M$ vector contains the known measurements
$\mathbf{x}$	$\in \mathbb{R}^N$ the target time sequence
$\mathbf{s}$	$\in \mathbb{R}^N$ sparse presentation of $\mathbf{x}$ in fixed basis
$\mathbf{O}$	$\in \mathbb{R}^{M\times N}$ observation matrix
$\mathbf{D}$	matrix of DCT basis functions
RD	report duration
$shift$	difference between the starts of adjacent reports
H	$\in \mathbb{R}^{(N-1)\times N}$ smoothness matrix

2.2.1 Historical Data Disaggregation

Data disaggregation considered in this work is a special case of data fusion as we aim to reconstruct an unknown time sequence from multiple aggregated observations with possible overlaps. For example, consider reconstructing the weekly counts of infection incidents (e.g., by measles) in the United States from reports aggregated over multiple weeks. In general, those aggregated reports could have overlaps, gaps or conflicts between them. Figure 1.1 shows an illustrative example of each case, respectively.

Formally, we want to reconstruct a detailed time sequence $\mathbf{x} = \{x_n\}_{n=1}^N$, given the aggregated observations $\mathbf{y} = \{y_m\}_{m=1}^M$, where y_m corresponds to the sum of multiple elements in $\mathbf{x}$. Thus, we specify a linear system $\mathbf{y} = \mathbf{O}\mathbf{x}$, where each row of an observation matrix $\mathbf{O} \in \mathbb{R}^{M\times N}$ is a binary vector that has ones for the elements of $\mathbf{x}$ that contribute in y_m (see Example in Eq. 1). We refer to the number of timeticks covered by a report as *Report Duration* (RD), i.e., ones in the m^{th} row of $\mathbf{O}$, and the difference between the starting points of two successive reports as $shift$ (marked in blue in Eq. 1). Observed reports may have different RD values, e.g., we may have one report covering a month and another covering two weeks. In any case, we can sort the reports according to their starting points. Below is an illustrative example containing three reports with $RD = 4, 4,$ and 2, and $shift = 1$ between the first two reports and $shift = 2$ between the 2^{nd} and 3^{rd} ones. Note that the reports in the example have overlaps (marked in green), however they could have gaps if the $shift$ between two reports is larger than RD of the first one. Moreover, conflict occurs when two rows of $\mathbf{O}$ are identical, but with different y_m values.

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}}_{\mathbf{O}} \times \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}}_{\mathbf{y}} \tag{2.1}$$

If **O** is square (i.e., $N = M$) and full rank, then the solution is trivial and error free. In practical settings, the resulting system of linear equations is under-determined (number of reports $\ll$ number of timeticks in the target sequence). In this case, the linear system has many solutions and Least Square (LS) solution finds $\mathbf{x}$ with minimum norm (min $\|\mathbf{x}\|_2^2$). However, there is no special reason why the best reconstructed sequence would have the minimum norm for this problem, which led researchers to add domain knowledge penalty terms to the linear system [69] to improve the LS solution.

Instead of solving for $\mathbf{x}$ directly, as it is common in the literature for this problem, we exploit an alternative signal representation and propose to solve for the DCT representation of the target sequence (as formulated in Section 2.3). We define the DCT in the next section before proceeding to the proposed methods.

2.2.2 Discrete Cosine Transform

Discrete Cosine Transform (DCT) transforms a finite-length discrete-time data sequence from the time (or spatial) domain into the frequency domain. In particular, DCT represents the finite-length sequence in terms of a sum of *basis sequences*. These basis are cosines oscillating at different frequencies [2, 77]. We focus here on one-dimensional DCT since the problem of our interest is the reconstruction of one-dimensional sequence. Formally, the most common DCT definition of a data sequence **x** of length N is as follows [56]:

$$s_k = \sum_{n=0}^{N-1} x_n \underbrace{\alpha(k) cos\left(\frac{\pi k(2n+1)}{2N}\right)}_{\phi(k,n)} = \sum_{n=0}^{N-1} x_n \phi(k,n) \tag{2.2}$$

for $0 \leq k \leq N-1$, where $\alpha(k)$ is a coefficient factor defined as follows:

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{N}}, & k = 0, \\ \sqrt{\frac{2}{N}}, & 1 \leq k \leq N-1. \end{cases} \tag{2.3}$$

Similarly, the original finite-length sequence can be uniquely recovered from its DCT using the inverse DCT (iDCT) defined as:

$$x_n = \sum_{k=0}^{N-1} s_k \phi(k,n) \tag{2.4}$$

for $0 \leq n \leq N-1$.

To facilitate concisely formulating the problem, we define a DCT matrix $\mathbf{D} \in \mathbb{R}^{N \times N}$ whose entries are the cosine basis functions:

$$\mathbf{D} = \begin{bmatrix} \phi(0,0) & \dots & \phi(0,N-1) \\ \vdots & \ddots & \\ \phi(N-1,0) & \dots & \phi(N-1,N-1) \end{bmatrix} = \begin{bmatrix} \mathbf{d}_0^T \\ \vdots \\ \mathbf{d}_{N-1}^T \end{bmatrix} \tag{2.5}$$

where, as it is clear from (2.2), $\phi(k,n) = \alpha(k)cos(\pi k(2n+1)/2N)$. The inner product of any row $\mathbf{d}_n$ with itself is 1, while the inner product of any two different rows is 0. Thus, $\mathbf{D}$ is orthogonal (and DCT is an orthogonal transform [77]), i.e., $\mathbf{D}^{-1} = \mathbf{D}^T$. Equations (2.2) and (2.4) can be written as:

$$\mathbf{s} = \mathbf{D}\mathbf{x} \tag{2.6}$$

$$\mathbf{x} = \mathbf{D}^T\mathbf{s} \tag{2.7}$$

Since cosines are *periodic* and *even symmetric*, the DCT transform imposes periodicity in the time domain signal [77]. An important property of DCT is energy compaction, which is the reason why DCT is widely used in many data compression applications, such as image compression [90]. Specifically, the DCT of a signal is usually concentrated in the coefficients of the low frequencies and the remaining coefficients can be discarded without a significant impact [77]. The degree of DCT energy compaction depends on how correlated the original signal

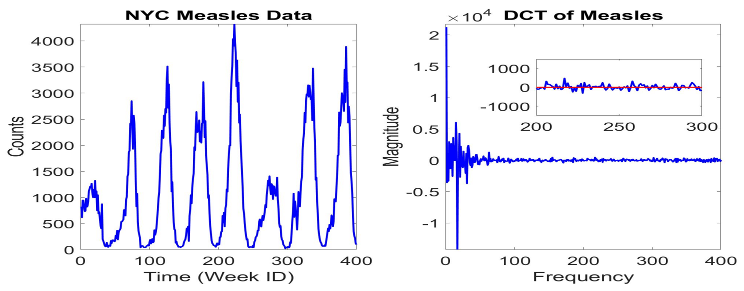


Figure 2.2: NYC measles data in time domain, **x** (left) and its spectrum, **s** (right).

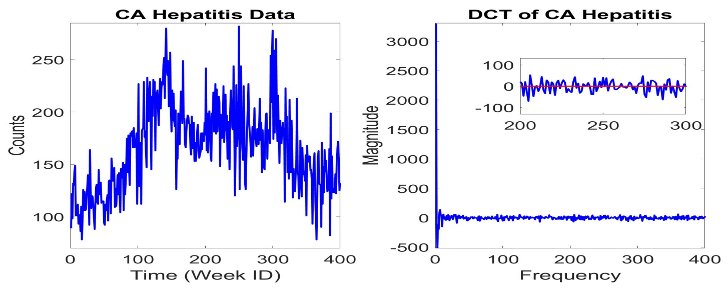


Figure 2.3: CA hepatitis data in time domain, **x** (left) and its spectrum, **s** (right).

is in time (or spatial) domain. For example, in image processing, the DCT energy compaction of an image relies on the correlation degree between its pixels. We demonstrate this phenomenon by showing New York (NYC) measles and California (CA) hepatitis weekly counts and their DCT in Figure 2.2 and 2.3, respectively. We can see that CA hepatitis sequence is less correlated (less smooth) in time domain, and therefore its DCT has high frequency components that are larger than in the case of NYC measles (*relative to their maximum values*) as clear in the zoomed parts.

If we discard the small coefficients of DCT, the DCT representation of the signal becomes sparse. In other words, although the reports of those diseases do not have zero values across all timeticks, most of their DCT coefficients are small, and the few large coefficients carry most of the energy and capture most of the information. We show an illustrative example by keeping only the largest 10% of the DCT coefficients of NYC measles and CA hepatitis weekly counts sequences and set the rest to zero, i.e., we pick the largest 10% elements in $\mathbf{s}$ and zero out the rest. In Figure 2.4, we show the time sequence of both data sets recovered from this 10% (using Equation (2.7)). It is clear that NYC measles has a better recovery since its DCT is sparser (compact and has less significant components). Finally, we should note that the ability of accurately estimating the DCT coefficients (**s**) of a sequence enables us to recover this sequence in time-domain (**x**). In the following section, we explain the basics of sparse reconstruction since it is essential to our methods.

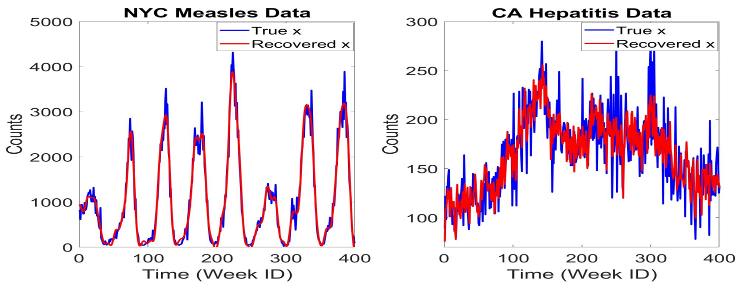


Figure 2.4: Data recovered from the largest 10% coefficients of their DCT – NYC measles (left) and CA hepatitis (right).

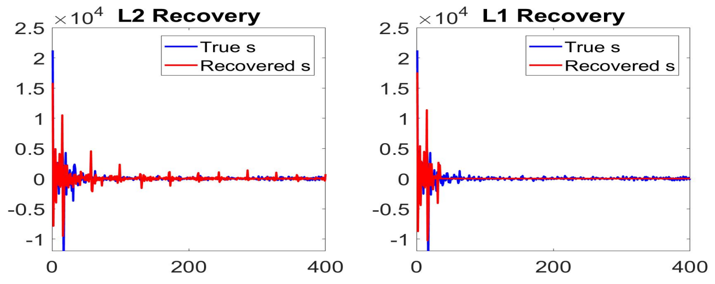


Figure 2.5: Recovery with $L1$ norm (left) and recovery with $L2$ norm by replacing $L1$ by $L2$ in (2.8) (right).

2.2.3 Sparse Signal Recovery

The goal of sparse reconstruction and compressive sensing [17] is to find a *sparse* approximate solution s of an under-determined linear system $\mathbf{As} = \mathbf{y}$, where $\mathbf{A} \in \mathbb{R}^{M \times N}, \mathbf{s} \in \mathbb{R}^{N}, \mathbf{y} \in \mathbb{R}^{M}$, with $M < N$. Then, s could be recovered by solving the following convex problem known as BP [20]:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{As} = \mathbf{y} \end{aligned} \tag{2.8}$$

where $\|s\|_1 = \sum_{n=1}^{N} |\mathbf{s}_n|$ is the $L1$ norm which promotes sparsity in the solution. In general, $\mathbf{A}$ may be a matrix containing the elements of an over-complete dictionary [30] and we seek to solve for the sparsest coefficient vector $\mathbf{s}$ to represent the observed measurements $\mathbf{y}$.

Now we will consider a more practical scenario: suppose we have a (non-sparse) signal in time-domain $\mathbf{x} \in \mathbb{R}^N$ under-sampled in such a way that we have fewer linear measurements $\mathbf{y} \in \mathbb{R}^M$ about $\mathbf{x}$ in the form $\mathbf{y} = \mathbf{\Phi x}$, where $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$. Thus, we are interested in solving for the unknown signal $\mathbf{x}$ given the observed measurements $\mathbf{y}$. In the case of $M \ll N$ where there are much fewer measurements than the unknowns, solving the linear problem may appear too challenging. However, if $\mathbf{x}$ can be compressed (accurately represented as sparse coefficients on some fixed basis) such that the number of the non-zero coefficients that carry most of the energy is less than N (the size of $\mathbf{x}$), then this changes the problem radically, making the search for solutions feasible [17]. In particular, suppose we have a sparse vector $\mathbf{s}$ that contains the coefficients of a time (spatial)-domain signal $\mathbf{x}$ in an orthonormal basis $\mathbf{\Psi}$, i.e., $\mathbf{x} = \mathbf{\Psi s}$. For example, $\mathbf{x}$ is the vector containing pixels of an image, $\mathbf{s}$ is the coefficient sequence of $\mathbf{x}$ in the wavelet basis, and $\mathbf{\Psi}$ is an $N \times N$ matrix containing the *wavelet basis functions* as its entries [19]. In this case, we would recover the coefficient sequence $\mathbf{s}$ with the minimum $L1$ norm that satisfies $\mathbf{As} = \mathbf{y}$, where $\mathbf{A} = \mathbf{\Phi\Psi}$ in problem (2.8).

As mentioned above, BP is often used as a heuristic algorithm for finding a sparse solution to an under-determined system of linear equations and $L1$ promotes sparsity in the solution. In Figure 2.5, we illustrate the advantage of using $L1$ norm by showing the solution we get from (2.8) and when replacing the $L1$ norm by $L2$ norm. In this example, we try to recover the DCT representation of NYC measles data (shown in Figure 2.2) from 29 measurements in the time domain (each measurement has the sum of counts over 21 weeks with overlaps). We can see that the two solutions are different. The $L2$ solution does not give a good approximation as it has spikes where the original signal is almost zero.

Alternating Direction Method of Multipliers: problem (2.8) can be recast as a linear program. However, we propose to use the ADMM algorithm as it is well suited for large-scale problems. ADMM solves the convex optimization of the following form

$$\begin{aligned} \min_{\mathbf{s},\mathbf{z}} \quad & f(\mathbf{s}) + g(\mathbf{z}) \\ s.t. \quad & \mathbf{As} + \mathbf{Ez} = \mathbf{c}. \end{aligned} \tag{2.9}$$

by iteratively updating the following blocks

$$\mathbf{s} \leftarrow arg\min_{\mathbf{s}} f(\mathbf{s}) + (\rho/2)\|\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2, \tag{2.10a}$$

$$\mathbf{z} \leftarrow arg\min_{\mathbf{z}} g(\mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2, \tag{2.10b}$$

$$\mathbf{u} \leftarrow \mathbf{u} + (\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c}) \tag{2.10c}$$

where **u** is a scaled version of the dual variable corresponding to the equality constraint in (2.9), and $\rho > 0$ is the augmented Lagrangian parameter specified by the user.

Problem (2.8) can be reformulated as follows after introducing the auxiliary variable $\mathbf{z} \in \mathbb{R}^N$:

$$\begin{aligned} \min_{\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{A}\mathbf{s}=\mathbf{y}\}} + \|\mathbf{z}\|_1 \\ s.t. \quad & \mathbf{s} - \mathbf{z} = \mathbf{0} \end{aligned} \tag{2.11}$$

where $\mathcal{I}_{\{\mathbf{A}\mathbf{s}=\mathbf{y}\}}$ is an indicator function such that:

$$\mathcal{I}_{\{\mathbf{A}\mathbf{s}=\mathbf{y}\}} = \begin{cases} 0 & \text{if } \mathbf{A}\mathbf{s} = \mathbf{y} \\ \infty & \text{otherwise.} \end{cases}$$

We will skip the derivation of the algorithm for brevity – refer to [16] for more comprehensive review of the ADMM algorithm. The solution to (2.8) is provided by the following iterative updates:

$$\mathbf{s}^{k+1} \leftarrow (\mathbf{I} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A})(\mathbf{z}^k - \mathbf{u}^k) + \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{y}$$

$$\mathbf{z}^{k+1} \leftarrow (\mathbf{s}^{k+1} + \mathbf{u}^k - 1/\rho)_+ - (-\mathbf{s}^{k+1} - \mathbf{u}^k - 1/\rho)_+$$

$$\mathbf{u}^{k+1} \leftarrow \mathbf{u}^k + (\mathbf{s}^{k+1} - \mathbf{z}^{k+1})$$

2.2.4 Related Work

Disaggregation: The aggregation of the data vector can be seen as representing or summarizing the data vector using linear transform. In [24, 26], the idea of sketches has been introduced as means of data aggregation or summarization. With the advance of the data collection technologies, we have been gaining more access to various sources of historical data in aggregated form.

This has led to an increasing interest in data integration and fusion including, in particular, disaggregation of the data sources [15, 29, 34, 69, 87, 116, 120]

The disaggregation problem is of interest in various domains. In the image and signal processing communities, for example, there have been works on solving under-determined problems for various applications, such as super-resolution reconstruction of image data [78], or information recovery from noisy or missing data [18].

In recent work [69], the authors proposed an algorithm called H-FUSE that enforces smoothness and periodicity constraints for the reconstruction of the historical data. The authors show that the proposed algorithm improves the reconstruction compared to the minimum-norm linear LS formulation, which is the most common formulation for solving the disaggregation problem. We will use this algorithm as our main baseline.

DCT and Sparse reconstruction: DCT is one of the most commonly used compression techniques in the signal processing community. It has been shown to be an effective transformation for compressing the data in large networks [61]. DCT is widely used especially for image compression [111] due to its energy compaction property, and for image denoising [31, 39]. DCT has been also used in databases community for answering queries in compressed form via DCT transform [46], representing the time series in spectral domain [28], etc.

The principle of finding a sparse signal representation in a basis dictionary has been used in various applications, such as denoising [20] and information recovery from incomplete and/or noisy measurements [18]. Basis Pursuit (BP) formulation is used to obtain such a sparse solution to the ill-posed problem. Expressing a signal in a proper basis (dictionary), where it is sparse for the purpose of recovering this signal from fewer linear measurements has been used in compressive sensing [17]. As an example, wavelet basis has been considered in order to sparsely represent an image to recover it from fewer measurements [19]. In [31], DCT is used as a dictionary for sparse representation of a noisy image for the purpose of removing the noise from the image.

To our knowledge, the application of DCT and sparse representation has not been exploited in historical data fusion domain. Table 2.2 summarizes our proposed HOMERUN method compared to the related approaches.

Table 2.2: HOMERUN satisfies all properties listed below.

Property	LS	H-FUSE	HOMERUN
Overlapping reports	✓	✓	✓
Smoothness reconstruction		✓	✓
Periodicity reconstruction		✓	✓
Multiple periods (quasi-periodic)			✓
Automatically detects periodicity			✓
Non-negativity			✓

2.3 Proposed Method: HomeRun

In this section, we explain our proposed method HOMERUN and the algorithmic solutions associated with it. Recollect, that the objective of reconstruction methods is finding the disaggregated sequence $\mathbf{x}$ that minimizes the following problem:

$$\min_{\mathbf{x}} \quad \|\mathbf{y} - \mathbf{O}\mathbf{x}\|_2^2 \tag{2.12}$$

where $\mathbf{O}, \mathbf{x}, \mathbf{y}$ have the same definition as in Section 2.2.1. More advanced methods are proposed to infuse domain knowledge, such as smoothness and periodicity, by penalizing (2.12) [69]. The role of this penalty is to make the under-determined linear system (that has infinite number of solution) an over-determined one, constraining the solution to adhere to some domain knowledge. All these methods solve the problem directly by the closed form of LS using *Moore-Penrose pseudo-inverse*. However, LS solution does not always give a good approximation, especially when the number of observations is much less than the number of unknown variables.

The main idea behind our proposed method is to deal with the under-determinacy of the linear system by solving for the coefficient vector $\mathbf{s}$ that represents the target sequence $\mathbf{x}$ in the DCT basis as the number of non-zero coefficients is much less than the length of the sequence. The accuracy of this reconstruction hinges on the degree of DCT energy compaction feature explained in Section 2.2.2. Moreover, DCT involves implicit assumptions of periodicity which makes it a good dictionary for this problem as the time sequence exhibits some degree of periodicity. A significant advantage of the proposed approach is that it automatically detects the prominent periodicities in the data, as opposed to assuming that there is only one or few *known* periodicities by constraining the LS as in [69]. In the rest of this section, we explain our proposed method in the order as it was derived.

- HOMERUN-0: the basic version of our method.
- HOMERUN-N: with added non-negativity constraint.
- **HOMERUN: the final and complete version of the proposed method.**

2.3.1 HOMERUN-0: The Basic Version

DCT matrix (defined in Equation (2.5)) offers a convenient way to compute the transform and its inverse as follows: $\mathbf{s} = \mathbf{D}\mathbf{x}$ (Equation (2.6)) is the DCT of the time sequence $\mathbf{x}$, and $\mathbf{x} = \mathbf{D}^T\mathbf{s}$ (Equation (2.7)) reconstructs the time sequence from $\mathbf{s}$. The following Insight shows the formulation of our HOMERUN-0 method.

Insight 2.1. *The historical data disaggregation problem can be formulated in the form of Basis Pursuit as follows:*

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{A}\mathbf{s} = \mathbf{y} \end{aligned} \tag{2.13}$$

Rationale 2.1. *Given* $\mathbf{O}\mathbf{x} = \mathbf{y}$*, we want to find the* sparse *vector that contains the DCT of the target sequence* $\mathbf{x}$*. Since minimizing the* $L1$ *norm promotes sparsity in the solution, we look for the minimum* $\|\mathbf{s}\|_1$ *that satisfies* $\mathbf{O}\mathbf{x} = \mathbf{y}$*. Replacing* $\mathbf{x}$ *by* $\mathbf{D}^T\mathbf{s}$*, we get the problem in the following form:*

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y} \end{aligned} \tag{2.14}$$

Note that (2.14) *is similar to* (2.13) *with* $\mathbf{A} = \mathbf{O}\mathbf{D}^T$.

We solve the above problem using the ADMM algorithm with the iterative updates presented in Section 2.2.3. After we get the solution to $\mathbf{s}$, the approximate solution of the target sequence is obtained as $\mathbf{x}_{\text{HOMERUN-0}} = \mathbf{D}^T\mathbf{s}$.

Conflicting reports: if there is a conflict between the reports (as explained in Fig. 1.1), then the linear system $\mathbf{O}\mathbf{x} = \mathbf{y}$ is inconsistent. As a result, the constraints in (2.14) can not be satisfied. We resolve this issue by the following preprocessing step: if $\mathbf{O} \in \mathbb{R}^{M \times N}$ is full row rank (i.e., rank($\mathbf{O}) = M$), then there is no conflict and we proceed with the algorithm to solve (2.14). If the rows of $\mathbf{O}$ have some linear dependency (i.e., rank($\mathbf{O}) < M$), then we have one of two cases: a) if $\mathbf{y} \in$ span($\mathbf{O}$) (column space of $\mathbf{O}$), then the system is consistent (there is no conflict) and we proceed with the algorithm; b) if $\mathbf{y} \notin$ span($\mathbf{O}$), then we have an inconsistent linear system. In this case, we replace $\mathbf{y}$ with its projection onto the span($\mathbf{O}$), $\overline{\mathbf{y}}$ as follows

$$\overline{\mathbf{y}} = proj_{\mathbf{O}}(\mathbf{y}) = \mathbf{O}(\mathbf{O}^T\mathbf{O})^{-1}\mathbf{O}^T\mathbf{y} \tag{2.15}$$

This orthogonal projection results in the nearest vector (set of reports) to $\mathbf{y}$ that is free of conflict. Previous methods for this problem (H-FUSE and LS) provide solutions that minimize the squared error in case of conflicts. Assuming that flawed reports are rare (correct reports are the norm), we would normally want to satisfy as many equations as possible, instead of using an inconsistent solution that minimizes the squared norm of the violations but could violate all equations. This turns out to be NP-hard however [11]. The advantage of our projection approach is that the solution satisfies *all* the equations in the linear system with $\overline{\mathbf{y}}$, which is the closest vector to $\mathbf{y}$ that belongs to the column space of $\mathbf{O}$. Note that this applies to the coming optimization formulations (HOMERUN-N, and HOMERUN).

2.3.2 HOMERUN-N: The Non-Negative Version

In the applications of our interest, the target sequence $\mathbf{x}$ is always non-negative. Therefore, we exploit this domain knowledge by adding the constraint $\mathbf{x} \geq \mathbf{0} \Leftrightarrow \mathbf{D}^T\mathbf{s} \geq \mathbf{0}$ to (2.14). The resulting formulation becomes:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{s} \geq \mathbf{0} \end{aligned} \tag{2.16}$$

Statement 2.1. *The ADMM algorithm can be adapted to solve the optimization formulation of* HOMERUN-N *in Equation* (2.16).

Proof. Problem (2.16) is convex and we reformulate it by introducing the auxiliary variables $\mathbf{r}, \mathbf{z} \in \mathbb{R}^N$ as follows

$$\begin{aligned} \min_{\mathbf{r},\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{z} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{z} = \mathbf{r}, \quad \mathbf{z} = \mathbf{s} \end{aligned} \tag{2.17}$$

where, again, $\mathcal{I}$ is an indicator function of $\{\mathbf{r} \in \mathbb{R}^N \colon \mathbf{r} \geq \mathbf{0}\}$ defined as:

$$\mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} = \begin{cases} 0 & \text{if } \mathbf{r} \geq \mathbf{0} \\ \infty & \text{otherwise.} \end{cases}$$

To facilitate concise notation and precisely present the algorithm using only matrix algebra, we define the following (components of $\mathbf{u}$ are defined and used later):

$$\mathbf{B} := \begin{bmatrix} \mathbf{OD}^T \\ \mathbf{D}^T \\ \mathbf{I} \end{bmatrix}; \quad \mathbf{b} := \begin{bmatrix} \mathbf{y} \\ \mathbf{r} \\ \mathbf{s} \end{bmatrix}; \quad \mathbf{u} := \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \end{bmatrix}, \tag{2.18}$$

where $\mathbf{I} \in \mathbb{R}^{N\times N}$ is the identity matrix. By concatenating the constraints as $\mathbf{Bz} - \mathbf{b} = \mathbf{0}$, it is straightforward to see that problem (2.17) above is in the form of the ADMM form defined in Equation (2.9) with $g(\mathbf{z}) = 0$, and $f(\mathbf{r}, \mathbf{s}) = \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \|\mathbf{s}\|_1$. Thus, we can derive the ADMM iterative updates, starting by forming the augmented Lagrangian of (2.17):

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}) = \|s\|_1 + \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \frac{\rho}{2}\big(&\|\mathbf{OD}^T\mathbf{z} - \mathbf{y} + \mathbf{u}_1\|_2^2 \\ &+ \|\mathbf{D}^T\mathbf{z} - \mathbf{r} + \mathbf{u}_2\|_2^2 + \|\mathbf{z} - \mathbf{s} + \mathbf{u}_3\|_2^2\big) \end{aligned} \tag{2.19}$$

where $\mathbf{u}_1 \in \mathbb{R}^M$, $\mathbf{u}_2, \mathbf{u}_3 \in \mathbb{R}^N$ are scaled versions of the dual variables, and ρ is the augmented Lagrangian parameter. We solve for $\mathbf{z}$, $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{u}$ by minimizing $\mathcal{L}_\rho$ in terms of one variable while fixing the rest in an alternating optimization fashion. We let $\mathbf{z}$ be the first block update, $(\mathbf{s}, \mathbf{r})$ is the second block update, and $\mathbf{u}$ is the third block update. Note that $\mathbf{s}$ and $\mathbf{r}$ can be updated independently since they do not appear together in one term in $\mathcal{L}_\rho$ (hence the parenthesis below). The solution to each block update is provided by solving the following optimization subproblems

$$\mathbf{z} \leftarrow arg \min_{\mathbf{z}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \tag{2.20a}$$

$$\begin{cases} \mathbf{s} \leftarrow arg \min_{\mathbf{s}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \\ \mathbf{r} \leftarrow arg \min_{\mathbf{r}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \end{cases} \tag{2.20b}$$

$$\mathbf{u} \leftarrow \mathbf{u} + (\mathbf{Bz} - \mathbf{c}) \tag{2.20c}$$

where the solution to $\mathbf{z}$ is the closed form solution of least squares via Moore-Penrose pseudo-inverse. The solutions to $\mathbf{s}$, and $\mathbf{r}$ are cases of the so-called *proximity operator* (see [48] for details), where $\mathbf{s}$ is solved using the *soft-thresholding* and the solution to $\mathbf{r}$ boils down to non-negative projection $(.)_+$ by zeroing out the negative values. Algorithm 3.1 states all these updates using linear algebraic notations. $\square$

Algorithm 2.1 : HOMERUN-N (2.16)

```
Initialization: set $k = 1$ and $\mathbf{z}^k$, $\mathbf{s}^k$, $\mathbf{r}^k$, and $\mathbf{u}^k$ to all zero vectors; $\mathbf{b}^k$ as defined in (2.18);
compute the pseudo-inverse of B and save it (i.e., $\mathbf{R} = \mathbf{B}^\dagger$)
Repeat
    • $\mathbf{z}^{k+1} = \mathbf{R}(\mathbf{b}^k - \mathbf{u}^k)$
    • $\mathbf{s}^{k+1} = (\mathbf{z}^{k+1} + \mathbf{u}_3^k - 1/\rho)_+ - (-\mathbf{z}^{k+1} - \mathbf{u}_3^k - 1/\rho)_+$
    • $\mathbf{r}^{k+1} = (\mathbf{D}^T\mathbf{z}^{k+1} + \mathbf{u}_2^k)_+$
    • update $\mathbf{b}^{k+1}$ as defined in (2.18) using $\mathbf{s}^{k+1}$ and $\mathbf{r}^{k+1}$
    • $\mathbf{u}^{k+1} = \mathbf{u}^k + (\mathbf{B}\mathbf{z}^{k+1} - \mathbf{b}^{k+1})$
    • Set $k := k + 1$.
Until maximum number of iterations K is reached (K = 3000)
```

Since the same pseudo-inverse ($\mathbf{B}^\dagger$) is used throughout the iterations in Algorithm 3.1, we compute it once in the initialization step and cache it in a variable we call $\mathbf{R}$ to save computation. After $\mathbf{s}$ is obtained, the approximate solution of the target sequence is $\mathbf{x}_{\text{HOMERUN-N}} = \mathbf{D}^T\mathbf{s}$.

2.3.3 HOMERUN: The Final version

The main idea here is to exploit the domain knowledge of smoothness as for most cases the solution sequence $\mathbf{x} = \mathbf{D}^T\mathbf{s}$ should be smooth. Thus, we penalize the large differences between adjacent timeticks by adding the smoothness penalty to HOMERUN-N, resulting in the formulation of HOMERUN as follows:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{s}\|_2^2 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{s} \geq \mathbf{0} \end{aligned} \tag{2.21}$$

where $\mathbf{H} \in \mathbb{R}^{(N-1)\times N}$ is a smoothness matrix with the n^{th} row has -1 and 1 in the n^{th} and $(n+1)^{th}$ columns, respectively. One could add a regularization (weighting) parameter λ with the smoothness penalty above, however, we observe that $\lambda = 1$ gives optimal or near-optimal performance.

Although both HOMERUN and H-FUSE in [69] have the same regularizer (smoothness constraint), their approach to the problem is very different (i.e., same domain knowledge constraint is added to different optimization cost functions). Again, the approach in [69] penalizes the under-determined LS system and solve for the sequence using the closed form solution. HOMERUN solves for $\mathbf{s}$, the sparse DCT representation of the sequence using the $L1$ norm.

We propose to solve Equation (2.21) in a similar manner as HOMERUN-N model in the previous section. Thus, we reformulate (2.21) as follows:

$$\begin{aligned} \min_{\mathbf{r},\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \|\mathbf{s}\|_1 + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{z}\|_2^2 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{z} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{z} = \mathbf{r}, \quad \mathbf{z} = \mathbf{s} \end{aligned} \tag{2.22}$$

where $\mathbf{r}, \mathbf{z} \in \mathbb{R}^N$ are auxiliary variables. The augmented Lagrangian of (2.22) is:

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{r},\mathbf{s},\mathbf{z},\mathbf{u}) = & \|s\|_1 + \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{z}\|_2^2 \\ & + \frac{\rho}{2}\big(\|\mathbf{O}\mathbf{D}^T\mathbf{z} - \mathbf{y} + \mathbf{u}_1\|_2^2 \\ & + \|\mathbf{D}^T\mathbf{z} - \mathbf{r} + \mathbf{u}_2\|_2^2 + \|\mathbf{z} - \mathbf{s} + \mathbf{u}_3\|_2^2\big) \end{aligned} \tag{2.23}$$

For the same reason as λ, we set the augmented Lagrangian parameter to $\rho = 1$ as it gives optimal or near-optimal performance, resulting in a parameter-free model. Recall the variables defined in (2.18), and similarly we define:

$$\mathbf{Q} := \begin{bmatrix} \mathbf{O}\mathbf{D}^T \\ \mathbf{D}^T \\ \mathbf{I} \\ \mathbf{H}\mathbf{D}^T \end{bmatrix}; \quad \mathbf{q} := \begin{bmatrix} \mathbf{y} \\ \mathbf{r} \\ \mathbf{s} \\ \mathbf{0} \end{bmatrix}; \quad \mathbf{v} := \begin{bmatrix} \mathbf{u} \\ \mathbf{0} \end{bmatrix}, \tag{2.24}$$

where $\mathbf{0} \in \mathbb{R}^{N-1}$ is a vector of all zeros. Similarly, we solve for $\mathbf{z}$, $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{u}$ by minimizing

$\mathcal{L}_\rho$ in (2.23) in terms of one variable while fixing the rest. We describe the implementation of HomeRun in what follows.

Direct Implementation of HomeRun

In Algorithm 2.2, we present the iterative updates that solve the optimization problem of HomeRun with *direct* implementation. These steps provide a convenient way to understand HomeRun using simple vector and matrix operations. Similarly, after we obtain $\mathbf{s}$, the approximate solution of the target sequence is $\mathbf{x}_{\text{HomeRun}} = \mathbf{D}^T\mathbf{s}$.

Algorithm 2.2 : HomeRun (2.21) (direct implementation)

```
Initialization: set $k = 1$ and $\mathbf{z}^k$, $\mathbf{s}^k$, $\mathbf{r}^k$, and $\mathbf{u}^k$ to all zero vectors; $\mathbf{b}^k$ as defined in (2.18); $\mathbf{q}^k$, $\mathbf{v}^k$ as defined in (2.24); compute the pseudo-inverse of $\mathbf{Q}$ and save it (i.e., $\mathbf{W} = \mathbf{Q}^\dagger$)
Repeat
    • $\mathbf{z}^{k+1} = \mathbf{W}(\mathbf{q}^k - \mathbf{v}^k)$
    • $\mathbf{s}^{k+1} = (\mathbf{z}^{k+1} + \mathbf{u}_3^k - 1)_+ - (-\mathbf{z}^{k+1} - \mathbf{u}_3^k - 1)_+$
    • $\mathbf{r}^{k+1} = (\mathbf{D}^T\mathbf{z}^{k+1} + \mathbf{u}_2^k)_+$
    • update $\mathbf{b}^{k+1}$ as defined in (2.18) using $\mathbf{s}^{k+1}$ and $\mathbf{r}^{k+1}$
    • update $\mathbf{q}^{k+1}$ as defined in (2.24) using $\mathbf{s}^{k+1}$ and $\mathbf{r}^{k+1}$
    • $\mathbf{u}^{k+1} = \mathbf{u}^k + (\mathbf{B}\mathbf{z}^{k+1} - \mathbf{b}^{k+1})$ ($\mathbf{B}$ as defined in (2.18))
    • update $\mathbf{v}^{k+1}$ as defined in (2.24) using $\mathbf{u}^{k+1}$
    • Set $k := k + 1$.
Until maximum number of iterations K is reached (K=3000)
```

Accelerated Implementation of HomeRun

In this section, we analyze the complexity of Algorithm 2.2, and derive a fast and memory efficient implementation of HomeRun.

The steps in Algorithm 2.2 provide a convenient way to implement HomeRun using simple vector and matrix operations. These steps involve a one time Moore-Penrose pseudo-inverse in the initialization step which have a complexity of $\mathcal{O}(N^3)$, and the iterative updates consist of matrix-vector multiplications, vector additions, and element-wise vector updates with complexity dominated by $\mathcal{O}(nnz(\mathbf{O})N)$, where $nnz(\mathbf{O})$ is the number of non-zero in the observation matrix

$\mathbf{O}$. Implementing these steps directly may be acceptable with small to moderate-size data since the matrix inversion need to be done only once. However, the direct implementation is not recommended for large data sets. Thus, we propose *accelerated* steps of HOMERUN explained in what follows.

Computing the pseudo-inverse of $\mathbf{Q}$ is the most time consuming step, which is used in the update of $\mathbf{z}$. We can state the update of $\mathbf{z}$ as

$$\begin{aligned}
\mathbf{z} &= \mathbf{Q}^{\dagger}(\mathbf{q}-\mathbf{v}) \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T(\mathbf{q}-\mathbf{v}) \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{D}\underbrace{\left(\mathbf{O}^T(\mathbf{y}-\mathbf{u}_1)+(\mathbf{r}-\mathbf{u}_2)+\overbrace{\mathbf{D}^T(\mathbf{s}-\mathbf{u}_3)}^{\mathbf{g}\in\mathbb{R}^N}\right)}_{\mathbf{p}\in\mathbb{R}^N} \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{D}\mathbf{p} \\
&= \left(\mathbf{D}\mathbf{O}^T\mathbf{O}\mathbf{D}^T+\mathbf{D}\mathbf{D}^T+\mathbf{I}+\mathbf{D}\mathbf{H}^T\mathbf{H}\mathbf{D}^T\right)^{-1}\mathbf{D}\mathbf{p} \\
&= \left(\mathbf{D}(\mathbf{O}^T\mathbf{O}+2\mathbf{I}+\mathbf{H}^T\mathbf{H})\mathbf{D}^T\right)^{-1}\mathbf{D}\mathbf{p} \\
&= \mathbf{D}(\underbrace{\mathbf{O}^T\mathbf{O}+2\mathbf{I}+\mathbf{H}^T\mathbf{H}}_{\mathbf{Z}\in\mathbb{R}^{\mathbf{N}\times\mathbf{N}}})^{-1}\underbrace{\mathbf{D}^T\mathbf{D}}_{\mathbf{I}}\mathbf{p} \\
&= \mathbf{D}\mathbf{Z}^{-1}\mathbf{p}
\end{aligned} \tag{2.25}$$

multiplying both side by $\mathbf{D}^T$, we get:

$$\underbrace{\mathbf{D}^T\mathbf{z}}_{\mathbf{t}\in\mathbb{R}^N} = \mathbf{Z}^{-1}\mathbf{p} \tag{2.26}$$

where the second equality in (2.25) is by the pseudo-inverse equation for $\mathbf{Q}$ (since it is tall with linearly independent columns due to $\mathbf{I}$; Eq. (2.24)); the third and fifth equalities follow from the definition of $\mathbf{Q}$, $\mathbf{q}$, and $\mathbf{v}$; the sixth equality is because $\mathbf{D}$ is orthogonal, i.e., $\mathbf{D}\mathbf{D}^T=\mathbf{D}^T\mathbf{D}=\mathbf{I}$; and the seventh equality is because $(\mathbf{ABC})^{-1}=\mathbf{C}^{-1}\mathbf{B}^{-1}\mathbf{A}^{-1}$ and $\mathbf{D}^{-1}=\mathbf{D}^T$.

The goal of the above derivation is to reduce the computational cost of updating $\mathbf{z}$. Note that the left side of (2.26) (called $\mathbf{t}$) is the inverse DCT of $\mathbf{z}$ – denoted as $iDCT(\mathbf{z})$. Fortunately, $\mathbf{Z}$ has a nice structure, it is a banded symmetric positive-definite matrix with bandwidth = $2RD_{max}-1$, where RD_{max} is the duration of the report with the maximum length. Thus, we exploit its structure to efficiently compute the matrix inversion needed to get $\mathbf{t}$ as follows. We use *Cholesky decomposition*, i.e., $\mathbf{Z} = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L}$ is a lower triangular matrix with the same bandwidth as $\mathbf{Z}$. Then, at every iteration, we perform forward substitution and back substitution steps to get $\mathbf{t}$, i.e., $\mathbf{t} = (\mathbf{L}^T)^{-1}\mathbf{L}^{-1}\mathbf{p}$. Moreover, we reduce the complexity by computing the required DCT and iDCT transforms more efficiently using Fast Fourier Transform (FFT) instead of using the matrix $\mathbf{D}$ (Matlab $fft(.)$ function is more efficient than multiplying by $\mathbf{D}$). The steps of the *accelerated* implementation of HOMERUN are presented in Algorithm 2.3.

Lemma 2.1. *For any report setting in the historical disaggregation problem, if $M \leq N$, then the total computational complexity of* HOMERUN *(Algorithm 2.3) is*

$$\mathcal{O}(b^2 N + NlogN) \tag{2.27}$$

where b is the bandwidth of $\mathbf{O}^T\mathbf{O}$ and equal to $2RD_{max} - 1$, and RD_{max} is the duration of the longest report.

Proof. The cost of computing Cholesky decomposition of a banded matrix in the initialization step is $\mathcal{O}(b^2N)$; performing the $iDCT$ in step 1 in Algorithm 2.3 and DCT in step 4 cost $\mathcal{O}(NlogN)$ using FFT; the matrix-vector multiplications in steps 2 and 7 have complexity of $\mathcal{O}(nnz(\mathbf{O}))$, where $nnz(\mathbf{O}) \leq (RD_{max} \times M)$; the rest are vector additions and element-wise updates in $(.)_+$ which are done with cost $\mathcal{O}(N)$.

Thus, the final complexity is:

$$\mathcal{O}(b^2 N + NlogN)$$

The dominant term in the above final cost depends on whatever is larger, $logN$ or b^2, which depends on the maximum report duration RD_{max}. □

In addition to reducing the running time, the resulting algorithmic steps above are very efficient in terms of memory consumption and can handle very large amounts of data as we demonstrate in Section 2.5.3.

2.4 Experimental Design

In this section, we explain the set up of the experiments performed to evaluate the proposed method. Data sets are explained in Section 2.4.1, the baselines and metrics are listed in Section 2.4.2, and Section 2.4.3 contains description of the input settings and configuration.

Algorithm 2.3 : HOMERUN (2.21) (accelerated implementation)

```
Initialization: set $k = 1$ and $\mathbf{g}^k$, $\mathbf{p}^k$, $\mathbf{t}^k$, $\mathbf{z}^k$, $\mathbf{s}^k$, $\mathbf{r}^k$, $\mathbf{u}_1^k$, $\mathbf{u}_2^k$ and $\mathbf{u}_3^k$ to all zero vectors; compute
$\mathbf{L}$ from the Cholesky decomposition of $\mathbf{Z}$
Repeat
    1. $\mathbf{g}^{k+1} = iDCT(\mathbf{s}^k - \mathbf{u}_3^k)$  %using fft in Matlab%
    2. $\mathbf{p}^{k+1} = \mathbf{O}^T(\mathbf{y} - \mathbf{u}_1^k) + (\mathbf{r}^k - \mathbf{u}_2^k) + \mathbf{g}^{k+1}$
    3. $\mathbf{t}^{k+1} = (\mathbf{L}^T)^{-1}\mathbf{L}^{-1}\mathbf{p}^{k+1}$  %Matlab: $t = L'\backslash(L\backslash p)$%
    4. $\mathbf{z}^{k+1} = DCT(\mathbf{t}^{k+1})$  % using fft Matlab%
    5. $\mathbf{s}^{k+1} = (\mathbf{z}^{k+1} + \mathbf{u}_3^k - 1)_+ - (-\mathbf{z}^{k+1} - \mathbf{u}_3^k - 1)_+$
    6. $\mathbf{r}^{k+1} = (\mathbf{t}^{k+1} + \mathbf{u}_2^k)_+$
    7. $\mathbf{u}_1^{k+1} = \mathbf{u}_1^k + \mathbf{O}\mathbf{t}^{k+1} - \mathbf{y}$
    8. $\mathbf{u}_2^{k+1} = \mathbf{u}_2^k + \mathbf{t}^{k+1} - \mathbf{r}^{k+1}$
    9. $\mathbf{u}_3^{k+1} = \mathbf{u}_3^k + \mathbf{z}^{k+1} - \mathbf{s}^{k+1}$
    10. Set $k := k + 1$.
Until maximum number of iterations K is reached (K=3000)
```

2.4.1 Data Sets

In order to study the performance of HOMERUN, we apply it to the data from project Tycho [105], which includes real epidemiological time sequences spanning over more than 100 years. We select the data about measles in NYC to be our main data set for analyzing and evaluating the performance of the three proposed methods. Furthermore, since the effectiveness of the proposed method hinges upon the degree of energy compaction of the DCT of the data (refer to Section 2.2.2), we explore the performance using six more data sets with different behavior. We pick the intervals with the *least missing values* – the particular weeks used for testing for each data set are NYC measles (from Week 51 to Week 450), CA polio from (1659 – 2058), CA rubella (2805 – 3204), CA smallpox(501 – 900), CA mumps (2756 – 3155), CA hepatitis (2653 – 3051), CA pertussis (1649 – 2048). The behavior of the counts of each disease across these weeks is shown in Figure 2.6 (NYC measles and CA hepatitis are shown earlier in Figures 2.2 and 2.3). We can see here that each disease has notably different dynamic, and, as we observed, they have different DCT with different degree of sparsity and energy compaction, which provide us with a rich test to evaluate the performance of our method.

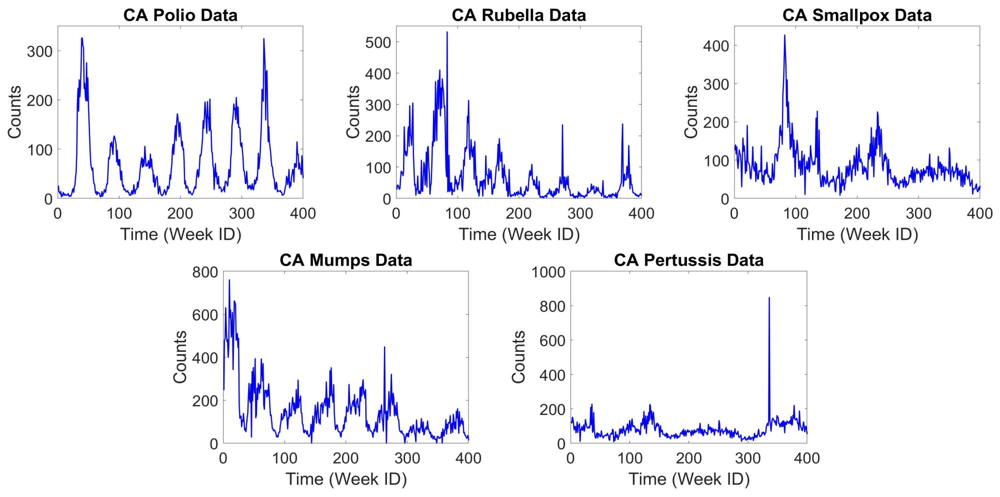


Figure 2.6: Disease Time Sequences

2.4.2 Baselines and Evaluation Metrics

We compare the performance of our method against two baselines, LS in (2.12) and H-FUSE [69], focusing on H-FUSE since it is a state-of-the-art for this problem and has better performance than LS. H-FUSE infuses domain knowledge to improve the reconstruction accuracy by penalizing large differences between adjacent timeticks to promote smoothness. Using our notation, H-FUSE can be written as follows:

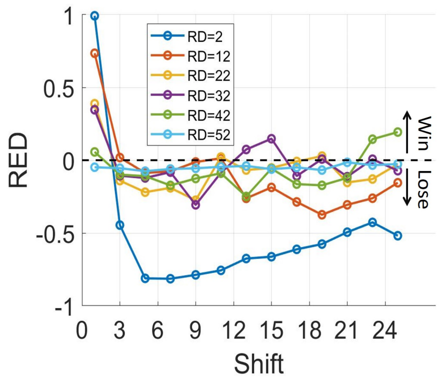
$$\min_{\mathbf{x}} \quad \|\mathbf{y} - \mathbf{O}\mathbf{x}\|_2^2 + \|\mathbf{H}\mathbf{x}\|_2^2 \tag{2.28}$$

where $\mathbf{H}$ is the smoothness matrix defined in Section 2.3.3.

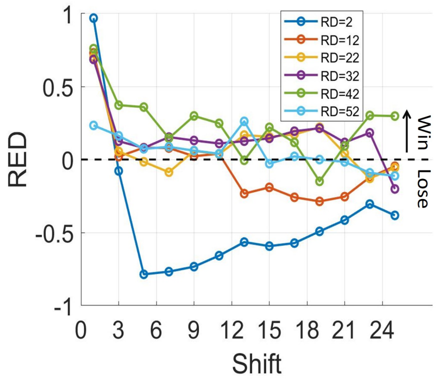
We use the Relative Error Difference (RED) defined below to compare the performance between the proposed and baseline methods.

$$RED = \frac{\text{RMSE}(baseline) - \text{RMSE}(proposed)}{\max(\text{RMSE}(baseline), \text{RMSE}(proposed))} \tag{2.29}$$

We use RED with the result figures in Section 3.5, ranging between -1 and 1. Clearly, positive RED means that the proposed method improves the baseline and vice versa.



(a) HOMERUN-0



(b) HOMERUN-N

Figure 2.7: Comparing HOMERUN-0 and HOMERUN-N versions against the baseline H-FUSE with NYC measles data.

2.4.3 Input Setting

The task is to reconstruct the weekly reports of each disease sequence. We generate different aggregated reports from the weekly counts. Each report covers multiple successive timeticks. As described earlier in Section 2.2.1, the number of weeks included in each observation is called *Report Duration* (RD). The difference between the starting week of two successive reports is the *shift*.

For most experiments, we generate reports with the same RD and $shift$ values and show the results on a wide range (RD spans from 2 to 52 weeks (1 year) with increment of 10 and the $shift$ span from 1 to 25 with increment of 2). Specifically, the first report (y_1) includes the weeks from 1 to RD; y_2 includes weeks from ($shift + 1$) to ($shift + 1 + RD$) and so on – refer to the example in Section 2.2.1. This methodology allows us to study the results for all , e.g., easy cases where RD and $shift$ are both small, more challenging cases where RD or/and $shift$ are large, overlapping reports, and reports with gaps ($RD < shift$). We also show results on experiments where each report covers different number of weeks (different RD) with random starting points, thus they could be overlapped or having gaps.

Table 2.3: RMSE of HOMERUN and the baselines (H-FUSE and LS) using NYC measles data.

	$RD = 2$					$RD = 22$					$RD = 42$				
$shift$	1	7	13	19	25	1	7	13	19	25	1	7	13	19	25
HOMERUN	20.30	**205.96**	**415.79**	**609.03**	**644.38**	**42.49**	**191.10**	**227.03**	**400.07**	894.11	**57.28**	**186.79**	**305.27**	**565.85**	**514.48**
H-FUSE	104.23	215.38	425.56	617.27	653.45	128.38	200.22	303.56	429.02	**891.49**	251.27	231.20	359.86	582.33	595.05
LS	**10.60**	1242.5	1358.4	1402.7	1391.5	75.45	346.49	559.55	654.32	976.30	230.32	371.29	617.42	768.31	762.65

2.5 Results and Analysis

In this section, we present and analyze the experimental results in the following order: 1) **effectiveness** of HOMERUN and its early versions (HOMERUN-0 and HOMERUN-N), 2) **discussion** of the observations and findings about the performance of the proposed methods, and 3) **scalability** to demonstrate how HOMERUN scales in terms of running time with data size.

2.5.1 Effectiveness

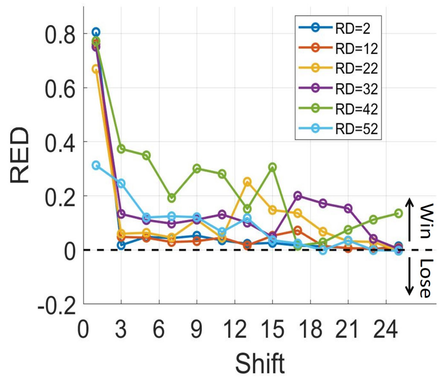
In this section, we show the performance of all three versions of the proposed method in the same order as we explained them in Section 2.3. Presenting results of the earlier versions of HOMERUN demonstrates the benefit of the added constraints (non-negativity and smoothness).

Performance of HOMERUN-0

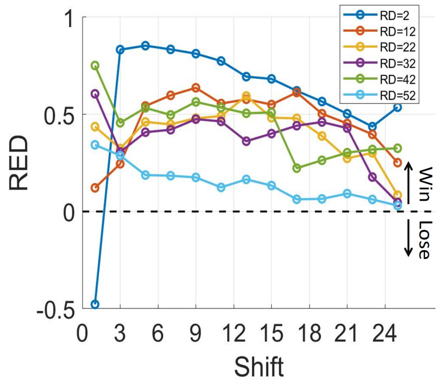
We compare the reconstruction accuracy of HOMERUN-0 with the baseline H-FUSE using NYC measles data in Figure 2.7 (a). We can see that HOMERUN-0 improves the baseline significantly for most RD values, but only with $shift$ smaller than 3. For small RD, H-FUSE works better than HOMERUN-0 with a large difference. This is intuitively understandable as if each report covers only few timeticks (i.e, each report is highly localized), then penalizing the large jumps between two adjacent timeticks is good enough to recover the solution sequence. However, we note that HOMERUN-0 loses to H-FUSE with smaller difference with larger RD values (e.g., $RD = 52$). Larger RD means less available reports, hence the problem is more difficult. The performance of this basic version needs to be improved, which led to deriving the more advanced versions.

Performance of HOMERUN-N

In this approach, we leverage the knowledge that the patient counts in the target sequence is always non-negative. A comparison between HOMERUN-N and the baseline H-FUSE usin g



(a) Against H-FUSE



(b) Against LS

Figure 2.8: HOMERUN wins. Performance of HOMERUN versus the baselines H-FUSE and LS using NYC measles data.

NYC measles data is shown in Figure 2.7 (b). Adding non-negativity constraint to HOMERUN-0 improves the accuracy significantly as it is clear from comparing Fig. 2.7 (a) and (b). HOMERUN-N makes a significant improvement over the baseline for the majority of cases with the following remarks. For $RD = 22, 32, 42, 52$, HOMERUN-N outperforms the baseline except for few outliers. It is therefore clear that more improvement occurs with larger RD values. HOMERUN-N has similar behavior to HOMERUN-0 in the sense that very large improvement happens with small $shift$ for all durations. When $RD = 2$ or 12 and the $shift$ is larger than RD, HOMERUN-N does not improve the baseline, this is consistent with results of HOMERUN-0. Although HOMERUN-N gives encouraging results, we develop the final version of HOMERUN seeking a consistent improvement.

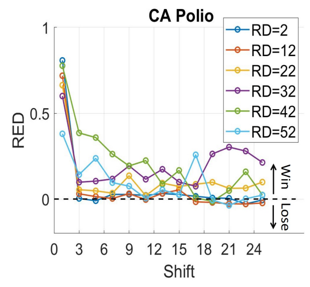
Performance of HOMERUN

In this section, we show the performance of the final version of the proposed method (HOMERUN) using NYC measles *and* the other six data sets described in Section 2.4.1. This model reaps the benefits of both the proposed method and the smooth reconstruction, thus it improves the estimation error of its earlier versions and considerably outperforms the competing baseline methods. Figure 2.8 (a) shows comparison between HOMERUN and the baseline H-FUSE with NYC measles data, HOMERUN is always superior with significant improvement. Note here that generally speaking, greater improvement happens with larger RD, which makes the problem more difficult as we have fewer observations (reports). Figure 2.8 (b) compares HOMERUN with the baseline LS, HOMERUN vastly outperforms LS at all the input settings (RD and $shift$) except for one. The reason of LS working better in this scenario is because each report covers only two weeks ($RD = 2$) and each variable (week), x_n, is involved in two reports since $shift = 1$ (except for the first and last weeks), resulting in an easy problem for LS solution since $\mathbf{O}$ is "almost" square and full rank. To give the reader an idea about the RED versus the actual RMSE of H-FUSE and the baselines (H-FUSE and LS), we present Table 2.3, showing the RMSE at various RD and $shift$ values (a subset of the input settings in Fig. 2.8 (a) and (b), but similar pattern with other RD values) – again, note that the improvement is higher as RD increases.

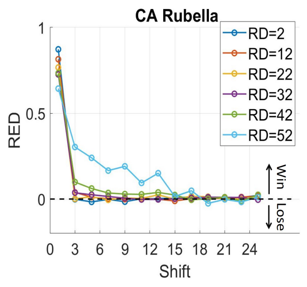
We compare HOMERUN with H-FUSE (since it is the best baseline) using the other six data sets in Figure 2.9. Polio data set has similar results to NYC measles because their time series have similar shapes, thus are similar in the DCT domain. HOMERUN significantly improves the baseline when RD is large with rubella and smallpox data sets. It also shows a large improvement with small $shift$ for all durations with rubella and smallpox. For data sets that do not have very smooth time sequence (mumps, hepatitis, and pertussis), HOMERUN improves the baseline significantly with $shift$ smaller than 3 and performs similar to it when the $shift$ is larger. Table 2.4 highlights the results comparing HOMERUN with the baseline H-FUSE in Fig. 2.8 (a) and Fig. 2.9. It shows the maximum and average improvement across the different input settings (RD and $shift$ values) with all the different data sets using RED percentage ($RED(\%)$). As we observe, the improvement is considerable and may be up to 94%.

Table 2.4: HOMERUN wins consistently. Comparing Performance of HOMERUN and H-FUSE using $RED(\%)$.

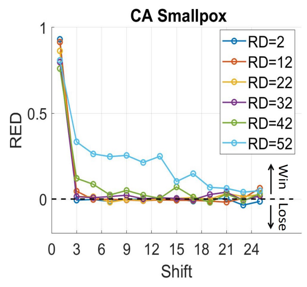
	RED (%)	
Data Name	Maximum	Average
NYC Measle	80.52	13.14
CA Polio	80.77	12.56
CA Rubella	87.20	8.19
CA Smallpox	93.04	9.90
CA Mumps	94.14	5.45
CA Hepatitis	91.25	4.73
CA Pertussis	88.24	4.43



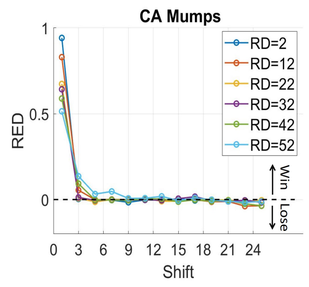
(a) CA Polio



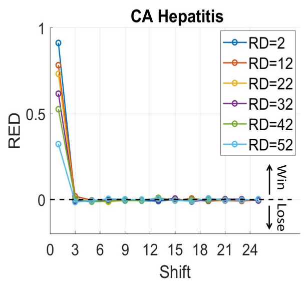
(b) CA Rubella



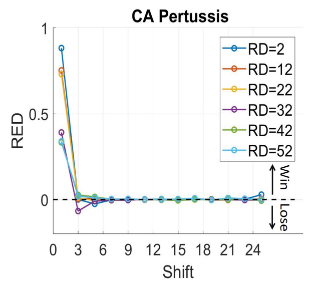
(c) CA Smallpox



(d) CA Mumps



(e) CA Hepatitis



(f) CA Pertussis

Figure 2.9: HOMERUN consistently wins. Performance of HOMERUN vs. the baseline H-FUSE with different data sets (positive=win and negative=lose).

So far, each single point in the result plots (or Table 2.3) corresponds to solving the problem given aggregated reports that have the same duration/resolution (i.e., same RD). In Figure 2.10, we compare the RMSE of HOMERUN and baselines in recovering the weekly counts of NYC measles data, given M reports with RD values randomly drawn from the range $(2-26)$ and the starting week of each report is also random. We test the performance across different M values ranging from 20 to 380 with increment of 10. Since the reports are drawn at random, we repeat each experiment 20 times and take the average RMSE with each data set at a given M. HOMERUN is always better than the baselines with $13\% - 23\%$ improvement depending on the data.

2.5.2 Discussion and Observations

In this section, we provide a higher level discussion about the results, and a detailed analysis of the performance with various input settings (RD and $shift$). We also provide observations about the scope of applicability that can give practitioners insights on when to expect good

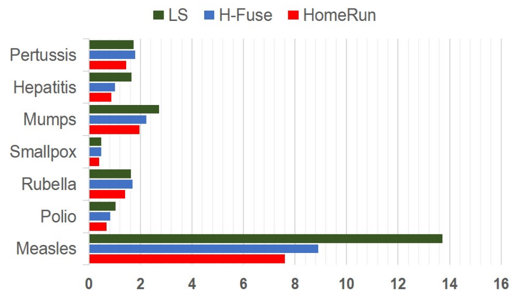


Figure 2.10: HOMERUN wins. RMSE of HOMERUN compared to H-FUSE and LS with reports having different RD.

reconstruction using the proposed method.

Quality of the disaggregation methods in general is affected by the report configurations. Note that as the RD and $shift$ increase, the number of available reports (or equations in the linear system $\mathbf{Ox} = \mathbf{y}$) decreases, resulting in a harder problem. In general, HOMERUN has greater improvement over the baseline H-FUSE as the RD increases as can be observed in Fig. 2.8 (a), Table 2.3, and top three data in Fig. 2.9. Nevertheless, more reports/equations in the system will constraint the solution of $\mathbf{s}$, bringing it closer to the actual DCT coefficients of the target sequence in the proposed method since the optimization problem is constrained by the linear system (Eq. (2.21)). This is also the case for the baselines (H-FUSE and LS), as the number of equations increases, the LS solution becomes closer to the true sequence. We also have the smoothness penalty in HOMERUN and H-FUSE to help bringing the solution closer to the true sequence *if* the sequence is in fact smooth. In both models, as the number of equations decreases (i.e., large $shift$ and/or RD), the degree of freedom of $||\mathbf{s}||_1$ in HOMERUN and $||\mathbf{y} - \mathbf{Ox}||_2^2$ in H-FUSE increases, and thus the quality of the solution relies more on the smoothness penalty. This is especially notable when $shift$ is larger than RD, resulting in gaps between reports which leaves some variables x_n out of the constraints. In that case, the solutions produced by HOMERUN and H-FUSE converge, justifying the similar performance with large $shift$ especially in the bottom three data in Fig. 2.9.

For more analysis, we show the performance of HOMERUN using NYC measles data set with RD spanning from 2 to 52 with increments of 2, and $shift$ ranging from 1 to 25 with increments of 2. In order to explain the comparison more clearly, we map the RED value (Eq.

(2.29)) to the logical value $RED_{logical}$ as follows:

$$RED_{logical} = \begin{cases} 1 & RED > threshold \\ -1 & RED < -threshold \\ 0 & else \end{cases} \tag{2.30}$$

where we empirically set the threshold to 0.015. In Figure 2.11, we show the $RED_{logical}$ across the different input settings. The yellow line shows when $x = y$, which separates reports with overlaps (above the line), i.e., *shift* $<$ *RD*, from reports with gaps (below the line), i.e., *shift* $>$ *RD*. Blue color means HOMERUN improves the baseline H-FUSE, light gray means they perform similarly, and red means the baseline works better. Figure 2.11 shows that HOMERUN never loses to the baseline and the improvement happens in *almost* all the cells in the upper area (overlapped reports), while it is not guaranteed in the lower part. This is because when we have large gaps between the available reports, reconstructing a higher resolution sequence using smoothness constraint may give the smallest error. One of the advantages of HOMERUN is that it reaps the benefits of its own and the baseline H-FUSE.

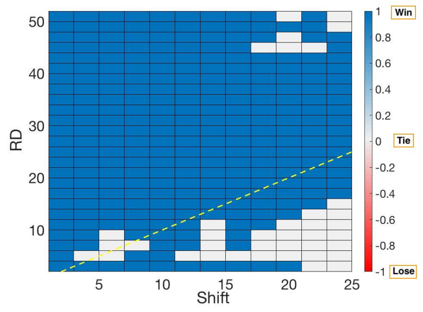


Figure 2.11: HOMERUN almost always wins and sometimes ties. HOMERUN performance against the baseline H-FUSE using NYC measles data.

Applicability of HOMERUN

Quasi-periodicity: if the target sequence is known to have few dominant periodicities, i.e., quasi-periodic (e.g., measles, and polio), then very few cosine functions are needed to approximate it. In other words, this sequence can be accurately represented by few DCT coefficients, i.e., its DCT is very compact and the sequence is very sparse in the DCT domain, thus HOMERUN achieves especially good reconstruction.

The sparser the spectrum of the data, the better the performance of HOMERUN. This is because we optimize for the sparsest spectrum representation of the sequence using $L1$ norm. Empirically, this can be observed by comparing the performance of HOMERUN when applied on data sets with different periodicity behavior in the time domain. When the data set is quasi-periodic, then HOMERUN improves the baseline H-FUSE significantly (e.g., measles in Fig. 2.8 (a) and polio in Fig. 2.9 (a)). With the less periodic sequences (e.g., hepatitis in Fig. 2.9 (e)), the accuracy of HOMERUN drops in comparison to its performance with quasi-periodic sequences, however, it still has a better performance than the baseline H-FUSE (significant improvement with small $shift$).

Smoothness: smoothness penalty has been shown to be effective for time sequences in many applications (e.g., epidemiological data from the Tycho project). Since HOMERUN includes smoothness term, its accuracy increases with data that exhibits higher degree of smoothness. Similarly, this can be seen by comparing the smoothness of measles data (Fig. 2.2) and hepatitis (Fig. 2.3) in their time domain, and their performance (Fig. 2.8 (a) and 2.9 (e)), respectively.

Non-negativity: moreover, if the data is non-negative in its nature (e.g., Tycho data set used in this work), then adding the non-negativity constraint improves the solution and reduces the error, as is clear from comparing the performance of HOMERUN-0 and HOMERUN-N (Fig. 2.7 (a) and (b), respectively).

It is important to point out that the smoothness and non-negativity assumptions are flexible in the proposed framework. For instant, if the target sequence in another application is quasi-periodic but not smooth (e.g., climate data), then HOMERUN-0 (or HOMERUN-N if the sequence is non-negative) can be applied and is expected to perform well.

2.5.3 Scalability

Experiments were performed using Matlab on a Linux server with an Intel Core i7 $-$ 4790 CPU 3.60 GHz processor and 32 GB memory. The accelerated implementation of HOMERUN following Algorithm 2.3 scales linearly with the length of the time sequence (see Figure 2.1 (c)). As clear from the comparison in Figure 2.1 (c), HOMERUN is always faster than H-FUSE. The simple LS method also gets slower than HOMERUN as the sequence size increases, this is due to the pseudo-inverse step in both LS and H-FUSE. We should point out here that the accelerated implementation of HOMERUN dramatically reduces the running time of the direct implementation, while keeping the same accuracy (Algorithm 2.2 versus Algorithm 2.3). Moreover, HOMERUN is very efficient in terms of memory and can handle very large sequences.

2.6 Conclusions

In this chapter, we proposed HOMERUN, a novel algorithm for solving historical data disaggregation problem via DCT-based sparse reconstruction with non-negativity and smoothness constraints. We leverage the ADMM algorithm to solve the resulting optimization problem. We demonstrated that HOMERUN outperforms the baseline methods with real data from the Tycho project. The contributions of this chapter are summarized as follows:

1. **Formulation and Algorithm**: we formulate the data disaggregation problem in the form of Basis Pursuit, add domain knowledge constraints (non-negative and smoothness), and derive the steps of the ADMM algorithm that solve HOMERUN optimization problem.
2. **Effectiveness**: HOMERUN improves the competing baselines and recovers the time sequences with up to 94% improvement in the accuracy of the baseline methods.
3. **Scalability**: We derive accelerated steps of HOMERUN, which scale well and have a complexity of $\mathcal{O}((2RD_{max} - 1)^2N + NlogN)$.
4. **Adaptability**: HOMERUN is parameter-free, and it adapts to the input signal, i.e., it automatically detects the prominent periodicities in the data without the need of assuming any known periodicity.

Chapter 3

Tensor Data Recovery from Multiple Aggregated Views

3.1 Introduction

Data aggregation is the process of summing (or averaging) multiple data samples from a certain dataset, which results in data resolution reduction and compression. The most common type of aggregation is *temporal aggregation*. For example, the annual income is the aggregate of the monthly salary. Aggregation over other attributes is also common, e.g., data get aggregated geographically (e.g., population of New York) or according to a defined affiliation (e.g., employees of Company X). The latter is known in economics as *contemporaneous aggregation* [97]. The different types of aggregation are often combined, e.g., the number of foreigners who visited different US states in 2019 can be aggregated in time, location (states), and affiliation (nationality).

In some cases, it is the data collection process that limits data resolution in the first place, e.g., Store X records item sales only on a monthly basis. Aggregated data also exist for other reasons, the most important being data summarization. In particular, aggregated data enjoy concise representations, which is critical in the era of data deluge. Aggregation also benefits various other purposes, including scalability [106], communication and storage costs [83], and privacy [95]. Aggregated data are common in a wide range of domains, such as economics, health care [81], education [32], wireless communication, signal and image processing, databases [75], energy [5, 6] and smart grid systems [33].

Unfortunately, the favorable properties of data aggregation come with major shortcomings. A plethora of data mining and machine learning tasks strive for data in finer granularity (disaggregated), thus data aggregation is undesirable. Along the same lines, algorithms designed for personalized analysis and accurate prediction significantly benefit from enhanced data resolution. Analysis results can differ substantially when using aggregated versus disaggregated data. Particularly, studies in the field of economics show that data aggregation results in information loss and misleading conclusions at the individual level [23, 38]. Furthermore, in supply chain management, researchers have concluded that aggregating sales over time, products, or locations has a negative impact on demand forecasting [50]. On the other hand, disaggregation prior to analysis is very effective in environmental studies [62], and leads to richer findings in learning analytics [25].

The previous discussion reveals a clear *trade-off* between the need for data aggregation and the benefit of disaggregated data. This has motivated numerous works in developing algorithms for data disaggregation. In general, the task of data disaggregation is an inverse ill-posed problem. In order to handle the problem, classic techniques exploit side information or domain knowledge, in their attempt to make the problem overdetermined and consequently enhance the disaggregation accuracy. Some common prior models, imposed on the target higher resolution data, involve smoothness, periodicity [69], and non-negativity plus sparsity over a given dictionary [10]. Such prior constraints are invoked when no other information is available about the data to be disaggregated.

An interesting question arises when a dataset is aggregated over more that one dimension. This is a popular research problem in the area of business and economics going back to the 70's [22, 89]. In this case temporal *and* contemporaneous aggregated data are available [84]. For instance, given a country consisting of regions, we are interested in estimating the quarterly gross regional product (GRP) values, given the annual GRP per region (temporal aggregate) *and* the quarterly national series (contemporaneous aggregate) [85]. Another notable example appears in healthcare, where data are collected by national, regional, and local government agencies, health and scientific organizations, insurance companies and other entities, and are often aggregated in many dimensions (e.g., temporally, geographically, or by groups of hospitals), often to preserve privacy [81].

Algorithms have been developed to integrate the multiple aggregates in the disaggregation task [22, 27, 84, 85, 89]. The general disaggregation problem is ill-posed, which is clearly

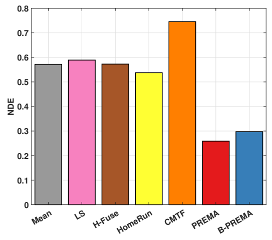


Figure 3.1: PREMA is effective with real data.

undesirable, even with multiple aggregates. Therefore, the majority of these works resort to adopting linear regression models with priors and additional information. However, it is unclear whether these formulations can identify the true disaggregated dataset under reasonable conditions. In this context, identifiability has not received the attention it deserves, likely because guaranteed recovery is considered mission impossible under realistic conditions. With multiview data aggregated in different ways, however, the problem can be well-posed, as we will show in this chapter.

Our work is inspired by the following question: *Is the disaggregation task possible when the data are: 1) multidimensional, and 2) observed by different agencies via diverse aggregation mechanisms?* This is a well motivated problem due to the ubiquitous presence of data with multiple dimensions (three or more), also known as tensors, in a large number of applications. Note that aggregation often happens in more than one dimensions *of the same data* as in the previously explained examples. The informal definition of the problem is given as follows:

Informal Problem 3.1 (Multidimensional Disaggregation)**.**

- **Given:** *two (or more) observations of a multidimensional dataset, each representing a different coarse view of the same data aggregated in one dimension (e.g., temporal and contemporaneous aggregates).*
- **Recover:** *the data in higher resolution (disaggregated) in all the dimensions.*

We propose PREMA: a framework for fusing the multiple aggregates of multidimensional data. The proposed approach represents the target high resolution data as a *tensor*, and models that tensor using the *canonical polyadic decomposition* (CPD) to reduce the number of unknowns, while capturing correlations and higher-order statistical dependencies across dimensions. PREMA employs a coupled CPD approach and estimates the low-rank factors of the target data, to perform the disaggregation task. This way, the originally ill-posed disaggregation problem is transformed to an overdetermined one, by leveraging the uniqueness properties of the CPD. PREMA is flexible in the sense that it can perform the disaggregation task on partially observed data, or data with missing entries. This is practically important as partially observed data appear often in real-world applications.

Our PREMA approach takes into account several well-known challenges that often emerge in real-life databases: the available measurements can have different scales (e.g., mixed monthly and yearly sums), gaps in the timeline (i.e., periods with no value reported), or time overlap (i.e., periods covered by more that one measurement). We also propose a variant of PREMA called B-PREMA that handles the disaggregation task in cases where the aggregation pattern is unknown. The proposed framework not only provides a disaggregation algorithm, but it also gives insights that can be exploited in creating more accurate data summaries for database applications. Interestingly, our work also provides insights on when aggregation *does not* preserve anonymity.

We evaluated PREMA on real data from different domains, i.e., retail sales, crime counts, and weather observations. Experimental results show that the proposed algorithm reduces the disaggregation error of the best baseline by up to 67%. Figure 3.1 shows the Normalized Disaggregation Error (NDE) of PREMA and the baselines with real data of the weekly sales counts of items in different stores of a retail company (CRA dataset, described in Section 5.4.1). We are given two observations: 1) monthly sales aggregates per store, and 2) weekly sales aggregated over groups of stores (94 stores are geographically divided into 18 areas). PREMA outperforms all the competitors, even if the aggregation pattern is unknown (B-PREMA)—all the baselines use the aggregation information. The fact that the naive mean (Mean) gives a large error, indicates that the data are not smooth and the task is difficult.

In summary, the contributions of our work are:

- **Formulation**: we formally define the multidimensional data disaggregation task from multiple views, aggregated across different dimensions, and provide an efficient algorithm.
- **Identifiability:** the considered model can provably transform the original ill-posed disaggregation problem to an identifiable one.
- **Effectiveness:** PREMA recovers data with large improvement over the competing methods on real data.

- **Unknown aggregation:** the proposed model works even when the aggregation mechanism is unknown.
- **Flexibility :** PREMA can disaggregate partially observed data.

Reproducibility: The datasets we use are publicly available; our code is also available online[1].

Preliminary results of part of this work were presented in the *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) 2020* [3]. In this journal version, the problem formulation is generalized to handle aggregated data with missing entries. Although accounting for missing entries makes the problem more complicated, our proposed models and careful algorithmic design yield an algorithmic framework that is efficient and comparable to [3] (which does not handle missing entries), both in terms of accuracy and computational complexity. We also provide identifiability proofs, detailed model and complexity analysis, and conduct extensive experiments.

The rest of the chapter is structured as follows. We explain the needed background and the related work in Section 3.2, and introduce our proposed method in Section 5.3. Then, we explain our experimental setup in Section 5.4 and show the experimental results in Section 3.5. Finally, we summarize conclusions and take-home points in Section 3.6.

3.2 Background & Related Work

In this section, we review some tensor algebraic tools utilized by the proposed framework, define the disaggregation problem, and provide an overview of the related work.

3.2.1 Tensor Algebra

Tensors are multidimensional arrays indexed by three or more indices, $(i, j, k, ...)$. A third-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ consists of three modes: rows $\underline{\mathbf{X}}(:, j, k)$, columns $\underline{\mathbf{X}}(i, :, k)$, and fibers $\underline{\mathbf{X}}(i, j, :)$. Moreover, $\underline{\mathbf{X}}(i, :, :)$, $\underline{\mathbf{X}}(:, j, :)$, and $\underline{\mathbf{X}}(:, :, k)$ denote the i^{th} horizontal, j^{th} lateral, and k^{th} frontal slabs of $\underline{\mathbf{X}}$, respectively.

Tensor decomposition (CPD/PARAFAC): The outer product of two vectors $(\mathbf{a} \circ \mathbf{b})$ results in a rank-one matrix. A rank-one third-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ is an outer product of three vectors: $\underline{\mathbf{X}}(i, j, k) = \mathbf{a}(i)\mathbf{b}(j)\mathbf{c}(k),\ \forall i \in \{1, ..., I\},\ j \in \{1, ..., J\}$, and $k \in \{1, ..., K\}$, i.e.,

[1]Code is available at `https://github.com/FaisalAlmutairi/Prema`

$\underline{\mathbf{X}} = (\mathbf{a} \circ \mathbf{b} \circ \mathbf{c})$, where $\mathbf{a} \in \mathbb{R}^I$, $\mathbf{b} \in \mathbb{R}^J$, and $\mathbf{c} \in \mathbb{R}^K$. The Canonical Polyadic Decomposition (CPD) (also known as PARAFAC) of a third-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ decomposes it into a sum of R rank-one tensors [44], i.e.,

$$\underline{\mathbf{X}} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \tag{3.1}$$

where R is the *tensor rank* and represents the minimum number of outer products needed, and $\mathbf{a}_r \in \mathbb{R}^I$, $\mathbf{b}_r \in \mathbb{R}^J$, and $\mathbf{c}_r \in \mathbb{R}^K$. For brevity, we use $\underline{\mathbf{X}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ to denote the relationship in (3.1). $\mathbf{A} \in \mathbb{R}^{I \times R}$, $\mathbf{B} \in \mathbb{R}^{J \times R}$, and $\mathbf{C} \in \mathbb{R}^{K \times R}$ are the factor matrices with columns $\mathbf{a}_r$, $\mathbf{b}_r$ and $\mathbf{c}_r$ respectively, i.e., $\mathbf{A} = [\mathbf{a}_1 \;\; \mathbf{a}_2 \ldots \mathbf{a}_R]$, and likewise for $\mathbf{B}$ and $\mathbf{C}$.

CPD uniqueness: An important property of the CPD is that $\mathbf{A},\ \mathbf{B},\ \mathbf{C}$ are essentially unique under mild conditions. CPD identifiability is established by the following theorem:

Theorem 3.1. *[21] Let $\underline{\mathbf{X}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ with $\mathbf{A} : I \times R$, $\mathbf{B} : J \times R$, and $\mathbf{C} : K \times R$. Assume $I \geq J \geq K$ without loss of generality. If $R \leq \frac{1}{16}JK$, then the decomposition of $\underline{\mathbf{X}}$ in terms of $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$ is essentially unique, almost surely – i.e., for almost every ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$) except for a set of Lebesgue measure zero.*

Essential uniqueness means that $\mathbf{A},\ \mathbf{B},\ \mathbf{C}$ are unique up to common column permutation and scaling (scaling a column of one matrix that is compensated by counter-scaling the corresponding column of another matrix).

The CPD is also essentially unique, even if the tensor is incomplete (has missing entries). Several results have established CPD identifiability of tensors with missing entries, considering fiber sampled [99], regularly sampled [52] or randomly sampled tensors [13]. The conditions for uniqueness are in general stricter compared to the case where the full tensor is available.

Tensor matricization (unfolding): There are three different ways to unfold (obtain a matrix view of) a third-order tensor $\underline{\mathbf{X}}$ of size $I \times J \times K$. First, the mode-3 unfolding is obtained by the vectorization and parallel stacking of the frontal slabs $\underline{\mathbf{X}}(:,:,k)$ as follows [96]

$$\mathbf{X}_3 = [\mathrm{vec}(\underline{\mathbf{X}}(:,:,1)), ..., \mathrm{vec}(\underline{\mathbf{X}}(:,:,K))] \quad \in \mathbb{R}^{IJ \times K}. \tag{3.2}$$

Equivalently, we can express $\mathbf{X}_3$ using the CPD factor matrices as $\mathbf{X}_3 = (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T$. In the

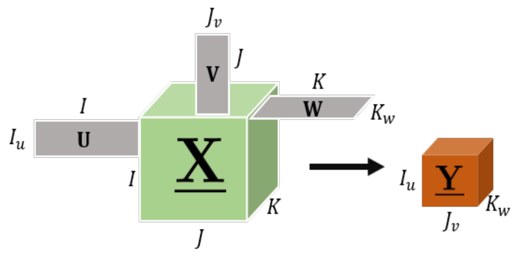


Figure 3.2: Illustration of mode product with $(I_u < I)$, $(J_v < J)$, and $(K_w < K)$.

same vein, we may consider horizontal slabs to express the matricization over the first mode

$$\begin{aligned}\mathbf{X}_1 &:= [\text{vec}(\underline{\mathbf{X}}(1,:,:)), ..., \text{vec}(\underline{\mathbf{X}}(I,:,:))] \\ &= (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T \in \mathbb{R}^{JK \times I}\end{aligned} \tag{3.3}$$

or lateral slabs to obtain mode-2 unfolding

$$\begin{aligned}\mathbf{X}_2 &:= [\text{vec}(\underline{\mathbf{X}}(:,2,:)), ..., \text{vec}(\underline{\mathbf{X}}(:,J,:))] \\ &= (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T \in \mathbb{R}^{IK \times J}.\end{aligned} \tag{3.4}$$

Mode product: It is the operation of multiplying a tensor by a matrix in one particular mode, e.g., mode-1 product of matrix $\mathbf{U} \in \mathbb{R}^{I_u \times I}$ and tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ corresponds to multiplying every column $\underline{\mathbf{X}}(i,:,k)$ of the tensor by $\mathbf{U}$ [57]. Similarly, mode-2 (mode-3) product corresponds to multiplying every row (fiber) of $\underline{\mathbf{X}}$ by a matrix. Applying mode-1, mode-2, and mode-3 products to a third-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ jointly is represented using the following notation:

$$\underline{\mathbf{Y}} = \underline{\mathbf{X}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W} \quad \in \mathbb{R}^{I_u \times J_v \times K_w} \tag{3.5}$$

where "$\times_n$" denotes the product over the n^{th} mode, $\mathbf{U} \in \mathbb{R}^{I_u \times I}$, $\mathbf{V} \in \mathbb{R}^{J_v \times J}$, and $\mathbf{W} \in \mathbb{R}^{K_w \times K}$. Mode-1 multiplication results in reducing the tensor size in the first dimension *if* $(I_u < I)$, similarly with the other modes; see Fig. 3.2. Moreover, if rows of $\mathbf{U}$ are binary vectors with more than one 1, then each horizontal slab of $\underline{\mathbf{Y}}$ is a sum of horizontal slabs of $\underline{\mathbf{X}}$ that correspond to the 1's in a particular row in $\mathbf{U}$. In the same vein, $\mathbf{V}$ and $\mathbf{W}$ could aggregate the lateral and frontal slabs, respectively. The mode product is also reflected in the CPD of the tensor, i.e., if $\underline{\mathbf{X}}$ in the operation in (3.5) admits $\underline{\mathbf{X}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, then $\underline{\mathbf{Y}} = [\![\mathbf{UA}, \mathbf{VB}, \mathbf{WC}]\!]$.

3.2.2 Disaggregation Problem

The goal of the disaggregation task is to estimate a particular dataset in a higher resolution, given observations in lower resolution. In this subsection we present a high level linear algebraic view of disaggregation. This reveals the challenge of the task, which is the relationship between equations versus unknowns; detailed analysis follows in the next section.

In the disaggregation problem, we are given a set of measurements $\mathbf{y} \in \mathbb{R}^{I_u}$ aggregated over the dataset $\mathbf{x} \in \mathbb{R}^{I}$, and our goal is to find $\mathbf{x}$. This can be cast as a linear inverse problem $\mathbf{y} = \mathbf{U}\mathbf{x}$, where $\mathbf{U} \in \mathbb{R}^{I_u \times I}$ is a 'fat' *aggregation matrix* that relates the measurements to the unknown variables. In this chapter, we consider the case where the target high-resolution data are multidimensional (tensor). Specifically, let $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ be the target high-resolution third-order tensor. In the considered problem, we are given two sets of observations, each aggregated over one or more different dimension(s). This is common when data are reported by different agencies, resulting in multiple views of the same information. The key insight is that the given aggregates can be modeled as mode product(s) of $\underline{\mathbf{X}}$ by an aggregation matrix in a particular mode(s). To see this, consider tensor $\underline{\mathbf{X}} \in \mathbb{R}^{4\times2\times2}$, a simple example of a set of observations aggregated over the first mode can be expressed as

$$\underbrace{\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}}_{\mathbf{U} \in \mathbb{R}^{2\times4}} \times \underbrace{\begin{bmatrix} x_{111} & x_{121} & x_{112} & x_{122} \\ x_{211} & x_{221} & x_{212} & x_{222} \\ x_{311} & x_{321} & x_{312} & x_{322} \\ x_{411} & x_{421} & x_{412} & x_{422} \end{bmatrix}}_{\mathbf{X}_1^T \in \mathbb{R}^{4\times(2\times2)}}$$

$$= \underbrace{\begin{bmatrix} y_{111} & y_{121} & y_{212} & y_{122} \\ y_{211} & y_{221} & y_{212} & y_{222} \end{bmatrix}}_{\mathbf{Y}_1^T \in \mathbb{R}^{2\times(2\times2)}} \tag{3.6}$$

where $\mathbf{X}_1$ and $\mathbf{Y}_1$ are mode-1 unfolding of $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$, respectively. The same idea applies when the aggregation is over the second (third) mode using mode-2 (mode-3) product, respectively. In practical settings, the number of available aggregated measurements is much smaller than the number of variables (i.e., $I_u \ll I$), resulting in an under-determined, ill-posed problem. This is the major challenge of disaggregation, even when more than one set of aggregates are available. An even more challenging case appears when one of the available observation sets is aggregated over more than one mode/dimension simultaneously (e.g., $\underline{\mathbf{Y}} \in \mathbb{R}^{I_u \times J_v \times K}$, where $I_u < I$ and $J_v < J$). For instance, sales are reported for categories rather than individual items *and* over groups of stores. This is a double aggregation over stores and items, and the proposed method can work under such a challenging scenario. Moreover, the aggregated observations might be partially observed (i.e., $\mathbf{Y}_1$ in (3.6) has missing entries). This makes the problem more complicated, however our approach efficiently handles data with missing entries.

3.2.3 Related Work

Data disaggregation and fusion: The problem of data integration and fusion [29, 63] from multiple sources has attracted the attention of several communities, due to the increasing access to all kinds of data, especially in database applications. A very challenging task in data integration, is that of recovering a sequence of events (e.g., time series) from multiple aggregated reports [10, 34, 92, 115]. A common approach is to formulate the problem as linear least squares as in (3.6). In practice, however, the number of available aggregated samples is often significantly smaller than the length of the target series, resulting in an under-determined system of equations. To resolve this, previous algorithms have resorted to Tikhonov-type regularization of the ill-posed problem to impose some domain knowledge constraints, e.g., smoothness and periodicity [69].

Fusing multiple observations aggregated in different dimensions for disaggregation purposes is a well studied task in the field of finance and economics [22, 27, 84, 85, 89]. The considered approaches try to exploit linear relations between the target series in high resolution and the available aggregated measurements. However, this results in an under-determined linear system, even with multiple aggregates. Therefore, the majority of these works assume linear regression models with priors and additional information. Moreover, it is unclear whether the assumed models are identifiable, i.e., the model is not guaranteed to disaggregate the data.

(Coupled) tensor factorization: Time series analysis, for various applications, is moving towards modern high-dimensional methods. For example, matrix and tensor factorization have been used in demand forecasting [119], mining and information extraction from complex time-stamped series [73], and prediction of unknown locations in spatio-temporal data [102].

Data share common dimension(s) in a wide spectrum of applications. In such cases, coupled factorization techniques are commonly used to fuse the information for various objectives. For example, coupled factorization is often employed to integrate contextual information into the

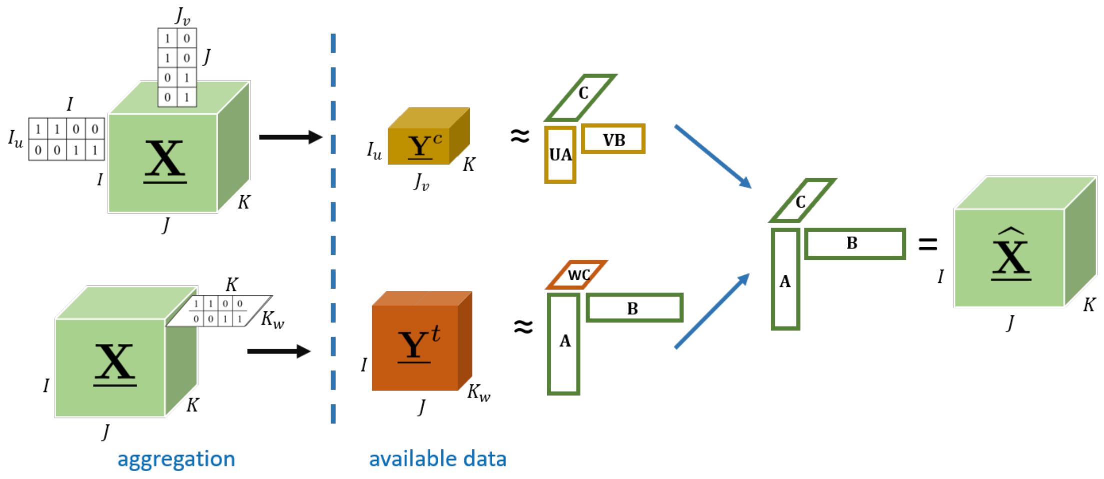


Figure 3.3: Overview of PREMA.

main data [7]. In recommender systems, for instance, we have a (user $\times$ item $\times$ time) tensor *and* a (user $\times$ features) matrix. In this case, the tensor and the features matrix are coupled in the user mode [79]. Coupled tensor factorization has also been proposed for image processing [53], remote sensing [54], and medical imaging problems [52, 55]. Closest to our work is the approach in [51], which employs a coupled CPD to fuse a hyperspectral image with a multispectral image, to produce a high spatial and spectral resolution image. To our knowledge, this work and its conference version [3] are the first that propose a tensor factorization approach to tackle data disaggregation applications.

3.3 Proposed Framework: PREMA

Multidimensional data are indexed by multiple indices, e.g., (i, j, k). Therefore, they can naturally be represented as a tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$. The different modes represent the physical dimensions of the data (e.g., time stamps, locations, items, users). For the sake of simplicity of exposition, we focus on three-dimensional data in our formulations and algorithms. However, the proposed framework can handle more general cases with data of higher dimensions.

In the remainder of this section, we give a detailed description and analysis of PREMA. Particularly, we state the problem and explain the proposed model in high level in Section 3.3.1, formulate PREMA in Section 4.3.1, and present the main algorithm in Section 4.3.4. We discuss the complexity of PREMA in Section 3.3.4, and identifiability in section 3.3.5. Finally, we introduce B-PREMA in Section 3.3.6, to tackle the disaggregation problem in the case where the aggregation matrices are unknown.

3.3.1 Problem & Model Overview

Multidimensional aggregation is common when data are collected or released by different agencies, resulting in multiple views of the same dataset. We will explain the concept with the example of retail sales, which we use in the experiments. Estimating the retail sales in higher resolution enables accurate forecasting of future demand, and planing of economically efficient commerce. There are two sources of data used for this forecasting task: 1) Point of Sale (POS) data at the store-level, commonly aggregated in time (temporal aggregate $\underline{\mathbf{Y}}^t$); and 2) historical orders made to the suppliers by the retailers' Distribution Centers (DC orders), aggregated over their multiple stores (contemporaneous aggregate $\underline{\mathbf{Y}}^c$). In particular, DC order data are immediately available to the suppliers, whereas the POS data are owned by the retailers. Both DC order and POS data are used to forecast demand, and especially POS data are vital in predicting future orders [112]. For that reason, many retailers share POS with their suppliers to assist in forecasting orders and avoid shortage or excess in inventory [49]. In a more restricted scenario, the second source collects information about each category of items rather than each item individually. Oftentimes, data are partially observed, i.e., $\underline{\mathbf{Y}}^t$ and $\underline{\mathbf{Y}}^c$ have missing entries. In this example, not all items are offered in all stores during all the considered time stamps. The question that arises is whether we can fuse these sources to reconstruct high-resolution data in stores, items, and time dimensions.

Formally, we are interested in the following:

Problem 3.1 (Multidimensional Disaggregation).

- **Given:** *two aggregated views of three-dimensional data* $\underline{\mathbf{X}} \in \mathbb{R}^{I\times J\times K}$*:* $\underline{\mathbf{Y}}^t \in \mathbb{R}^{I\times J\times K_w}$*, and* $\underline{\mathbf{Y}}^c \in \mathbb{R}^{I_u\times J\times K}$ *(or* $\underline{\mathbf{Y}}^c \in \mathbb{R}^{I_u\times J_v\times K}$*), with* $I_u < I$*,* $J_v < J$*, and* $K_w < K$*, and possibly missing entries.*
- **Recover:** *the original disaggregated multidimensional data* $\underline{\mathbf{X}} \in \mathbb{R}^{I\times J\times K}$.

Note that each aggregated view is the result of the mode product of the target data with an aggregation matrix. In particular $\underline{\mathbf{Y}}^t = \underline{\mathbf{X}} \times_3 \mathbf{W}$, where $\mathbf{W} \in \mathbb{R}^{K_w\times K}$ is an aggregation matrix with $K_w < K$, and $\underline{\mathbf{Y}}^c = \underline{\mathbf{X}} \times_1 \mathbf{U}$, where $\mathbf{U} \in \mathbb{R}^{I_u \times I}$ is an aggregation matrix with $I_u < I$. In the case where one view is jointly aggregated in 2 dimensions, e.g., sales are aggregated over groups of stores and groups of items, $\underline{\mathbf{Y}}^c = \underline{\mathbf{X}} \times_1 \mathbf{U} \times_2 \mathbf{V}$, where $\mathbf{V} \in \mathbb{R}^{J_v \times J}$ is an aggregation matrix with $J_v < J$.

PREMA aims to fuse the different available aggregates in order to estimate the multidimensional data in the desired higher resolution. At a higher level, the main idea behind the proposed method is that the target multidimensional data, $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$, admits a CPD model. Therefore, it can be well approximated using its CPD factors $\mathbf{A}, \mathbf{B}, \mathbf{C}$ (i.e., $\underline{\mathbf{X}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$). Exploiting the low-rank modeling helps in reducing the number of unknown variables, especially if the data are highly correlated. Then, the CPD factors of the two aggregated observations are

$$\underline{\mathbf{Y}}^t = [\![\mathbf{A}, \mathbf{B}, \mathbf{W}\mathbf{C}]\!], \tag{3.7}$$

$$\underline{\mathbf{Y}}^c = [\![\mathbf{U}\mathbf{A}, \mathbf{V}\mathbf{B}, \mathbf{C}]\!]. \tag{3.8}$$

PREMA learns the factor matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ by applying a coupled CPD model on the available aggregates with respect to the available observations. Note that up to this point, we have not explained how missing entries in $\underline{\mathbf{Y}}^t$ and $\underline{\mathbf{Y}}^c$ are treated, which will be discussed in the next section. Figure 3.3 illustrates the high level picture of our model.

3.3.2 PREMA: Formulation

If we have the original (disaggregated) data in the tensor $\underline{\mathbf{X}}$ with missing entries, a common way to estimate its CPD factors is by adopting a least squares criterion to minimize the difference between the original tensor $\underline{\mathbf{X}}$ and its CPD $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ with respect to the available (observed) entries. This can be done by adding a weight tensor that masks the available entries, i.e.,

$$\underset{\mathbf{A},\mathbf{B},\mathbf{C}}{\text{minimize}} \quad \|\underline{\mathbf{\Omega}} \circledast (\underline{\mathbf{X}} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!])\|_F^2 \tag{3.9}$$

where $\underline{\mathbf{\Omega}}$ is defined as

$$\underline{\mathbf{\Omega}}(i,j,k) = \begin{cases} 1, & \text{if } \underline{\mathbf{X}}(i,j,k) \text{ is available} \\ 0, & \text{otherwise.} \end{cases} \tag{3.10}$$

Fortunately, many real life data exhibit low-rankness due to the correlation between the elements within each dimension (e.g., stores, items, time stamps), i.e., R in (3.1) is small relative to the size of the tensor.

In the considered disaggregation task, we only have aggregated views of the multidimensional data (i.e., compressed version of the target tensor $\underline{\mathbf{X}}$). These aggregated views can have missing elements for various application-specific reasons such as privacy, lack of data collection, or absence of events. We use the fact that the aggregated tensors share the same factors (up to aggregation) as shown in equations (3.7) and (3.8) to jointly decompose $\underline{\mathbf{Y}}^t$ and $\underline{\mathbf{Y}}^c$ by means of coupled tensor factorization. To this end, we obtain the following formulation:

$$\begin{aligned} \underset{\mathbf{A},\mathbf{B},\mathbf{C}}{\text{minimize}} \quad \mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) := \|\underline{\mathbf{\Omega}}^t \circledast (\underline{\mathbf{Y}}^t - ([\![\mathbf{A},\mathbf{B},\mathbf{W}\mathbf{C}]\!])\|_F^2 \\ + \|\underline{\mathbf{\Omega}}^c \circledast (\underline{\mathbf{Y}}^c - ([\![\mathbf{U}\mathbf{A},\mathbf{V}\mathbf{B},\mathbf{C}]\!])\|_F^2 \end{aligned} \tag{3.11}$$

where $\underline{\mathbf{\Omega}}^t \in \{0,1\}^{I\times J\times K_w}$ and $\underline{\mathbf{\Omega}}^c \in \{0,1\}^{I_u\times J_v\times K}$ are weight tensors with ones at the indices of the available entries in $\underline{\mathbf{Y}}^t$ and $\underline{\mathbf{Y}}^c$, respectively, and zeros elsewhere. As a result, the CPD factors $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are learned with respect to the available data. One could add a regularization parameter λ to control the weight between the two terms, however, we observed that it does not significantly affect the disaggregation performance. Enforcing non-negativity constraints on the factors seems natural *if* we are dealing with count data, however, we empirically observed that it does not improve the disaggregation accuracy. Note that if we have additional aggregated observations, we can incorporate them using the same concept. Although (3.11) assumes that the tensors are three-dimensional, we can handle higher-dimensional data following the same idea of coupling factors and mode product over any aggregated mode by the respective aggregation matrix. For example, assume that the data are four-dimensional and we observe an additional tensor $\underline{\mathbf{Y}}^a = \underline{\mathbf{X}} \times_4 \mathbf{L}$, where $\mathbf{L}$ is an aggregation matrix. Then, we add a fourth factor matrix $\mathbf{D}$ to the factorization terms in (3.11) (i.e., the first term becomes $\underline{\mathbf{Y}}^t = [\![\mathbf{A},\mathbf{B},\mathbf{W}\mathbf{C},\mathbf{D}]\!]$). In this case, we also add a term that minimizes the squared error in $\underline{\mathbf{Y}}^a = [\![\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{L}\mathbf{D}]\!]$.

3.3.3 PREMA: Algorithm

The optimization in (3.11) is non-convex, and NP-hard in general. To tackle it, we derive a *Block Coordinate Descent* (BCD) algorithm that updates the three variables in an alternating fashion. Starting from initial factors $\mathbf{A}^{(0)}$, $\mathbf{B}^{(0)}$, and $\mathbf{C}^{(0)}$, at every iteration $k \in \mathbb{N}$, we cyclically update each factor while fixing the other two. Each update is a step in the direction of the negative gradient of $\mathcal{F}$ with respect to the corresponding factor. To simplify the expressions, let us define $\widetilde{\mathbf{A}} = \mathbf{UA}$, $\widetilde{\mathbf{B}} = \mathbf{VB}$, and $\widetilde{\mathbf{C}} = \mathbf{WC}$. The partial derivative of the above objective function $\mathcal{F}$ w.r.t. $\mathbf{A}$ is as follows—the derivations are deferred to Appendix A.1.

$$\begin{aligned}\frac{\partial \mathcal{F}}{\partial \mathbf{A}} = \nabla_{\mathbf{A}} \mathcal{F} = 2\big(\underbrace{\mathbf{\Omega}_1^t \circledast ((\widetilde{\mathbf{C}} \odot \mathbf{B})\mathbf{A}^T - \mathbf{Y}_1^t)}_{\mathbf{E}_t}\big)^T \big(\widetilde{\mathbf{C}} \odot \mathbf{B}\big) \\ + 2\mathbf{U}^T\big(\underbrace{\mathbf{\Omega}_1^c \circledast ((\mathbf{C} \odot \widetilde{\mathbf{B}})\widetilde{\mathbf{A}}^T - \mathbf{Y}_1^c)}_{\mathbf{E}_c}\big)^T \big(\mathbf{C} \odot \widetilde{\mathbf{B}}\big)\end{aligned} \tag{3.12}$$

where $\mathbf{Y}_1^t$, $\mathbf{Y}_1^c$, $\mathbf{\Omega}_1^t$, and $\mathbf{\Omega}_1^c$ are mode-1 unfolding of the corresponding tensors. Similarly, we derive the derivatives of $\mathcal{F}$ w.r.t. $\mathbf{B}$ and $\mathbf{C}$ using mode-2 and mode-3 unfoldings of the tensors, respectively, and get the following equations:

$$\begin{aligned}\nabla_{\mathbf{B}} \mathcal{F} = 2\big(\mathbf{\Omega}_2^t \circledast ((\widetilde{\mathbf{C}} \odot \mathbf{A})\mathbf{B}^T - \mathbf{Y}_2^t)\big)^T \big(\widetilde{\mathbf{C}} \odot \mathbf{A}\big) \\ + 2\mathbf{V}^T\big(\mathbf{\Omega}_2^c \circledast ((\mathbf{C} \odot \widetilde{\mathbf{A}})\widetilde{\mathbf{B}}^T - \mathbf{Y}_2^c)\big)^T \big(\mathbf{C} \odot \widetilde{\mathbf{A}}\big),\end{aligned} \tag{3.13}$$

$$\begin{aligned}\nabla_{\mathbf{C}} \mathcal{F} = 2\mathbf{W}^T\big(\mathbf{\Omega}_3^t \circledast ((\mathbf{B} \odot \mathbf{A})\widetilde{\mathbf{C}}^T - \mathbf{Y}_3^t)\big) \big(\mathbf{B} \odot \mathbf{A}\big) \\ + 2\big(\mathbf{\Omega}_3^t \circledast ((\widetilde{\mathbf{B}} \odot \widetilde{\mathbf{A}})\mathbf{C}^T - \mathbf{Y}_3^c)\big)^T \big(\mathbf{C} \odot \widetilde{\mathbf{A}}\big).\end{aligned} \tag{3.14}$$

In the case of higher-dimensional data, mode-4 unfolding is used to derive the gradient w.r.t. the fourth factor, and so on for more dimensions. With the above gradient expressions at hand, we have established the update direction for each block (factor), which is the negative gradient of $\mathcal{F}$ with respect to each factor:

$$\mathbf{A} = \mathbf{A} - \alpha \nabla_{\mathbf{A}} \mathcal{F}, \tag{3.15}$$

$$\mathbf{B} = \mathbf{B} - \beta \nabla_{\mathbf{B}} \mathcal{F}, \tag{3.16}$$

$$\mathbf{C} = \mathbf{C} - \gamma \nabla_{\mathbf{C}} \mathcal{F}. \tag{3.17}$$

We now seek to select the step-size terms α, β, and γ. We use the *exact line search* approach for this task. At every iteration $k \in \mathbb{N}$, α is chosen to minimize $\mathcal{F}$ along the line $\{\mathbf{A} - \alpha \nabla_{\mathbf{A}} \mathcal{F} | \alpha \geq 0\}$

$$\underset{\alpha \geq 0}{\arg\min} \quad \mathcal{F}\big(\mathbf{A} - \alpha \nabla_{\mathbf{A}} \mathcal{F}\big). \tag{3.18}$$

Luckily, in our case, the above optimization can be solved optimally without extra heavy computations. The optimal solution to (3.18) is as follows (refer to Appendix A.2 for derivations).

$$\alpha = max\big(0, \frac{\mathbf{e}_t^T \mathbf{g}_t + \mathbf{e}_c^T \mathbf{g}_c}{\mathbf{g}_t^T \mathbf{g}_t + \mathbf{g}_c^T \mathbf{g}_c}\big), \tag{3.19}$$

where $\mathbf{e}_t = \text{vec}(\mathbf{E}_t)$, $\mathbf{e}_c = \text{vec}(\mathbf{E}_c)$, with $\mathbf{E}_t$ and $\mathbf{E}_c$ are as defined in (3.12), and

$$\mathbf{g}_t = \text{vec}(\boldsymbol{\Omega}_1^t \circledast ((\widetilde{\mathbf{C}} \odot \mathbf{B})\nabla_{\mathbf{A}}\mathcal{F}^T)), \tag{3.20}$$

$$\mathbf{g}_c = \text{vec}(\boldsymbol{\Omega}_1^c \circledast ((\mathbf{C} \odot \widetilde{\mathbf{B}})(\mathbf{U}\nabla_{\mathbf{A}}\mathcal{F})^T)). \tag{3.21}$$

Note that $\mathbf{E}_t$ and $\mathbf{E}_c$ are already computed in (3.12). We have also computed $(\widetilde{\mathbf{C}} \odot \mathbf{B})$ and $(\mathbf{C} \odot \widetilde{\mathbf{B}})$ in (3.12), which are needed to obtain $\mathbf{g}_t$ amd $\mathbf{g}_c$, respectively. Thus, the exact line search step only requires:

- Multiplying the transpose of the gradient $\nabla_{\mathbf{A}}\mathcal{F} \in \mathbb{R}^{I \times R}$ by a $K_w J \times R$ matrix in (3.20) (and $\mathbf{U}\nabla_{\mathbf{A}}\mathcal{F} \in \mathbb{R}^{I_u \times R}$ by a $KJ_v \times R$ matrix in (3.21)).
- Computing the inner products in (3.19).

In a similar fashion, β and γ are obtained by solving the following optimization functions, respectively:

$$\beta = \underset{\beta \geq 0}{\arg\min} \quad \mathcal{F}\big(\mathbf{B} - \beta\nabla_{\mathbf{B}}\mathcal{F}\big), \tag{3.22}$$

$$\gamma = \underset{\gamma \geq 0}{\arg\min} \quad \mathcal{F}\big(\mathbf{C} - \gamma\nabla_{\mathbf{C}}\mathcal{F}\big). \tag{3.23}$$

The solutions to the above are similar to the case of α, but with mode-2 and mode-3 tensor unfoldings. We provide an illustrative example of deriving the solution to (3.18), (3.22)-(3.23) in Appendix A.2. The overall steps of PREMA are summarized in Algorithm 3.1.

We observed empirically that a careful initialization for the factor matrices in Algorithm 3.1 results in a better disaggregation accuracy, and substantially reduces the operational time (i.e., reduces the required number of iterations). Thus, we design a careful initialization method based on CPD. First, we set the missing entries to zero, then perform CPD on one tensor to get initial estimates of two factors. Then, we solve a system of linear equations using the other tensor to obtain an initial estimate of the third factor. For instance, from CPD($\underline{\mathbf{Y}}^t$) we get $\mathbf{A}$, $\mathbf{B}$, and $\widetilde{\mathbf{C}}$. Then, we obtain $\mathbf{C}$ by solving the linear system $\mathbf{Y}_3^c = \big((\mathbf{VB}) \odot (\mathbf{UA})\big)\mathbf{C}^T$. This

Algorithm 3.1 : PREMA (3.11)

```
input: $\underline{\mathbf{Y}}^t$, $\underline{\mathbf{Y}}^c$, $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$, $R$
Initialize: $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ (refer to Appendix A.3)
Repeat
    • Update $\mathbf{A}$ using (3.15), (3.12), and (3.19)
    • Update $\mathbf{B}$ using (3.16), (3.13), and (3.22)
    • Update $\mathbf{C}$ using (3.17), (3.14), and (3.23)
Until criterion is met (max. #iterations)
output: $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$
```

way, we establish an initial guess for $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$. We provide detailed initialization steps in Appendix A.3.

3.3.4 PREMA: Complexity Analysis

The complexity of PREMA is determined by the matrix multiplication operations required to obtain the gradients and the step size terms. The products in the gradient expressions have the dominant computational cost. Therefore, we break down the computational complexity below using the gradient w.r.t. $\mathbf{A}$ in (3.12); the complexity of computing the gradients w.r.t $\mathbf{B}$ and $\mathbf{C}$ are similar. Recall (3.12):

$$\begin{aligned}\nabla_{\mathbf{A}}\mathcal{F} &= 2\big(\underbrace{\boldsymbol{\Omega}_1^t \circledast ((\widetilde{\mathbf{C}} \odot \mathbf{B})\mathbf{A}^T - \mathbf{Y}_1^t)}_{\mathbf{E}_t \in \mathbb{R}^{JK_w \times I}}\big)^T \big(\widetilde{\mathbf{C}} \odot \mathbf{B}\big) \\ &\quad + 2\mathbf{U}^T\big(\underbrace{\boldsymbol{\Omega}_1^c \circledast ((\mathbf{C} \odot \widetilde{\mathbf{B}})\widetilde{\mathbf{A}}^T - \mathbf{Y}_1^c)}_{\mathbf{E}_c \in \mathbb{R}^{J_v K \times I_u}}\big)^T \big(\mathbf{C} \odot \widetilde{\mathbf{B}}\big).\end{aligned}$$

1. Computing the two Khatri-Rao products costs $\mathcal{O}(K_w JR + KJ_v R)$, where R is the rank.
2. The cost of multiplying $(\widetilde{\mathbf{C}} \odot \mathbf{B})$ with $\mathbf{A}^T$, and $(\mathbf{C} \odot \widetilde{\mathbf{B}})$ with $\widetilde{\mathbf{A}}^T$ is $\mathcal{O}(IJK_w R + I_u J_v KR)$.
3. The element-wise products ($\circledast$) cost $\mathcal{O}(nnz(\underline{\boldsymbol{\Omega}}^t) + nnz(\underline{\boldsymbol{\Omega}}^c))$.
4. Multiplying $\mathbf{E}_t^T$ and $\mathbf{E}_c^T$ with the Khatri-Rao products costs $\mathcal{O}(R(nnz(\underline{\boldsymbol{\Omega}}^t) + nnz(\underline{\boldsymbol{\Omega}}^c)))$.
5. In the worst case where $\mathbf{U}$ and $\boldsymbol{\Omega}_1^c$ have no zeros, the cost of multiplying $\mathbf{U}^T$ with

$\mathbf{E}_c^T(\mathbf{C} \odot \widetilde{\mathbf{B}})$ is $\mathcal{O}(II_uR)$.

The dominant cost terms are in the 2^{nd} point above. Thus, the overall complexity is $\mathcal{O}(IJK_wR + I_uJ_vKR)$. Since R is usually very small relative to the size of the tensors in real data, the complexity is linear with the size of $\underline{\mathbf{Y}}^t$ and $\underline{\mathbf{Y}}^c$.

3.3.5 PREMA: Identifiability Analysis

After introducing the model and the algorithm, we establish the identifiability of the PREMA model. As mentioned earlier, the multidimensional disaggregation task is an inverse ill-posed problem. Considering a low rank CPD model on the data, results in a tensor disaggregation problem with a unique solution. In other words, the optimal solution of (3.11) is guaranteed to be unique, under mild conditions, and identify the original fine-resolution tensor almost surely. For the sake of simplicity we first assume that $\underline{\mathbf{Y}}^t$ does not have any missing values.

Proposition 3.1. *Let $\underline{\mathbf{X}} \in \mathbb{R}^{I\times J\times K}$ be the target tensor to disaggregate with CPD $\underline{\mathbf{X}} = [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$ of rank R. Also let $\underline{\mathbf{Y}}^t \in \mathbb{R}^{I\times J\times K_w} = \underline{\mathbf{X}} \times_3 \mathbf{W}$ and $\underline{\mathbf{Y}}^c \in \mathbb{R}^{I_u\times J_v\times K} = \underline{\mathbf{\Omega}}^c \circledast (\underline{\mathbf{X}} \times_1 \mathbf{U} \times_2 \mathbf{V})$ be the two aggregated sets of observations. Assume that $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are drawn from some absolutely continuous joint distribution with respect to the Lebesque measure in $\mathbb{R}^{(I\times J\times K)R}$, and that $(\mathbf{A}^\star,\mathbf{B}^\star,\mathbf{C}^\star)$ is an optimal solution to problem* (3.11). *Also assume that the number of observed entries at each frontal slab of $\underline{\mathbf{Y}}^c$ is greater than or equal to R. Then, $\widehat{\underline{\mathbf{X}}} = [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$ disaggregates $\underline{\mathbf{Y}}^t$, $\underline{\mathbf{Y}}^c$ to $\underline{\mathbf{X}}$ almost surely if $R \leq \frac{1}{16}\min\{IJ, IK_w, JK_w, 16I_uJ_v\}$.*

The proof is intuitive and parallels recent results obtained in the hyperspectral imaging literature [51]. **Proof sketch:** We use Theorem 3.1 to claim identifiability of $\underline{\mathbf{Y}}^t$. Then factors $\mathbf{A}$, $\mathbf{B}$ can be identified up to common permutation and scaling. The solution for $\mathbf{C}$ is obtained via solving an overdetermined linear system of equations using $\underline{\mathbf{Y}}^c$. This way permutation and scaling is preserved and the target tensor is recovered as $\underline{\mathbf{X}} = [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$. In the case where $\underline{\mathbf{Y}}^t$ has missing entries, identifiability depends on the pattern of missings. Specifically, the results in [99], [52], [13] can be employed, when the available measurements are fiber, regularly or randomly sampled respectively. The conditions are more restrictive compared to the case of fully observed tensor, but guarantee identifiability of $\mathbf{A}$, $\mathbf{B}$ up to common permutation and scaling. The solution for $\mathbf{C}$ is the same as in the previous case. The detailed proof is presented in Appendix A.4.

3.3.6 B-PREMA: PREMA with Unknown Aggregation

In most practical applications, the aggregation details are known. However, there exist cases with limited knowledge on how the data are aggregated, i.e., we do not know (or have partial knowledge of) $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$. We consider the case where each available view is aggregated in one dimension, and propose the following formulation to get the factors of the disaggregated tensor ($\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$):

$$\begin{aligned}\min_{\mathbf{A},\mathbf{B},\mathbf{C},\widetilde{\mathbf{A}},\widetilde{\mathbf{C}}} \mathcal{L}(\mathbf{A},\mathbf{B},\mathbf{C},\widetilde{\mathbf{A}},\widetilde{\mathbf{C}}) &:= \|\underline{\mathbf{\Omega}}^t \circledast (\underline{\mathbf{Y}}^t - [\![\mathbf{A},\mathbf{B},\widetilde{\mathbf{C}}]\!])\|_F^2 \\ &+ \|\underline{\mathbf{\Omega}}^c \circledast (\underline{\mathbf{Y}}^c - [\![\widetilde{\mathbf{A}},\mathbf{B},\mathbf{C}]\!])\|_F^2 + \mu\mathcal{R}(\mathbf{C},\widetilde{\mathbf{C}})\end{aligned} \tag{3.24}$$

where $\widetilde{\mathbf{A}} = \mathbf{U}\mathbf{A}$, and $\widetilde{\mathbf{C}} = \mathbf{W}\mathbf{C}$ are treated as separate variables since we do not know $\mathbf{U}$ and $\mathbf{W}$, and $\mathcal{R}$ is a regularization function. This problem is more challenging than (3.11) as the number of variables has been increased, with the same number of equations. Another challenge is that there is a scaling ambiguity between the factors of the two tensors *if* we omit the regularization term in (3.24). Scaling and counter-scaling the factors of the tensor $\underline{\mathbf{Y}}^t$ (or $\underline{\mathbf{Y}}^c$) does not change its estimated value, or the value of the cost function in (3.24). For example, scaling $\mathbf{A}$ by a λ, and $\widetilde{\mathbf{C}}$ by $1/\lambda$ does not change the value of $\widehat{\mathbf{Y}}_1^t = (\widetilde{\mathbf{C}} \odot \mathbf{B})\mathbf{A}^T$, and as a result, it gives the same cost value. However, this scaling changes the estimated value of the disaggregated tensor $\widehat{\mathbf{X}}_1 = (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T$. This is because tensor $\underline{\mathbf{X}}$ shares factors with both $\underline{\mathbf{Y}}^t$ and $\underline{\mathbf{Y}}^c$. To overcome this, we observe that the temporal aggregation $\mathbf{W}$ in most aggregated data is non-overlapping and includes all the time ticks[2]. This means that the respective column sums of $\mathbf{C}$ and $\widetilde{\mathbf{C}}$ should be equal. We exploit this observation by choosing the following regularization term for (3.24)

$$\mathcal{R}(\mathbf{C},\widetilde{\mathbf{C}}) = \|\mathbf{1}^T\mathbf{C} - \mathbf{1}^T\widetilde{\mathbf{C}}\|_2^2,$$

which reconciles for the scaling ambiguity.

In order to tackle the problem above, we derive a BCD algorithm, in the same fashion as Algorithm 3.1. The steps are summarized in Algorithm 3.2. We alternate between updating the five variables. In each update, we take a step in the direction of the negative gradient w.r.t. the corresponding variable. The derivations of the gradients are shown in Appendix A. The step size parameters $\alpha, \rho, \beta, \gamma$, and σ are chosen using the exact line search explained in Sec. 4.3.4 above,

[2] *Known* overlap, e.g., 50%, can be treated similarly – as in this case every atom is counted twice.

and Appendix A.2.

To initialize the factors in Algorithm 3.2, we set the missing entries to zero, then we use `Tensorlab` and compute (`CPD`($\underline{\mathbf{Y}}^c$)) to get $\widetilde{\mathbf{A}}$, $\mathbf{B}$, and $\mathbf{C}$. To get an initial estimate of $\widetilde{\mathbf{C}}$, we exploit the fact that the temporal aggregates are the summation over consecutive time stamps in most real data. Therefore, we sum every consecutive $w = \frac{K}{K_W}$ rows in $\mathbf{C}$. This way we approximate the temporal aggregation process in a very intuitive way, the true aggregation matrix being unknown[3].

3.4 Experimental Design

In this section, we provide a detailed description of the setup we use in our experiments. First, we describe the data used in the experiments. Then, we explain the aggregation applied on these data to generate aggregated views. Last, we present the evaluation metrics and baselines used for comparison.

3.4.1 Datasets

We evaluate PREMA using the following public datasets, which are readily available online:

DFF[4]: Retail sales data, called Dominick's Finer Foods (DFF), collected by the James M. Kilts Center, University of Chicago Booth School of Business. DFF used to be a grocery store chain based in the Chicago area until all of its stores were closed. Sales, in this dataset, are divided into category-specific files. In particular, each file contains the weekly sales (i.e., number of sold units) of items belonging to a specific category (e.g., cheese, cookies, soft drinks, etc) in about 100 stores. DFF data contain the geographical locations of the different stores, which we use to aggregate stores into groups. We create ground truth three-dimensional tensors, using 10 different category-specific datasets. This way, a (stores $\times$ items $\times$ weeks) tensor is formed *for each category*. These 10 department-specific datasets are listed as the first group in Table 3.1—we use the three bold letters acronym for these categories in the results. We pick the 50 most popular items from each category. Note that this results in an 'incomplete' tensor, owing to the fact that not all items were offered in all stores, or they were offered only for part of the time in some stores. These tensors have varying statistics (see Table 3.1), which allows thorough

[3]In the experiments, we make sure that the true temporal aggregation and the estimated one do not align.
[4]https://www.chicagobooth.edu/research/kilts/datasets/dominicks

Algorithm 3.2 : B-PREMA

```
input: $\underline{\mathbf{Y}}^t$, $\underline{\mathbf{Y}}^c$, $R$, $\mu$
Initialize: $\widetilde{\mathbf{A}}$, $\mathbf{B}$, $\mathbf{C}$, $\leftarrow$ CPD($\underline{\mathbf{Y}}^c$)
$\widetilde{\mathbf{C}}(k_w, :) \leftarrow \sum_{k=w(k_w-1)+1}^{w \times k_w} \mathbf{C}(k, :)$
$\mathbf{A} \leftarrow$ solve $\mathbf{Y}_3^t = \mathbf{A}(\widetilde{\mathbf{C}} \odot \mathbf{B})^T$
Repeat
    • $\alpha \leftarrow \arg\min_{\alpha \geq 0} \mathcal{L}(\mathbf{A} - \alpha \nabla_{\mathbf{A}} \mathcal{L})$;   $\mathbf{A} = \mathbf{A} - \alpha \nabla_{\mathbf{A}} \mathcal{L}$
    • $\rho \leftarrow \arg\min_{\rho \geq 0} \mathcal{L}(\widetilde{\mathbf{A}} - \rho \nabla_{\widetilde{\mathbf{A}}} \mathcal{L})$;   $\widetilde{\mathbf{A}} = \widetilde{\mathbf{A}} - \rho \nabla_{\widetilde{\mathbf{A}}} \mathcal{L}$
    • $\beta \leftarrow \arg\min_{\beta \geq 0} \mathcal{L}(\mathbf{B} - \beta \nabla_{\mathbf{B}} \mathcal{L})$;   $\mathbf{B} = \mathbf{B} - \beta \nabla_{\mathbf{B}} \mathcal{L}$
    • $\gamma \leftarrow \arg\min_{\gamma \geq 0} \mathcal{L}(\mathbf{C} - \gamma \nabla_{\mathbf{C}} \mathcal{L})$;   $\mathbf{C} = \mathbf{C} - \gamma \nabla_{\mathbf{C}} \mathcal{L}$
    • $\sigma \leftarrow \arg\min_{\sigma \geq 0} \mathcal{L}(\widetilde{\mathbf{C}} - \sigma \nabla_{\widetilde{\mathbf{C}}} \mathcal{L})$;   $\widetilde{\mathbf{C}} = \widetilde{\mathbf{C}} - \sigma \nabla_{\widetilde{\mathbf{C}}} \mathcal{L}$
Until termination criterion is met (max. #iterations)
output: $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$
```

testing and analysis. We also form an additional (stores $\times$ items $\times$ weeks) tensor that contains items from all the 10 different categories combined, 50 items from each (namely **Mixed DFF** in Table 3.1).

Walmart[5]: Historical weekly sales data for 99 different departments in 45 Walmart stores located in different regions. A (stores $\times$ departments $\times$ weeks) tensor is created from these data. The resulting tensor is complete and has no missing entries. The size of each store (in square feet) is included in the data (we use this information to form groups of stores).

Crime[6]: Reported incidents of crimes that occurred in the city of Chicago from 2001 to present. Each incident is marked with its beat (police geographical area), and a code indicating the crime type. There are 304 geographical areas and 388 crime types in total. Using this dataset, we form a (locations (by beat) $\times$ crime types $\times$ months) tensor.

Weather[7]: Daily weather observations from 49 stations in Australia. These observations contain 17 different variables, e.g., min temperature, max temperature, cloud, humidity, wind, etc. We form a (station (location) $\times$ variables $\times$ days) tensor using one year of daily observations.

Table 3.1 summarizes the different datasets described above, with their size, maximum and average values, Standard Deviation (SD), and percentage of missing entries and zeros. These

[5]https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/data
[6]https://www.kaggle.com/chicago/chicago-crime/activity
[7]http://www.bom.gov.au/climate/data/

Table 3.1: Summary of datasets and their statistics.

Dataset ($\underline{\mathbf{X}}$)	**Size**	**Max**	**Avg**	**SD**	**% (missing entries)**	**% (zero entries)**
BATh Soap	$93 \times 50 \times 266$	52	0.79	1.34	44.73	33.37
Bottled **J**ui**C**es	$93 \times 50 \times 393$	12288	13.76	50.08	8.79	9.19
CHEeses	$93 \times 50 \times 393$	18176	26.65	88.29	8.59	5.51
COOkies	$94 \times 50 \times 390$	14080	16.00	56.86	9.81	7.57
CRAckers	$94 \times 50 \times 382$	14080	8.21	29.61	14.21	7.57
Canned **SO**up	$93 \times 50 \times 379$	34494	40.46	133.42	8.64	4.54
Fabric **S**o**F**teners	$93 \times 50 \times 397$	7168	5.68	18.84	18.64	27.48
GROoming	$93 \times 50 \times 272$	232	1.94	2.94	7.66	32.66
Paper **T**o**W**els	$93 \times 50 \times 389$	19712	45.36	117.82	36.72	23.49
Soft **DR**inks	$93 \times 50 \times 391$	18944	48.81	155.09	8.58	11.18
Mixed DFF	$93 \times 500 \times 230$	17610	19.01	71.30	15.30	17.83
Walmart	$45 \times 81 \times 143$	6.93e+05	1.29e+04	2.14e+04	0	19.38
Crime	$304 \times 388 \times 221$	325	0.26	1.47	0	91.56
Weather	$49 \times 17 \times 365$	1038	10.23	95.65	0	93.30

datasets are the ground truth in our experiments, and represented by $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$.

3.4.2 Aggregation Configuration

The aggregated observations (compressed tensors), that are used as inputs to the disaggregation methods, are generated from $\underline{\mathbf{X}}$ following two practical scenarios described below:

Scenario A: The multidimensional data, we aim to disaggregate, are represented by $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$. Instead of the full tensor $\underline{\mathbf{X}}$, we are given two aggregated views: 1) temporally aggregated tensor $\underline{\mathbf{Y}}^t = \underline{\mathbf{X}} \times_3 \mathbf{W}$, i.e., aggregated in the third dimension; and 2) contemporaneously aggregated tensor $\underline{\mathbf{Y}}^c = \underline{\mathbf{X}} \times_1 \mathbf{U}$, aggregated in the first mode (e.g., stores/locations dimension). We use the 10 category-specific datasets from DFF and Walmart data to test this scenario. The stores are aggregated according to their geographical locations in the DFF datasets, and based on their sizes in Walmart data. We also test this scenario on Weather data, where the temporal aggregate represents the weather observations averaged over a course of time, and the contemporaneous aggregate is the average of the observations over a geographical region.

Scenario B: In this scenario, two aggregated views of $\underline{\mathbf{X}}$ are given: 1) similar to the previous scenario, temporally aggregated tensor $\underline{\mathbf{Y}}^t = \underline{\mathbf{X}} \times_3 \mathbf{W}$; and 2) contemporaneously aggregated tensor $\underline{\mathbf{Y}}^c = \underline{\mathbf{X}} \times_1 \mathbf{U} \times_2 \mathbf{V}$, aggregated in the first *and* second dimensions (e.g., sales counts that are *jointly* aggregated over groups of stores *and* groups of items). We use Mixed DFF and Crime data to test this scenario. The stores are aggregated into groups according to their locations in Mixed DFF data, whereas items are aggregated according to their categories. In

Crime data, locations and types are grouped based on the closeness in geographical location and similarity in crime type, respectively. Note that when $\mathbf{V} = \mathbf{I}$, this yields to Scenario A. Evidently, this scenario is more challenging since the second observation is aggregated in two modes, i.e., double aggregation, resulting in fewer measurements.

The difficulty of the problem also depends on the *aggregation level*, i.e., the number of data points (e.g., weeks, items, or stores) in one sum. Fewer aggregated measurements result in more challenging problems from an "equations versus unknowns" standpoint. We test the disaggregation performance using different aggregation levels for each dimension.

3.4.3 Evaluation Baselines & Metrics

We evaluate the disaggregation performance of the proposed method using the Normalized Disaggregation Error (NDE = $\|\underline{\mathbf{X}} - \widehat{\underline{\mathbf{X}}}\|_F^2/\|\underline{\mathbf{X}}\|_F^2$), where $\widehat{\underline{\mathbf{X}}}$ is the estimated data. The baseline methods are described next. Note that we compare to state-of-art approaches in the time series disaggregation literature as well as methods developed to fuse multiple views of multidimensional data, but for different tasks. To the best of our knowledge our work is the first to perform disaggregation on multidimensional data from multiple views.

Mean: This baseline assumes that the constituent data atoms (entries in $\underline{\mathbf{X}}$) have equal contribution in their aggregated samples. The final estimate of Mean is the average of the estimation from the temporal and the contemporaneous aggregates. For example, the contemporaneous aggregate reports 100 units sold in 10 stores in the first week of January, and the temporal one tells us that 80 units were sold in January (4 weeks) in Store 1. Then, Mean estimation of week 1 and store 1 is $(100/10 + 80/4)/2 = 15$

LS: This baseline is inspired by [27, 85], where a least squares criterion is adopted on the linear relationship between the target time series in high resolution and the available aggregates. The resulting linear system is underdetermined, thus, these works assume a linear regression model between the target series and some set of indicators. In their context, indicators are time series available in high resolution that are expected to display similar fluctuations to the target series. For example, the stock price of an oil company is a linear combination of the stock prices of other relevant companies. This assumption requires additional data that are not available in our datasets. Therefore, we resort to the minimum-norm solution

$$\begin{aligned}\min_{\underline{\mathbf{X}}} \quad & \|\text{vec}({\boldsymbol{\Omega}_3^t}^T) \circledast \big(\text{vec}({\mathbf{Y}_3^t}^T) - \widetilde{\mathbf{W}}\text{vec}({\mathbf{X}_3}^T)\big)\|_2^2 \\ & + \|\text{vec}(\boldsymbol{\Omega}_3^c) \circledast \big(\text{vec}(\mathbf{Y}_3^c) - \widetilde{\mathbf{U}}\text{vec}(\mathbf{X}_3)\big)\|_2^2\end{aligned} \tag{3.25}$$

where $\widetilde{\mathbf{W}} = \mathbf{I} \otimes \mathbf{W}$ and $\widetilde{\mathbf{U}} = \mathbf{I} \otimes \mathbf{V} \otimes \mathbf{U}$.

H-Fuse: [69] This baseline constrains the solution to the LS baseline above to be smooth, i.e., it penalizes large differences between adjacent time ticks.

HomeRun: [10] To circumvent the indeterminacy of the linear system in the time series disaggregation problem, this baseline solves for the disaggregated series in the frequency domain. More specifically, HomeRun searches for the coefficients of the Discrete Cosine Transform (DCT) that represent the target high-resolution series. The key point is that the number of non-negligible DCT coefficients of the time series is much smaller than its length. In other words, the DCT is used as a sparsifying dictionary to reduce the number of variables. HomeRun also imposes smoothness and non-negativity constraints.

CMTF: Couple Matricized Tensor Factorization has been widely used, to fuse multiple views of multidimensional data, in the hyperspectral imaging application [98, 118]—the work in [118] adds non-negativity constraints. These images are three-dimensional tensors, and the motivation behind these works is to exploit the low-rankness of the matricized image. We compare to this model because real world multidimensional data are often well-approximated using low-rank, as we will show empirically. Using our notation, CMTF solves

$$\begin{aligned}\min_{\mathbf{A},\mathbf{B}} \quad & \|\boldsymbol{\Omega}_3^t \circledast (\mathbf{Y}_3^t - \mathbf{A}(\mathbf{W}\mathbf{B})^T)\|_F^2 \\ & + \|\boldsymbol{\Omega}_3^c \circledast (\mathbf{Y}_3^c - (\mathbf{V} \otimes \mathbf{U})\mathbf{A}\mathbf{B}^T)\|_F^2.\end{aligned} \tag{3.26}$$

We solve (3.26) using a BCD algorithm with exact line search. Similar to PREMA, a good initialization for the low-rank factors improves the performance of CMTF. To ensure fair comparison, we initialize using SVD with missing entries set to be zeros.

Note that all the baselines described above use the aggregation information; B-PREMA is the only method that disaggregates without using the aggregation matrices. In addition to the above baseline methods, we also test the estimation of the target disaggregated data with the following *oracle* baseline.

Table 3.2: NDE of the proposed methods and the baselines using the 10 category-specific datasets.

Dataset	**BAT**	**BJC**	**CHE**	**COO**	**CRA**	**CSO**	**FSF**	**GRO**	**PTW**	**SDR**
% (missings)	44.73%	8.79%	8.59%	9.81%	14.21%	8.64%	18.64%	7.66%	36.72%	8.58%
SD	1.34	50.08	88.29	56.86	29.61	133.42	18.84	2.94	117.82	155.09
Mean	**0.3284**	0.4441	0.3118	0.3596	0.5217	0.3309	0.5609	0.2464	0.2994	0.2860
LS	0.3328	0.6077	0.4650	0.6224	0.5889	0.4664	0.5982	0.2831	0.4593	0.5420
H-FUSE	0.3411	0.6437	0.4870	0.6414	0.5726	0.4885	0.6451	0.2863	0.4719	0.5644
HomeRun	0.3461	0.6453	0.4818	0.6284	0.5376	0.4856	0.6496	0.2877	0.4662	0.5594
CMTF	0.4254	0.1818	0.1954	0.1783	0.7455	0.1564	0.1930	0.2908	0.2577	0.1633
PREMA, R=10	0.5203	0.1978	0.1756	0.1757	**0.2587**	0.2057	0.2019	0.3198	0.2844	0.2039
PREMA, R=25	0.5079	0.1684	0.1516	0.1371	0.2624	0.1373	0.1790	0.2581	0.2132	0.1438
PREMA, R=40	0.4972	**0.1572**	**0.1491**	**0.1318**	0.2589	**0.1332**	**0.1747**	**0.2458**	**0.1969**	**0.1348**
CPD (oracle), R=10	0.4782	0.0937	0.0723	0.1205	0.0776	0.0776	0.0810	0.2919	0.2356	0.1329
CPD (oracle), R=25	0.4345	0.0586	0.0419	0.0676	0.0518	0.0476	0.0494	0.2448	0.1358	0.0822
CPD (oracle), R=40	0.4109	0.0443	0.0321	0.0532	0.0438	0.0345	0.0399	0.2284	0.1007	0.0605
B-PREMA, R=10	0.5242	0.3012	0.3525	0.2207	0.3080	0.1752	0.2090	0.3156	0.3594	0.2008
B-PREMA, R=25	0.5002	0.3583	0.3553	0.2496	0.2976	0.1756	0.1892	0.2557	0.3758	0.1539
B-PREMA, R=40	0.4914	0.3909	0.3823	0.2942	0.3042	0.1825	0.1846	0.2472	0.3963	0.1620

CPD: We fit a CPD model directly to the ground truth tensor $\underline{\mathbf{X}}$ *with respect to the observed entries*. We use the Matlab-based package `Tensorlab` to compute the CPD. Then, we reconstruct $\widehat{\underline{\mathbf{X}}}$ from the learned factors $(\mathbf{A}, \mathbf{B}, \mathbf{C})$. This baseline can also serve as a lower bound for the error produced by the proposed method PREMA.

3.5 Experimental Results

In this section, we evaluate the performance of PREMA and B-PREMA in terms of disaggregation accuracy using real data. The two aforementioned aggregation scenarios (refer to Section 3.4.2) are considered with different aggregation levels. In the experiments, we choose the rank R for PREMA (and the CPD baseline) based on Proposition 3.1, unless stated otherwise. On the other hand, for CMTF, we perform a grid search and show the results with the best R. We run 10 iterations of the CPD step in the initialization of PREMA in Algorithm 3.1 (or B-PREMA in Algorithm 3.2) using `Tensorlab`, then run 10 iterations of the iterative procedure in the algorithms. We set $\mu = 100$ for B-PREMA in Algorithm 3.2. All experiments were performed using Matlab on a Linux server with an Intel Core i7–4790 CPU 3.60 GHz processor and 32 GB memory.

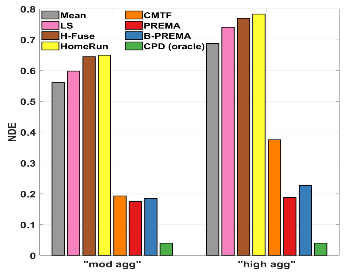
3.5.1 Results on Scenario A

Two aggregated views $\underline{\mathbf{Y}}^t$, $\underline{\mathbf{Y}}^c$ are observed. Table 3.2 shows the disaggregation error in terms of NDE, achieved by the proposed method and the baselines on the 10 category-specific datasets from DFF. The proposed methods, PREMA and B-PREMA, along with the CPD oracle are shown under 3 different ranks ($R = 10,\ R = 25,\ R = 40$). In $\underline{\mathbf{Y}}^t$, the weekly sales counts are observed on a monthly basis, while in $\underline{\mathbf{Y}}^c$, the 93 (or 94 for some categories) stores are clustered geographically into 18 areas. This means that the measurements in the temporal aggregate $\underline{\mathbf{Y}}^t$ are about 25% of the original size, and the number of the contemporaneously aggregated measurements in $\underline{\mathbf{Y}}^c$ is only 19.35% of the disaggregated data size.

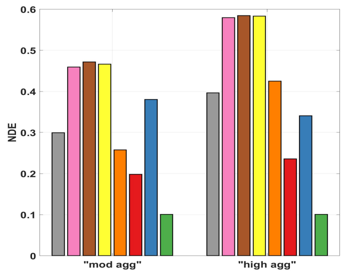
For all datasets in Table 3.2, except BAT, PREMA markedly outperforms the baselines—to highlight the improvement, we make the smallest error in bold and underline the second smallest. The naive mean (Mean) is good enough with BAT dataset because it is smooth (SD = 1.34) and has the largest percentage of missing entries, compared to the other datasets. The time series methods, H-Fuse and HomeRun, do not perform well with these datasets because they are designed for smooth and quasi-periodic data, respectively. To provide an example, we noticed that HomeRun improves the error of LS and H-Fuse baselines with CRA data, and found that CRA exhibit more periodicity compared to the rest of the categories. Comparing PREMA with CPD, we see that PREMA achieves error very close to CPD of the ground truth data with the same rank, e.g., with GRO, PTW, and SDR datasets. By looking at the performance of B-PREMA in the table, we can see that the proposed algorithm works remarkably well when the aggregation matrices are unknown. For example, with GRO data and $R = 40$, the NDE of B-PREMA is 0.2472, while NDE $= 0.2284$ with CPD. B-PREMA disaggregates with smaller, or very similar, error compared to the baselines that uses the aggregation pattern information—see results with CRA, FSF, GRO, and SDR datasets. With all datasets, there is always a wide range of R under which the proposed algorithm works similarly well.

Next, we examine the performance when we change the level of aggregation from moderate ("mod agg") to very high ("high agg"). The disaggregation error is shown with two datasets from DFF data, FSF and PTW, in Figure 4.1, and with Walmart and Weather datasets in Figure 3.5.

The aggregation levels in Figure 4.1 are: 1) monthly basis measurements (every 4 weeks) in $\underline{\mathbf{Y}}^t$, and the 93 stores are divided geographically into 18 areas ("mod agg"); and 2) quarterly samples (every 12 weeks) in $\underline{\mathbf{Y}}^t$, and the stores are divided into only 9 areas ("high agg"). The rank R for PREMA, B-PREMA, and CPD is set to 40 in this figure. By comparing the

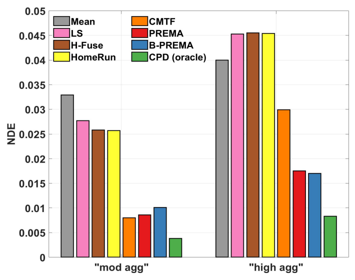


(a) FSF dataset

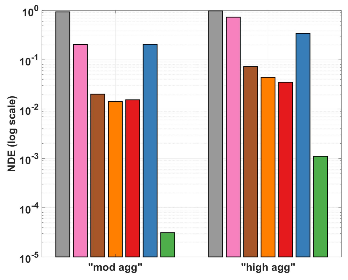


(b) PTW dataset

Figure 3.4: PREMA works well with extreme aggregation.



(a) Walmart dataset



(b) Weather dataset

Figure 3.5: PREMA works well with different data.

moderate and high aggregation levels in Figure 4.1, we conclude that PREMA is more robust with aggressive aggregation where only few samples are available. With "high agg", the number of aggregation samples is only 8.56% of the original size in the temporal aggregate, and 9.68% in the contemporaneous aggregate. In this case, the NDE of the best baseline is 3.04 (1.68)x the error of PREMA with FSF (PTW) dataset, respectively. PTW dataset is more challenging as it has relatively high percentage of missing entries (36.72%). Moreover, with no knowledge of the aggregation pattern, B-PREMA outperforms all baselines that have access to the aggregation information with FSF data. Although, B-PREMA has NDE larger than Mean and CMTF with "mod agg" on PTW data, it becomes superior to all baselines when the aggregation level is high.

With Walmart data in Figure 3.5 (a), "mod agg" means that weeks are aggregated into months in $\underline{\mathbf{Y}}^t$, and the 45 stores are divided into 15 groups, whereas time is aggregated quarterly (12 weeks) and stores are clustered into 9 groups in "high agg". CMTF works slightly better when the aggregation is moderate, owing to the fact that the second mode in Walmart data is departments as apposed to items in DFF data. Departments are less correlated than items from the same category. As a result, the advantage of tensor models over the matricized tensor in capturing the higher-order dependencies becomes less clear. However, PREMA is more immune to aggressive aggregation. In the "high level" case, The NDE of CMTF is 1.71 times the error of PREMA. Even without access to the aggregation information, B-PREMA significantly reduces the error of the baselines.

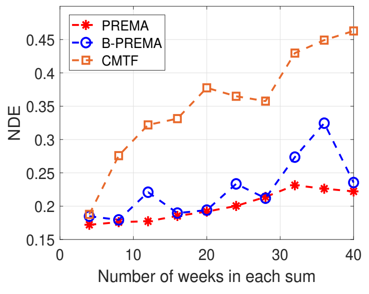
In Figure 3.5 (b), "mod agg" corresponds to the daily weather measurements averaged into weekly samples, and the 49 stations are averaged over 13 stations. On the other hand, the daily measurements are averaged over monthly samples, and the 49 stations are clustered into 7 stations in the "high agg" case. PREMA, CMTF, and H-Fuse perform similarly with Weather data[8] (it has 93.30% zeros) with moderate aggregation. The size of the second dimension of Weather data is small ($J = 17$), thus, the advantage of a tensor model over a matricized tensor model is less clear. H-Fuse works well with this data as it penalizes the large jumps between the adjacent time ticks (i.e., days), and weather data are well suited for such constraint. Nevertheless, PREMA improves the error of CMTF and H-Fuse when the aggregation level is high. Although B-PREMA does not work as well as with other data, it still has smaller error than the simple baselines (Mean and LS), especially with aggressive aggregation.

Next, we show the disaggregation performance on a wider range of aggregation levels using FSF dataset. The results are shown in Figure 3.6. The number of areas in $\underline{\mathbf{Y}}^c$ is fixed to 18 in Figure 3.6 (a) and 9 in Figure 3.6 (b), whereas the number of weeks in each sum in $\underline{\mathbf{Y}}^t$ ranges from 4 to 40 (x-axis). The total number of weeks in the dataset is 397; thus, we only have 10 temporally aggregated samples if we have 40 weeks in each sum. In this set of results, we focus on comparing the proposed models with CMTF since it is the best performing among the baselines. The rank is set to $R = 40$ for PREMA and B-PREMA, while for CMTF we use a grid search to select the best rank. One can see that the proposed models are less affected as the aggregation level increases, even when the aggregation matrices are unknown with B-PREMA.

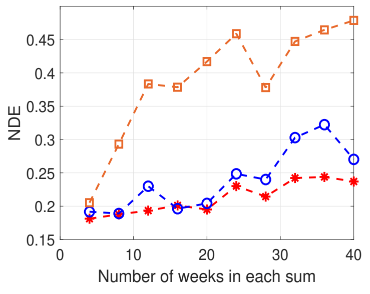
3.5.2 Results on Scenario B

The contemporaneous aggregate $\underline{\mathbf{Y}}^c$ in this scenario is aggregated in two dimensions: stores

[8]HomeRun is excluded from the results with Weather data as it has non-negativity constraints.

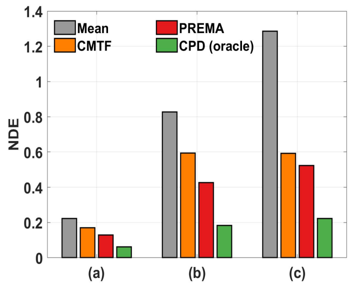


(a) 18 areas in $\underline{\mathbf{Y}}^c$

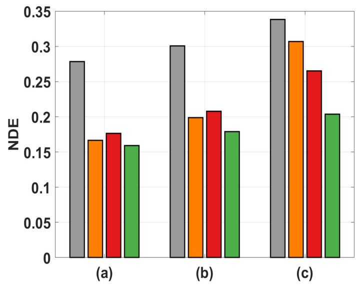


(b) 9 areas in $\underline{\mathbf{Y}}^c$

Figure 3.6: PREMA is more immune to aggressive aggregation.



(a) Mixed DFF dataset



(b) Crime dataset

Figure 3.7: PREMA works well with double aggregation (Scenario B).

and items with Mixed DFF data, or crime locations and types with Crime data[9]. We test this with three different aggregation levels with each data. Difficulty (i.e., level of aggregation), increases as we move from case (a) to (c)—Figure 3.7 shows the performance for these three cases. B-PREMA is not included in this set of experiments as it does not perform well. The reason is because double aggregation significantly reduces the number of equations, and the number of unknown parameters in B-PREMA is almost doubled since $\widetilde{\mathbf{A}}$ and $\widetilde{\mathbf{C}}$ are treated as separate variables from $\mathbf{A}$ and $\mathbf{C}$. Combining double aggregation and blind disaggregation makes it hard for the identifiability conditions to be satisfied.

With Mixed DFF data, these levels are: a) $\underline{\mathbf{Y}}^t$ aggregates weeks into monthly samples, while $\underline{\mathbf{Y}}^c$ groups the 93 stores into 18 areas with no aggregation over the items, b) samples in $\underline{\mathbf{Y}}^t$ have monthly resolution, and $\underline{\mathbf{Y}}^c$ groups the stores into 18 areas *and* items into groups of 10, and c) $\underline{\mathbf{Y}}^t$ contains temporal aggregates for each quarter of the year, and $\underline{\mathbf{Y}}^c$ groups stores into 18 areas *and* items into groups of 25. One can see that the naive mean totally fails and its error exceeds 1

[9]LS, H-Fuse, and HomeRun are excluded from this comparison as they run out of memory.

in case (c) with Mixed DFF data in Figure 3.7 (a). Notwithstanding, PREMA works well with double aggregation *and* few available samples.

With Crime data, the aggregation levels are: a) $\underline{\mathbf{Y}}^t$ aggregates the months into quarterly resolution, while $\underline{\mathbf{Y}}^c$ clusters both the crime locations and types into groups of 5, b) $\underline{\mathbf{Y}}^t$ has a quarterly time resolution, and $\underline{\mathbf{Y}}^c$ aggregates both the locations and types into groups of 10, and c) $\underline{\mathbf{Y}}^t$ aggregates the months into bi-yearly resolution, and $\underline{\mathbf{Y}}^c$ groups the crime locations and types into groups of 20. Figure 3.7 (b) shows the performance with these levels using Crime data. These data are challenging as they have 91.56% zero values and small SD. PREMA reduces the error of Mean significantly. Although CMTF performs slightly better with the first two levels, PREMA becomes superior with extreme aggregation.

3.5.3 Run time Comparison

In Table 3.3, we compare the run time of all the different methods for disaggregating the FSF dataset with the same setup as in Table 3.2 and $R = 40$. We can see that PREMA and B-PREMA are very scalable and faster than all the baselines (except for Mean, which only requires simple averaging). Our methods handle the missing entries very efficiently compared to the plain vanilla CPD using `TensorLab`.

Table 3.3: Run time comparisons.

Method	Run time (seconds)
Mean	0.10
LS	222.53
H-Fuse	6116.34
HomeRun	117.10
CMTF	1.26
CPD	13.85
PREMA	0.90
B-PREMA	0.89

3.6 Conclusions

In this work, we proposed a novel framework, called PREMA, for fusing multiple aggregated views of multidimensional data. The proposed method leverages the properties of tensors in estimating the low-rank factors of the target data in higher resolution. The assumed model is provably transforming a highly ill-posed problem to an identifiable one. PREMA works with partially observed data, and can disaggregate effectively, even without any knowledge of the aggregation mechanism (B-PREMA). Experimental results on real data show that the proposed algorithm is very effective, even in challenging scenarios, such as data with double aggregation and high level of aggregation. The contributions of our work in this chapter are summarized as follows:

- **Formulation**: we formally defined the problem of multidimensional data disaggregation from views aggregated in different dimensions.
- **Identifiability:** The considered tensor model provably converts a highly ill-posed problem to an identifiable one.
- **Effectiveness:** PREMA reduced the disaggregation error of the competing alternatives by up to 67%.
- **Unknown aggregation:** B-PREMA works even when the aggregation mechanism is unknown.
- **Flexibility :** PREMA can perform disaggregation on partially observed data.

Chapter 4

Learning Tree-structured Embeddings

4.1 Introduction & Related Work

In many applications, the categories of items exhibit a hierarchical tree structure. For instance, human diseases can be divided into coarse categories, e.g., bacterial, and viral. These categories can be further divided into finer categories, e.g., viral infections can be respiratory, gastrointestinal, and exanthematous viral diseases. In e-commerce, products, movies, books, etc., are grouped into hierarchical categories, e.g., clothing items are divided by gender, then by type (formal, casual, etc.). While the tree structure and the categories of the different items may be known in some applications, they have to be learned together with the embeddings in many others.

Incorporating tree structures in machine learning models has been recently considered, mostly in recommender systems [65, 76, 123] and also in other applications such as image processing [35], clustering and classification [104], and natural language processing (NLP) [94]. For example, the recommender system model in [117] penalizes MF with the distance between users who share common traits based on hierarchically-organized features. In another MF model [101], the item embeddings are assumed to form a tree, where each leaf node represents a single item and the parent nodes contain subsets of items (categories). The embeddings of parent nodes and leaf nodes are learned jointly. The final item feature vector is modeled as a weighted sum of its embedding and those of the categories it belongs to. Regularizing MF with a pre-defined tree prior has been also explored in response prediction in online advertising [74]. In this example, the tree groups the set of ads according to their campaigns, and the campaigns are further grouped based on the advertisers running them.

All the aforementioned methods assume that the tree structure is known apriori, or learned separately via side information. Recently, [107, 108] proposed to capture the unknown implicit tree structure via a model based on nonnegative matrix factorization (NMF). In a three-layer tree, the embedding of a leaf node (item/user) is assumed to be a linear combination of *all* the parent nodes (subcategories) in the intermediate layer, and each subcategory is a linear combination of all the categories in the root nodes. The weights that determine the memberships of a child node to the parent nodes are non-negative and learned by the model. This results in a fully connected tree, thus, a clear tree clustering can not be obtained. Moreover, [107, 108] imposes the implicit tree as a hard constraint, which can be restrictive if the data do not exactly follow the imposed prior.

In this chapter, we propose eTREE (Learning Tree-structured Embeddings), a framework that integrates the unknown implicit tree structure into a low-rank nonnegative factorization model to improve the quality of embeddings. eTREE does not require any extra information and jointly learns: i) the embeddings of all the tree nodes (items, subcategories, and main categories), and ii) the tree clustering in an unsupervised fashion. Unlike [107, 108], the obtained tree provides clear hierarchical clusters as each node belongs to exactly one parent node, e.g., an item belongs to one subcategory, and a subcategory belongs to one main category. The formulation of eTREE handles partially observed data matrices, which appear often in real-world applications. We derive an efficient algorithm to compute eTREE with a scalable implementation that leverages parallel computing, computation caching, and warm-start strategies. Our contributions can be summarized as follows:

- **Formulation:** eTREE provides an intuitive formulation that: i) exploits the tree structure, and ii) learns the hierarchical clustering in an unsupervised data-driven fashion.
- **Identifiability:** We leverage the special uniqueness properties of NMF to prove identifiability of eTREE.
- **Effectiveness:** eTREE significantly improves the quality of the embeddings in terms of matrix completion error on data from recommender systems, healthcare, and education.
- **Interpretability:** We demonstrate the meaningfulness of the tree clusters learned by eTREE using real-data interpreted by domain experts.

4.2 Background

In this section, we provide the background needed before presenting eTREE. We review NMF and related recent identifiability results, followed by a brief background on the alternating direction method of multipliers (ADMM).

4.2.1 Non-negative Matrix Factorization

Assume we have a healthcare data matrix $\mathbf{X} \in \mathbb{R}^{N \times M}$ indexed by (patient, medical service), where $\mathbf{X}(i,j)$ denotes the number of times the i^{th} patient has received the j^{th} service. In other applications, $\mathbf{X}$ may contain the ratings given by users to items, or the grades received by students in their courses. In some parts of this chapter, we refer to patients, users, students, etc. as individuals, and to medical services, products, etc. as items. NMF models aim to decompose the data matrix into low-rank latent factor matrices as $\mathbf{X} = \mathbf{A}\mathbf{B}^T$, where $\mathbf{A} \in \mathbb{R}^{N \times R}$, $\mathbf{B}^{M \times R}$ only have non-negative values, and $R \leq \min(N, M)$ is the matrix rank. NMF has gained considerably special attention as it tends to produce interpretable representations. For instance, it has been shown that the columns of $\mathbf{A}$ produce clear parts of human faces (e.g., nose, ears, and eyes) when NMF in applied on a matrix $\mathbf{X}$ whose columns are vectorized face images [60]. In practice, NMF is often formulated as a bilinear optimization problem:

$$\min_{\mathbf{A} \geq \mathbf{0}, \mathbf{B} \geq \mathbf{0}} \quad \frac{1}{2}\mathcal{F}(\mathbf{A}, \mathbf{B}) = \|\mathbf{W} \odot (\mathbf{X} - \mathbf{A}\mathbf{B}^T)\|_F^2 \tag{4.1}$$

where $\mathbf{W} \in \{0,1\}^{N \times M}$ has ones at the indices of the observed entries in $\mathbf{X}$, and zeros otherwise. Each row of $\mathbf{A}$ corresponds to the embedding/latent representation of the corresponding individual, whereas the rows of $\mathbf{B}$ are the embeddings of the items.

Identifiability of NMF: The interpretability of NMF is intimately related to its uniqueness properties – the latent factors are identifiable under some conditions (up to trivial ambiguity, e.g., scaling/counter-scaling or permutation). To facilitate our discussion of the uniqueness of eTREE, we present the following definitions and established identifiability results.

Definition 4.1 (Identifiability)**.** *The NMF of* $\mathbf{X} = \mathbf{A}\mathbf{B}^T$ *is said to be (essentially) unique if* $\mathbf{X} = \widetilde{\mathbf{A}}\widetilde{\mathbf{B}}^T$ *implies* $\widetilde{\mathbf{A}} = \mathbf{A}\mathbf{\Pi}\mathbf{D}$ *and* $\widetilde{\mathbf{B}} = \mathbf{B}(\mathbf{\Pi}\mathbf{D})^{-1}$*, where* $\mathbf{\Pi}$ *is a permutation matrix, and* $\mathbf{D}$ *is a diagonal positive matrix.*

Definition 4.2 (Sufficiently Scattered)**.** *A nonnegative matrix* $\mathbf{B} \in \mathbb{R}^{M\times R}$ *is said to be sufficiently scattered if: 1) cone*$\{\mathbf{B}^T\} \supseteq \mathcal{C}$*, and 2) cone*$\{\mathbf{B}^T\} \cap$ *bd*$\{\mathcal{C}^\star\} = \{\lambda \mathbf{e}_k | \lambda \geq 0, k = 1,\ldots,R\}$*, where* $\mathcal{C} = \{\mathbf{x}|\mathbf{x}^T\mathbf{1} \geq \sqrt{R-1}\|\mathbf{x}\|_2\}$*,* $\mathcal{C}^\star = \{\mathbf{x}|\mathbf{x}^T\mathbf{1} \geq \|\mathbf{x}\|_2\}$*, cone*$\{\mathbf{B}^T\} = \{\mathbf{x}|\mathbf{x} = \mathbf{B}^T\boldsymbol{\theta},\ \forall\boldsymbol{\theta} \geq \mathbf{0},\ \mathbf{1}^T\boldsymbol{\theta} = 1\}$*, and cone*$\{\mathbf{B}^T\}^\star = \{\mathbf{y}|\mathbf{x} = \mathbf{x}^T\mathbf{y},\ \forall \mathbf{x} \in$ *cone*$\{\mathbf{H}^T\}\}$ *are the conic hull of* $\mathbf{B}^T$ *and its dual cone, respectively, and bd is the boundary of a set.*

The works in [37, 66] prove that the so-called volume minimization (VolMin) criterion can identify the factor matrices if $\mathbf{A}$ is full-column rank, and the rows of $\mathbf{B}$ are sufficiently scattered (Definition 4.2) and sum-to-one (row stochastic). Recently, Fu *et al.* shifted the row stochastic condition on rows of $\mathbf{B}$ to the *columns* of $\mathbf{B}$.

Theorem 4.1 (NMF Identifiability)**.** *[36]* $\mathbf{A}$ *and* $\mathbf{B}$ *are essentially unique under the criterion of minimizing det*$(\mathbf{A}^T\mathbf{A})$ *w.r.t.* $\mathbf{A} \in \mathbb{R}^{N\times R}$ *and* $\mathbf{B} \in \mathbb{R}^{M\times R}$*, subject to* $\mathbf{X} = \mathbf{A}\mathbf{B}^T$ *and* $\mathbf{B}^T\mathbf{1} = \mathbf{1}, \mathbf{B} \geq \mathbf{0}$ *if* $\mathbf{B}$ *is sufficiently scattered, and rank(*$\mathbf{X}$*) = rank(*$\mathbf{A}$*) = R.*

Theorem 4.1 provides an intriguing generalization of NMF, as it pertains to a more general factorization. Note that $\mathbf{A}$ is *not* restricted to be non-negative. Also note that the column-sum-to-one constraint on $\mathbf{B}$ is without loss of generality, as one can always assume the columns of $\mathbf{B}$ are scaled by a diagonal matrix $\mathbf{D}$, and compensate for this scaling in the columns of $\mathbf{A}$, i.e., $\mathbf{X} = (\mathbf{A}\mathbf{D}^{-1})(\mathbf{B}\mathbf{D})^T$.

4.2.2 Alternating Direction Method of Multipliers

ADMM is a primal-dual algorithm that solves convex optimization problems in the form

$$\begin{aligned} \min_{\mathbf{x},\mathbf{z}} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c} \end{aligned} \tag{4.2}$$

by iterating the following updates

$$\begin{aligned} \mathbf{x} &\leftarrow \arg\min_{\mathbf{x}} \quad f(\mathbf{x}) + \rho/2\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2 \\ \mathbf{z} &\leftarrow \arg\min_{\mathbf{z}} \quad g(\mathbf{z}) + \rho/2\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2 \\ \mathbf{u} &\leftarrow \mathbf{u} + (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c}) \end{aligned} \tag{4.3}$$

where $\mathbf{u}$ is a scaled version of the dual variable, and $\rho > 0$ is a Lagrangian parameter. A comprehensive review of ADMM can be found in [16].

4.3 Proposed Framework: eTREE

In this section, we present our proposed framework. We start with the mathematical formulation of eTREE, then we discuss the theoretical uniqueness of the proposed model. Next, we work out some design considerations of eTREE, then we derive the algorithmic solution.

4.3.1 eTREE: Formulation

In many applications, categories of items exhibit a hierarchical tree structure – as we showed in the introduction. For ease of notation, let us denote the embedding matrix of items resulting from NMF in (4.1) as $\mathbf{B}_1 \in \mathbb{R}^{M_1 \times R}$, where M_1 is the number of items. Assume that the embeddings of the M_1 items (rows of $\mathbf{B}_1$) are the leaf nodes at the very bottom layer in a tree. A subset of items that belong to the same category is grouped together via one parent node, where the parent node is the embedding of the corresponding category. Assuming that the embeddings are fully inherited (replicated verbatim) from one's parent category, we can further decompose $\mathbf{B}_1$ into

$$\mathbf{B}_1 = \mathbf{S}_1 \mathbf{B}_2 \tag{4.4}$$

where each row of $\mathbf{B}_2 \in \mathbb{R}^{M_2 \times R}$ is the embedding of one category, M_2 is the number of categories with $M_2 \leq M_1$, and $\mathbf{S}_1 \in \{0,1\}^{M_1 \times M_2}$, $\|\mathbf{S}_1(i,:)\|_0 = 1, \forall i \in [M_1]$, i.e., values in $\mathbf{S}_1$ are binary with only one 1 per row to ensure that each item belongs to exactly one category. Note that M_2 is the number of parent nodes (categories) in the second from bottom layer. The M_2 categories can be grouped into coarser categories, i.e., we decompose $\mathbf{B}_2$ into $\mathbf{B}_2 = \mathbf{S}_2\mathbf{B}_3$, where rows of $\mathbf{B}_3 \in \mathbb{R}^{M_2 \times M_3}$ represent the embeddings of the coarse categories, and $\mathbf{S}_2$ maps the M_2 fine-level categories into the M_3 coarse-level categories in the same fashion as $\mathbf{S}_1$. Up to here, we have constructed a three-layer tree, and we can use the same concept to create a Q-layer tree. Fig. 4.1 illustrates the mapping between $\mathbf{B}_1$ and $\mathbf{B}_2$ in matrix notation (left), and shows a 3-layer tree (right). Generalizing to Q layers, we obtain

$$\mathbf{B}_1 = \mathbf{S}_1 \mathbf{S}_2 \ldots \mathbf{S}_{Q-1} \mathbf{B}_Q \tag{4.5}$$

Substituting the embedding matrix of items in (4.1) with the right term in (4.5) above may seem natural, however there is a solution ambiguity in the cases where $Q > 2$. To see this, the route $\mathbf{B}_1(1,:) \rightarrow \mathbf{B}_2(2,:) \rightarrow \mathbf{B}_3(1,:)$ in Fig. 4.1 (right) would give the same cost value as

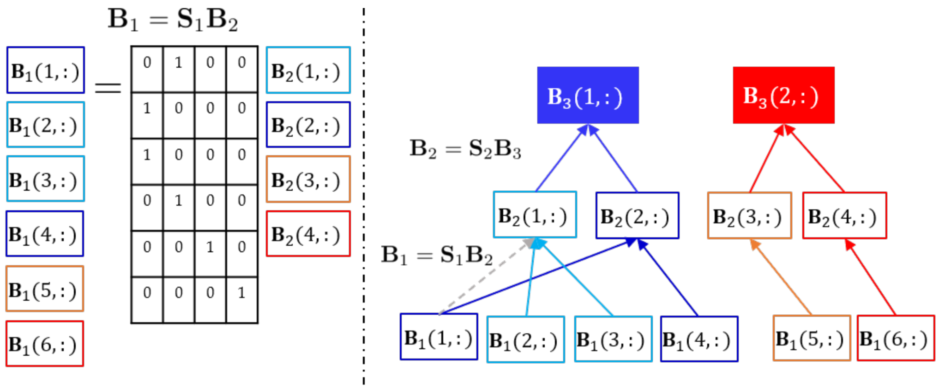


Figure 4.1: Illustration of the tree prior in eTREE.

$\mathbf{B}_1(1,:) \to \mathbf{B}_2(1,:) \to \mathbf{B}_3(1,:)$ (the dotted gray arrow). Moreover, imposing the tree structure as a hard constraint can be too intrusive when the data do not exactly follow the assumed prior. Thus, we propose to: i) incorporate the tree prior as a soft constraint, and ii) explicitly solve for the embedding of the intermediate layers to resolve the (immaterial for our purposes) solution ambiguity. This yields the following formulation:

$$\begin{aligned}
\min_{\mathcal{U}} \quad & \mathcal{F}(\mathbf{A}, \mathbf{B}_1) + \frac{\mu}{2}\sum_{q=1}^{Q-1} \|\mathbf{B}_q - \mathbf{S}_q\mathbf{B}_{q+1}\|_F^2 \\
\text{s.t.} \quad & \mathbf{S}_q \in \{0,1\}^{M_q \times M_{q+1}}, q \in [Q-1] \\
& \|\mathbf{S}_q(i,:)\|_0 = 1, \ \forall i \in [M_q], q \in [Q-1] \\
& \mathbf{A} \geq \mathbf{0}, \ \mathbf{B}_1 \geq \mathbf{0}
\end{aligned} \tag{4.6}$$

where $\mathcal{U} := \left\{\mathbf{A}, \{\mathbf{B}_q\}_{q=1}^{Q}, \{\mathbf{S}_q\}_{q=1}^{Q-1}\right\}$ is the set of all variables, and $\mathcal{F}$ is the NMF cost function as defined in (4.1) with $\mathbf{B}_1$ as the embedding matrix of items. The second term is to minimize the difference between the embedding of each child node and its parent node in the tree structure. In other words, it minimizes the difference between the embeddings of each item and its category, or between each fine category and its coarse category. $\mu \geq 0$ is a regularization parameter to balance the data fidelity and the tree prior.

There is an intriguing connection between the proposed tree regularizer and k-means formulation. The variables $\{\mathbf{S}_q\}_{q=1}^{Q-1}$ are equivalent to the assignment variables in k-means for clustering the rows of $\{\mathbf{B}_q\}_{q=1}^{Q-1}$, respectively, and $\mathbf{B}_Q$ is equivalent to the centroid variable in k-means for clustering the rows of $\mathbf{B}_{Q-1}$. On the other hand, each variable in $\{\mathbf{B}_q\}_{q=2}^{Q-1}$ is involved in two terms: 1) $\|\mathbf{B}_{q-1} - \mathbf{S}_{q-1}\mathbf{B}_q\|_F^2$ where its rows are centroids, and 2) $\|\mathbf{B}_q - \mathbf{S}_q\mathbf{B}_{q+1}\|_F^2$ where its rows are the points to be clustered. This can be thought of as a *regularized* k-means. Interestingly, the joint NMF and latent k-means model in [113] is a special case of eTREE with $Q = 2$.

Note that if the tree structure is known apriori, it can be seamlessly incorporated using our formulation. A direct way is to fix $\{\mathbf{S}_q\}_{q=1}^{Q-1}$ to the known tree and solve (4.6) w.r.t. the rest of the variables. This way we learn the embedding of the categories and penalize the distance between items and their corresponding categories. When the tree is *partially* known, one can fix the known parts and learn the unknowns. Another way to integrate a known tree is to penalize the difference between the embeddings of items that share similar paths to the root nodes. The latter method does not require us to learn $\{\mathbf{B}_q\}_{q=2}^{Q}$. In this work, we focus on the more challenging scenario where the tree structure is unknown and to be learned from the data.

eTREE has the following advantages: i) it incorporates the tree structure to improve the quality of embeddings, ii) unlike most methods in the literature, it assumes the tree is unknown and learns it through the solution of $\{\mathbf{S}_q\}_{q=1}^{Q-1}$ in an unsupervised fashion, and iii) it provides the embedding of the parent nodes (categories) in addition to the item embeddings. The tree clustering can be useful in broader applications such as classification and data labeling. The embeddings of categories provide extra information for some applications, e.g., web personalization and category-based recommendation [42].

4.3.2 eTREE: Theoretical Identifiability

Identifiability in machine learning problems that require parameter estimation is essential in guaranteeing sensible results, especially in applications where model identifiability is entangled with interpretability, such as topic modeling [12], image processing [60], and social network clustering [72]. Nevertheless, the majority of MF-based methods in practice do not have known identifiability guarantees. In the following theorem, we establish the identifiability of eTREE for the case where $\mathbf{X}$ is fully observed.

Theorem 4.2. *Assume that a data matrix follows $\mathbf{X} = \mathbf{A}\mathbf{B}_1^T$, where $\mathbf{A} \in \mathbb{R}^{N\times R}$, and $\mathbf{B}_1 \in \mathbb{R}^{M_1\times R}$ are the ground-truth factors, and assume that $\mathbf{B}_1 = \mathbf{S}_1\mathbf{S}_2\ldots\mathbf{S}_{Q-1}\mathbf{B}_Q$, where $\mathbf{S}_q \in \{0,1\}^{M_q\times M_{q+1}}$, $\|\mathbf{S}_q(i,:)\|_0 = 1, \forall i \in [M_q], q \in [Q-1]$. Let $\mathbf{S} = \mathbf{S}_1\mathbf{S}_2\ldots\mathbf{S}_{Q-1}$, then, $\mathbf{B}_1 = \mathbf{S}\mathbf{B}_Q$. Also, assume that rank$(\mathbf{X})$ = rank$(\mathbf{A})$ = R, and, without loss of generality, $\mathbf{M}_Q \geq R$. If $\mathbf{A}$ and $\mathbf{S}$ are full-column rank, and rows of $\mathbf{B}_Q$ are sufficiently scattered, then rows* *of* $\mathbf{B}_1$ *are sufficiently scattered, and* $\mathbf{A}$*,* $\mathbf{B}_1$*,* $\mathbf{B}_Q$*, and* $\mathbf{S}$ *are essentially unique.*

Proof Sketch: The factors $\mathbf{A}$ and $\mathbf{B}_Q$ in $\mathbf{M} = \mathbf{A}\mathbf{B}_Q^T$ are essentially unique by Theorem 4.1, since $\mathbf{A}$ is full-column rank, and rows of $\mathbf{B}_Q$ are sufficiently scattered (Definition 4.2). If $\mathbf{S}$ is full-column rank, then all the rows of $\mathbf{B}_Q$ will appear in $\mathbf{B}_1 = \mathbf{S}\mathbf{B}_Q$. Thus, rows of $\mathbf{B}_1$ are sufficiently scattered *iff* the rows of $\mathbf{B}_Q$ are sufficiently scattered. Now, $\mathbf{A}$ and $\mathbf{B}_1$ in $\mathbf{X} = \mathbf{A}\mathbf{B}_1^T$ are essentially unique by Theorem 4.1, since $\mathbf{A}$ is full-column rank, and rows of $\mathbf{B}_1$ are sufficiently scattered. Next, the factor $\mathbf{S}$ in $\mathbf{B}_1 = \mathbf{S}\mathbf{B}_Q$ is also essentially unique because the rows of $\mathbf{B}_1$ are rows of $\mathbf{B}_Q$, and every row of $\mathbf{B}_Q$ appears in $\mathbf{B}_1$, hence $\mathbf{S}$ can be determined based on the correspondence (identifiability of $\mathbf{S}$ also follows as a very special instance of Theorem 4.1).

In plain words, in addition to the NMF identifiability conditions ($\mathbf{A}$ to be full-rank, and rows of $\mathbf{B}_Q$ to be sufficiently scattered), we only require $\mathbf{S}$ to be full-column rank, which means that every root node (main category) must have at least one leaf node — this is a natural condition in a tree. Interestingly, the rows of $\mathbf{B}_Q$ are likely to be sufficiently scattered as they are the embeddings of the coarsest categories and encouraged to be distant (think, e.g., in a 3-layer tree, each row is the centroid of multiple subcategories, where each subcategory is the centroid of a set of items). We point out that there is an inherent column permutation ambiguity in $\{\mathbf{S}_q\}_{q=1}^{Q-1}$ in $\mathbf{S} = \mathbf{S}_1\mathbf{S}_2 \dots \mathbf{S}_{Q-1}$, however, this is immaterial in our context.

4.3.3 eTREE: Model Engineering

In this section, we discuss some caveats that need to be addressed in the formulation (4.6) before moving to the algorithmic derivation.

The first point is the scaling between the low-rank factors $\mathbf{A}$ and $\mathbf{B}_1$. The tree structure regularizer implicitly favors $\mathbf{B}_1$ to have a small norm. On the other hand, the first term is not affected by the scaling of $\mathbf{B}_1$, as long as this scaling is compensated for in $\mathbf{A}$. This motivates introducing norm regularization on $\mathbf{A}$, i.e., $||\mathbf{A}||_F$.

The second consideration is regarding the tree structure term. It has been shown that the cosine similarity metric is superior over the Euclidean distance in clustering [100] and latent clustering [113] in many applications. We also observed that constraining the rows of $\{\mathbf{B}\}_{q=1}^{Q-1}$ to be in the unit l_2-norm ball, which is equivalent to using cosine similarity in clustering, gives better performance. Taking these points into account, we obtain the following formulation:

$$\begin{aligned}
\min_{\mathcal{Y}} \;& \mathcal{F}_d(\mathbf{A},\mathbf{B},\mathbf{D}) + \frac{\mu}{2}\sum_{q=1}^{Q-1}\|\mathbf{B}_q - \mathbf{S}_q\mathbf{B}_{q+1}\|_F^2 + \frac{\lambda}{2}\|\mathbf{A}\|_F^2 \\
\text{s.t. }\; & \|\mathbf{B}_q(i,:)\|_2 = 1 \; \forall i \in [M_q],\; \forall q \in [Q-1] \\
& \mathbf{S}_q \in \{0,1\}^{M_q \times M_{q+1}}, \forall q \in [Q-1] \\
& \|\mathbf{S}_q(i,:)\|_0 = 1,\; \forall i \in [M_q], q \in [Q-1] \\
& \mathbf{D} = \text{Diag}(d_1,\ldots,d_{M_1}) \\
& \mathbf{A} \geq \mathbf{0},\; \mathbf{B}_1 \geq \mathbf{0}
\end{aligned} \tag{4.7}$$

where $\mathcal{Y} := \left\{\mathbf{A},\mathbf{D},\{\mathbf{B}_q\}_{q=1}^{Q},\{\mathbf{S}_q\}_{q=1}^{Q-1}\right\}$ is the set of all variables, $\lambda \geq 0$, $\mathcal{F}_d := 1/2\|\mathbf{W} \odot (\mathbf{X} - \mathbf{A}\mathbf{B}^T\mathbf{D})\|_F^2$, and $\mathbf{D}$ is a diagonal matrix that is introduced to allow us to fix the rows of $\mathbf{B}_1$ onto the unit l_2-norm ball without loss of generality of the factorization model.

4.3.4 eTREE: Algorithm

The optimization problem in (4.7) is NP-hard (as it contains both NMF and k-means as special cases, and both are known to be NP-hard). We therefore present a carefully designed alternating optimization (AO) algorithm. The proposed algorithm leverages ADMM (reviewed in the background section above) and utilizes parallel computing, computation caching, and warm-start to provide a scalable implementation. The high level algorithmic strategy is to employ AO to update $\mathbf{A}$, $\mathbf{D}$, $\{\mathbf{B}_q\}_{q=1}^{Q}$, and $\{\mathbf{S}_q\}_{q=1}^{Q-1}$ one at a time, while fixing the others. The resulting sub-problems w.r.t. a single variable can be solved optimally.

We propose a variable-splitting strategy by introducing slack variables $\{\mathbf{Z}_q \in \mathbb{R}^{M_q \times R}\}_{q=1}^{Q-1}$ to handle the unit l_2 norm ball constraints on $\{\mathbf{B}_q \in \mathbb{R}^{M_q \times R}\}_{q=1}^{Q-1}$ in (4.7). Specifically, we consider the following optimization surrogate:

$$\begin{aligned}
\min_{\mathcal{H}} \quad & \mathcal{F}_d(\mathbf{A},\mathbf{B},\mathbf{D}) + \frac{\mu}{2}\textstyle\sum_{q=1}^{Q-1}\|\mathbf{B}_q - \mathbf{S}_q\mathbf{B}_{q+1}\|_F^2 \\
& + \frac{\eta}{2}\textstyle\sum_{q=1}^{Q-1}\|\mathbf{B}_q - \mathbf{Z}_q\|_F^2 + \frac{\lambda}{2}\|\mathbf{A}\|_F^2 \\
\text{s.t.} \quad & \|\mathbf{Z}_q(i,:)\|_2 = 1, \forall i \in [M_q], q \in [Q-1] \\
& \mathbf{S}_q \in \{0,1\}^{M_q \times M_{q+1}}, \forall q \in [Q-1] \\
& \|\mathbf{S}_q(i,:)\|_0 = 1,\ \forall i \in [M_q], q \in [Q-1] \\
& \mathbf{D} = \text{Diag}(d_1,\ldots,d_{M_1}) \\
& \mathbf{A} \geq \mathbf{0},\ \mathbf{B}_1 \geq \mathbf{0}
\end{aligned} \tag{4.8}$$

where $\mathcal{H} := \left\{\mathbf{A},\mathbf{D},\{\mathbf{B}_q\}_{q=1}^{Q},\{\mathbf{Z}_q\}_{q=1}^{Q-1},\{\mathbf{S}_q\}_{q=1}^{Q-1}\right\}$ is the set of all the variables, and $\eta \geq 0$. Note that when $\eta = +\infty$, then (4.8) is equivalent to (4.7). In practice, we choose a large η to enforce $\mathbf{B}_q \approx \mathbf{Z}_q$ (we set it to $\eta = 1000$ in all experiments). We handle problem (4.8) as follows. First, we update $\mathbf{A}$ by solving the following non-negative least squares

$$\min_{\mathbf{A}\geq\mathbf{0}} \quad \frac{1}{2}\|\mathbf{W}\odot(\mathbf{X}-\mathbf{A}\mathbf{B}_1^T\mathbf{D})\|_F^2 + \frac{\lambda}{2}\|\mathbf{A}\|_F^2 \tag{4.9}$$

using ADMM. Due to space limitation, we use the update of $\mathbf{A}$ as a working example for the two updates that uses ADMM ($\mathbf{A}$ and $\mathbf{B}_1$). Problem (4.9) can be reformulated by introducing an auxiliary variable $\widetilde{\mathbf{A}}$

$$\begin{aligned}
\min_{\mathbf{A},\widetilde{\mathbf{A}}} \quad & \frac{1}{2}\|\mathbf{W}^T\odot(\mathbf{X}^T-\widetilde{\mathbf{B}}\widetilde{\mathbf{A}})\|_F^2 + \frac{\lambda}{2}\|\widetilde{\mathbf{A}}\|_F^2 + \mathcal{R}(\mathbf{A}) \\
\text{s.t.} \quad & \mathbf{A} = \widetilde{\mathbf{A}}^T
\end{aligned} \tag{4.10}$$

where $\widetilde{\mathbf{B}} = \mathbf{D}\mathbf{B}_1$, and $\mathcal{R}(.)$ is the indicator function of the nonnegative orthant. Next, we derive the ADMM updates

$$\begin{aligned}
\widetilde{\mathbf{A}}(:,i) \leftarrow & (\widetilde{\mathbf{B}}(\mathcal{J}_i,:)^T\widetilde{\mathbf{B}}(\mathcal{J}_i,:) + c\mathbf{I}_R)^{-1}\big(\widetilde{\mathbf{B}}(\mathcal{J}_i,:)^T\cdot \\
& \quad \mathbf{X}(i,\mathcal{J}_i)^T + \rho(\widetilde{\mathbf{A}}(:,i)+\mathbf{U}(i,:)^T)\big) && \text{(4.11a)} \\
\mathbf{A}(i,:) \leftarrow & [\widetilde{\mathbf{A}}(:,i)^T - \mathbf{U}(i,:)]_+ && \text{(4.11b)} \\
\mathbf{U}(i,:) \leftarrow & \mathbf{U}(i,:) + \mathbf{A}(i,:) - \widetilde{\mathbf{A}}(:,i)^T && \text{(4.11c)}
\end{aligned}$$

where $c := \lambda + \rho$, $[.]_+$ is the projection on $\mathbb{R}_+$ by zeroing the negative entries, and $\mathcal{J}_i$ is the set of items that have observations for the i^{th} individual. We use the adaptive $\rho = \|\widetilde{\mathbf{B}}\|_F^2/NR$, which is a scaled version of ρ suggested in [48]. The ADMM steps in (5.6) are performed until a termination criterion is met. We adopt the criterion in [16, 48], namely, the primal and dual residuals

$$\begin{aligned} p_i &= \|\mathbf{A}(i,:) - \widetilde{\mathbf{A}}(:,i)^T\|_F^2/\|\mathbf{A}(i,:)\|_F^2; \\ d_i &= \|\mathbf{A}(i,:) - \mathbf{A}_0(i,:)\|_F^2/\|\mathbf{U}(i,:)\|_F^2; \end{aligned} \tag{4.12}$$

where $\mathbf{A}_0$ is $\mathbf{A}$ from the previous iteration. We iterate between the ADMM updates until p and d are smaller than a predefined threshold, or we reach the maximum number of iterations K – in our experiments we set $K = 5$.

Scalability Considerations: There are some important observations regarding the implementation of the ADMM updates in (5.6). First, we do not compute the matrix inversion in (4.11a) explicitly. Instead, the *Cholesky decomposition* of the Gram matrix $\mathbf{G}_i := \widetilde{\mathbf{B}}(\mathcal{J}_i,:)^T\widetilde{\mathbf{B}}(\mathcal{J}_i,:) + c\mathbf{I}_R$ is computed, i.e., $\mathbf{G}_i = \mathbf{L}_i\mathbf{L}_i^T$, where $\mathbf{L}_i$ is a lower triangular matrix. Then, at each ADMM iteration, we only need to perform a forward and a backward substitution to get the solution of $\widetilde{\mathbf{A}}(:,i)$. Thus, the step in (4.11a) is replaced with:

$$\mathbf{G}_i \leftarrow \widetilde{\mathbf{B}}(\mathcal{J}_i,:)^T\widetilde{\mathbf{B}}(\mathcal{J}_i,:) + c\mathbf{I}_R;\ \mathbf{L}_i \leftarrow \text{Cholesky}(\mathbf{G}_i) \tag{4.13a}$$

$$\mathbf{F}_i \leftarrow \widetilde{\mathbf{B}}(\mathcal{J}_i,:)^T\mathbf{X}(i,\mathcal{J}_i)^T \tag{4.13b}$$

$$\widetilde{\mathbf{A}}(:,i) \leftarrow \mathbf{L}_i^{-T}\mathbf{L}_i^{-1}(\mathbf{F}_i + \rho(\widetilde{\mathbf{A}}(:,i) + \mathbf{U}(i,:)^T)) \tag{4.13c}$$

Computing the Cholesky decomposition requires $\mathcal{O}(R^3)$ flops, and the back and forward substitution steps cost $\mathcal{O}(NR^2)$. The matrix multiplication in $\widetilde{\mathbf{B}}(\mathcal{J}_i,:)^T\widetilde{\mathbf{B}}(\mathcal{J}_i,:)$ and in computing $\mathbf{F}_i$ in (4.13b) takes $\mathcal{O}(|\mathcal{J}_i|R^2)$ and $\mathcal{O}(|\mathcal{J}_i|R)$, respectively, where, $|\mathcal{J}_i| \leq N$ is the cardinality of the set $\mathcal{J}_i$. An important implication is that $\mathbf{L}_i$ and $\mathbf{F}_i$ do not change throughout the ADMM iterations, thus can be cached to save computation. The overall complexity to update $\mathbf{A}$ is $\mathcal{O}(NR^2)$. Moreover, the ADMM updates enjoy row separability, allowing parallel computation. In the case where $\mathbf{X}$ is fully observed, $\mathbf{G} := \widetilde{\mathbf{B}}^T\widetilde{\mathbf{B}} + c\mathbf{I}_R$ and $\mathbf{F} := \widetilde{\mathbf{B}}^T\mathbf{X}^T$ are shared not only across the ADMM iterations, but also among the N parallel sub-problems corresponding to the rows of $\mathbf{A}$. Finally, the outer AO routine naturally provides a good initial point (warm-start) to the inner ADMM iterations (for both $\mathbf{A}$ and its dual variable $\mathbf{U}$), resulting in a faster convergence.

Next, we update $\mathbf{B}_1$ using ADMM in the same fashion as $\mathbf{A}$. The updates of $\mathbf{Z}_1$ and $\mathbf{D}$ admit closed form solutions

$$\mathbf{Z}_1(j,:) = \mathbf{B}_1(j,:)/\|\mathbf{B}_1(j,:)\|_2;\ d_j = \mathbf{h}_j^T\mathbf{X}(\mathcal{I}_j, j)/\mathbf{h}_j^T\mathbf{h}_j \tag{4.14}$$

where $\mathbf{h}_j = (\mathbf{B}_1(j,:)\mathbf{A}^T)^T$, and $\mathcal{I}_j$ is the set of individuals that have observations for the j^{th} item.

In the next step, we perform few inner iterations to alternate between updating the tree structure triplets $\{\mathbf{S}_{q-1}, \mathbf{B}_q, \mathbf{Z}_q)\}_{q=2}^{Q-1}$ and $\mathbf{B}_Q$ in a cyclic fashion (we call it the tree loop) – in the experiments we set the maximum number of tree iterations $T = 5$. The updates w.r.t. $\{\mathbf{B}_q\}_{q=2}^{Q-1}$ are unconstrained least squares problems. These problems are column separable with a common mixing matrix. Thus, the complexity can be reduced by computing *one* Cholesky decomposition. Then, at each iteration, the update of each column only requires a forward and a backward substitution as follows

$$\mathbf{H} \leftarrow \mu\mathbf{S}_{q-1}^T\mathbf{S}_{q-1} + v\mathbf{I}_{M_q};\ \mathbf{L} \leftarrow \text{Cholesky}(\mathbf{H}) \tag{4.15a}$$

$$\begin{aligned}\mathbf{B}_q(:,j) \leftarrow \mathbf{L}^{-T}\mathbf{L}^{-1}\big(&\mu\mathbf{S}_{q-1}\mathbf{B}_{q-1}(:,j) + \mu\mathbf{S}_q\mathbf{B}_{q+1}(:,j)\cdot \\ &+ \eta\mathbf{Z}_q(:,j)\big)\end{aligned} \tag{4.15b}$$

where $v := \mu + \eta$. The updates of $\mathbf{B}_Q$ and the matrices $\{\mathbf{S}\}_{q=1}^{Q-1}$ are similar to solving for the centroids and the assignment variables in the k-means algorithm, respectively. Let $\mathcal{T}_{m_Q} = \{i|\mathbf{S}_{Q-1}(i, m_Q) = 1\}$, then each row in $\mathbf{B}_Q$ is

$$\mathbf{B}_Q(m_Q,:) \leftarrow \textstyle\sum_{i\in\mathcal{T}_{m_Q}} \mathbf{B}_{Q-1}(i,:)/|\mathcal{T}_{m_Q}| \tag{4.16}$$

And the i^{th} row of the assignment matrices is updated using

$$\mathbf{S}_q(i,k) \leftarrow \begin{cases} 1, & k = \arg\min_{m_q} \|\mathbf{B}_{q-1}(i,:) - \mathbf{B}_q(m_q,:)\|_2 \\ 0, & \text{otherwise} \end{cases} \tag{4.17}$$

The overall algorithm is summarized in Algorithm 4.1. One nice property of the proposed algorithm is that all the updates are row separable and can be computed in a distributed fashion (with the exception of $\{\mathbf{B}_q\}_{q=2}^{Q-1}$, which are *column* separable)[1].

Algorithm 4.1 Algorithmic Solution to eTREE

```
Initialize: $\mathbf{A}, \mathbf{B}_1 \leftarrow$ NMF;
$\{\mathbf{B}_q\}_{q=2}^{Q}$ and $\{\mathbf{S}\}_{q=1}^{Q-1} \leftarrow$ random; $\mathbf{D} \leftarrow \mathbf{I}$; $\mathbf{U} \leftarrow \mathbf{0}$
repeat
       Compute $\mathbf{L}_i$, and $\mathbf{F}_i$, $\forall i$ using (4.13a) and (4.13b)
       Set $k = 1$                                        // counter of ADMM loop
1      while $p_i, d_i$ in (4.12) $> \epsilon$ and $k < K$ do
2          Update $\widetilde{\mathbf{A}}(:,i)$, $\mathbf{A}(i,:)$, and $\mathbf{U}(i,:)$, $\forall i$ using (4.13c), (4.11b), and (4.11c), respectively
            $k = k + 1$
3      end
4      Update $\mathbf{B}_1$ using ADMM loop (similar to $\mathbf{A}$)
        Update $\mathbf{D}$ and $\mathbf{Z}_1$ using (4.14)
        Set $t = 1$                                       // counter of tree loop
5      while $t < T$ do
6          for $q = 2, \ldots, Q-1$ do
7              compute $\mathbf{L}$ using (4.15a)  update $\mathbf{B}_q(:,j)$, $\forall j$ using (4.15b)  update $\mathbf{S}_{q-1}(i,:)$, $\forall i$
                using (4.17)  $\mathbf{Z}_q(i,:) = \mathbf{B}_q(i,:)/\|\mathbf{B}_q(i,:)\|_2$, $\forall i$
8          end
9          update $\mathbf{B}_Q(i,:)$, $\forall i$ using (4.16)  t = t + 1
10     end
until convergence
```

4.4 Experiments

In this section, we evaluate the proposed framework on real data from various application domains: healthcare analytics, movie recommendations, and education. This section aims to answer the following questions:

Q1. Accuracy: Does eTREE improve the quality of embeddings for the downstream tasks?

Q2. Interpretability: How meaningful is the tree structure learned by eTREE from an application domain knowledge viewpoint?

4.4.1 Datasets

We evaluate eTREE and the competing baselines on the following real datasets:

[1]Code is at: `https://github.com/FaisalAlmutairi/eTREE`

(i) **Med-HF:** These data are provided by IQVIA Inc. and include the counts of medical services performed on patients with heart failure (HF) conditions, including patients with preserved ejection fraction (pEF), and reduced ejection fraction (rEF). We include the $5,000$ patients with the most records in our experiments. The total number of medical services is 411. The majorities of the counts fall in the range of small numbers, with a small percentage of larger numbers, resulting in a "long tail" in the histogram of the data. To circumvent this, we apply a logarithmic transform on $\mathbf{X} + 1$ (we add the 1 to be slightly above the zero as we have nonnegativity constraints). The resulting range is $log(2) - 7.79$, and the sparsity of the data matrix is 78.01%.

(ii) **Med-MCI:** These data are also provided by IQVIA Inc. and similar to Med-HF, but they include patients with mild cognitive impairment (MCI) conditions. Similarly, we include $5,000$ patients and the total number of medical services is 412. We also apply a logarithmic transform on the data. The final range of data is $log(2) - 6.98$, with a 77.76% sparsity.

(iii) **Movielens:** Movielens [41] is a movie rating dataset and a popular baseline in recommender systems literature. It contains $\sim 10^5$ ratings. The data only include users with at least 20 ratings. We also filter out movies with less than 20 ratings. The total number of users is 943 and the total number of movies is $1,152$. The rating range is $1 - 5$, with 0.5 increments. The sparsity of this dataset is 90.98%.

(iv) **College Grades:** These data contain the grades of students from the College of Science and Engineering at the University of Minnesota spanning from Fall 2002 to Spring 2013. The total number of students is $5,703$, and the number of courses is 837. The grades take 11 discrete values (0, and 1 to 4 with increments of $1.\overline{33}$), and the sparsity of the data matrix is 96.28%.

4.4.2 Baselines

We compare to the plain NMF and following state-of-the-art methods from the literature: (i) **NMF:** nonnegative Matrix factorization (4.1) regularized with ($\|\mathbf{A}\|_F^2 + \|\mathbf{B}\|_F^2$), and implemented using ADMM [48], (ii) **BMF**: matrix factorization with rank-1 factors specified to capture items' and individuals' biases [58, 82]; implemented using Stochastic Gradient Descent (SGD). The is a well-known approach in recommender systems and is considered a state-of-the-art method in student grade prediction [7], (iii) **AdaError:** a collaborative filtering model based on matrix factorization with learning rate that adaptively adjusts based on the prediction error

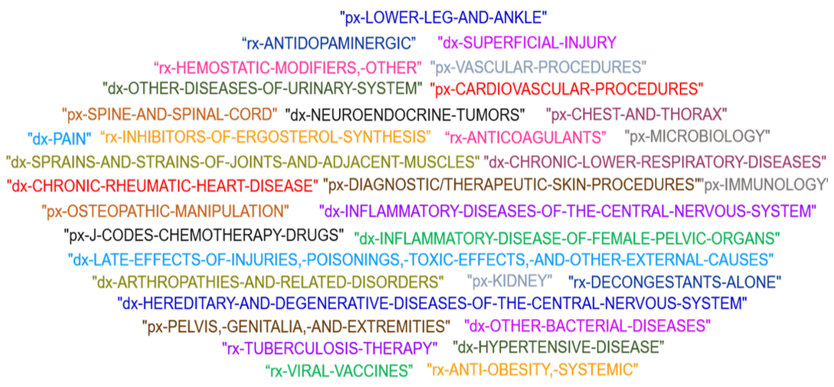
[64]. AdaError is reported to have a state-of-the-art results on MovieLens [88], (iv) **HSR:** a hierarchical structure recommender system model that captures the tree structure in items (users) via factorizing the item (user) embeddings matrix into a product of matrices, i.e., $\mathbf{X} = \mathbf{A}_1\mathbf{A}_2\ldots\mathbf{A}_P(\mathbf{B}_1\mathbf{B}_2\ldots\mathbf{B}_Q)^T$ [107, 108]. $\mathbf{B}_Q \in \mathbb{R}^{M_Q \times R}$ can be interpreted as the embedding of the coarsest items categories, whereas the matrices $\{\mathbf{B}_q \in \mathbb{R}^{M_q \times M_{q+1}}\}_{q=1}^{Q-1}$ indicate the affiliation of the M_q subcategories (or items) with the M_{q+1} coarser categories. Note that an item can belong to all the subcategories with different scales since no constraints are imposed on $\mathbf{B}_q$'s matrices (except for nonnegativity). The same analysis also applies to user embeddings. We are unaware of other algorithms that incorporate the tree structure while learning the embeddings simultaneously. We used the Matlab code sample provided by the authors for a 3-layer tree and generalized it to handle Q layers, and (v) **NMF+KM:** is a simple two-stage procedure where we first apply NMF, then we obtain the embeddings of the root nodes $\mathbf{B}_Q$ and the product of the assignment matrices $\mathbf{S} = \mathbf{S}_1 \ldots \mathbf{S}_{Q-1}$ via k-means' centroids and assignment variable, respectively – we include NMF+KM to demonstrate the advantage of learning the embeddings and tree structure simultaneously.

Table 4.1: Matrix Completion Errors. eTREE significantly improves the prediction accuracy.

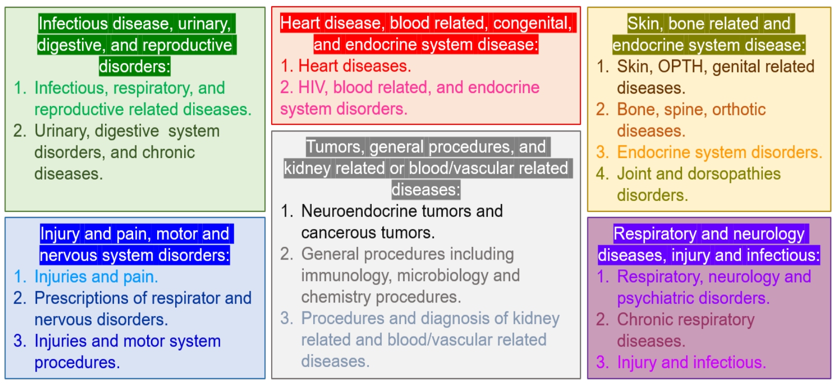
	BMF		**AdaError**		**HSR**		**NMF**		**eTREE**		**NMF+KM**	
Data	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**
Med-HF	0.9875	0.7721	0.9147	0.6858	0.9287	0.7094	0.9031	**0.6788**	**0.8873**	0.6808	1.0797	0.8094
Med-MCI	0.8034	0.6232	0.7468	0.5680	0.7807	0.5990	0.7445	0.5612	**0.7317**	**0.5611**	0.8781	0.6578
MovieLens	0.9300	0.7312	0.9123	0.7165	0.9216	0.7226	0.9286	0.7286	**0.9106**	**0.7136**	1.0182	0.8250
College Grades	0.5765	0.4254	0.5777	0.4206	0.5844	0.4339	0.5755	0.4229	**0.5601**	**0.4126**	0.5991	0.4476

4.4.3 Q1. Accuracy of Embeddings

The quality of embeddings can be evaluated by testing their performance with a particular task, e.g., classification or regression. Here we take a more generic approach and evaluate the embedding quality on matrix completion. The philosophy is: *if the embeddings predict missing data with high accuracy, then they must be good representations of items and individuals.* We split each dataset into 5 equal folds. After training the models on 4 folds (80% of the data), we test the trained models on the held-out fold. The hyper-parameters of all methods are chosen via cross validation (10% of training data). Due to random initialization, the results can differ for different runs. Thus, after choosing the hyper-parameters, we run the training and testing on each fold 20 times and report the average error of the total 100 experiments.


"px-LOWER-LEG-AND-ANKLE"
"rx-ANTIDOPAMINERGIC"
"dx-SUPERFICIAL-INJURY
"rx-HEMOSTATIC-MODIFIERS,-OTHER"
"px-VASCULAR-PROCEDURES"
"dx-OTHER-DISEASES-OF-URINARY-SYSTEM"
"px-CARDIOVASCULAR-PROCEDURES"
"px-SPINE-AND-SPINAL-CORD"
"dx-NEUROENDOCRINE-TUMORS"
"px-CHEST-AND-THORAX"
"dx-PAIN"
"rx-INHIBITORS-OF-ERGOSTEROL-SYNTHESIS"
"rx-ANTICOAGULANTS"
"px-MICROBIOLOGY"
"dx-SPRAINS-AND-STRAINS-OF-JOINTS-AND-ADJACENT-MUSCLES"
"dx-CHRONIC-LOWER-RESPIRATORY-DISEASES"
"dx-CHRONIC-RHEUMATIC-HEART-DISEASE"
"px-DIAGNOSTIC/THERAPEUTIC-SKIN-PROCEDURES"
"px-IMMUNOLOGY'
"px-OSTEOPATHIC-MANIPULATION"
"dx-INFLAMMATORY-DISEASES-OF-THE-CENTRAL-NERVOUS-SYSTEM"
"px-J-CODES-CHEMOTHERAPY-DRUGS"
"dx-INFLAMMATORY-DISEASE-OF-FEMALE-PELVIC-ORGANS"
"dx-LATE-EFFECTS-OF-INJURIES,-POISONINGS,-TOXIC-EFFECTS,-AND-OTHER-EXTERNAL-CAUSES"
"dx-ARTHROPATHIES-AND-RELATED-DISORDERS"
"px-KIDNEY"
"rx-DECONGESTANTS-ALONE"
"dx-HEREDITARY-AND-DEGENERATIVE-DISEASES-OF-THE-CENTRAL-NERVOUS-SYSTEM"
"px-PELVIS,-GENITALIA,-AND-EXTREMITIES"
"dx-OTHER-BACTERIAL-DISEASES"
"rx-TUBERCULOSIS-THERAPY"
"dx-HYPERTENSIVE-DISEASE"
"rx-VIRAL-VACCINES"
"rx-ANTI-OBESITY,-SYSTEMIC"

(a) Sample of the 412 medical services in Med-MCI.



Infectious disease, urinary, digestive, and reproductive disorders:
1. Infectious, respiratory, and reproductive related diseases.
2. Urinary, digestive system disorders, and chronic diseases.

Heart disease, blood related, congenital, and endocrine system disease:
1. Heart diseases.
2. HIV, blood related, and endocrine system disorders.

Skin, bone related and endocrine system disease:
1. Skin, OPTH, genital related diseases.
2. Bone, spine, orthotic diseases.
3. Endocrine system disorders.
4. Joint and dorsopathies disorders.

Tumors, general procedures, and kidney related or blood/vascular related diseases:
1. Neuroendocrine tumors and cancerous tumors.
2. General procedures including immunology, microbiology and chemistry procedures.
3. Procedures and diagnosis of kidney related and blood/vascular related diseases.

Injury and pain, motor and nervous system disorders:
1. Injuries and pain.
2. Prescriptions of respirator and nervous disorders.
3. Injuries and motor system procedures.

Respiratory and neurology diseases, injury and infectious:
1. Respiratory, neurology and psychiatric disorders.
2. Chronic respiratory diseases.
3. Injury and infectious.

(b) A tree learned by eTREE and labeled by domain experts.

Figure 4.2: eTREE Provides Meaningful Clusters.

Table 4.1 shows the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) of all methods on the different datasets. We highlight the smallest error in bold and underline the second smallest. eTREE significantly improves the best baseline with all datasets. Note that MovieLens and the grade datasets are challenging and an improvement in the second digit is considered significant in the literature. By comparing NMF and eTREE, we can conclude that the tree prior enhances the accuracy. Moreover, we can see the clear advantage of simultaneously learning the embeddings and tree clusters when we compare eTREE with MNF+KM. We point out that HSR baseline works better with MovieLens, compared to the medical datasets. This is likely because a movie usually belongs to a mix of genres, which suits the complete tree assumption in HSR. Nevertheless, the proposed tree formulation in eTREE provides better accuracy.

4.4.4 Q2. Interpretability of Learned Trees

We ran a 3-layer eTREE on Med-MCI with the following parameters: $R = 9$, $\lambda = 1$, $\mu = 50$ (more emphasis on the tree term), $M_2 = 27$ (number of subcategories), and $M_3 = 9$ (number of main categories). A sample of the 412 medical services is shown in Fig. 4.2 (a), where 'dx' stands for a diagnosis, 'rx' is a prescription, and 'px' is a procedure. Note that eTREE assigns each medical service to one subcategory, and each subcategory to a main category. Services with the *same* color in Fig. 4.2 (a) belong to the same subcategories, whereas services with *similar* colors (e.g., light and dark blue) belong to the same main category but to different subcategories. These unsupervised tree clusters were then shown to medical professionals. The domain experts were able to find coherence in the groups and they labeled both the main categories and their subcategories. The tables in Fig. 4.2 (b) shows the names of the main categories and their subcategories as labeled by the medical professionals – we show the top 6 coherent categories. Similar interpretability was observed on Med-HF data, but not shown due to space limitation.

4.5 Conclusions

In this chapter, we proposed eTREE, a framework that incorporates the tree structure while learning the embeddings of a data matrix. eTREE not only exploits the tree structure, but also learns the hierarchical clustering in an supervised fashion. We leveraged the special properties of NMF to prove the uniqueness of the proposed model. We employed ADMM, parallel computing, and computation caching to derive a lightweight algorithm with scalable implementation to solve eTREE. We showed the effectiveness and interpretability of eTREE on real data.

Chapter 5

Explainable Embeddings for Feature-based Collaborative Filtering

5.1 Introduction

In the context of recommendation engines, collaborative filtering (CF) is the process of filtering information using techniques involving collaboration among multiple viewpoints. CF models can be divided into *neighbor-based* and *feature-based* (e.g., latent factor) categories; latent factor methods have been the state-of-the art in CF. One of the very successful latent factor CF techniques is matrix factorization (MF) due to its ability to capture correlations and higher-order statistical dependencies across dimensions. MF automatically predicts a person's affinity for items by connecting that person's historical interests with the interests of similar users, while taking inter-dependencies among items into account. More specifically, given a sparse user $\times$ item rating matrix, MF uses the observed ratings to learn dense latent representations (embeddings) of users and items in a lower dimensional space. In the inference phase, the *unknown* entry corresponding to the i^{th} user and j^{th} item is predicted by the dot product of their embeddings. The more similar the user's and item's embeddings (closer to each other in the latent space), the larger their dot product (predicted rating). Although this provides a geometric interpretation of the prediction of MF, we still cannot explain how the latent vectors are formed. MF methods tend to be black-box machine learning models that lack interpretability and do not provide a straightforward explanation for their predictions; this is the main drawback of latent factor methods compared to neighbor-based CF.

Researchers have recently found that interpretations and explainability in recommendation systems play a significant role to improve the transparency, persuasiveness, effectiveness, trustworthiness, and user satisfaction [121]. They also enable system designers to diagnose, debug, and refine the recommendation algorithm. Interpretable recommendations are of interest in many applications, especially in business-to-business (B2B) scenarios where the recipient of the recommendation is a salesperson responsible for the client. A salesperson has to decide whether to pursue a sales opportunity (i.e., recommendation), and (s)he relies on evaluating the *reasoning* behind a generated recommendation [43]. Explainable recommendations have also been proven effective in business-to-client (B2C) e-commerce settings [122].

In this chapter, we propose XPL-CF, a CF approach that augments the classical MF model with a new type of prior information. The proposed prior not only improves the prediction accuracy of MF, but it also underpins the latent factors and explains how the resulting recommendations are formed. Unlike most recent explainable recommendation methods, XPL-CF does not require additional data. The main intuition behind our modeling is that a user preference profile (latent factor) is determined by their experience with a subset of items. The strength of this association can differ, e.g., a user might strongly associate herself with Sci-Fi movies and mildly with horror movies. Our proposed prior encodes a user's embedding as a sparse linear combination of item embeddings. Conversely, an item's embedding is determined by a subset of users (i.e., a sparse linear combination of user embeddings). We demonstrate the effectiveness of the proposed model on real datasets from investment and recommender system domains.

5.2 Related Work

Explainable recommendation methods can be grouped into two broad types: post-hoc and embedded methods. In post-hoc approaches, explanations and recommendations are generated from separate models [67, 80, 86]. Embedded methods, on the other hand, aim to explain the recommendation model itself [59, 122]. Here we focus on the embedded category; we refer the reader to [121] for an in-depth literature review. In the case of neighbor-based CF methods, the recommendations are directly based on similarities between users and/or items [91], which also serve to explain the recommendation in a rather straightforward way - but these methods are far from the state-of-art in terms of quality of recommendation. The explanation task is trickier with latent factor models. Their internal decision processes cannot be directly interpreted by humans, since by finding lower dimensional representations of users and items they abstract away from the interactions between users and items [86]. The two predominant approaches in the recommendation literature are: **i)** adding constraints to the latent factor models (our approach belongs to this class), and **ii)** using external data.

In the latter category, **external data** such as product reviews (e.g., in TF/IDF form) and rating data are jointly factored with shared latent factors via topic modelling [14] or coupled MF [122]. For instance, in the case of topic modeling, the learned latent topics can be leveraged to provide an interpretation of the latent factors. Although exploiting additional information provides valuable insight, such information is not always available – especially in B2B settings. Moreover, if the additional sources used for explanation are not correlated with the rating data, then the explanations will not accurately reflect the reasons for the recommendation and will degrade the rating prediction accuracy.

Heckel et al. [43] proposed a **constrained latent factor** model that explicitly detects the user's and item's participation to overlapping co-clusters. Their model is designed to predict the probability of a user/item to belong to a cluster; thus, it cannot predict values (e.g., ratings). Closest to our work is the Explainable MF (EMF) approach in [1]. EMF modifies the cost function of MF by penalizing the Euclidean distance of the latent vectors of similar users and items. The similarity is predefined by a user $\times$ item similarity matrix and is measured by the ratio of the neighbors of user i who have rated item j – the neighborhood is calculated using cosine similarity. EMF is essentially a hybrid method between neighbor- and feature-based CF. Although EMF has the advantage of not requiring extra data views to generate explanations, it still employs a rather restrictive predefined neighborhood model. We point out that EMF explains the recommendation via the distance in the latent space and does not attempt to explain the embeddings of users/items. XPL-CF, on the other hand, explains the embedding of a user in relation to item embeddings and vice versa. In contrast to [1], the explainability relationships in XPL-CF are automatically revealed by the model and not predefined apriori.

5.3 Proposed Method

5.3.1 Formulation

Assume we have a data matrix $\mathbf{X} \in R^{N \times M}$, with the user $\times$ item rating data. The matrix factorization CF models assume that $\mathbf{X}$ can be approximated using low-rank factor matrices, i.e.,

$\mathbf{X} \approx \mathbf{A}\mathbf{B}^T$, where rows of $\mathbf{A} \in \mathbb{R}^{N\times R}$ and $\mathbf{B} \in \mathbb{R}^{M\times R}$ are the embeddings of users and items, respectively, and $R \leq \min(N, M)$ is the matrix rank. After obtaining $\mathbf{A}$ and $\mathbf{B}$, the unknown rating of the i^{th} user for the j^{th} item is predicted by the dot product of their embeddings, i.e., $\mathbf{X}(i, j) = \mathbf{a}_i\mathbf{b}_j^T$. In other words, the MF model produces latent representations of users and items in a lower dimensional space. If a user likes an item, the distance between their embeddings will be small and therefore their dot product is larger.

In the original data domain, the user-item relationships are clear: users are represented by their ratings of a subset of items, and items are represented by ratings given by a subset of users. However, the user-item relationships are not clear in the latent space. Why is the embedding of a user (or item) more/less similar to certain items (users)? Our framework addresses this question. In our proposed formulation, we rely on MF to obtain user and item embeddings and impose a prior on these embeddings. The prior encodes each user's embedding as a sparse linear combination of item embeddings, and vice versa for each item embedding. This leads to the following problem formulation.

$$\begin{aligned} \min_{\mathbf{A},\mathbf{B},\mathbf{S},\mathbf{Z}} \quad & \|\mathbf{\Omega} \odot (\mathbf{X} - \mathbf{A}\mathbf{B}^T)\|_F^2 + \mu_a\|\mathbf{A} - \mathbf{S}\mathbf{B}\|_F^2 \\ & + \mu_b\|\mathbf{B} - \mathbf{Z}\mathbf{A}\|_F^2 + \lambda\mathbf{1}^T(\mathbf{S} + \mathbf{Z}^T)\mathbf{1} \\ \text{s.t.} \quad & \mathbf{S}, \mathbf{Z}^T \geq \mathbf{0} \end{aligned} \tag{5.1}$$

where $\odot$ is the element-wise product, and $\mathbf{\Omega}$ is a zero-one matrix indicating the availability of the corresponding entries in $\mathbf{X}$. $\mathbf{1}$ is a vector of all ones of the appropriate size and $\mu_a \geq 0$, $\mu_b \geq 0$, and $\lambda \geq 0$ are regularization hyper-parameters. The first term in (5.1) is the least squares data fitting, while the second and third terms represent the user-item relationships in the latent space. The last term is introduced to promote sparsity in $\mathbf{S}$ and $\mathbf{Z}$ (l_1 norm with non-negativity boils down to the sum of entries).

The variables $\mathbf{S}$ and $\mathbf{Z}$ reveal the user-item relationships in the latent space and explain the resulting recommendations. For easier interpretability, we model $\mathbf{S}$ and $\mathbf{Z}$ as element-wise non-negative. Assume that $\mathbf{a}_i$ and $\mathbf{s}_i$ are rows in $\mathbf{A}$ and $\mathbf{S}$, respectively. Then, $\mathbf{a}_i = \mathbf{s}_i\mathbf{B}$ and $\mathbf{s}_i$ is a *sparse* vector that selects (and scales) some item embeddings to form user i's embedding. The motivation behind this assumption is that the features that the user cares about are characterized by her experience and knowledge about a subset of items. Similarly, $\mathbf{b}_j = \mathbf{z}_j\mathbf{A}$ assumes that the item embedding is characterized by a subset of user embeddings.

5.3.2 Explainability Analysis

In this subsection, we present how XPL-CF can be used to explain a recommendation. The prediction of a value in $\mathbf{X}$, $\hat{x}_{ij}$, is

$$\hat{x}_{ij} = \mathbf{a}_i \mathbf{b}_j^T \approx \mathbf{s}_i \mathbf{B} \mathbf{b}_j^T \tag{5.2a}$$

$$\approx \mathbf{s}_i \mathbf{Z} \mathbf{A} \mathbf{A}^T \mathbf{z}_j^T = \mathbf{u}_i \mathbf{A} \mathbf{A}^T \mathbf{z}_j^T \tag{5.2b}$$

where in (5.2a), the recommendation boils down to the similarity between the target item $\mathbf{b}_j$ and a subset of items selected by $\mathbf{s}_i$. Because $\mathbf{s}_i$ is fixed across all items for user i, we can interpret this subset of items as the "lens" that user i sees all items through. Equation (5.2b) provides another intriguing insight by explaining the prediction as a (sparse) linear combination of user $\times$ user similarity encoded in $\mathbf{A}\mathbf{A}^T$—note that vector $\mathbf{u}_i := \mathbf{s}_i \mathbf{Z}$ may be dense. In the same vein, we can write

$$\hat{x}_{ij} = \mathbf{b}_j \mathbf{a}_i^T \approx \mathbf{z}_j \mathbf{A} \mathbf{a}_i^T \tag{5.3a}$$

$$\approx \mathbf{z}_j \mathbf{S} \mathbf{B} (\mathbf{s}_i \mathbf{B})^T = \mathbf{v}_j \mathbf{B} \mathbf{B}^T \mathbf{s}_i^T \tag{5.3b}$$

where $\mathbf{v}_j := \mathbf{z}_j \mathbf{S}$. The prediction in (5.3a) is explained as the similarity between the target user with a subset of users selected by the model, whereas (5.3b) explains the prediction as a (sparse) linear combination of item $\times$ item similarity. Combining (5.2a) and (5.3a), we can say

$$\hat{x}_{ij} = \mathbf{a}_i \mathbf{b}_j^T \approx \mathbf{s}_i \mathbf{B} (\mathbf{z}_j \mathbf{A})^T = \mathbf{s}_i \mathbf{B} \mathbf{A}^T \mathbf{z}_j^T \tag{5.4}$$

where each prediction is explained through a sparse linear combination of user-item similarity encoded in $\mathbf{B}\mathbf{A}^T$. Thus, the explanation associated with a recommendation can list the items (and users if applicable) that contribute to the prediction the most (i.e., items with highest values in $\mathbf{s}_i$).

Another benefit of $\mathbf{S}$ and $\mathbf{Z}$ is that they can be used to extract communities *in the latent space*. For instance, a community includes users whose embeddings are characterized by the same items; however, this is out of the scope of this chapter and we leave it for future work.

5.3.3 Model Engineering

We add two modifications to the problem formulation in (5.1). The first point we address is the scaling between the low-rank factors $\mathbf{A}$ and $\mathbf{B}$. Since the embedding of a user is a linear combination of item embeddings and vice versa, it is important for $\mathbf{A}$ and $\mathbf{B}$ to be within the same scale. Thus, we constrain the columns of $\mathbf{A}$ and $\mathbf{B}$ to be on the unit l_2 norm ball. We introduce a diagonal matrix to allow us to fix the scale without loss of generality of the factorization model [113], i.e., $\mathbf{X} \approx \mathbf{ADB}^T$, where $\mathbf{D}$ is a diagonal matrix.

The second addition to the model is the user and item bias terms. These biases capture how well an item is rated compared to the average, across all items. Similarly, a user's bias corresponds to the user's tendency to give better/worse ratings relative to the average. Taking these points into account, we obtain the following:

$$\begin{aligned}\min_{\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{a},\mathbf{b},\mathbf{S},\mathbf{Z}} \quad & \|\mathbf{\Omega} \odot (\mathbf{X} - \mathbf{ADB}^T - \mathbf{a1}^T - \mathbf{1b}^T)\|_F^2 + \lambda \mathbf{1}^T(\mathbf{S} + \mathbf{Z}^T)\mathbf{1} \\ & + \mu_a \|\mathbf{A} - \mathbf{SB}\|_F^2 + \mu_b \|\mathbf{B} - \mathbf{ZA}\|_F^2 + \eta(\|\mathbf{d}\|^2 + \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2) \\ \text{s.t.} \quad & \mathbf{S}, \mathbf{Z}^T \geq \mathbf{0}, \ \ \mathbf{D} = \text{Diag}(\mathbf{d}) \\ & \|\mathbf{A}(:,r)\|_2 = \|\mathbf{B}(:,r)\|_2 = 1, \ \forall r \in [R]\end{aligned} \tag{5.5}$$

5.3.4 Optimization

The formulation in (5.5) is non-convex and a very challenging optimization problem. An additional challenge stems from the fact that $\mathbf{X}$ is partially observed. We employ a carefully designed alternating optimization (AO) algorithm. The proposed algorithm leverages the Alternating Direction Method of Multipliers (ADMM) and utilizes parallel computing, computation caching, and warm-start to provide a scalable and efficient implementation[1]. The high level algorithmic strategy is to employ AO to update $\mathbf{A}$, $\mathbf{B}$, $\mathbf{d}$, $\mathbf{a}$, $\mathbf{b}$, $\mathbf{S}$ and $\mathbf{Z}$ one at a time, while fixing the others. Let us consider the subproblem w.r.t. $\mathbf{A}$. We introduce an auxiliary variable $\widetilde{\mathbf{A}}$ to handle the unit $l2$ norm ball constraint. The ADMM updates for $\mathbf{A}$ are:

$$\begin{aligned}\widetilde{\mathbf{A}}(:,i) \leftarrow \min_{\widetilde{\mathbf{A}}(:,i)} \quad & \frac{1}{2}\|\mathbf{X}_s(i,\mathcal{J}_i)^T - \widetilde{\mathbf{B}}(\mathcal{J}_i,:)\widetilde{\mathbf{A}}(:,i)\|_F^2 + \frac{\mu_a}{2}\|\widetilde{\mathbf{A}}(:,i)^T - \\ & \mathbf{S}(i,:)\mathbf{B}\|_F^2 + \frac{\rho_a}{2}\|\mathbf{A}(i,:) - \widetilde{\mathbf{A}}(:,i)^T + \mathbf{U}(i,:)\|_F^2, \ \ \forall i \in [N]\end{aligned} \tag{5.6a}$$

[1]Code is available at `https://github.com/FaisalAlmutairi/explainable_recommendation`.

$$\mathbf{A}(:,r) \leftarrow \mathbf{A}_u(:,r)/\|\mathbf{A}_u(:,r)\|_2, \ \ \forall r \in [R] \tag{5.6b}$$

$$\mathbf{U} \leftarrow \mathbf{U} + \mathbf{A} - \widetilde{\mathbf{A}}^T \tag{5.6c}$$

where $\mathbf{X}_s = \mathbf{X} - \mathbf{a}\mathbf{1}^T - \mathbf{1}\mathbf{b}^T$, $\widetilde{\mathbf{B}} = \mathbf{BD}$, $\mathbf{A}_u = \widetilde{\mathbf{A}}^T - \mathbf{U}$ and $\mathcal{J}_i$ is the set of items that have observations for user i. Equation (5.6b) is a simple column scaling, whereas (5.6c) is the dual variable update. Problem (5.6a) is a *weighted* least squares problem (weighted by the binary matrix $\mathbf{\Omega}$). An important implication is that (5.6a) corresponds to solving N separable least squares problems, which enables parallel computation. One point that requires more care is handling the missing entries in $\mathbf{X}$ (or the zeros in $\mathbf{\Omega}$). The way we handle this is by removing the equations that correspond to the indices of the missing entries, i.e., we remove rows in $\widetilde{\mathbf{B}}$ and entries in $\mathbf{X}_s(i,:)$ when solving for each $\widetilde{\mathbf{A}}(:,i)$. Moreover, for each least squares problem, we do not compute the matrix inversion explicitly. Instead, the Cholesky decomposition of a Gram matrix is computed. Then, back and forward substitution steps are performed to obtain $\widetilde{\mathbf{A}}(:,i)$. Matrix $\mathbf{B}$ is updated using the ADMM in the same fashion as $\mathbf{A}$ with the appropriate transpose.

Next, we update vector $\mathbf{d}$ by by minimizing $(\|\mathbf{x}(\mathcal{T}) - \mathbf{K}(\mathcal{T},:)\mathbf{d}\|_F^2 + \eta\|\mathbf{d}\|_2^2)$ w.r.t $\mathbf{d}$, where $\mathbf{K} = \mathbf{B} \otimes \mathbf{A}$, $\otimes$ is the Khatri–Rao product, $\mathbf{x} = \text{vec}(\mathbf{X}_s)$ and $\mathcal{T}$ is the set of observed entries in $\mathbf{x}$.

Next, we update the bias variables for users and items. The update for the bias of user i, $\mathbf{a}(i)$, corresponds to solving:

$$\min_{\mathbf{a}(i)} \quad \frac{1}{2}\|\mathbf{X}_b(i,\mathcal{J}_i)^T - \mathbf{1}\mathbf{a}(i)\|_F^2 + \frac{\eta}{2}(\mathbf{a}(i))^2 \tag{5.7}$$

where $\mathbf{X}_b = \mathbf{X} - \mathbf{ADB}^T - \mathbf{1}\mathbf{b}^T$. The items' biases in $\mathbf{b}$ are updated similarly. Note that the updates of the bias variables across users (and items) are independent; thus, they can be computed in parallel.

Finally, we update the latent mapping variables $\mathbf{S}$ and $\mathbf{Z}$ using the ADMM (we present the update of $\mathbf{S}$ as a running example). We omit the terms in (5.5) that do not include $\mathbf{S}$ and introduce an auxiliary variable $\widetilde{\mathbf{S}}$ to split the effort of handling the least squares terms and the non-negativity constraint. The ADMM updates for the resulting problem are the following:

$$\widetilde{\mathbf{S}} \leftarrow \min_{\widetilde{\mathbf{S}}} \quad \frac{\mu_a}{2}\|\mathbf{A}^T - \mathbf{B}^T\widetilde{\mathbf{S}}\|_F^2 + \lambda\mathbf{1}^T\widetilde{\mathbf{S}}\mathbf{1} + \frac{\rho_s}{2}\|\mathbf{S} - \widetilde{\mathbf{S}}^T + \mathbf{V}\|_F^2 \tag{5.8a}$$

$$\mathbf{S} \leftarrow \underset{\mathbf{S}\geq\mathbf{0}}{\arg\min} \quad \|\mathbf{S} - \widetilde{\mathbf{S}}^T + \mathbf{V}\|_F^2 \tag{5.8b}$$

$$\mathbf{V} \leftarrow \mathbf{V} + \mathbf{S} - \widetilde{\mathbf{S}}^T \tag{5.8c}$$

Equation (5.8b) is a simple element-wise non-negative projection (i.e., zero out the negative elements in $\widetilde{\mathbf{S}}^T - \mathbf{V}$). Equation (5.8c) is the dual variable update. Similar to the case of $\mathbf{A}$, the update in (5.8a) corresponds to solving N separable least squares problems that can be solved in parallel. Unlike (5.6a), the N problems in (5.8a) share the same mixing matrix $\mathbf{B}^T$. This means that we need to compute the Cholesky decomposition of $(\mu_a\mathbf{B}\mathbf{B}^T + \rho_s\mathbf{I})$ only once.

5.4 Experimental Results

5.4.1 Datasets

We evaluate XPL-CF using the following datasets.

i) B2B [114], an investor holding-position dataset (an example of B2B applications). The data are organized into a company vs investor matrix where the entries are the percentage of shares that one investor holds in each company among all the shares issued. We use the data as collected and preprocessed in [114].

ii) ML100K [41], a movie rating dataset known as MovieLens and a popular baseline in recommender systems literature. It contains $\sim 10^5$ ratings. The original data only include users with at least 20 ratings. We also filter out movies with less than 20 ratings. The final number of users is 943 and the total number of movies is $1,152$. The sparsity of this dataset is 90.98%.

5.4.2 Baselines

We evaluate XPL-CF against the following baselines.
i) BMF, a matrix factorization approach with rank-1 factors specified to capture items' and users' biases [58, 82] implemented using Stochastic Gradient Descent (SGD). Our approach in (5.5) boils down to BMF when $\mu_a = \mu_b = 0$.

ii) AdaErr, a CF model based on MF with a learning rate that adaptively adjusts based on the prediction error [64]. AdaErr is reported to have state-of-the-art results on ML100K (MovieLens)

Table 5.1: Matrix completion error of all methods.

		R = 10		R = 50		R = 100	
Data	Model	RMSE	MAE	RMSE	MAE	RMSE	MAE
B2B	BMF	**1.2026**	**0.8547**	1.2100	0.8565	1.2334	0.8704
	AdaErr	1.4536	1.0230	1.4391	1.0142	1.4593	1.0496
	EMF	1.2373	0.8843	1.2360	0.8821	1.2231	0.8598
	XPL-CF	1.2482	0.8779	**1.1915**	**0.8392**	**1.1892**	**0.8339**
ML100K	BMF	0.9228	0.7261	0.9188	0.7245	0.9158	0.7228
	AdaErr	0.9432	0.7514	0.9326	0.7422	1.1119	0.9244
	EMF	0.9393	0.7504	0.9355	0.7491	0.9339	0.7479
	XPL-CF	**0.9123**	**0.7150**	**0.9132**	**0.7182**	**0.9156**	**0.7166**

Table 5.2: Explanability evaluation using ML100K (R = 10).

	k = 10		k = 15		k = 20	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
BMF	0.9228	0.7261	0.9228	0.7261	0.9228	0.7261
BMF-RandU	0.9342	0.7348	0.9433	0.7412	0.9521	0.7476
BMF-S	0.9452	0.7432	0.9598	0.7544	0.9756	0.7660
BMF-RandI	0.9414	0.7383	0.9550	0.7467	0.9705	0.7565
BMF-Z	0.9569	0.7498	0.9822	0.7659	1.0064	0.7815

[88]. **iii) EMF**, an explainable CF model based on MF [1]; see Sec. 5.2 for more details.

5.4.3 Matrix Completion

In order to evaluate the quality of the embeddings, we take a generic approach by evaluating the embedding quality on the matrix completion task. The philosophy is: *if the embeddings predict missing data with high accuracy, then they must be good representations of items and users*. Accurate prediction, e.g., predicting holding-positions in the B2B dataset, not only gives a relative ranking of the likelihood of interest, but it also enables deriving useful information (e.g., percentage of investment).

We split each dataset into 5 equal folds. After training the models on 4 folds, we test the trained models on the held-out fold. The hyper-parameters of all methods are chosen via cross validation (10% of training data). Due to random initialization, the results can differ for different runs; thus, after choosing the hyper-parameters, we run the training and testing on each fold 20 times and report the average error of the total 100 experiments.

Table 5.1 shows the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE)

Table 5.3: Lists of movies that explain the prediction of "Get Shorty".

User 1 ($U1$)	User 2 ($U2$)
Alphaville	Showgirls
Showgirls	Ready to Wear (Pret-A-Porter)
Striking Distance	Vampire in Brooklyn
Dead Presidents	Miami Rhapsody
Bloodsport 2	Party Girl
Fair Game	The Fog
High School High	Four Days in September
Steel	Little Big League
The Jackal	Free Willy 3: The Rescue
April Fool's Day	Exit to Eden

using B2B and ML100K data with various ranks R (number of features). Explainable methods usually suffer from accuracy-interpretability trade-off, which can be seen by comparing the explainable method EMF and BMF. Nevertheless, XPL-CF significantly improves all the baselines, especially when $R = 10$ with ML100K and when $R = \{50, 100\}$ with B2B. The fact that XPL-CF improves BMF suggests that the data follow the proposed prior.

5.4.4 Explainability Evaluation

There is no well-defined methodology for evaluating the model's explainability. There are two main approaches in the literature: online and offline. Online evaluation tests the performance by adding explanation to the recommendation loop on a live recommendation platform, e.g., e-commerce website [103, 122]. Offline evaluation usually either quantifies the importance of the explanation provided by the model [1, 59], or demonstrates the quality of the explainability by examples [47, 93]—we adopt both strategies. Following the approach in [59], we remove the k ratings in the training data with the highest values in $\mathbf{s}_i$ (for each user). Then, we train a BMF model using the resulting training set—we call this model BMF-S. We perform the same strategy and remove the k ratings with the highest values in $\mathbf{z}_j$ for each item (we call it BMF-Z). Table 5.2 shows that the performance degradation of BMF-S and BMF-Z (relative to BMF) is significantly higher compared to when we *randomly* remove k training ratings from each user (BMF-RandU) or from each item (BMF-RandI). This suggests that the items (users) identified by $\mathbf{S}$ ($\mathbf{Z}$) are important in defining a user (item).

We chose two users from ML100K data: $U1$ who has a clear interest in action, adventure and thriller movies and $U2$ who is more interested in comedy and romance - we determine their interest based on movies they have rated. Table 5.3 shows the list of movies that explain the rating prediction for "Get Shorty" for these two users. To generate these explanations, we selected the top 20 movies with the highest values in $\mathbf{s}_i$ for $U1$ and $U2$ (we denote these sets as $\mathcal{S}_{U1}$ and $\mathcal{S}_{U2}$, respectively). Then, in Table 5.3, we list the top 10 movies with the highest values in $\mathbf{B}(\mathcal{S}_{U1}, :)\mathbf{b}_j^T$ for $U1$ (and similarly for $U2$). These explanations are based on (5.2a); note that in this case $\mathbf{s}_i$ is user-specific, while $\mathbf{b}_j$ is item-specific. Get Shorty is an action *and* comedy movie. We highlight action/adventure/thriller movies in red, while comedy movies are in blue. One can see that the prediction is explained from an action viewpoint for $U1$, while it is explained by comedy movies for $U2$. Note that our model uses the rating data only and does not have access to the movies' genres.

5.5 Conclusions

In this chapter, we proposed XPL-CF, a CF model that augments the classical MF framework for CF with a prior that encodes each user's embedding as a sparse linear combination of item embeddings, and vice versa for each item embedding. XPL-CF not only improves the prediction accuracy of MF, but it also automatically reveals the user-item relationships in the latent space (without requiring additional data). These relationships underpin the latent factors and explain how the resulting recommendations are formed.

Chapter 6

Dissertation Summary and Future Directions

In this dissertation, we developed frameworks for data reconstruction and completion from aggregated and partial observations. We developed concise models that are intuitive, insightful and effective. We derived efficient algorithms and scalable implementation to solve the proposed formulations. We addressed important and challenging problems that arise in real world applications and provided thorough experimental examination.

6.1 Summary

In chapter 2 we tackled the time series disaggregation problem from a new perspective. The proposed approach exploits an alternative representation of the time series and finds the spectrum of the target series (DCT coefficients). We showed that real world time series have sparse spectrum representations as most of the energy is compacted in the coefficients of low frequencies. The proposed algorithm (HOMERUN) is parameter-free, and it adapts to the input signal, i.e., it automatically detects the prominent periodicities in the data without the need of assuming any known periodicity. We derived a light weight and memory efficient algorithmic steps and demonstrated the effectiveness of the approach using real epidemiological datasets.

In chapter 3, we proposed a novel framework, called PREMA, for fusing multiple aggregated views of multidimensional (tensor) data. The proposed method leverages the properties of tensors in estimating the low-rank factors of the target data in higher resolution. The assumed model is provably transforming a highly ill-posed problem to an identifiable one. PREMA works with partially observed data, and can disaggregate effectively, even without any knowledge of the aggregation mechanism (B-PREMA). Experimental results on real data show that the proposed algorithm is very effective, even in challenging scenarios, such as data with double aggregation and high level of aggregation.

In chapter 4, we proposed eTREE, a framework that incorporates the tree structure while learning the embeddings of a data matrix. eTREE not only exploits the tree structure, but also learns the hierarchical clustering in an supervised fashion. We leveraged the special properties of NMF to prove the uniqueness of the proposed model. We employed ADMM, parallel computing, and computation caching to derive a lightweight algorithm with scalable implementation to solve eTREE. We showed the effectiveness and interpretability of eTREE on real data.

In chapter 5, we proposed XPL-CF, a CF model that augments the classical MF framework for CF with a prior that encodes each user's embedding as a sparse linear combination of item embeddings, and vice versa for each item embedding. XPL-CF not only improves the prediction accuracy of MF, but it also automatically reveals the user-item relationships in the latent space (without requiring additional data). These relationships underpin the latent factors and explain how the resulting recommendations are formed.

6.2 Future Directions

- **Tree-structured Non-linear embeddings:** In chapter 4, we imposed a tree clustering structure on item embeddings that are learned by a NMF term. A limitation of this approach is that the low-dimensional representations are learned through a linear mapping from the data domain. Neural network methods such as autoencoders have been successfully employed to learn nonlinear mappings from the data domain to high quality low-dimensional latent spaces. Our goal is to generalize our eTREE formulation to handle nonlinear latent representations. We also intend to explore other types of tree structures, such as soft tree clustering (e.g., an item can belong to more than one category).

- **Explainable embeddings for tensor data:** In chapter 5, we introduced a prior that encodes the embedding of a user to be a linear combination of item embeddings (and vice versa for each item embedding). We showed that the proposed prior not only improves the prediction of missing entries, but also provides an intriguing explainability to these predictions. We plan to generalize the factorization term to the case of multi-dimensional arrays (tensors). Using this generalization, we intend to explore applications in knowledge graphs, education and medical data. For example: What are the symptoms/diagnosis that explain the prediction of a specific disease? What are the questions/skills that determine the predictions of whether or not a student is likely to answer a question?

References

[1] B. Abdollahi and O. Nasraoui, "Explainable matrix factorization for collaborative filtering," in *Proceedings of the 25th International Conference Companion on World Wide Web*, 2016, pp. 5–6.

[2] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[3] F. M. Almutairi, C. I. Kanatsoulis, and N. D. Sidiropoulos, "Tendi: Tensor disaggregation from multiple coarse views," in *Proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2020)*, Singapore, May 2020.

[4] ——, "Prema: Principled tensor data recovery from multiple aggregated views," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 3, pp. 535–549, 2021.

[5] F. M. Almutairi, A. Konar, and N. D. Sidiropoulos, "Scalable energy disaggregation via successive submodular approximation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2676–2680.

[6] F. M. Almutairi, A. Konar, A. S. Zamzam, and N. D. Sidiropoulos, "Phased: Phase-aware submodularity-based energy disaggregation," in *Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring*, 2020, pp. 94–98.

[7] F. M. Almutairi, N. D. Sidiropoulos, and G. Karypis, "Context-aware recommendation-based learning analytics using tensor and coupled matrix factorization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 5, pp. 729–741, 2017.

[8] F. M. Almutairi, N. D. Sidiropoulos, and B. Yang, “Xpl-cf: Explainable embeddings for feature-based collaborative filtering,” in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021.

[9] F. M. Almutairi, Y. Wang, D. Wang, E. Zhao, and N. D. Sidiropoulos, “etree: Learning tree-structured embeddings,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, 2021, pp. 6609–6617.

[10] F. M. Almutairi, F. Yang, H. A. Song, C. Faloutsos, N. Sidiropoulos, and V. Zadorozhny, “Homerun: scalable sparse-spectrum reconstruction of aggregated historical data,” *Proceedings of the VLDB Endowment*, vol. 11, no. 11, pp. 1496–1508, 2018.

[11] E. Amaldi and V. Kann, “The complexity and approximability of finding maximum feasible subsystems of linear relations,” *Theoretical computer science*, vol. 147, no. 1-2, pp. 181–210, 1995.

[12] S. Arora, R. Ge, Y. Halpern, D. Mimno, A. Moitra, D. Sontag, Y. Wu, and M. Zhu, “A practical algorithm for topic modeling with provable guarantees,” in *International Conference on Machine Learning*, 2013, pp. 280–288.

[13] M. Ashraphijuo and X. Wang, “Fundamental conditions for low-cp-rank tensor completion,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2116–2145, 2017.

[14] Y. Bao, H. Fang, and J. Zhang, “Topicmf: Simultaneously exploiting ratings and reviews for recommendation,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, 2014.

[15] J. Bleiholder and F. Naumann, “Data fusion,” *ACM Computing Surveys (CSUR)*, vol. 41, no. 1, pp. 1–41, Jan. 2009.

[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*, vol. 3, no. 1, pp. 1–122, 2011.

[17] E. J. Candès, “Compressive sampling,” in *Proceedings of the international congress of mathematicians*, vol. 3. Madrid, Spain, 2006, pp. 1433–1452.

[18] E. J. Candes, J. K. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” *Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[19] E. J. Candès and M. B. Wakin, “An introduction to compressive sampling,” *IEEE signal processing magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[20] S. S. Chen, D. L. Donoho, and M. A. Saunders, “Atomic decomposition by basis pursuit,” *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[21] L. Chiantini and G. Ottaviani, “On generic identifiability of 3-tensors of small rank,” *SIAM J. on Matrix Analys. and App.*, vol. 33, no. 3, pp. 1018–1037, 2012.

[22] G. C. Chow and A.-l. Lin, “Best linear unbiased interpolation, distribution, and extrapolation of time series by related series,” *The Review of Econ. and Stats.*, pp. 372–375, 1971.

[23] W. A. Clark and K. L. Avery, “The effects of data aggregation in statistical analysis,” *Geo. Analysis*, vol. 8, no. 4, pp. 428–438, 1976.

[24] E. Cohen and H. Kaplan, “Bottom-k sketches: Better and more efficient estimation of aggregates,” in *ACM SIGMETRICS Performance Evaluation Review*, vol. 35, no. 1. ACM, 2007, pp. 353–354.

[25] D. Cole, “The effects of student-faculty interactions on minority students’ college grades: Differences between aggregated and disaggregated data.” *J. of the Professoriate*, vol. 3, no. 2, 2010.

[26] G. Cormode, M. Garofalakis, P. J. Haas, and C. Jermaine, “Synopses for massive data: Samples, histograms, wavelets, sketches,” *Foundations and Trends in Databases*, vol. 4, no. 1–3, pp. 1–294, 2012.

[27] T. Di Fonzo, “The estimation of m disaggregate time series when contemporaneous and temporal aggregates are known,” *The Rev. of Econ. and Stats.*, pp. 178–182, 1990.

[28] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, “Querying and mining of time series data: experimental comparison of representations and distance measures,” *PVLDB*, vol. 1, no. 2, pp. 1542–1552, 2008.

[29] X. L. Dong and F. Naumann, “Data fusion: resolving data conflicts for integration,” *PVLDB*, vol. 2, no. 2, pp. 1654–1655, 2009.

[30] D. L. Donoho and M. Elad, “Optimally sparse representation in general (nonorthogonal) dictionaries via $l1$ minimization,” *Proceedings of the National Academy of Sciences*, vol. 100, no. 5, pp. 2197–2202, 2003.

[31] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *IEEE Transactions on Image processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[32] R. H. Ellaway, M. V. Pusic, R. M. Galbraith, and T. Cameron, “Developing the role of big data and analytics in health professional education,” *Medical teacher*, vol. 36, no. 3, pp. 216–222, 2014.

[33] Z. Erkin, J. R. Troncoso-Pastoriza, R. L. Lagendijk, and F. Pérez-González, “Privacy-preserving data aggregation in smart metering systems: An overview,” *IEEE Signal Process. Mag.*, vol. 30, no. 2, pp. 75–86, 2013.

[34] C. Faloutsos, H. V. Jagadish, and N. Sidiropoulos, “Recovering information from summary data,” *PVLDB*, vol. 1, no. 1, pp. 36–45, 1997.

[35] J. Fan and J. Cheng, “Matrix completion by deep matrix factorization,” *Neural Networks*, vol. 98, pp. 34–41, 2018.

[36] X. Fu, K. Huang, and N. D. Sidiropoulos, “On identifiability of nonnegative matrix factorization,” *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 328–332, 2018.

[37] X. Fu, W.-K. Ma, K. Huang, and N. D. Sidiropoulos, “Blind separation of quasi-stationary sources: Exploiting convex geometry in covariance domain,” *IEEE Transactions on Signal Processing*, vol. 63, no. 9, pp. 2306–2320, 2015.

[38] T. A. Garrett, “Aggregated versus disaggregated data in regression analysis: implications for inference,” *Economics Letters*, vol. 81, no. 1, pp. 61–65, 2003.

[39] O. G. Guleryuz, “Weighted overcomplete denoising,” in *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, vol. 2. IEEE, 2003, pp. 1992–1996.

[40] R. C. Gunning and H. Rossi, *Analytic functions of several complex variables*. American Mathematical Soc., 2009, vol. 368.

[41] F. M. Harper and J. A. Konstan, “The movielens datasets: History and context,” *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[42] J. He, X. Li, and L. Liao, “Category-aware next point-of-interest recommendation via listwise bayesian personalized ranking.” in *IJCAI*, vol. 17, 2017, pp. 1837–1843.

[43] R. Heckel, M. Vlachos, T. Parnell, and C. Dünner, “Scalable and interpretable product recommendations via overlapping co-clustering,” in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 2017, pp. 1033–1044.

[44] F. L. Hitchcock, “The expression of a tensor or a polyadic as a sum of products,” *Journal of Mathematics and Physics*, vol. 6, no. 1-4, pp. 164–189, 1927.

[45] P. O. Hoyer, “Non-negative matrix factorization with sparseness constraints,” *Journal of machine learning research*, vol. 5, no. Nov, pp. 1457–1469, 2004.

[46] M.-J. Hsieh, W.-G. Teng, M.-S. Chen, and P. S. Yu, “Dawn: an efficient framework of dct for data with error estimation,” *The VLDB Journal—The International Journal on Very Large Data Bases*, vol. 17, no. 4, pp. 683–702, 2008.

[47] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *2008 Eighth IEEE International Conference on Data Mining*. Ieee, 2008, pp. 263–272.

[48] 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*, vol. 64, no. 19, pp. 5052–5065, 2016.

[49] Y. Jin, B. D. Williams, T. Tokar, and M. A. Waller, “Forecasting with temporally aggregated demand signals in a retail supply chain,” *Journal of Business Logistics*, vol. 36, no. 2, pp. 199–211, 2015.

[50] Y. Jin, B. D. Williams, M. A. Waller, and A. R. Hofer, “Masking the bullwhip effect in retail: the influence of data aggregation,” *Intl. J. of Physical Distrib. & Logistics Mgmt.*, vol. 45, no. 8, pp. 814–830, 2015.

[51] C. I. Kanatsoulis, X. Fu, N. D. Sidiropoulos, and W. Ma, "Hyperspectral super-resolution: A coupled tensor factorization approach," *IEEE T. on Signal Process.*, vol. 66, no. 24, pp. 6503–6517, 2018.

[52] C. I. Kanatsoulis, X. Fu, N. D. Sidiropoulos, and M. Akçakaya, "Tensor completion from regular sub-nyquist samples," *arXiv preprint arXiv:1903.00435*, 2019.

[53] C. I. Kanatsoulis, X. Fu, N. D. Sidiropoulos, and W.-K. Ma, "Hyperspectral super-resolution: Combining low rank tensor and matrix structure," in *2018 25th IEEE International Conference on Image Processing (ICIP).* IEEE, 2018, pp. 3318–3322.

[54] ——, "Hyperspectral super-resolution via coupled tensor factorization: Identifiability and algorithms," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2018, pp. 3191–3195.

[55] C. I. Kanatsoulis, N. D. Sidiropoulos, M. Akçakaya, and X. Fu, "Regular sampling of tensor signals: Theory and application to fmri," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2019, pp. 2932–2936.

[56] S. A. Khayam, "The discrete cosine transform (dct): Theory and application. department of electrical and computing engineering," 2003.

[57] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[58] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 426–434.

[59] C. Lawrence, T. Sztyler, and M. Niepert, "Explaining neural matrix factorization with gradient rollback," *arXiv preprint arXiv:2010.05516*, 2020.

[60] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[61] J.-H. Lee, D.-H. Kim, and C.-W. Chung, "Multi-dimensional selectivity estimation using compressed histogram information," in *ACM SIGMOD Record*, vol. 28, no. 2. ACM, 1999, pp. 205–214.

[62] M. Lenzen, "Aggregation versus disaggregation in input–output analysis of the environment," *Econ. Sys. Research*, vol. 23, no. 1, pp. 73–89, 2011.

[63] M. Lenzerini, "Data integration: A theoretical perspective," in *Proc. of the 21st ACM SIGMOD-SIGACT-SIGART symposium on Princ. of database sys.* ACM, 2002, pp. 233–246.

[64] D. Li, C. Chen, Q. Lv, H. Gu, T. Lu, L. Shang, N. Gu, and S. M. Chu, "Adaerror: An adaptive learning rate method for matrix approximation-based collaborative filtering," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 741–751.

[65] H. Li, Y. Liu, Y. Qian, N. Mamoulis, W. Tu, and D. W. Cheung, "Hhmf: hidden hierarchical matrix factorization for recommender systems," *Data Mining and Knowledge Discovery*, vol. 33, no. 6, pp. 1548–1582, 2019.

[66] C.-H. Lin, W.-K. Ma, W.-C. Li, C.-Y. Chi, and A. Ambikapathi, "Identifiability of the simplex volume minimization criterion for blind hyperspectral unmixing: The no-pure-pixel case," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 10, pp. 5530–5546, 2015.

[67] H. Liu, J. Wen, L. Jing, J. Yu, X. Zhang, and M. Zhang, "In2rec: Influence-based interpretable recommendation," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 1803–1812.

[68] M. Liu, Y. Luo, D. Tao, C. Xu, and Y. Wen, "Low-rank multi-view learning in matrix completion for multi-label image classification," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, pp. 2778–2784.

[69] Z. Liu, H. A. Song, V. Zadorozhny, C. Faloutsos, and N. Sidiropoulos, "H-fuse: Efficient fusion of aggregated historical data," in *Proceedings of the 2017 SIAM International Conference on Data Mining*, Houston, Texas, USA, April 2017, pp. 786–794.

[70] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[71] M. Maleszka, B. Mianowska, and N. T. Nguyen, “A method for collaborative recommendation using knowledge integration tools and hierarchical structure of user profiles,” *Knowledge-Based Systems*, vol. 47, pp. 1–13, 2013.

[72] X. Mao, P. Sarkar, and D. Chakrabarti, “On mixed memberships and symmetric nonnegative matrix factorizations,” in *International Conference on Machine Learning*, 2017, pp. 2324–2333.

[73] Y. Matsubara, Y. Sakurai, C. Faloutsos, T. Iwata, and M. Yoshikawa, “Fast mining and forecasting of complex time-stamped events,” in *Proc. of the 18th ACM SIGKDD intl. conf. on Knowl. discovery and data mining*. ACM, 2012, pp. 271–279.

[74] A. K. Menon, K.-P. Chitrapura, S. Garg, D. Agarwal, and N. Kota, “Response prediction using collaborative filtering with hierarchies and side-information,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 141–149.

[75] I. Motakis and C. Zaniolo, “Temporal aggregation in active database rules,” in *Proc. of the 1997 ACM SIGMOD*, ser. SIGMOD ’97. New York, NY, USA: ACM, 1997, pp. 440–451.

[76] A. N. Nikolakopoulos, M. A. Kouneli, and J. D. Garofalakis, “Hierarchical itemspace rank: Exploiting hierarchy to alleviate sparsity in ranking-based recommendation,” *Neurocomputing*, vol. 163, pp. 126–136, 2015.

[77] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009.

[78] A. Panagiotopoulou and V. Anastassopoulos, “Super-resolution image reconstruction techniques: Trade-offs between the data-fidelity and regularization terms,” *Information Fusion*, vol. 13, no. 3, pp. 185–195, 2012.

[79] E. E. Papalexakis, C. Faloutsos, and N. D. Sidiropoulos, “Tensors for data mining and data fusion: Models, applications, and scalable algorithms,” *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 2, pp. 16:1–16:44, Oct. 2016.

[80] N. Park, A. Kan, C. Faloutsos, and X. L. Dong, “J-recs: Principled and scalable recommendation justification,” *arXiv preprint arXiv:2011.05928*, 2020.

[81] Y. Park and J. Ghosh, “Ludia: An aggregate-constrained low-rank reconstruction algorithm to leverage publicly released health data,” in *Proc. of the 20th ACM SIGKDD*. ACM, 2014, pp. 55–64.

[82] A. Paterek, “Improving regularized singular value decomposition for collaborative filtering,” in *Proceedings of KDD cup and workshop*, vol. 2007, 2007, pp. 5–8.

[83] N. S. Patil and P. Patil, “Data aggregation in wireless sensor network,” in *IEEE intl conf. on comput. intell. and computing research*, vol. 6, 2010.

[84] J. M. Pavía-Miralles, “A survey of methods to interpolate, distribute and extra-polate time series,” *J. of Service Science and Mgmt.*, vol. 3, no. 04, p. 449, 2010.

[85] J. M. Pavía-Miralles and B. Cabrer-Borrás, “On estimating contemporaneous quarterly regional gdp,” *J. of Forecasting*, vol. 26, no. 3, pp. 155–170, 2007.

[86] G. Peake and J. Wang, “Explanation mining: Post hoc interpretability of latent factor models for recommendation systems,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2060–2069.

[87] T. Rekatsinas, M. Joglekar, H. Garcia-Molina, A. Parameswaran, and C. Ré, “Slimfast: Guaranteed results for data fusion and source reliability,” in *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 2017, pp. 1399–1414.

[88] S. Rendle, L. Zhang, and Y. Koren, “On the difficulty of evaluating baselines: A study on recommender systems,” *arXiv preprint arXiv:1905.01395*, 2019.

[89] N. Rossi *et al.*, “A note on the estimation of disaggregate time series when the aggregate is known,” *The Review of Econ. and Stats.*, vol. 64, no. 4, pp. 695–696, 1982.

[90] S. Saha, “Image compression – from dct to wavelets: A review,” *Crossroads*, vol. 6, no. 3, pp. 12–21, Mar. 2000. [Online]. Available: http://doi.acm.org/10.1145/331624.331630

[91] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 285–295.

[92] C. Sax and P. Steiner, “Temporal disaggregation of time series,” *The R Journal*, vol. 5, no. 2, pp. 80–87, 2013.

[93] S. Seo, J. Huang, H. Yang, and Y. Liu, “Interpretable convolutional neural networks with dual local and global attention for review rating prediction,” in *Proceedings of the eleventh ACM conference on recommender systems*, 2017, pp. 297–305.

[94] Y. Shen, S. Tan, A. Sordoni, and A. Courville, “Ordered neurons: Integrating tree structures into recurrent neural networks,” in *International Conference on Learning Representations*, 2018.

[95] E. Shi, H. Chan, E. Rieffel, R. Chow, and D. Song, “Privacy-preserving aggregation of time-series data,” in *Annual Network & Distributed System Security Symposium (NDSS)*. Internet Society., 2011.

[96] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, “Tensor decomposition for signal processing and machine learning,” *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582.

[97] A. Silvestrini and D. Veredas, “Temporal aggregation of univariate and multivariate time series models: a survey,” *J. of Econ. Surveys*, vol. 22, no. 3, pp. 458–497, 2008.

[98] M. Simões, J. Bioucas-Dias, L. B. Almeida, and J. Chanussot, “A convex formulation for hyperspectral image superresolution via subspace-based regularization,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 6, pp. 3373–3388, 2014.

[99] M. Sørensen and L. De Lathauwer, “Fiber sampling approach to canonical polyadic decomposition and application to tensor completion,” *SIAM Journal on Matrix Analysis and Applications*, vol. 40, no. 3, pp. 888–917, 2019.

[100] A. Strehl, J. Ghosh, and R. Mooney, "Impact of similarity measures on web-page clustering," in *Workshop on artificial intelligence for web search (AAAI 2000)*, vol. 58, 2000, p. 64.

[101] Z. Sun, J. Yang, J. Zhang, and A. Bozzon, "Exploiting both vertical and horizontal dimensions of feature hierarchy for effective recommendation," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[102] K. Takeuchi, H. Kashima, and N. Ueda, "Autoregressive tensor factorization for spatio-temporal predictions," in *2017 IEEE Intl. Conf. on Data Mining (ICDM)*. IEEE, 2017, pp. 1105–1110.

[103] Y. Tao, Y. Jia, N. Wang, and H. Wang, "The fact: Taming latent factor models for explainability with factorization trees," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 295–304.

[104] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. W. Schuller, "A deep matrix factorization method for learning attribute representations," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 3, pp. 417–429, 2016.

[105] Tycho, "Project tycho: Data for health," https://www.tycho.pitt.edu, 2013.

[106] S. Uludag, K.-S. Lui, K. Nahrstedt, and G. Brewster, "Analysis of topology aggregation techniques for qos routing," *ACM Computing Surveys (CSUR)*, vol. 39, no. 3, p. 7, 2007.

[107] S. Wang, J. Tang, Y. Wang, and H. Liu, "Exploring implicit hierarchical structures for recommender systems." in *IJCAI*, 2015, pp. 1813–1819.

[108] ——, "Exploring hierarchical structures for recommender systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1022–1035, 2018.

[109] X. Wang, W. Pan, and C. Xu, "Hgmf: Hierarchical group matrix factorization for collaborative recommendation," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, 2014, pp. 769–778.

[110] Y. Wang, T. Wu, Y. Wang, and G. Wang, "Enhancing model interpretability and accuracy for disease progression prediction via phenotype-based patient similarity learning," in *Pac Symp Biocomput*. World Scientific, 2019.

[111] A. B. Watson, "Image compression using the discrete cosine transform," *Mathematica journal*, vol. 4, no. 1, p. 81, 1994.

[112] B. D. Williams and M. A. Waller, "Creating order forecasts: point-of-sale or order history?" *J. of Business Logistics*, vol. 31, no. 2, pp. 231–251, 2010.

[113] B. Yang, X. Fu, and N. D. Sidiropoulos, "Learning from hidden traits: Joint factor analysis and latent clustering," *IEEE Transactions on Signal Processing*, vol. 65, no. 1, pp. 256–269, 2016.

[114] B. Yang, K. Huang, and N. D. Sidiropoulos, "Identifying potential investors with data driven approaches," in *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, 2020, pp. 235–243.

[115] F. Yang, F. M. Almutairi, H. A. Song, C. Faloutsos, N. D. Sidiropoulos, and V. Zadorozhny, "Turbolift: fast accuracy lifting for historical data recovery," *The VLDB Journal*, pp. 1–20, 2020.

[116] F. Yang, H. A. Song, Z. Liu, C. Faloutsos, V. Zadorozhny, and N. Sidiropoulos, "Ares: Automatic disaggregation of historical data," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 2018, pp. 65–76.

[117] J. Yang, Z. Sun, A. Bozzon, and J. Zhang, "Learning hierarchical feature influence for recommendation by recursive regularization," in *Proceedings of the 10th ACM Conference on Recommender Systems*, 2016, pp. 51–58.

[118] N. Yokoya, T. Yairi, and A. Iwasaki, "Coupled nonnegative matrix factorization unmixing for hyperspectral and multispectral data fusion," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 2, pp. 528–537, 2011.

[119] H.-F. Yu, N. Rao, and I. S. Dhillon, "Temporal regularized matrix factorization for high-dimensional time series prediction," in *Advances in neural information processing systems*, 2016, pp. 847–855.

[120] V. Zadorozhny and M. Lewis, “Information fusion for usar operations based on crowd-sourcing,” in *Information Fusion (FUSION), 2013 16th International Conference on*. IEEE, 2013, pp. 1450–1457.

[121] Y. Zhang and X. Chen, “Explainable recommendation: A survey and new perspectives,” *arXiv preprint arXiv:1804.11192*, 2018.

[122] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, “Explicit factor models for explainable recommendation based on phrase-level sentiment analysis,” in *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, 2014, pp. 83–92.

[123] E. Zhong, W. Fan, and Q. Yang, “Contextual collaborative filtering via hierarchical matrix factorization,” in *Proceedings of the 2012 SIAM International Conference on Data Mining*. SIAM, 2012, pp. 744–755.

Appendix A

Supplementary Material for Chapter 3

A.1 Derivation of Gradient Expressions

The terms in (3.11) and (3.24) can be divided into two types: 1) CPD of a tensor, with some aggregation matrices multiplied with the factors; and 2) the regularization term $\mathcal{R}$ in (3.24). Because the gradient of a sum is the sum of the gradients, it is enough to show the derivation of the gradients using the function below. This function consists of two terms, each represents one of the terms types listed above. Consider:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \underbrace{\|\underline{\mathbf{\Omega}} \circledast (\underline{\mathbf{X}} - ([\![\mathbf{UA}, \mathbf{VB}, \mathbf{WC}]\!])\|_F^2}_{\mathcal{T}} + \underbrace{\|\mathbf{1}^T\mathbf{C} - \mathbf{1}^T\widetilde{\mathbf{C}}\|_2^2}_{\mathcal{R}} \tag{A.1}$$

where $\underline{\mathbf{\Omega}}$ is as defined in (3.10), and $\underline{\mathbf{X}} \in \mathbb{R}^{I\times J\times K}$ is our data tensor. Note that all the CPD terms in (3.11) and (3.24) are similar to the term $\mathcal{T}$, with one (or more) of the aggregation matrices $\{\mathbf{U}, \mathbf{V}, \mathbf{W}\}$ is equal to $\mathbf{I}$. Thus, the term $\mathcal{T}$ generalizes all the CPD terms in our models. Using mode-3 unfolding, $\mathcal{T}$ is equivalent to

$$\mathcal{T} = \|\mathbf{\Omega}_3 \circledast \big(\mathbf{X}_3 - ((\mathbf{VB}) \odot (\mathbf{UA}))(\mathbf{WC})^T\big)\|_F^2. \tag{A.2}$$

Vectorizing the above, we get

$$\mathcal{T} = \|\mathbf{Sx} - \mathbf{S}((\mathbf{VB}) \odot (\mathbf{UA}) \odot (\mathbf{WC}))\mathbf{1}\|_F^2 \tag{A.3}$$

where $\mathbf{x} = \text{vec}(\mathbf{X}_3)$, and $\mathbf{S} \in \{0,1\}^{N \times IJK}$ is a fat matrix with one 1 in each row to select the available entries in $\mathbf{x}$, where $N = nnz(\underline{\mathbf{\Omega}})$. Thus, the role of $\mathbf{S}$ with $\mathbf{x}$, is similar to the role of $\underline{\mathbf{\Omega}}$ with the tensor form $\underline{\mathbf{X}}$. Equation (A.3) is also equivalent to

$$\mathcal{T} = \|\mathbf{S}\mathbf{x} - \mathbf{S}\big(\mathbf{I} \otimes ((\mathbf{V}\mathbf{B}) \odot (\mathbf{U}\mathbf{A}))\big)(\mathbf{W} \otimes \mathbf{I})\mathbf{c}\|_F^2 \tag{A.4}$$

where $\mathbf{c} = \text{vec}(\mathbf{C}^T)$. We show the derivative of $\mathcal{T}$ and $\mathcal{R}$ w.r.t. $\mathbf{C}$ (derivatives w.r.t. $\mathbf{A}$ and $\mathbf{B}$ are derived similarly by using mode-1 and mode-2 unfolding and rotating the factors accordingly). First, we derive the gradient of $\mathcal{T}$ w.r.t. $\mathbf{C}$:

$$\begin{aligned}
\nabla_{\mathbf{C}}\mathcal{T} &= 2(\mathbf{W}^T \otimes \mathbf{I})(\mathbf{I} \otimes (\mathbf{VB} \odot \mathbf{UA})^T)\mathbf{S}^T\mathbf{S}\ (\mathbf{I} \otimes (\mathbf{VB} \odot \cdot \\
&\mathbf{UA}))(\mathbf{W} \otimes \mathbf{I})\mathbf{c} - 2(\mathbf{W}^T \otimes \mathbf{I})(\mathbf{I} \otimes (\mathbf{VB} \odot \mathbf{UA})^T)\mathbf{S}^T\mathbf{S}\mathbf{x} &\text{(A.5)} \\
&= 2(\mathbf{I} \otimes (\mathbf{VB} \odot \mathbf{UA})^T)(\mathbf{W}^T \otimes \mathbf{I})\mathbf{S}^T\mathbf{S}(\mathbf{I} \otimes (\mathbf{VB} \odot \cdot \\
&\mathbf{UA}))(\mathbf{W} \otimes \mathbf{I})\mathbf{c} - 2(\mathbf{I} \otimes (\mathbf{VB} \odot \mathbf{UA})^T)(\mathbf{W}^T \otimes \mathbf{I})\mathbf{S}^T\mathbf{S}\mathbf{x} &\text{(A.6)} \\
&= 2(\mathbf{I} \otimes (\mathbf{VB} \odot \mathbf{UA})^T)(\mathbf{W}^T \otimes \mathbf{I})\mathbf{S}^T\mathbf{S}\big((\mathbf{I} \otimes (\mathbf{VB} \odot \cdot \\
&\mathbf{UA}))(\mathbf{W} \otimes \mathbf{I})\mathbf{c} - \mathbf{x}\big). &\text{(A.7)}
\end{aligned}$$

We can use mode-3 unfolding to rewrite (A.7) above as

$$\nabla_{\mathbf{A}}\mathcal{T} = 2\mathbf{W}^T\big(\mathbf{\Omega}_3 \circledast (\widehat{\mathbf{X}}_3 - \mathbf{X}_3)\big)^T\big((\mathbf{VB}) \odot (\mathbf{UA})\big) \tag{A.8}$$

where $\widehat{\mathbf{X}}_3 = \big((\mathbf{VB}) \odot (\mathbf{UA})\big)(\mathbf{WC})^T$. The gradient above can be computed efficiently by the following steps:

1. Compute $\mathbf{L} = \mathbf{\Omega}_3 \circledast (\widehat{\mathbf{X}}_3 - \mathbf{X}_3)$.

2. Compute $\mathbf{M} = \mathbf{L}^T\big((\mathbf{VB}) \odot (\mathbf{UA})\big)$.

3. Compute $2\mathbf{W}^T\mathbf{M}$

Next, the derivative of $\mathcal{R}$ w.r.t. $\mathbf{C}$ is

$$\nabla_{\mathbf{C}}\mathcal{R} = 2(\mathbf{1}\mathbf{1}^T\mathbf{C} - \mathbf{1}\mathbf{1}^T\widetilde{\mathbf{C}}). \tag{A.9}$$

A.2 Derivation of Step Size Expressions

The step size terms in both Algorithm 1 and 2 are chosen using the exact line search optimization method. Recall (A.1)

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \underbrace{\|\underline{\boldsymbol{\Omega}} \circledast (\underline{\mathbf{X}} - ([\![\mathbf{UA}, \mathbf{VB}, \mathbf{WC}]\!])\|_F^2}_{\mathcal{T}} + \underbrace{\|\mathbf{1}^T\mathbf{C} - \mathbf{1}^T\widetilde{\mathbf{C}}\|_2^2}_{\mathcal{R}}.$$

As mentioned earlier in Appendix A.1, the function above generalizes all the terms in PREMA and B-PREMA models. Thus, we use (A.1) to show how to find the step size γ associated with updating $\mathbf{C}$ as an illustrative example. In this case, the exact line search chooses γ to be the minimizer of

$$\arg\min_{\gamma\geq 0} \quad \mathcal{F}\big(\mathbf{C} - \gamma\nabla_{\mathbf{C}}\mathcal{F}\big) \tag{A.10}$$

where $\mathcal{F} = \mathcal{T} + \mathcal{R}$, which are as defined in (A.1). Plugging the variable $\mathbf{C} - \gamma\nabla_{\mathbf{C}}\mathcal{F}$ into (A.1) and rearranging the terms, we get

$$\begin{aligned}\arg\min_{\gamma\geq 0} \quad & \| \underbrace{\boldsymbol{\Omega}_3 \circledast \big(\mathbf{Y}_3 - \big((\mathbf{VB}) \odot (\mathbf{UA})\big)\mathbf{W}^T\mathbf{C}^T\big)}_{\mathbf{E}} \\ & + \gamma \underbrace{\boldsymbol{\Omega}_3 \circledast \big((\mathbf{VB} \odot \mathbf{UA})\mathbf{W}^T\nabla_{\mathbf{C}}\mathcal{F}^T\big)}_{\mathbf{D}} \|_F^2 \\ & + \| \underbrace{\mathbf{1}^T\mathbf{C} - \mathbf{1}^T\widetilde{\mathbf{C}}}_{\mathbf{e}^T} - \gamma \underbrace{\mathbf{1}^T\nabla_{\mathbf{C}}\mathcal{F}}_{\mathbf{d}^T} \|_2^2.\end{aligned} \tag{A.11}$$

One can see that at the optimal solution to (A.11), we have:

$$-\text{vec}(\mathbf{E})^T = \gamma \text{vec}(\mathbf{D})^T \tag{A.12}$$

$$\mathbf{e}^T = \gamma \mathbf{d}^T \tag{A.13}$$

Multiplying (A.12) by vec($\mathbf{D}$) and (A.13) by $\mathbf{d}$, and summing up the resulting two equations, we get

$$-\text{vec}(\mathbf{E})^T\text{vec}(\mathbf{D}) + \mathbf{e}^T\mathbf{d} = \gamma(\text{vec}(\mathbf{D})^T\text{vec}(\mathbf{D}) + \mathbf{d}^T\mathbf{d}) \tag{A.14}$$

Respecting the non-negativity constraint, we can see that the optimal solution is

$$\gamma = max\Big(0, \frac{-\text{vec}(\mathbf{E})^T \text{vec}(\mathbf{D}) + \mathbf{e}^T\mathbf{d}}{\text{vec}(\mathbf{D})^T \text{vec}(\mathbf{D}) + \mathbf{d}^T\mathbf{d}}\Big) \tag{A.15}$$

A.3 Initialization Algorithm

The initialization steps of Algorithm 3.1 are as follows

```
Set missing entries in $\underline{\mathbf{Y}}^t$, and $\underline{\mathbf{Y}}^c$ to zeros.
**if** $\mathbf{V} = \mathbf{I}$ and $K > I$ **then**
    $\widetilde{\mathbf{A}}, \mathbf{B}, \mathbf{C} \leftarrow$ CPD($\underline{\mathbf{Y}}^c$);
    $\mathbf{A} \leftarrow$ solve $\mathbf{Y}_1^t = ((\mathbf{WC}) \odot \mathbf{B})\mathbf{A}^T$
**else**
    $\mathbf{A}, \mathbf{B}, \widetilde{\mathbf{C}} \leftarrow$ CPD($\underline{\mathbf{Y}}^t$);
    $\mathbf{C} \leftarrow$ solve $\mathbf{Y}_3^c = ((\mathbf{VB}) \odot (\mathbf{UA}))\mathbf{C}^T$
**end if**
```

Note that the missing entries are set to 0 only in the initialization steps. We use the Matlab-based package `Tensorlab` to compute the CPD in the initialization (e.g., CPD($\underline{\mathbf{Y}}^c$)).

A.4 Proof of Proposition 3.1

Let $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ be the target tensor to disaggregate with CPD $\underline{\mathbf{X}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ of rank R and $\underline{\mathbf{Y}}^t \in \mathbb{R}^{I \times J \times K_w} = \underline{\mathbf{X}} \times_3 \mathbf{W}$. Then, under the conditions of Theorem 3.1, $\underline{\mathbf{Y}}^t$ admits a unique CPD $\underline{\mathbf{Y}}^t = [\![\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t]\!]$. Since it is unique, it holds that:

$$\mathbf{A}_t = \mathbf{A\Pi\Lambda}_1, \mathbf{B}_t = \mathbf{B\Pi\Lambda}_2, \mathbf{C}_t = \mathbf{WC\Pi\Lambda}_3, \tag{A.16}$$

where $\mathbf{\Pi}$ is a permutation matrix and $\mathbf{\Lambda}_1,\ \mathbf{\Lambda}_2,\ ,\mathbf{\Lambda}_3$ are diagonal matrices such that $\mathbf{\Lambda}_1\mathbf{\Lambda}_2\mathbf{\Lambda}_3 = \mathbf{I}$. In the case where $\underline{\mathbf{Y}}^t$ has missing entries the conditions under which $[\![\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t]\!]$ are identifiable are stricter and depend on the pattern of misses. We can use the conditions in [13, 52, 99] for fiber, regular and random sampling respectively. So far, factors $\mathbf{A},\ \mathbf{B}$ have been identified up to column permutation and scaling. What remains to be proven is that:

$$\underline{\mathbf{\Omega}}^c \circledast \underline{\mathbf{Y}}^c = \underline{\mathbf{\Omega}}^c \circledast (\underline{\mathbf{X}} \times_1 \mathbf{U} \times_2 \mathbf{V}) = \underline{\mathbf{\Omega}}^c \circledast ([\![\mathbf{UA}, \mathbf{VB}, \mathbf{C}]\!]) \tag{A.17}$$

yields a solution for $\mathbf{C}_c$ such that $\mathbf{C}_c = \mathbf{C\Pi\Lambda}_3$. Equation (A.17) can be equivalently written as:

$$\mathbf{S}_c\mathbf{y}_c = \mathbf{S}_c(\mathbf{C} \odot \mathbf{VB} \odot \mathbf{UA})\mathbf{1} = \mathbf{S}_c(\mathbf{I} \otimes (\mathbf{VB} \odot \mathbf{UA}))\mathbf{c}, \tag{A.18}$$

where $\mathbf{y}_c, \mathbf{c}$ are vectorized versions of $\underline{\mathbf{Y}}^c,\ \mathbf{C}^T$, and $\mathbf{S}_c \in \{0,1\}^{N_c \times I_u J_v K}$ is a fat selection matrix that selects the available entries in $\mathbf{y}_c$, where $N_c = nnz(\underline{\mathbf{\Omega}}^c)$.

Now let $\widetilde{\mathbf{A}} = \mathbf{UA}$ and $\widetilde{\mathbf{B}} = \mathbf{VB}$. Following [51, Lemma 1] $\widetilde{\mathbf{A}},\ \widetilde{\mathbf{B}}$ are drawn from absolutely continuous non-singular distributions. Also let $\mathbf{P} = \widetilde{\mathbf{B}} \odot \widetilde{\mathbf{A}}$. Since $I_u J_v \geq R$ the determinant of any $R \times R$ submatrix of $\mathbf{P}$ is a non-trivial analytic function of $\widetilde{\mathbf{A}},\ \widetilde{\mathbf{B}}$. Therefore any $R \times R$ minor of $\mathbf{P}$ is non-zero almost surely [40, Lemma 3] and any R rows of $\mathbf{P}$ are independent.

Taking a closer look at matrix $\mathbf{G} = \mathbf{I} \otimes (\mathbf{VB} \odot \mathbf{UA}) = \mathbf{I} \otimes (\widetilde{\mathbf{B}} \odot \widetilde{\mathbf{A}})$ we observe that it is an $I_u J_v K \times KR$ block diagonal matrix of the form:

$$\mathbf{G} = \begin{bmatrix} \mathbf{P} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{P} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{P} \end{bmatrix} = \begin{bmatrix} \mathbf{G}_1 \\ \mathbf{G}_2 \\ \vdots \\ \mathbf{G}_K \end{bmatrix} \tag{A.19}$$

Each block $\mathbf{G}_k$ corresponds to the $k-$th frontal slab of $\underline{\mathbf{Y}}^c$ and the rows between different $\mathbf{G}_k$'s are independent by construction. Since we have assumed that the minimum number of observed entries for each frontal slab is greater than or equal to R, then $\mathbf{S}_c\mathbf{G}$ has full column rank equal to KR and the solution for $\mathbf{c}$ in (A.18) is unique with probability 1. Plugging $\mathbf{A}_t,\ \mathbf{B}_t$ in equation (A.18) we get:

$$\begin{aligned} \mathbf{S}_c\mathbf{y}_c &= \mathbf{S}_c(\mathbf{C} \odot \mathbf{VB}_t \odot \mathbf{UA}_t)\mathbf{1} \\ &= \mathbf{S}_c(\mathbf{C} \odot \mathbf{VB\Pi\Lambda}_2 \odot \mathbf{UA\Pi\Lambda}_1)\mathbf{1} \end{aligned} \tag{46}$$

Then the unique solution for $\mathbf{C}$ satisfies $\mathbf{C}_c = \mathbf{C\Pi\Lambda}_3$ and $\widehat{\underline{\mathbf{X}}} = [\![\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_c]\!]$ disaggregates $\underline{\mathbf{Y}}^t,\ \underline{\mathbf{Y}}^c$ to $\underline{\mathbf{X}}$ almost surely.

Appendix B

Supplementary Material for Chapter 4

B.1 Proof of Theorem 4.2

Here, we provide the detailed proof of the theorem that establishes the identifiability of eTREE.

Theorem 4.2 *Assume that a data matrix follows* $\mathbf{X} = \mathbf{A}\mathbf{B}_1^T$*, where* $\mathbf{A} \in \mathbb{R}^{N \times R}$*, and* $\mathbf{B}_1 \in \mathbb{R}^{M_1 \times R}$ *are the ground-truth factors, and assume that* $\mathbf{B}_1 = \mathbf{S}_1\mathbf{S}_2 \ldots \mathbf{S}_{Q-1}\mathbf{B}_Q$*, where* $\mathbf{S}_q \in \{0,1\}^{M_q \times M_{q+1}}$*,* $\|\mathbf{S}_q(i,:)\|_0 = 1, \forall i \in [M_q], q \in [Q-1]$*. Let* $\mathbf{S} = \mathbf{S}_1\mathbf{S}_2 \ldots \mathbf{S}_{Q-1}$*, then,* $\mathbf{B}_1 = \mathbf{S}\mathbf{B}_Q$*. Also, assume that rank*$(\mathbf{X})$ *= rank*$(\mathbf{A})$ *=* R*, and, without loss of generality,* $\mathbf{M}_Q \geq R$*. If* $\mathbf{A}$ *and* $\mathbf{S}$ *are full-column rank, and rows of* $\mathbf{B}_Q$ *are sufficiently scattered, then rows of* $\mathbf{B}_1$ *are sufficiently scattered, and* $\mathbf{A}$*,* $\mathbf{B}_1$*,* $\mathbf{B}_Q$*, and* $\mathbf{S}$ *are essentially unique.*

Proof. First, we prove the identifiability of $\mathbf{A}$ and $\mathbf{B}_1$. Let $\mathbf{X}$ be the data matrix that follows $\mathbf{X} = \mathbf{A}\mathbf{B}_1^T$. Since $\mathbf{A}$ is full-column rank, we only need to prove that the rows of $\mathbf{B}_1$ are sufficiently scattered (Definition 2) to satisfy the identifiability conditions of Theorem 1 [36]. Since $\mathbf{S}_q \in \{0,1\}^{M_q \times M_{q+1}}$, $\|\mathbf{S}_q(i,:)\|_0 = 1, \forall i \in [M_q], q \in [Q-1]$, then all the rows of $\mathbf{B}_Q$ will appear in $\mathbf{B}_1 = \mathbf{S}\mathbf{B}_Q$ *if* $\mathbf{S}$ is full-column rank. Thus, rows of $\mathbf{B}_1$ are sufficiently scattered *iff* the rows of $\mathbf{B}_Q$ are sufficiently scattered. So far, the factors $\mathbf{A}$ and $\mathbf{B}_1$ have been identified. Next, the factor $\mathbf{B}_Q$ and $\mathbf{S}$ in $\mathbf{B}_1 = \mathbf{S}\mathbf{B}_Q$ are also essentially unique because every row of $\mathbf{B}_Q$ appears in $\mathbf{B}_1$. Thus, the rows of $\mathbf{B}_Q$ are the unique rows in $\mathbf{B}_1$, and hence $\mathbf{S}$ can be determined based on the correspondence. Interestingly, identifiability of the factorization $\mathbf{B}_1 = \mathbf{S}\mathbf{B}_Q$ also follows as a very special instance of Theorem 1. This is because we can argue that $\mathbf{S}$ is full-column rank, and the rows of $\mathbf{B}_Q$ are sufficiently scattered. □

B.2 Experiments

In this section, we provide implementation details for the experiments on the matrix completion accuracy, a demo to run the code, and additional analysis to demonstrate the effectiveness of the tree clustering in eTREE.

B.2.1 Implementation Details for Q1 Experiments (Matrix Completion Accuracy)

Our algorithm was implemented in Matlab and all the experiments (including the baselines) were performed using Matlab v2020a on a Linux server with Intel Core i7–4790 CPU 3.60 GHz processor and 32 GB memory.

As we described in the main paper, the parameters in the matrix completion experiments are selected via a validation set. Table B.2 lists the ranges of all the hyper-parameters for eTREE and the baselines on the different datasets. In the table, λ corresponds to the Frobenius-norm regularization parameter in all methods, μ is the tree regularization parameter in eTREE, and Q is the number of layers in the tree. We set the maximum number of epochs to 1000 for all models, however, eTREE typically does not require more than 100 epochs. With all methods, we employ an early stopping strategy to terminate the algorithm once the prediction accuracy (on the validation set) starts degrading.

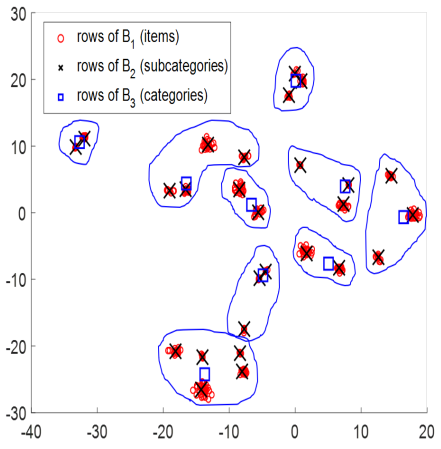
B.2.2 Additional Analysis of Q2 Experiments (Interpretability)

The tree shown in the interpretability results on Med-MCI dataset (Figure 3) contains 3 layers with $M_1 = 412$ items in the bottom layer, $M_2 = 27$ subcategories in the intermediate layer, and $M_3 = 9$ main categories in the top layer. As we showed in the main paper, eTREE produces meaningful hierarchical tree categories.

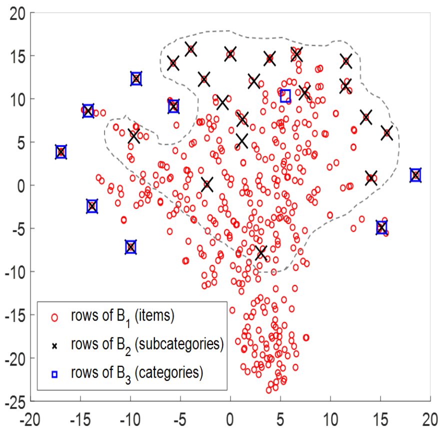
Here, we demonstrate the power of learning the tree clusters jointly with obtaining the embeddings, as opposed to learning the tree clusters after obtaining the embeddings in a two phase fashion. For the two phase algorithm, we apply NMF to produce the item embeddings ($\mathbf{B}_1$). Then, we perform k-means with $k = 27$ to get the embeddings of the subcategories ($\mathbf{B}_2$), then we apply another k-means with $k = 9$ on the obtained subcategories to get the main categories ($\mathbf{B}_3$), i.e., recursive k-means (RKM) – we call this method NMF+RKM. We perform t-SNE [70] to produce a 2-dimensional visualization of the node embeddings produced by eTREE (Figure

Table B.2: The ranges of hyper-parameters.

Parameter	**Model**	**Data**	**Range**
R (rank)	All	Med-HF, Med-MCI, College Grades	$\{2, 5, 7, 10, 15, 20, 25\}$
R (rank)	All	MovieLens	$\{5, 10, 15, 20, 25, 50, 100, 150, 200, 250\}$
λ (norm-reg)	All	All	{0, 1e-05, 1e-03, 1e-02, 0.1 to 1 with 0.1 increments, 5, 10, 15, 20}
μ (tree-reg)	eTREE	All	{0, 1e-05, 1e-03, 1e-02, 0.1 to 1 with 0.1 increments, 5, 10, 15, 20}
learning rate	BMF, AdaError	All	{1e-05, 1e-03, 1e-02, 0.1 to 1 with 0.1 increments, 5, 10}
Q (number of layers)	HSR, eTREE	All	$\{2, 3, 4\}$



(a) eTREE



(b) NMF+RKM

Figure B.4: t-SNE visualization of the embeddings of the tree nodes. eTREE produces clear clusters.

B.4 (a)) and NMF+RKM (Figure B.4 (b)). The scatter plot of the medical services (red circles), their subcategories (black x's), and the main categories (blue squares) produced by eTREE is shown in Figure B.4 (a). Here, we circle the subcategories that belong to the same main category together with their centroid (main category). Note that this is a 2-dimensional visualization of the same clusters presented in the main paper in Figure 3. On the other hand, the groups produced by NMF+RKM did not give meaningful clusters that could be interpreted by the domain experts. For instance, in Figure B.4 (b), one main category includes 19 subcategories (inside the dotted gray curve), and each one of the remaining main categories has one subcategories. This is counter-intuitive with the meaningful tree shown in Figure 3. Similarly, the items (red circles) do not exhibit a clustering structure around their centroids (black x's) in Figure B.4 (b). These results prove that the tree structure in the item embeddings can not be naturally captured by basic method (e.g., NMF). On the other hand, the tree prior in eTREE succeeds to extract and exploit the hierarchical categorical structure.