Reconstructing and Mining Signals: Algorithms and Applications

Hyun Ah Song

May 2020
CMU-ML-20-101

Machine Learning Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Christos Faloutsos, Chair
Roni Rosenfeld
Nicholas D. Sidiropoulos (University of Virginia)
Vladimir Zadorozhny (University of Pittsburgh)

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was sponsored by Defense Threat Reduction Agency award HDTRA11010120, National Science Foundation award IIS1247489, U.S. Army Research Laboratory award W911NF0920053, U.S. Army Research Office award W911NF11C0088, and the Boeing Company.

Keywords: signal reconstruction, signal mining, super resolution reconstruction, historical information fusion, brain data, power grid, aircraft signals, anomaly detection, forecasting, interpretability, domain knowledge

To my mom.

Abstract

Given two different brain-scan modalities, like fMRI and MEG, how can we combine them, to achieve better resolution in both space and time, than each? Given power grid measurements (voltage, current), how can we spot patterns, anomalies and do forecasts? We answer these types of questions in the two parts of the thesis.

In the first part of the thesis, we discuss our work on signal **reconstruction**. We live in a world that is flooded by streaming signals that come from different sources in different resolutions which often account for the same information. One example is historical records on patient counts; one TV source may report the patient counts in weekly intervals, while a newspaper source may report in monthly intervals. Another example is the brain activity signals; while fMRI has high spatial resolution and low temporal resolution, it is vice versa for MEG/EEG. We are interested in constructing higher resolution signals that complement those signals from different sources or modes in different resolutions. We discuss our work on signal reconstruction in two applications: *historical epidemiological data* and *brain data*. 1) *Historical epidemiological data*: We introduce constraints such as smoothness and periodicity, utilizing the well-known epidemiological model 'susceptible-infected-susceptible (SIS)'. Also we discuss how we can utilize techniques like annihilating filters and discrete cosine transforms to discover linear or multiple periodicities in the sequences. 2) *Brain data*: We discuss our proposed approaches that employ various assumptions such as sparsity, low rank, or smoothness and show that reconstructed brain signal displays richer information of fMRI and MEG, interpolating in time and space in a principled way.

In the second part of the thesis, we discuss our work on signal **mining**. Raw signals can contain unnoticeable hidden information that is not observable in their raw forms. One example is power grid signals (voltages, currents); signals in its raw form do not lead to straightforward interpretation. Another example is aircraft sensor signals; sensor signals are result of complex physical models which do not provide straight-forward interpretation. If we want to understand the data, we should explore it deeper (if you *mind* it, *mine* it!), preferably with domain knowledge. We are interested in mining signals that can provide us with better interpretation of the data, and aid us with various data mining tasks such as forecasting, anomaly detection, etc. We discuss our signal mining work on two different application domains: *power grid data* and *aircraft data*. 1) *Power grid data*: We introduce our works that incorporate a physics model, the BIG model, to better interpret the data, and utilize tensor factorization and Holt-Winters to model the data for anomaly detection and forecasting. Experimental results on CMU and Lawrence Berkeley National Laboratory (LBNL) power grid dataset demonstrate *32%* and *27%* error reduction in forecasting compared to the latest algorithm. Also we show that proposed algorithm successfully detects anomalous events. 2) *Aircraft data*: We discuss our proposed method on analyzing aircraft sensor signals for detecting anomalous events using coupled tensor factorization.

Acknowledgments

All the work presented in this thesis could not have been possible without the help from my advisor, Professor Christos Faloutsos. Christos has been the best advisor anyone could hope for. He always cares for his students not only academically but also personally. In every project, Christos was always there from the brainstorming stage throughout the detailed technical discussions and to the paper writing and presentation stages. After five years of working with him, his ways of academical thinking and writing styles have been adopted to me. Apart from research, Christos would also make sure that I am feeling well, balancing work and life. Christos has provided me with an opportunity to explore various research topics, which helped me learn a lot in broad scope of applications. I was fortunate to be given a great opportunity to collaborate with smart colleagues within and outside CMU (University of Pittsburgh, University of Minnesota, University of Virginia, University of Porto, Data Insights Laboratories) and also within and outside SCS (ECE, Statistics, Informatics). This wide collaboration across departments and schools has been one of the fun experiences I had in my PhD program. I will always remember his endless passion for various research topics and positivity, as well as his passion for pictionary, charades, and jokes at the group parties. *efcharisto* (thanks)!

In addition to Christos, I have been working closely with other professors as well: Professor Vladimir Zadorozhny and Professor Nicholas D. Sidiropoulos. I came to know Vlad when he introduced me a project on historical epidemiology when I was in my third year. Thanks to his guidance on the project, we were able to publish five papers together. I enjoyed brain-storming new ideas on this project with Vlad and other colleagues. The works from this project constitute the largest sections in this thesis. I appreciate Vlad for introducing and guiding me on this project that makes up a large portion of my works. I have worked with Nikos on many different projects including the brain project, and the historical epidemiology project. Nikos has been the-go-to-person whenever we got stuck formalizing mathematical concepts. During discussions, Nikos would ask sharp questions to clear any misunderstandings and suggest detailed ideas on how to approach the solution. I was always amused by his broad knowledge on various research topics and ability to come up with solutions regardless of the problem given to us. I fortunate enough to learn a lot of machine learning concepts to solve for the problems and gravely expanded my problem solving skillset by working with Nikos.

Throughout my PhD program, I was fortunate to collaborate with smart colleagues in many different projects. I would like to thank Bryan Hooi. I have known him since my first year at CMU and we literally went through the program together – taking courses, working on the same projects with Christos, etc. Many of the courses and the projects presented in this thesis are with Bryan, and I wouldn't have made it without him. I would like to thank my collaborators from the historical epidemiology project where we had fun gradually developing the ideas, from the most naive forms to more complex and completed forms: Zongge Liu, Fan Yang, and Faisal M. Almutairi. I enjoyed learning about power grid systems and electrical engineering

domain knowledge by working with our ECE colleagues: Amritanshu Pandey and Marko Jereminov. I learned a lot about brain scans by working with the colleagues in the brain project: Xiao Fu, Kejun Huang, Otilia Stretcu, Partha Pratim Talukdar and Tom Mitchell. Lastly I would like to thank the DB group members for the fruitful discussions: Miguel Araujo, Evangelos E. Papalexakis, Alex Beutel, Neil Shah, Dhivya Eswaran, Kijung Shin, Hemank Lamba and Srijan Kumar.

I've had a great pleasure of interacting with so many nice and smart friends at CMU during my PhD years. I would like to thank Su Zhou and Yu-Xiang Wang for being there to talk about everything – from academics to some random thoughts. The time I shared my office with them was one of the most memorable times I had at CMU. I would like to also thank Natalie Klein and Yotam Hechtlinger for being there in my first year at CMU, working on a project together. I would like to thank Qiong Zhang, Xun Zheng, Guangyu (Gus) Xia, and Jisu Kim for all the academic and friendly conversations we had. Lastly, many thanks to Ifigeneia Apostolopoulou for sharing an office with me, and all the fun discussions, both in academics and arts.

I would like to specially thank the university staff members for all the supports. Diane Stidle was always there to help me out with all sorts of administrative tasks – friendly reminders for the deadlines, requirements, MLD events, announcements, and baby photos! Diane makes it possible for MLD to maintain, coordinate, and connect. Thanks to Diane, I was able to stay on track. Also, special thanks to Marilyn Walgora, Ann Stetser, and Tony Mareino for all the help with the document works regarding attending the conferences and management of the meetings, and talks.

Away from CMU, I had a great opportunity to work at Amazon during summer 2018 with Xin Luna Dong and Andrey Kan. I enjoyed working on a new problem domain and learned a lot from the discussions with Luna and Andrey. In addition to scientific discussions, Luna also provided me helpful tips and comments regarding industry careers. I would like to thank them for all the valuable comments and feedback.

Last but not least, I would like to thank my mom for her unconditional support and love. My life would not have been what it is now without her.

Contents

List of Figures

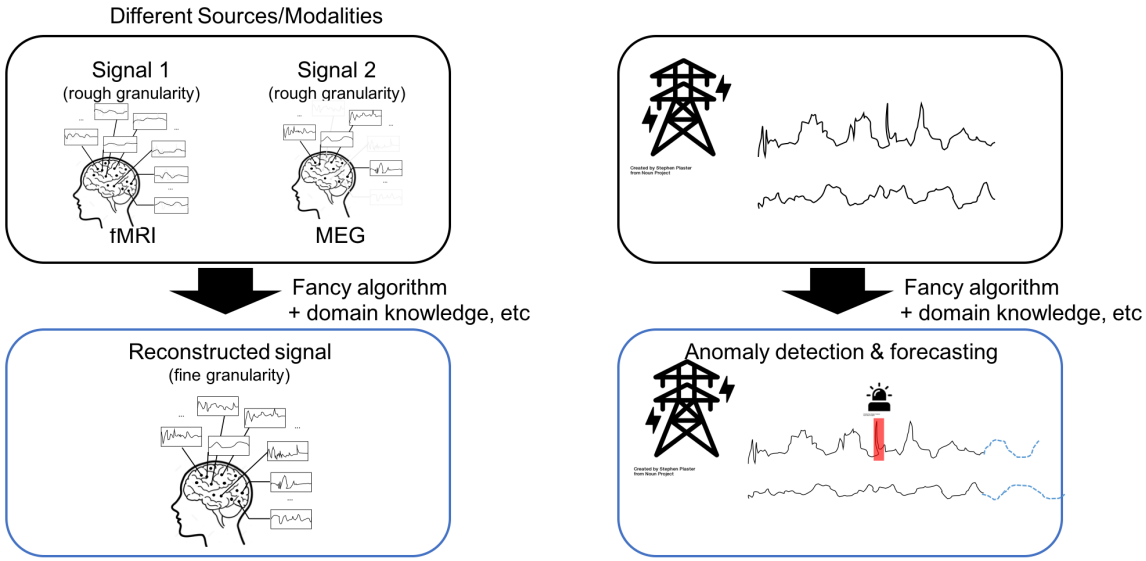
List of Tables

Chapter 1

Introduction

This thesis consists of two parts: 1) signal reconstruction and 2) signal mining. In each of the chapter, we introduce our proposed methods and share our insights in different application domains.

An overview of our works is demonstrated in Figure 1.1, in a simplified diagram. In (a), the signal reconstruction process is demonstrated. Given low resolution signals, we try to recover high resolution signals utilizing domain knowledge, etc. In (b), signal mining is described. Given complex sensor signals, we try to understand the pattern, so that we can detect anomalies, predict future points, etc. Details of each task are provided in the following chapters.



(a) Signal resonctruction

(b) Signal mining

Figure 1.1: **Overview**. (a) Signal reconstruction and (b) signal mining.

In the first chapter, we discuss our work on signal reconstruction. We live in a world that is flooded by streaming signals that come from different sources in different resolutions which

often account for the same information. One example is historical records on patient counts; one TV source may report the patient counts in weekly intervals, while a newspaper source may report in monthly intervals. Another example is the brain activity signals; while fMRI has high spatial resolution and low temporal resolution, it is vice versa for MEG/EEG. Even though they all account for the same information and we have abundant amount of data, unless we harness them to put together in a principled manner to come up with a big picture, they are almost not so useful pieces of information – we cannot utilize them. Therefore, it is of great interest to combine those different sources of signals to come up with richer information.

This is called "information fusion" or super resolution reconstruction problem; reconstructing higher resolution signals given low resolution signals. Because this is an under-determined problem, one has to utilize regularizers or constraints suitable for the domain. Exploring domain specific assumptions and optimizing the problem constitute the major challenges in the super resolution problem.

Super-resolution via fusing multi-modal data is an active research area in the image processing community. Specifically, fusion of remotely sensed hyper-spectral and multi-spectral images is of great interest [79]. Both hyper- and multi-spectral images are special images whose pixels span many frequency bands and can provide spectral information of the materials contained in the pixels – which finds many applications in geoscience, mineral detection, and food safety. Much effort has been spent on fusing these two types of images that captured on the sites, to create super-resolution data in both space and frequency.

Although the super resolution problem has been studied extensively in the image processing community, it is not straightforward to apply the methods to other domains as it requires domain specific assumptions and optimizations. We discuss our work on signal reconstruction in two applications where the super resolution problem has not been studied before: *historical epidemiological data* and *brain data*. 1) *Historical epidemiological data*: We start by introducing the most simple constraints such as smoothness and periodicity. As our next attempt, we apply domain-specific constraints, utilizing the well-known epidemiological model 'susceptible-infected-susceptible (SIS)' to reconstruct the data and also interpret the epidemiological time evolving characteristics. As a next step, we discuss how we can utilize techniques like annihilating filters and discrete cosine transforms to discover linear or multiple periodicities in the data.2) *Brain data*: Although there have been previous works on learning the intersection of two different brain image modalities, there had not been a work that tries to recover a super resolution image from them. In our work, we try to reconstruct finer resolution brain image from two different brain image modalities, fMRI and MEG, that have opposite resolution characteristics; fMRI has high spatial resolution and low temporal resolution while it is vice versa for MEG. We discuss our proposed approaches that employ various assumptions such as sparsity, low rank, or smoothness and show that the reconstructed brain signal displays richer information of fMRI and MEG, providing more accurate interpolation in both time and space, close to the ground-truth data.

In the second chapter, we discuss our work on signal mining. Raw signals can contain unnoticeable hidden information that is not observable in their raw forms. One example is the power grid signals (voltages, currents); signals in their raw form do not lead to the straightforward interpretations. Another example is the aircraft sensor signals; sensor signals reflect the pilot controls and the complex physics models in the aircraft system, which is rarely understandable without

proper knowledge. If we want to understand the data, we should explore it deeper (if you *mind* it, *mine* it!), preferably with domain knowledge. We are interested in mining signals that can provide us with better interpretation of the data, and aid us with various data mining tasks such as forecasting, anomaly detection, etc.

Data mining has been actively applied to a wide variety of data, in many different domains. Mining methods depend on the goal (what we want to do with the data – is it forecasting, learning patterns, detecting anomalies, etc), and the data characteristics (is it time evolving, is it binary/-continuous, is it multi-dimensional, etc).

We discuss our signal mining work on two different application domains: *power grid data* and *aircraft data*. 1) *Power grid data*: We introduce our proposed approaches that incorporate a physics model, the BIG model, to better represent and interpret the data. On top of the BIG model, we construct and factorize the data in tensors and use the Holt-Winters for anomaly detection and forecasting. 2) *Aircraft data*: We introduce our method for analyzing aircraft sensor signals for detecting anomalous events using coupled tensor factorization method.

The structure of this thesis is as follows: In the following Part 1 I, we introduce our work on signal reconstruction in two application domains: historical epidemiological data (Chapter 2) and brain image (Chapter 3). In Part 2 II, we introduce our work on signal mining in power grid domain (Chapter 4) and aircraft systems (Chapter 5).

A summary of our works in this thesis is shown in Table 1.1.

Tasks	Applications
Signal reconstruction	Historical epidemiological data (Chapter 2)
	Brain data (Chapter 3)
Signal mining	Power grid data (Chapter 4)
	Aircraft systems data (Chapter 5)

Table 1.1: Overview of the thesis

Part I

Part 1: Signal reconstruction

In Part I, we show how to reconstruct time sequences from multiple sources in larger granularity. We will introduce our proposed methods and applications for 1-dimensional signals (historical epidemiology data in Chapter 2) as well as high-dimensional signals (brain scans in Chapter 3).

Chapter 2

Historical epidemiological data

In this Chapter, we describe the methods that we have proposed to solve the signal reconstruction problem in historical epidemiological domain.

We introduce H-FUSE (section 2.1), GB-R (section 2.2), ARES (section 2.3), and HOMERUN (section 2.4). The methods are introduced in the order of increasing complexity. Here we list some of the distinctions between the methods.

- H-FUSE: This is the *simplest* signal reconstruction method. If the time sequences of your interest evolve *smoothly* over time, with a known *period*, then the most naive method you can try is H-FUSE.
- GB-R: If the target sequences follow the *epidemics model (*SIS*)*, then GB-R, which incorporates *nonlinear domain knowledge from epidemiology* is the better choice.
- ARES: If the target sequences are not necessarily epidemiological data, and you would like to *discover notable linear patterns* in your target sequences, we suggest using ARES.
- HOMERUN: If you know that the target sequences should be non-negative and are sure that the sequences consist of *multiple periods* (or wish to represent your target sequences using sparse dictionaries), then HOMERUN is the right choice.

In Table 2.1 below, we summarize a few main properties of the methods.

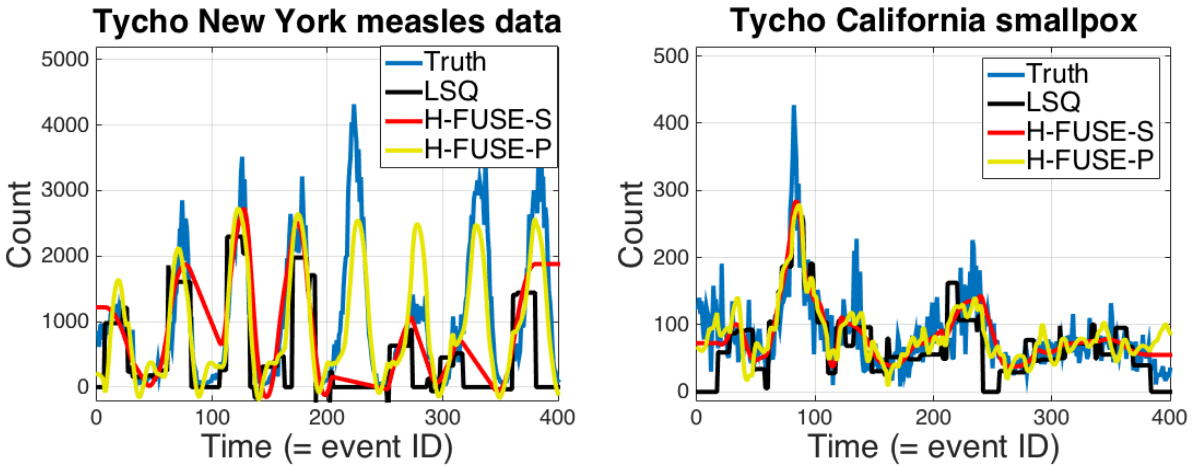
Table 2.1: Comparison of H-FUSE, GB-R, ARES, HOMERUN

Distinctions	LSQ	H-FUSE (2.1)	GB-R (2.2)	ARES (2.3)	HOMERUN (2.4)
Smoothness reconstruction		✓	✓	✓	✓
Periodicity reconstruction		✓	✓	✓	✓
Epidemics model (SIS)			✓		
Auto-detect linear patterns				✓	
Auto-detect multiple periods (quasi-periodic)					✓

2.1 H-FUSE: assuming smoothness and periodicity [77]

H-FUSE is our first attempt at solving signal reconstruction problem for epidemics data. The main idea behind our H-FUSE is to apply naive regularizers: 1) smoothness and 2) periodicity constraint. Our experimental results on the Tycho dataset shows that our choice of regularizers help signal reconstruction compared to using no regularizers at all.

2.1.1 Introduction



(a) Tycho New York measles data - $30\%, 81\%$ improvement by H-FUSE-S, H-FUSE-P over LSQ

(b) Tycho California smallpox data - $58\%, 41\%$ improvement by H-FUSE-S, H-FUSE-P over LSQ

Figure 2.1: **H-FUSE *is effective***: Tycho [100] (a) New York measles and (b) California smallpox counts per week (in *blue*). H-FUSE (in *red* and *yellow*) captures the cycles, outperforming top competitor 'LSQ' (in *black*).

Information fusion is the process of reconstructing objects from multiple observations. The need for information fusion appears in numerous domains, including multi-sensor data fusion [48], information fusion for data integration [14], and human-centered information fusion [49]. It is particularly important for interdisciplinary research, where a comprehensive picture of the subject requires large amounts of historical data from disparate data sources from a variety of disciplines. For example, epidemiological data analysis often relies upon knowledge of population dynamics, climate change, migration of biological species, drug development, etc. Similarly, information fusion is vital for exploring long-term and short-term social changes, where we need to consolidate data on social-scientific, health, and environmental dynamics. Such historical data sets are available from numerous groups worldwide such as the Institute for Quantitative Social Science and the Center for Geographic Analysis at Harvard, or World-Historical Database at the University of Pittsburgh.

In all cases, we have an unknown, target time sequence $\mathbf{x} = \{x_1, x_2, \ldots, x_T\}$ (say, of count of measles incidents in the USA, per week), and we want to reconstruct it, from aggregated information.

Informal Problem 1 (Information Fusion) *Informally, the problem is as follows:*

- Given: *several (aggregate) reports for the target sequence $\vec{x}$, (for example, some of the monthly sums from source 'A', and some of the weekly sums from source 'B')*
- Reconstruct *the target sequence $\vec{x}$, as accurately as possible, and*
- Confidence: *indicate whether we should trust the reconstruction.*

The challenges are the following:

- *Conflicts/Overlaps*: the reports may overlap; and even worse, conflict, e.g., the sum of the monthly reports for year, say 1993, might **not** be the same as the count for that year, from source 'B'.
- *Missing values*: Maybe none of our sources ('A' and 'B', above), covers the period of 1940-1944 (because, say, of World War II).
- *Trust in results*: How confident should one be on the reconstructed values? As we show later, in most cases our proposed method is very good, but in some cases, *no* method can do good reconstruction (see Recommendation 3 in Section 2.1.6).

We introduce H-FUSE, a systematic and efficient approach to address the problem of fusing aggregated historical data sources. Our H-FUSE consistently dis-aggregates historical data reports, addressing all the challenges mentioned above. The main idea is to take into account domain knowledge (e.g., smoothness, periodicity) and to infuse it as constraints in an optimization problem, when merging several historical reports.

The advantages of our method are

- **effective**: our experimental result on real data show that H-FUSE reconstructs the original data with good accuracy, outperforming top competitors: (see figure 2.1).
- **self-aware** : H-FUSE provides an assessment of when the recovery is not reliable.
- **scalable** : H-FUSE provably scales almost linearly on the length T of the target time sequence.

Figure 2.1 deserves some discussion: it plots the New York measles data (in blue, 'Truth'), the reconstruction of the top competitor (in black, 'LSQ', for least squares, see Section 2.1.2), and the two versions of H-FUSE ('Smoothness' in red, and 'Periodicity' in yellow). Our reconstructions are visibly better than 'LSQ', with up to 80% better reconstruction (see Section 2.1.4 for more details).

Reproducibility: the Tycho dataset is publicly available [100]; our code is open-sourced at `https://github.com/zonggel/GFusion`.

The outline of this section is typical: background, proposed method, experiments, theoretical analysis and conclusions.

2.1.2 Background and Related Work

In this section, we briefly introduce our problem domain, and related works.

Challenge of Historical Information Fusion

A major challenge in historical information fusion is estimating number of historical events from multiple aggregated reports while handling redundant, and possibly, inconsistent information.

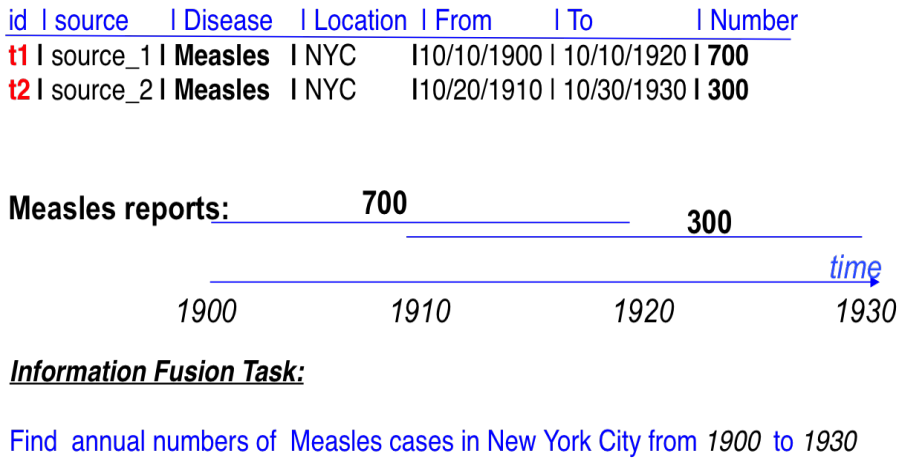


Figure 2.2: **Illustration on the nature of the data**. Time-overlapping historical reports

Figure 2.2 shows an example of a database with two historical reports on total cases of measles in NYC overlapping in time. Either of the reports are covering time intervals of 20 years. The task of information fusion would be estimating the population dynamics within smaller time units (e.g., what was the most likely annual numbers of measles cases in NYC from 1900 to 1930?). Granularity of the reports may differ. For example, we may need to estimate weekly numbers from monthly reports, or daily values from weekly aggregates. The process of information fusion requires efficient dis-aggregation of the reported data. In general, this problem can be stated in wider context of fusion and making sense of data obtained from a variety of sources, with gaps and overlaps in time and space, and uncertainty in trust of sources.

In our approach, we represent the overlapping historical report as a system of linear equations, – a *characteristic linear system*, as shown in Figure 2.3. Each report generates a binary row vector for coverage in an observation matrix with "ones" corresponding to the time units covered by the report. The characteristic system is commonly under-determined and we need to find a reasonably accurate approximate solution that would correspond to the dis-aggregated information.

Related Work

Our work is related to a more general problem of a large-scale information integration to cope with diverse data sources. A prominent example of a large-scale information integration project is Tycho [100, 126, 140]. Currently, Tycho consolidates information from approximately 50,000 reports on United States epidemiological data spanning more than 100 years. We used Tycho data for experimental evaluation of our method in Section 2.1.4.

Historical information fusion task often requires dis-aggregation of historical reports and solving an under-determined linear system. Temporal dis-aggregation methods have been studied in time series analysis of mostly economic data (see [20, 28, 107] for review). Given a low frequency time series (e.g. annual sales, weekly stock market index, etc.) the goal of temporal

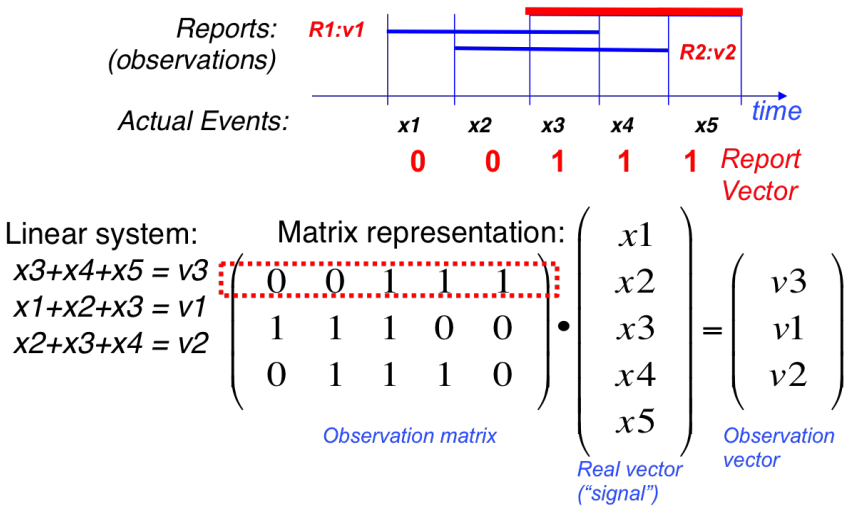


Figure 2.3: **Illustration on our problem setting**. Characteristic Linear System of time-overlapping historical reports

dis-aggregation is to produce a high-resolution series (e.g. quarterly sales, daily stock market index, etc.) while satisfying temporal aggregation constraints.

Temporal aggregation constraints ensure that the sum, average, the first or the last value of the resulting high frequency time series is consistent with the low frequency series. If available, related series observed at the required high frequency can be used to dis-aggregate the original observations. Such series are called indicators. However, care must be taken when selecting indicators since two strongly correlated low frequency time series may not be correlated at a higher frequency [44]. Thus, choosing good indicator series is not a straightforward task. Temporal dis-aggregation methods have been used for the cases of non-overlapping aggregated reports and cannot be directly applied to the task of historical information fusion.

Table 2.2 contrasts our H-FUSE method against the state-of-the-art competitors. We present our method in the next section.

The challenge of data fusion has been studied in various domains such as image processing, signal processing, geology, etc, that deals with uncertainty in the task. The image and signal processing community has been dealing with related ill-posed problems such as edge detection, surface reconstruction [143], vehicle detection [106], or super-resolution image reconstruction [91, 98]. A common way to find an approximate solution of an under-determined linear system is to apply least squares method (LSQ) and Tikhonov regularization [43, 45], by introducing additional constraints such as smoothness in space and/or temporal domain so that reconstruction reflects some aspects of the target data. Although Tikhonov regularization has been widely used in solving ill-posed problem in various communities, to our knowledge the application of Tikhonov regularization has not been addressed in historical data fusion domain.

In 'VLDB97', Faloutsos et. al. [41] use smoothness, for information fusion. The summaries were expected to be consecutive and non-overlapping; we believe that it could handle overlaps and missing values (indicated as '?' in Table 2.2) but the method is completely incapable of handling conflicts.

Property	*LSQ*	*VLDB97[41]*	*TDisaggregation*	*H-FUSE*
Scalability	✓	✓		✔
Self-awareness		✓		✔
Overlapping reports	✓	?		✔
Missing values		?		✔
Conflicting reports				✔
Domain knowledge		?	✓	✔

Table 2.2: **H-FUSE matches all specs**, while competitors miss one or more of the features.

2.1.3 Method

In this section, we explain our H-FUSE in more details. Table 2.3 gives the list of symbols we use.

The common requirement in all reconstruction methods, is that the reconstructed sequence $\vec{x}$ should satisfy the reports/facts, that is

$$\mathcal{F}(\vec{x}) = \sum_{n=1}^{N} (v_n - \sum_{t=1}^{T} \mathbf{O}_{nt} x_t)^2 \tag{2.1}$$

and, in matrix form:

$$\mathcal{F}(\vec{x}) = ||\vec{v} - \mathbf{O}\vec{x}||_2^2 \tag{2.2}$$

Ideally, the deviation from the facts should be zero, unless the facts/reports are conflicting. The top competitor, 'LSQ', stopped here, and tried to minimize $\mathcal{F}()$; since the problem is (usually) under-determined, 'LSQ' proposed to find the minimum-norm solution ($\min\ ||\vec{x}||_2^2$) that satisfies $\mathcal{F}$. This is a well-understood problem, and 'LSQ' can find that unique solution using the so-called *Moore-Penrose pseudo-inverse*.

But there is no reason why the solution would have minimum norm, which leads to our proposed solution.

Intuition

The main idea behind our H-FUSE is to infuse domain knowledge. For example, in most cases where the solution sequence $\vec{x}$ should be smooth, we propose to penalize large differences between adjacent timeticks; if we know that the periodic, we propose to also impose periodicity constraints.

More formally, our approach is to find the values (x_t, $t = 1, \ldots, T$) that (a) can be aggregated to generate observed report ($\vec{v}$) and (b) minimize a domain-dependent penalty functions. Thus,

Symbols	Definitions
$\vec{x}$	$= (x_1, \ldots, x_T)$: target time series (*unknown.*)
T	total number of timeticks in $\vec{x}$
PD	(smallest) period of $\vec{x}$
N	total number of reports
D	report duration
$\vec{v}$	$= (v_1, \ldots, v_N)$: values of reports (*observed* - aggregated form of *unknown* $\vec{x}$)
$\mathbf{O}$	$N \times T$ observation matrix ($\vec{v} = \mathbf{O}\vec{x}$)
$\mathcal{F}(\vec{x})$	deviation from facts/reports
$\mathcal{C}(\vec{x})$	domain-imposed soft constraint
$\mathcal{L}(\vec{x})$	total penalty ('loss')
$\mathbf{H}_s$	$(T-1) \times T$ smoothness matrix
$\mathbf{H}_p$	$(T-PD) \times T$ periodicity matrix

Table 2.3: Symbols and Definitions

we propose to formulate the optimization problem as follows:

$$\min_{\vec{x}} \mathcal{L}(\vec{x}) = \min_{\vec{x}} \ (\mathcal{F}(\vec{x}) + \mathcal{C}(\vec{x})) \tag{2.3}$$

where $\mathcal{L}(\vec{x})$ stands for the total penalty ('loss', hence the symbol $\mathcal{L}$), and consists of two components: The first is $\mathcal{F}(\vec{x})$, the deviation from the reports ('facts'), that we defined before (Eq. 2.2). The second component, $\mathcal{C}(\vec{x})$, infuses domain knowledge, in the form of soft constraints, like smoothness and periodicity, that we explain below. It could also infuse other types of domain knowledge, like sparsity, adherence to an epidemiology model like SIS (susceptible-infected-susceptible, like the flu, as we do in Section 2.2), but we will not elaborate here. Let us focus on the two constraints that we propose, since they proved to be the most successful in our experiments.

- Smoothness constraint $\mathcal{C}_s$ This constraint penalizes big jumps between successive timeticks. Formally:

$$\mathcal{C}_s(\vec{x}) = \sum_{t=1}^{T} (x_t - x_{t+1})^2 = ||\mathbf{H}_s\vec{x}||_2^2 \tag{2.4}$$

  where $\mathbf{H}_s$ is a $\mathbb{R}^{(T-1)\times T}$ matrix whose t^{th} row has 1 and -1 in the t^{th} and $(t+1)^{th}$ column, respectively.
- Periodicity constraint $\mathcal{C}_p$: If there is a period (say PD=52 weeks = 1 year) in our data, we can penalize deviations from that, as follows:

$$\mathcal{C}_p(\vec{x}) = \sum_{t=1}^{T} (x_t - x_{t+PD})^2 = ||\mathbf{H}_p\vec{x}||_2^2 \tag{2.5}$$

  where $\mathbf{H}_p$ is a $\mathbb{R}^{(T-PD)\times T}$ matrix whose t^{th} row has 1 and -1 in the t^{th} and $(t+PD)^{th}$

column respectively. The intuition here is to make the event at timetick t to be close to the one at $t + PD$.

Subtle issue: relative weights. A careful reader may wonder whether we should give different weight to deviations from the facts $\mathcal{F}(\cdot)$, as opposed to the (soft) constraints/conjectures $\mathcal{C}(\cdot)$.

The short answer is '*no*'. The long answer is that we tried a weighting parameter λ, and we tried to minimize the loss function $\mathcal{L}(\cdot) = \mathcal{F}(\cdot) + \lambda\mathcal{C}(\cdot)$. However, we recommend to set λ=1, for the following reasons: (a) it gives optimal, or near-optimal results, for all real cases we tried (as compared to the ground truth $\vec{x}$); (b) the results are insensitive to the exact value of λ, when $\lambda \leq 1$; (c) it is hard for a practitioner to set the value of λ, given that the target sequence is unknown.

Methods

In short, smoothness is the constraint that we propose to use as default, if the domain expert has nothing else to tell us. The reason is that it performs well, as we see in the experiments, as well as for theoretical reasons (Lemma 2).

If the domain expert believes that there is a periodicity with period PD, then we can do even better. The exact problem formulations are as follows.

H-FUSE-S (Smoothness Method)

The proposed loss function $\mathcal{L}_s(\cdot)$ is given by Equation 2.3

$$\mathcal{L}_s(\vec{x}) = \mathcal{F}(\vec{x}) + \mathcal{C}_s(\vec{x})$$

which gives, in matrix form:

$$\min_{\vec{x}} \mathcal{L}_s(\vec{x}) = \min_{\vec{x}} \ (||\vec{v} - \mathbf{O}\vec{x}||_2^2 + ||\mathbf{H}_s\vec{x}||_2^2) \tag{2.6}$$

H-FUSE-P (Periodicity Method)

Time sequence data are often periodic. Historical events such as epidemics, weather measurements demonstrate repeating cycles of patterns. For example, flu outbreak records may have a seasonal pattern where there is a peak in the winter and recovery in the summer season. Weather measurements may demonstrate cyclic patterns in days, and seasons.

For such cases, we propose to impose both smoothness, as well as periodicity constraints, with the period PD that a domain expert will provide.

Then, we propose the loss function $\mathcal{L}_p(\cdot)$ to be

$$\mathcal{L}_p(\vec{x}) = \mathcal{F}(\vec{x}) + 1/2\mathcal{C}_s(\vec{x}) + 1/2\mathcal{C}_p(\vec{x})$$

which leads to the optimization problem below:

$$\min_{\vec{x}} ||\vec{v} - \mathbf{O}\vec{x}||_2^2 + 1/2||\mathbf{H}_s\vec{x}||_2^2 + 1/2||\mathbf{H}_p\vec{x}||_2^2 \tag{2.7}$$

We chose equal weights of $1/2$ for each of the constraints, so that they will not overwhelm the facts-penalties $\mathcal{F}(\cdot)$. In any case, the reconstruction quality is rather insensitive to the exact choice of weights, with similar arguments as we discussed earlier about the λ weight (see subsection 'Intuition' 2.1.3).

2.1.4 Experiments

In this section we report experimental results of our H-FUSE on the real data.

Experimental setup

To prove the effectiveness of H-FUSE on the real data, we apply H-FUSE on Tycho [100] New York measles data which is our default main dataset. The full dataset contains 3952 weekly records with some missing values. We carefully select the time period without missing values, Week 51 to Week 450, which gives us 400 records in total as our selected dataset. To test the generality of H-FUSE, we also select 400 weekly records from California smallpox data, ranging from Week 501 to Week 900.

We refer to the observations or records as *reports*, and the number of observations as *report numbers*, and the timeticks that each report covers as *report duration*. We vary the report numbers and the report duration to conduct sensitivity test of H-FUSE. The reports are generated randomly for specific report number and report duration combination, i.e., the mixing matrix $\mathbf{O}$ is constructed to reflect the report number and duration that we set.

Effectiveness of H-FUSE-S

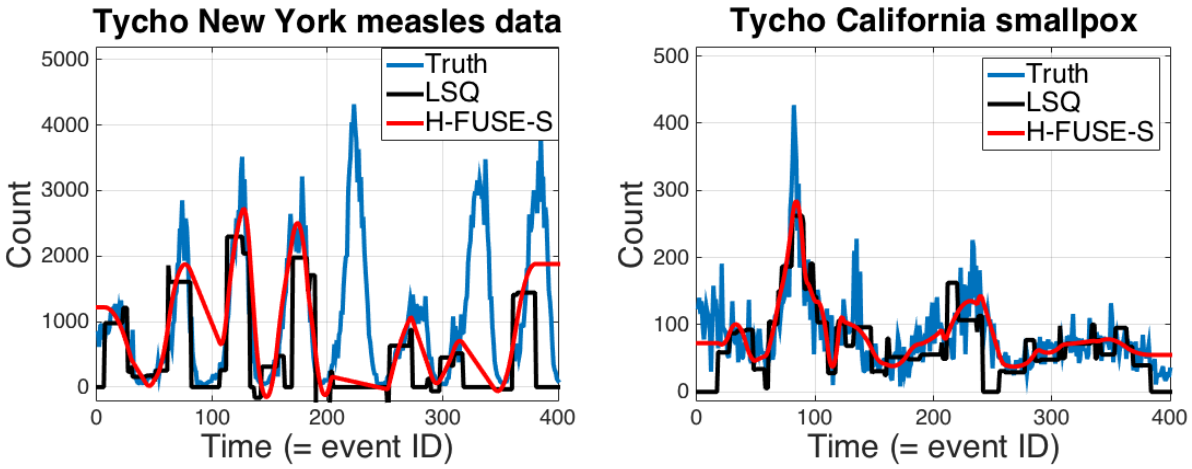
In this section, we compare the reconstruction performance of H-FUSE-S and the conventional approach, LSQ method. In Figure 2.4, the reconstruction comparisons of LSQ method and H-FUSE-S are shown for (a) Tycho New York measles data, and (b) Tycho California smallpox data. We see that generally, H-FUSE with smoothness constraint gives better reconstruction than the LSQ method.

We further conducted experiment under various configurations of report number and report duration ranging from 10 to 80. For each configuration, we repeat the experiment 100 times, and average over the reconstruction MSE. The error dynamics of H-FUSE with smoothness constraint on various settings of report number and report duration is shown in Figure 2.5 (a). We vary the report number and the report duration in the x-axis and y-axis, respectively. Here brighter color indicates higher MSE in reconstruction. From the figure, we observe a clear trend of decrease in reconstruction MSE as the report number increases, and MSE reach its minimum for a combination of large report number and long report duration.

Effectiveness of H-FUSE-P

In the case of Tycho New York measles data, it is known that it has a periodicity of one year. Therefore, we apply $\mathcal{C}_p$ periodicity constraint in addition to $\mathcal{C}_s$ smoothness constraint on the measles data, i.e. H-FUSE-P as described in Section 2.1.3. Earlier in Figure 2.1, the reconstruction comparisons of LSQ method, H-FUSE-S and H-FUSE-P give us a clear picture on how periodicity constraint improves the performance in addition to that of the smoothness constraint. We observe that in almost all cases the additional periodicity constraint results in smaller MSE than the smoothness constraint alone.

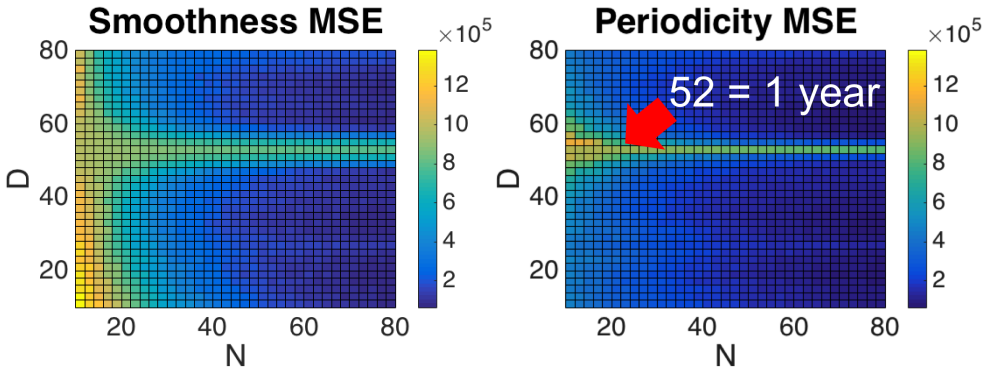
The error dynamics of H-FUSE-P on various settings of report number and report duration is shown in Figure 2.5 (b). We observe a similar trend as in the simple smoothness constraint



(a) Tycho New York measles data - 30% improvement over LSQ

(b) Tycho California smallpox data - 58% improvement over LSQ

Figure 2.4: **H-FUSE-S reconstructs well:** H-FUSE-S wins over LSQ method in reconstruction all along. (a) Tycho New York measles data ($N = 20, D = 20$) - H-FUSE-S reconstructs 30% better than LSQ. (b) Tycho California smallpox data ($N = 30, D = 30$) - H-FUSE-S reconstructs 58% better than LSQ.



(a) H-FUSE-S

(b) H-FUSE-P

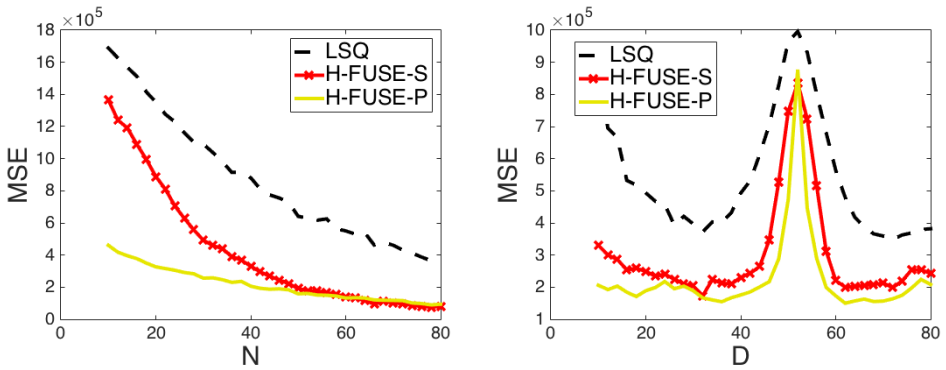
Figure 2.5: **H-FUSE with various configurations**. Both versions of H-FUSE reconstruct well (*blue*) unless the report duration matches exactly with the period $D = P(= 52)$.

case: the MSE decreases as the report number increases. However, we observe that the additional periodicity constraint significantly improves the accuracy in terms of MSE.

In fact, among all of the report number and report duration configurations, there were only a very few case when H-FUSE-S outperformed the H-FUSE-P. The result is illustrated in more detail in Figure 2.5 (b). These cases arise because when we have many reports, neither assumption is required to be strong any more. Therefore they have similar level of performance.

In Figure 2.6 (a), we study the change in MSE with varying report numbers. The overall observation is that the MSE decreases with the increase of the number of reports.

We also analyzed the change in MSE with varying report duration in Figure 2.6 (b). We observe that MSE displays a periodic pattern, reaching its peaks when $D = 52$. The reason is that when report duration matches the periodicity of the data, all of the values in any of the report will have similar values, leading to deficit of information.



(a) error vs. report number (b) error vs. report duration

Figure 2.6: **H-FUSE wins consistently** (a) Error decreases consistently with report number N. (here $D = 40$) (b) Error is almost constant with respect to report duration D unless $D = P = 52$ (here $N = 40$).

2.1.5 Theoretical analysis: Self-awareness, and Scalability

In this section we provide theoretical analysis on H-FUSE. From the theoretical analysis, we induce a set of observations that provide an assessment of the cases when the reconstruction is not reliable, which we refer to as "self-awareness" of H-FUSE. Also, we demonstrate the scalability property of H-FUSE in terms of both theory and empirical aspects.

Background

Consider

$$\min_{\vec{x}} ||\vec{v} - \mathbf{O}\vec{x}||_2^2 + ||\mathbf{H}\vec{x}||_2^2 \tag{2.8}$$

which is equivalent to

$$\min_{\vec{x}} \left|\left| \begin{bmatrix} \vec{v} \\ - \\ \vec{0} \end{bmatrix} - \begin{bmatrix} \mathbf{O} \\ - \\ \mathbf{H} \end{bmatrix} \vec{x} \right|\right|_2^2 . \tag{2.9}$$

Here $\mathbf{H}$ is of type

$$\mathbf{H}_s = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}, \tag{2.10}$$

Signals in the right null space of the matrix are smooth. In the case of $\mathbf{H}_s$, the signals are constant. More generally, notice that if $|\vec{x}(n) - \vec{x}(n-1)| \leq \epsilon$, then $||\mathbf{H}_s\vec{x}||_2^2 \leq \epsilon^2(T-1)$, where $T = \text{length}(\vec{x})$.

In short, the inclusion of the regularizer $||\mathbf{H}\vec{x}||_2^2$, nudges $\vec{x}$ into a smoother solution vector.

Self-awareness

What can we say about the error of reconstruction? We give the theoretical analysis here, and later on, show how they translate to practical recommendations.

The first lemma states that if our target sequence $\vec{x}$ satisfies a smoothness condition, and if we have enough 'suitable' equations, then we can have error-free reconstruction.

Formally, let $\mathbf{O}$ denote the $N \times T$ observation matrix, $\vec{x}$ be the target time series, and $\mathbf{H}$ is one of the smoothness regularization matrices in Eq. 2.10.

Lemma 1 *With the above notations, if (a)* $\begin{bmatrix} \mathbf{O} \\ - \\ \mathbf{H} \end{bmatrix}$ *is tall or square and full column rank, and (b)* $\mathbf{H}\vec{x} = 0$*, then we can have error-free reconstruction.*

Proof 1 *Special case of the upcoming Lemma 2.* ■

The first condition (full column rank) is almost always true; the second condition is rather strict. The next Lemma relaxes it, effectively stating that if our target sequence is close to smooth ($\mathbf{H}\vec{x} \approx 0$), then the reconstruction error is small. Formally, with the same notations as above, we have:

Lemma 2 *If* $\begin{bmatrix} \mathbf{O} \\ - \\ \mathbf{H} \end{bmatrix}$ *is tall or square and full column rank, the squared error for our smoothness reconstruction is given by*

$$SE = ||(\mathbf{O}^T\mathbf{O} + \mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{H}\vec{x}||_2^2 \tag{2.11}$$

Proof 2 *Let $\hat{\vec{x}}$ be the solution we obtain from solving Equation 2.8. This is an over-determined system (more rows/equations than unknowns/columns), and the solution is given by*

$$\hat{\vec{x}} = (\mathbf{O}^T\mathbf{O} + \mathbf{H}^T\mathbf{H})^{-1}\mathbf{O}^T\vec{v}$$

Then for $\vec{x}$, we have trivially that

$$\vec{x} = \mathbf{I}\vec{x}$$

with $\mathbf{I}$ *being the* $T \times T$ *identity matrix. Then:*

$$\begin{aligned}\vec{x} &= \left(\begin{bmatrix} \mathbf{O}^T | \mathbf{H}^T \end{bmatrix} \begin{bmatrix} \mathbf{O} \\ - \\ \mathbf{H} \end{bmatrix} \right)^{-1} \begin{bmatrix} \mathbf{O}^T | \mathbf{H}^T \end{bmatrix} \begin{bmatrix} \mathbf{O}\vec{x} \\ - \\ \mathbf{H}\vec{x} \end{bmatrix} \\ &= (\mathbf{O}^T\mathbf{O} + \mathbf{H}^T\mathbf{H})^{-1}(\mathbf{O}^T\vec{v} + \mathbf{H}^T\mathbf{H}\vec{x}) \\ &= \hat{\vec{x}} + \left(\mathbf{O}^T\mathbf{O} + \mathbf{H}^T\mathbf{H}\right)^{-1} \mathbf{H}^T\mathbf{H}\vec{x}\end{aligned}$$

Thus, the error vector $\vec{e} = \vec{x} - \hat{\vec{x}}$ *is*

$$\vec{e} = \left(\mathbf{O}^T\mathbf{O} + \mathbf{H}^T\mathbf{H}\right)^{-1} \mathbf{H}^T\mathbf{H}\vec{x}$$

and its squared norm is given by Eq. 2.11 ■

Clearly, when $\mathbf{H}\vec{x} = 0$, the loss becomes zero, which is exactly what Lemma 1 says. Otherwise, the error depends on how much $\vec{x}$ deviates from smoothness ($\mathbf{H}\vec{x}$), as well as the specifics of the $\mathbf{O}$ and $\mathbf{H}$ matrices.

The above lemmas hold for arbitrary starting points of the reports, and for arbitrary lengths. We can provide additional bounds and guide-lines, for the (very realistic) setting that the reports all have the same length D. We distinguish 3 settings, in this case:

- *semi-random report*: The starting points of the reports, are random. Thus, they may overlap, and/or coincide, and/or leave uncovered parts of the target signal $\vec{x}$.
- *tile report*: The reports have deterministic starting points (1, D+1, $2 * D$+1, ...) and thus they cover the whole time interval, without overlaps.
- *shingle report*: General case of *tile report*: successive reports overlap by o time-ticks (for *tile report*, o=0).

Then we can give additional guarantees.

Lemma 3 *Given an infinite target time series* $\vec{x}$*, with smallest period* PD*; in a* tile report *setting of duration* D*, if*

$$D < PD/2$$

then there exists a method for reconstructing the signal with no error.

Proof 3 *See [41]. The proof is closely related to the Nyquist sampling frequency.* ■

Lemma 4 *Given an infinite target time series* $\vec{x}$*, with smallest period* PD *in a* shingle report *setting of duration* D*, if*

$$D < PD/2$$

then there exists a method for reconstructing the signal with no error.

Proof 4 *Choose the subset of reports that form a* tile report *setting - by Lemma 3 we can have error-free reconstruction.* ■

Informally, the above Lemmas explains our empirical observations which show that if the target time series $\vec{x}$ is finite and periodic with smallest period P (52 weeks, in our measles data), and if we have *tile report* (or *shingle report*) reports with $D < PD/2$, H-FUSE (with its smoothness constraint) will result in small error.

When the $D \geq PD/2$, there are no recommendations any more - H-FUSE may, or may not, result in large errors.

Scalability

Our H-FUSE eventually needs to solve a sparse linear system, and, intuitively, this should be fast. This is indeed the case, as we show next. Let D_{max} be the duration of the longest report, and assume that $D_{max} \geq b$, where b is the bandwidth of the $\mathbf{H}$ matrix - b=2 for $\mathbf{H}_s$ as in Section 2.1.3. With the usual notation ($\vec{x}$ is the target sequence, of length T), we have the Lemma:

Lemma 5 *For any report setting, let D_{max} be the longest report duration. Then the total computation time for our* H-FUSE-S *is*

$$O(Tlog(T) + 4D_{max}^2)$$

Proof 5 *The most time consuming part is the matrix inversion. The $\mathbf{O}$ is a banded Toeplitz matrix with bandwidth D_{max}, thus $\mathbf{O}^T\mathbf{O}$ is likewise a banded Toeplitz matrix of bandwidth $2D_{max} - 1$; and the same holds for $\mathbf{H}$ and $\mathbf{H}^T\mathbf{H}$, with bandwidth $b \leq D_{max}$. Then, the result follows from [59].* ■

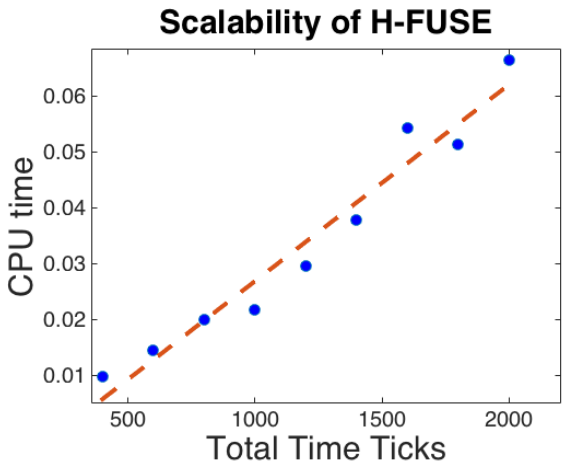


Figure 2.7: **H-FUSE is scalable:** H-FUSE-S scales near-linearly on the length T. CPU time (seconds) vs T, in lin-lin scales; dashed orange line is plotted for reference.

Empirically, our H-FUSE seems to have even better scalability, close to linear: Figure 2.7 shows the wall-clock time versus the sequence length T. We used "regular" reports, with duration $D = 200$, on the full New York measles data with $T = 3,952$ time ticks and applied H-FUSE with the smoothness constraint.

2.1.6 Practitioner's Guide

From the above discussion, smoothness has desirable theoretical properties, and, as the experiments show, it is a good choice for solving the information fusion problem. Moreover, if the domain expert knows that there is a periodicity of period, say PD, our H-FUSE can incorporate it and achieve even better reconstruction.

The question is when should the domain expert trust (or discard) the results of our reconstruction? We summarize our recommendations, next.

Recommendation 1 (Smoothness is effective) *If the target sequence $\vec{x}$ is smooth, and we have enough reports, then* H-FUSE *achieves good reconstruction.*

The error is given by Lemma 2, and it is zero, if $\vec{x}$ is perfectly smooth (Lemma 1) Figure 2.6 (a) provides evidence.

Recommendation 2 (Nyquist-like setting) *When we have regular reports frequently enough (i.e., with $D < PD/2$), then we can expect small error from our* H-FUSE.

This is the informal version of Lemma 3, and illustrated in Figure 2.6. PD is the smallest period of our target signal (eg., PD=52 weeks, in our measles data).

Finally, we did not provide a proof, but the recommendation is obvious. Given reports of the same length D, we have:

Recommendation 3 (Obliteration) *If the report length D coincides with the period PD of the signal, large errors are possible.*

The intuition is that, say, if all our reports span exactly D=52 weeks (=1 year), there is no way anyone can recover the annual (March) spikes of measles. Figure 2.5 gives us an arithmetic example where you have large MSE for D=52 weeks.

2.1.7 Conclusion

We proposed H-FUSE method that efficiently reconstructs historical counts from possibly overlapping aggregated reports. We propose a principled way of recovering a times sequence from its partial sums, by formulating it as an optimization problem with various constraints (Eq. 2.3). Our formulation allows the injection of domain knowledge (smoothness, periodicity, etc). Our method has the following major properties:

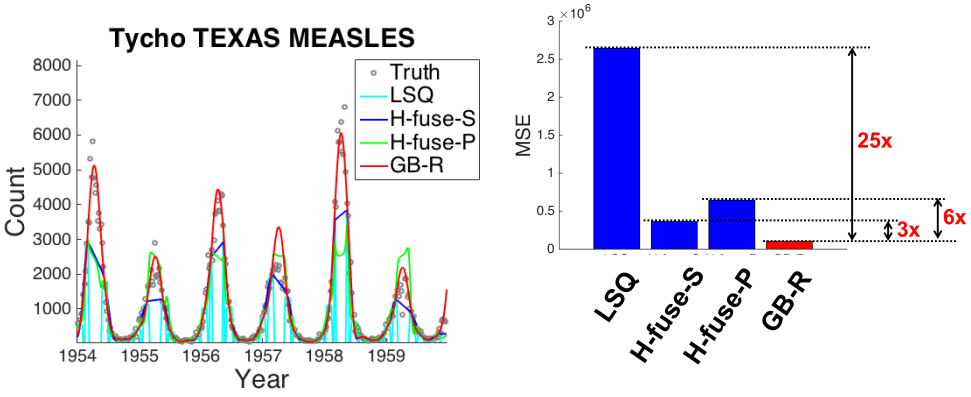
1. **Effectiveness**: The experimental result on the real-world Tycho New York measles data, outperformed the reconstruction by naive approach as shown in Section 2.1.4.
2. **Self-awareness**: We provide theoretical results that help evaluate the quality of the reconstruction as discussed in Section 2.1.5 'Self-awareness'.
3. **Scalability**: The computational cost for H-FUSE scales nicely as $O(Tlog(T) + 4D^2)$ as discussed in Section 2.1.5 'Scalability'.

2.2 GB-R: incorporating SIS epidemics model [116]

In this section, we introduce GB-R, a signal reconstruction method specifically designed for epidemics dataset. We utilize one of the prominent epidemics evolving model called "*SIS model*" as the constraint to fit the unknown time series x. Utilizing the domain knowledge allows for improvement over the baseline method that does not consider the data specific characteristics. With SIS model, we show how GB-R is able to make interpretations out of the data as well.

2.2.1 Introduction

Given reports from multiple data sources, in several granularities, (like *monthly* counts of sales from one source, and *weekly* counts of sales from another), with some of the reports missing, how can we reconstruct the *daily* sales counts? The problem is usually under-determined - how can we find a realistic solution, if experts advice us that the real sequence obeys a well-defined model, like the susceptible-infected-susceptible ('SIS') cascade model?



(a) (b)

Figure 2.8: **GB-R outperforms competitors:** (a) GB-R (*red*) is closer to truth (*gray dot*), in measles patient-counts over time (1954-1960); (b) it gives **3x** to **25x** better MSE in reconstruction.

More formally, the problem is as follows: Let $\mathbf{x} = (x_1, \ldots, x_T)$ be the unknown time sequence (say, weekly counts of flu patients); let $\mathbf{y} = (y_1, \ldots, y_N)$ be the aggregated information (say, reports of monthly sums, with some of them missing); and suppose that domain experts advise that the unknown sequence $\mathbf{x}$ obeys a differential equation, like the SIS model (see equation (2.13)), or some other model; such models have a few parameters, like the patient-recovery rate δ, the spreading-rate β, etc. Our goal is to reconstruct the time sequence $\mathbf{x}$ (and the model parameters β etc), so that $\mathbf{x}$ satisfies the aggregated information ($\mathbf{y}$), and also matches the gray-box model as best as it can.

Informal Problem 2 (Gray-Box Reconstruction)

- Given: *the reports* $\mathbf{y}$*; and a differential equation (gray box model, with some unknown parameters)*

- Reconstruct: *the true time series* $\mathbf{x}$*, and the gray-box-model parameters*

One especially challenging setting is when the peak values are wrong, or missing (*"flooding effect"*). For example, when there is an outburst of an epidemic in an developing regions (e.g. Ebola epidemic in West Africa[1]), clinical practitioners may prioritize clinical care over case reporting. Similarly, in computer traffic monitoring, during peak times, routers drop packets, resulting in incorrect count of packets.

We propose the "GB-R" method to solve the gray-box reconstruction problem. What sets GB-R apart from all other reconstruction methods is that it carefully infuses domain knowledge (like the SIS cascade model), with dual benefits: (a) *accuracy*: it better reconstructs the unknown time series $\mathbf{x}$; and (b) *interpretability*: it also estimates the gray-box model parameters (infection rate, etc), thus allowing a better understanding of the cascade.

Figure 2.8 compares GB-R against top contenders. The real data were the Project Tycho level1 measles weekly counts in the state of Texas, during 1954-1960 [100] (*gray dots* in Figure 2.8-(a)). We generated the 5-week sums, omitting the peak times ("flooding effect" scenario). From those *5-week sums*, we reconstructed the true data to *weekly* granularity, using GB-R in *red*, and the competitors, 'LSQ' ([140]), 'H-FUSE-S' ([77]), 'H-FUSE-P' ([77]) in *cyan*, *blue*, and *green*, respectively. The proposed GB-R method reconstructs the true data very closely, including the peaks, while other methods fail to reconstruct. Figure 2.8(b) shows GB-R achieves **3x**- **25x** better reconstruction than the competitors.

The contribution of our work is as follows:

1. **Effectiveness**: GB-R outperforms the competitors, often by **3x-25x** in reconstruction error.
2. **Scalability**: GB-R scales linearly with respect to the length of the time sequence.
3. **Interpretability**: GB-R also estimates the gray-box-model parameters (spreading-rate, patient-recovery rate, timing of peak, etc), providing a better understanding of the time sequence.

Reproducibility: Our code for GB-R is open-sourced at: `http://www.cs.cmu.edu/~hyunahs/code/GB_R.zip`. All the data we used are publicly available at `https://www.tycho.pitt.edu/`

The outline of this section is the typical one: Background and related work; method description; intuition behind our approach; experiments; and conclusions.

2.2.2 Background and Related Work

In this section, we provide a brief background on the basic epidemic model, and the information fusion problem. Then we introduce some of the previous work related to our work.

SIS epidemic model basics

Among the numerous epidemic models (SI, SIR, SIRS, etc [6]), we choose here the SIS model, because it is the simplest one and still gives good fits. The SIS model is suitable for, say, the flu,

[1] `http://www.nejm.org/doi/full/10.1056/NEJMsr1600236?af=R&rss=currentIssue#t=article`

where we assume no immunity: a person is either susceptible ('S', i.e., healthy but possible to get infected), or infected ('I', i.e., sick). An infected person heals with probability δ, and becomes susceptible again (no immunity); an infected person may infect a healthy person with probability β.

Let x_t be the count of infected people at time t. Then, the SIS model uses the following differential equation:

$$\frac{dx_t}{dt} = \beta x_t(P - x_t) - \delta x_t \tag{2.12}$$

The equivalent difference equation is: $x_t = x_{t-1} + \beta x_{t-1}(P - x_{t-1}) - \delta x_{t-1}$ where β, δ are the infection rate and recovery rate of the disease, and P is the population size.

In [85], the authors modified the SIS model to allow for periodicity (flu propagates faster during the colder months, etc). and they modeled the infection rate β as periodic function of time, β_t, as follows:

$$x_t = x_{t-1} + \beta_{t-1} x_{t-1}(P - x_{t-1}) - \delta x_{t-1} \tag{2.13}$$

where β_t is defined as

$$\beta_t = \beta_0\Big(1 + P_a \cos\big(\frac{2\pi}{P_p}(t + P_s)\big)\Big) \tag{2.14}$$

and β_0, P_a, P_s, P_p are the average infection rate of the disease, the amplitude, the phase shift, and the period of the infection rate, respectively.[2]

Information fusion basics

(We saw most of these materials in Section 2.1.3, but we repeat them here for the readers' convenience.)

As mentioned, the problem is to recover the *true data* $\mathbf{x}$=($x_1, \ldots, x_T$) (e.g., daily counts of infected people) given some aggregated counts or reports $\mathbf{y}$=($y_1, \ldots, y_N$), say, monthly sums. Next, we show that this becomes an under-specified, linear problem.

Figure 2.9 illustrates the concepts that we need, which are defined below:

- *true data* $\mathbf{x}$: the finer-scale data that we want to reconstruct, in length of T.
- *a report* (y): a sum of true data over a longer period (in Figure 2.9, report 1 ($y_{i=1}$) is a *sum of 30 weeks*) The *report coverage* of report i is the time interval that it spans (t^i_{start}, t^i_{end}), and the *report duration* is obviously the time length ($t^i_{end} - t^i_{start} + 1$).

The given N reports $\mathbf{y} = (y_1, \ldots, y_N)$ are the hard constraints that we need to satisfy.

The information fusion task can be formulated as a system of linear equations, the *characteristic linear system* [77]. We need the concept of mixing matrix $\mathbf{A}$: this is an $N \times T$ matrix, where N and T are the total number of reports and total number of timeticks in the true data, respectively; with all entries as zero, except $A_{it} = 1$ when the i-th report includes the t-th timetick (i.e., $t^i_{start} \leq t \leq t^i_{end}$, for each i^{th} report as shown in Figure 2.10).

[2]Note that the conventional formulation of the SIS model would be $\frac{\beta}{P}$, proportioned by the population instead of just β, but in terms of the objective function, there is no difference.

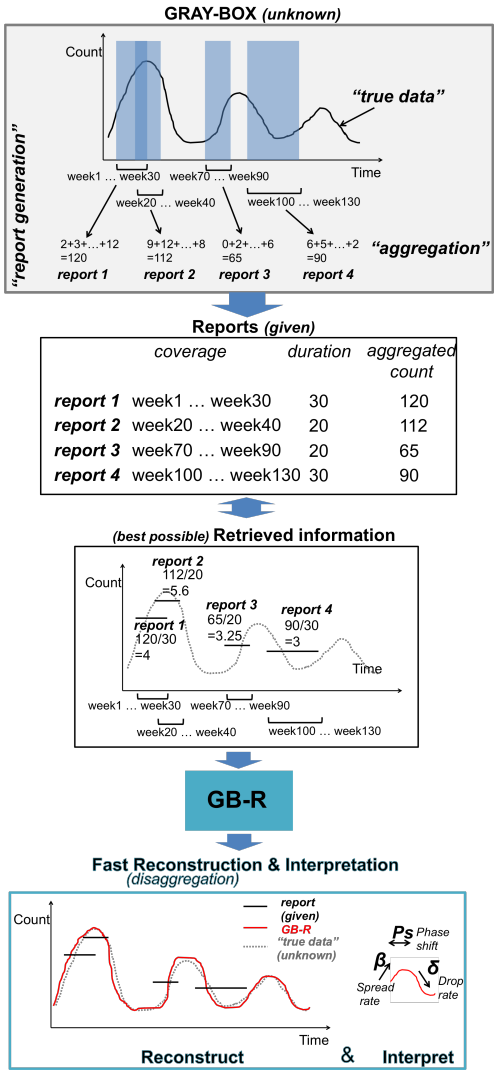


Figure 2.9: Information fusion basics.

With the above definition of the mixing matrix $\mathbf{A}$, we have

$$\mathbf{y} = \mathbf{A}\mathbf{x} \tag{2.15}$$

which is under-determined, since $N < T$.

Related works

(Most of the related works have been mentioned in Section 2.1.2, but we repeat them here for the readers' convenience.)

Previous works fall in the following groups: information fusion, super resolution reconstruction, and epidemics modeling.

Infofusion: One of the most well-known information fusion projects in the historical data domain is the Tycho project [100, 126, 140]. Project Tycho is an open-access repository for disease data for which over 6,500 weekly disease reports were digitized and curated, spanning

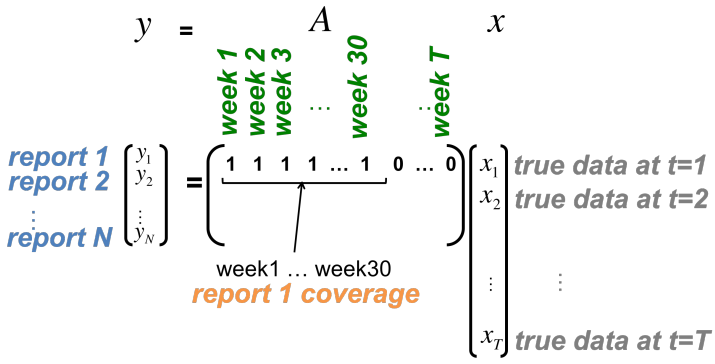


Figure 2.10: Information fusion formulation

125+ years and 100+ diseases. Data entry for Project Tycho took 200 million keystrokes, making it into a valuable real-world dataset and a testbed for the problem of information fusion.

A recent work appeared in [77], where the main idea is to assume that the unknown sequence $\mathbf{x}$ is smooth, and/or periodic. The resulting methods, H-FUSE-S and H-FUSE-P, give good results on Tycho data, and we use them as baselines here, together with the least squares method "LSQ". (throughout this section, we use the term LSQ to refer to the minimum norm solution to the under-determined system, following the convention of [77] - while in signal processing, it usually means the least squares solution to the over-determined problem.)

Super resolution reconstruction: Information fusion and super resolution reconstruction share a similar goal of reconstructing higher resolution data from low. The most basic algorithm for super resolution reconstruction is the least squares (LSQ) method along with Tikhonov regularization [43, 45] that enforce additional constraints such as smoothness. Super resolution reconstruction has been studied in various fields like signal processing ("F.J.S. method" [41]), or brain imaging (BRAINZOOM [42]).

Epidemics modeling: With respect to the epidemic data modeling, cascade and epidemic analysis have attracted huge interest [65], [99], also in biological/computer virus propagation, word-of-mouth marketing campaigns, etc.

In contrast to GB-R, none of the algorithms mentioned allows for information fusion using the domain knowledge, lacking the interpretability of the result (e.g. phase shift of the sinusoidal pattern, healing rate, etc).

In Table 2.4, a we compare GB-R and other competitors on the basis of selected properties.

2.2.3 Method

In this section, we provide the details of GB-R.

A table of symbols that is used throughout this section is shown in Table 2.5.

Our goal is to reconstruct the unknown true time series data $\mathbf{x}$ given the multiple reports $\mathbf{y}$ of aggregated sums and the mixing matrix $\mathbf{A}$ that tells us the coverage of each report. Because this is an underdetermined optimization problem ($N \ll T$), we need to incorporate additional constraints and/or regularization, based on and reflecting our domain knowledge about the nature

Table 2.4: GB-R captures all of the listed properties.

Properties	LSQ	F.J.S. method [41]	BRAINZOOM [42]	H-FUSE [77]	GB-R
Scalability	✓	✓	✓	✓	✔
Smooth reconstruction		✓	✓	✓	✔
Periodic reconstruction				✓	✔
Nonlinear modeling			✓		✔
Interpretability					✔
Infusion of domain knowledge					✔

Table 2.5: Symbols and definitions

Symbols	Definitions
x_t	true (unknown) time series data in fine scale that we want to reconstruct. $t = 1, \cdots, T$
y_i	a report count (an aggregated sum of $\mathbf{x}$). $i = 1, \cdots, N$
$\mathbf{A}$	mixing matrix used to generate report from $\mathbf{x}$: $\mathbf{y} = \mathbf{A}\mathbf{x}$. $\mathbf{A} \in \mathbb{R}^{N \times T}$
T	total timeticks of $\mathbf{x}$
N	total number of reports $\mathbf{y}$
β_0	average infection rate in SIS model
δ	recovery rate of the disease in SIS model
P_a	amplitude of the SIS model
P_s	phase shift of the SIS model
P_p	period of the SIS model
$\mathbf{\Theta}$	a set of parameters to optimize. $\{\mathbf{x}, \mathbf{z}, P_a, P_s, \beta_0, \delta\}$
$\mathcal{M}$	a set of SIS model parameters. $\{P_a, P_s, \beta_0, \delta\}$
P	the size of population

of the data.

In our work of GB-R, we utilize one of the prominent epidemic model called "*SIS model*" (Section 2.2.2) to fit the unknown time series $\mathbf{x}$. This expert-model explains how individuals in the population become infected, and how they recover and become susceptible to the disease again. The model uses parameters such as infection rate, peak phase, peak amplitude, etc that help interpret the evolving pattern of the time series data in terms of the model parameters, which capture how rapidly infection spreads, when it reaches its peak values, etc. Other models, like SIR (allowing immunity), SIRS (temporary immunity) etc, may reflect the nature of the disease even better, but the expert-model we propose to use here, gives very good accuracy and it is simpler to fit. Thus we decided to use the SIS model as our expert-model.

Details of our proposed algorithm GB-R are provided in the following section.

Proposed problem formulation

What is the simplest way to *reconstruct the true data from the aggregated reports*, i.e. to solve for infofusion problem? The most naive approach is to solve the following problem:

$$\min_{\mathbf{x}} \sum_{i=1}^{N} (y_i - \sum_{t=1}^{T} A_{it} x_t)^2 \tag{2.16}$$

where $\mathbf{x}$ is the true data that we want to reconstruct, $\mathbf{y}$ is a vector of N given reports, and $\mathbf{A}$ is the mixing matrix that generates the reports. However, as we mentioned in the previous section, this is under-determined problem ($N < T$), so we need to have proper regularizers that can reflect the nature of the true data.

Then how can we reconstruct the true data from the aggregated reports, in a way that it *obeys the domain knowledge (expert-model)*? The first attempt (the most naive approach) to include the expert-model is shown below:

$$\begin{aligned} \min_{\mathbf{\Theta}} \Big[& \sum_{i=1}^{N} (y_i - \sum_{t=1}^{T} A_{it} x_t)^2 \\ & + \mu \sum_{t=2}^{T} (x_t - x_{t-1} - \beta_{t-1} x_{t-1} (P - x_{t-1}) + \delta x_{t-1})^2 \Big] \end{aligned} \tag{2.17}$$

where $\mathbf{\Theta}$ is a set of parameters we optimize $\mathbf{\Theta} = \{\mathbf{x}, \mathbf{z}, P_a, P_s, \beta_0, \delta\}$ and β_t is defined as equation (2.14). The first term is for the reconstruction of the true data from the aggregated reports, and the second term is to fit the true data using our expert-model to incorporate our domain knowledge.

Then our next question is: *how can we optimize* the nonlinear equation given in equation (2.17)? In the second term (expert-model), we see that the key challenge is in the quadratic form of x_{t-1}, which makes the second term a fourth-order polynomial, which is very hard to optimize.

In GB-R, we reformulate our problem to turn it to a bilinear one, which can be handled using alternating optimization. This is the key step needed to derive a computationally tractable and effective algorithm. We use the variable splitting method, where we introduce an auxiliary

variable $\mathbf{z}$ that is a copy of $\mathbf{x}$, and replaced one of $\mathbf{x}$ with its copy $\mathbf{z}$ so that our problem becomes bilinear. Then we enforce that the copy $\mathbf{z}$ is same as $\mathbf{x}$. This reformulation of the problem reduces to the optimization problem below:

$$\begin{aligned}\min_{\mathbf{\Theta}} \Big[& \sum_{i=1}^{N}(y_i - \sum_{t=1}^{T} A_{it}x_t)^2 \\ & +\mu \sum_{t=2}^{T}(x_t - x_{t-1} - \beta_{t-1}x_{t-1}(P - z_{t-1}) + \delta x_{t-1})^2 \\ & +\xi \sum_{t=1}^{T}(x_t - z_t)^2 \Big]\end{aligned} \tag{2.18}$$

From the experimental results, we can simply set μ and ξ to 1.

Our optimization problem of equation (2.18) consists of three terms: T1: hard constraint (reconstruction), T2: soft constraint (expert-model), and T3: variable splitting. We will explain each term in more detail in the following.

T1: hard constraint (reconstruction)

$$\sum_{i=1}^{N}(y_i - \sum_{t=1}^{T} A_{it}x_t)^2 \tag{2.19}$$

The first term in our optimization problem is shown above. It is to ensure that when we aggregate our reconstruction $\hat{\mathbf{x}}$ based on the mixing matrix $\mathbf{A}$, we are able to generate as good matches to the observed reports $\mathbf{y}$ as possible. This is a hard constraint that must be met.

T2: soft constraint (expert-model)

$$\sum_{t=2}^{T}(x_t - x_{t-1} - \beta_{t-1}x_{t-1}(P - z_{t-1}) + \delta x_{t-1})^2 \tag{2.20}$$

The second term of the optimization problem shown above involves the domain knowledge modeling of $\mathbf{x}$. T2 nudges $\hat{\mathbf{x}}$ towards the solution that best obeys the domain knowledge. By this term, we enforce 1) smoothly but possibly peaky patterns, and 2) periodic patterns in $\hat{\mathbf{x}}$. T2 enables us to recover the underlying evolving characteristics of given data by learning basic parameters of β_0, δ, P_a, and P_s.

As discussed briefly earlier in the section, in our reformulation, we introduce an auxiliary variable $\mathbf{z}$ that is a copy of $\mathbf{x}$ and replace one of $\mathbf{x}$ with its copy $\mathbf{z}$ (*variable splitting method*). This reformulated optimization problem becomes bilinear and becomes easier to solve using alternating optimization method. This is to replace the quadratic nonlinear term in the cost function by one that is conditionally linear in one of the two variables given the other. This greatly facilitates optimization, as we will see.

T3: Variable splitting

$$\sum_{t=1}^{T}(x_t - z_t)^2 \tag{2.21}$$

With our formulation in GB-R, in T3 we enforce that $\mathbf{z}$, a copy of $\mathbf{x}$, matches $\mathbf{x}$.

Proposed optimization steps

In this section, we explain how we solve our optimization problem shown in equation (2.18). Given that our optimization problem is reformulated into bilinear form, we propose to use an alternating optimization method. The iterative update rule of GB-R algorithm is shown in Algorithm 1. We alternatively solve for each of the variables in the parameter set $\mathbf{\Theta} = \{\mathbf{x}, \mathbf{z}, P_a, P_s, \beta_0, \delta\}$ one at a time, while fixing the rest. The update equations are derived by setting the corresponding partial derivative of equation (2.18) to zero.

Notice that thanks to the variable splitting method (i.e. the introduction of an auxiliary variable $\mathbf{z}$), each alternating optimization step of our problem can be solved as a least-squares problem, using the Moore-Penrose pseudoinverse. We will explain the update equations in the following section.

How to optimize for $\mathbf{x}$: The solution is explained in Lemma 6 below.

Lemma 6 $\mathbf{x}$ *can be solved via Moore-Penrose pseudoinverse:*

$$\mathbf{x} = \begin{bmatrix} \mathbf{A} \\ - \\ \mathbf{M}_1 \\ - \\ \mathbf{I} \end{bmatrix}^{\dagger} \begin{bmatrix} \mathbf{y} \\ - \\ \mathbf{0} \\ - \\ \mathbf{z} \end{bmatrix} \tag{2.22}$$

where $\dagger$ *means Moore-Penrose pseudoinverse and* $\mathbf{M}_1$ *is a 2-band diagonal matrix of size* $(T - 1 \times T)$*:*

$$\begin{aligned} \mathbf{M}_1(t,t) &= -1 - \beta_t(P - z_t) + \delta \\ \mathbf{M}_1(t,t+1) &= 1 \end{aligned} \tag{2.23}$$

for $\forall t = 1 \cdots, T-1$. $\mathbf{I}$ *is an identity matrix of size* $(T \times T)$.

Proof 6 *To update* $\mathbf{x}$*, consider only the terms involving* $\mathbf{x}$*. Then we can reformulate the optimization problem as follows.*

$$\min_{\mathbf{x}} \left\| \begin{bmatrix} \mathbf{y} \\ - \\ \mathbf{0} \\ - \\ \mathbf{z} \end{bmatrix} - \begin{bmatrix} \mathbf{A} \\ - \\ \mathbf{M}_1 \\ - \\ \mathbf{I} \end{bmatrix} \mathbf{x} \right\|_2^2 . \tag{2.24}$$

Solving for above optimization problem enables us to solve $\mathbf{x}$ *via Moore-Penrose pseudoinverse as in equation (2.22).*

How to optimize for $\mathbf{z}$: $\mathbf{z}$ can be solved in the similar manner. The solution is shown in Lemma 7.

Lemma 7 *$\mathbf{z}$ can be solved via Moore-Penrose pseudoinverse:*

$$\mathbf{z} = \begin{bmatrix} \mathbf{I} \\ - \\ \mathbf{M}_2 \end{bmatrix}^{\dagger} \begin{bmatrix} \mathbf{x} \\ - \\ \mathbf{v}_1 \end{bmatrix} \tag{2.25}$$

Here, $\mathbf{v}_1$ is a vector of size $(T-1)$:

$$\mathbf{v}_1(t) = -x_{t+1} + x_t + \beta_t P x_t - \delta x_t, \tag{2.26}$$

$\forall t = 1, \cdots, T-1$. *$\mathbf{M}_2$ is a diagonal matrix of size $(T-1 \times T)$:*

$$\mathbf{M}_2(t,t) = \beta_t x_t, \tag{2.27}$$

$\forall t = 1, \cdots, T-1$, *and $\mathbf{I}$ is an identity matrix of size $(T \times T)$.*

Proof 7 *Considering only the terms involving $\mathbf{z}$, we can reformulate optimization problem regarding $\mathbf{z}$ as follows.*

$$\min_{\mathbf{z}} \left\| \begin{bmatrix} \mathbf{x} \\ - \\ \mathbf{v}_1 \end{bmatrix} - \begin{bmatrix} \mathbf{I} \\ - \\ \mathbf{M}_2 \end{bmatrix} \mathbf{z} \right\|_2^2. \tag{2.28}$$

Solving for above optimization problem enables us to solve $\mathbf{z}$ via Moore-Penrose pseudoinverse as in equation (2.25).

How to optimize for expert-model parameters $\mathcal{M}$: The rest of the variables $\mathcal{M} = \{P_a, P_s, \beta_0, \delta\}$ can be updated in a similar manner:
Update equation for P_a:

$$P_a = \sum_{t=2}^{T} \frac{x_t - x_{t-1} - \beta_0 x_{t-1}(P - z_{t-1}) + \delta x_{t-1}}{\beta_0 \cos(\frac{2\pi}{P_p}(t - 1 + P_s)) x_{t-1}(P - z_{t-1})} \tag{2.29}$$

Update equation for P_s: For better interpretability of P_s, we seek P_s to be an integer (week). Thus we perform exhaustive search of 52 possible positions (weeks throughout a year) and pick the one with the smallest cost.

$$\begin{aligned} P_s = \arg \min_{P'_s = 1,2,\cdots,52} & \sum_{t=2}^{T} x_t - x_{t-1} \\ & - \beta_0(1 + P_a cos(\frac{2\pi}{P_p}(t - 1 + P'_s)))x_{t-1}(P - z_{t-1}) + \delta x_{t-1} \end{aligned} \tag{2.30}$$

Update equation for β_0:

$$\beta_0 = \sum_{t=2}^{T} \frac{x_t - x_{t-1} - \beta_{t-1} x_{t-1}(P - z_{t-1})}{(1 + P_a cos(\frac{2\pi}{P_p}(t - 1 + P_s)))} \tag{2.31}$$

Update equation for δ:

$$\delta = \sum_{t=2}^{T} \frac{x_t - x_{t-1} - \beta_{t-1} x_{t-1}(P - z_{t-1})}{-x_{t-1}} \tag{2.32}$$

Algorithm 1: GB-R.

Data: mixing matrix $\mathbf{A}$, reports $\mathbf{y}$, the population P, the periodicity P_p
Result: A set of parameters $\mathbf{\Theta}$: unknown true data $\{\mathbf{x}\}$ of finer scale, and expert-model parameters $\{P_a, P_s, \beta_0, \delta\}$
for $iter = 1, \cdots$, *until convergence* **do**
    (block coordinate descent);
    Update each variable from $\mathbf{\Theta} = \{\mathbf{x}, \mathbf{z}, P_a, P_s, \beta_0, \delta\}$, while fixing the rest;

2.2.4 Experiments

In this section, we demonstrate experimental results of GB-R to address three questions: ***Q1***: how **effective** is the method; ***Q2***: how **scalable**; and ***Q3***: how easy it is to **interpret** its answer.

Data description: Tycho data [100, 126, 140] is an epidemic database that is integrated from more than $6,500$ weekly reports of the United States epidemic data spanning over 100 years from multiple sources in different granularities. There are 3 types of dataset: Level 1, 2, and 3 based on the level of standardization of the data. Level 1 dataset is in the most tailored and standardized, thus the most complete form, but with limitations in the types of diseases and the location of the data. There are 7 state-level diseases (hepatitis A, measles, mumps, pertussis, polio, rubella, and smallpox) and 1 city-level disease (diphtheria) for 50 states and 122 cities, respectively. The total time coverage of the counts varies for different types of diseases, some of which starts as early as 1916 and ends as recent as 2009. The counts are weekly-basis, and there exist missing values.

We applied GB-R on Tycho Level 1 dataset, and 7 state-level diseases.

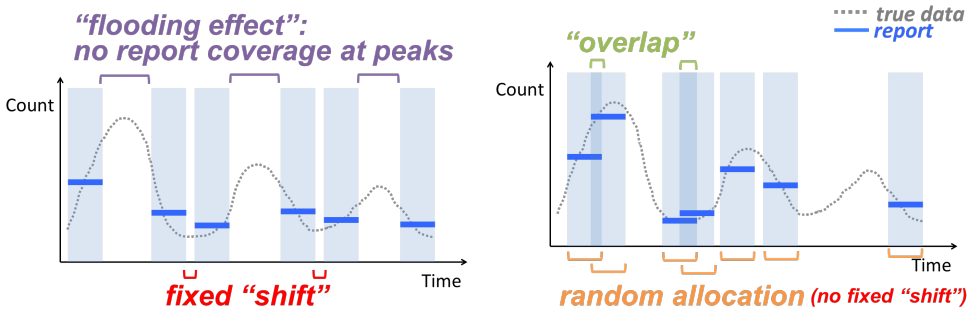
Experimental setup: We incorporate our knowledge on the yearly periodicity of the epidemic data, and set $P_p = 52$ (since the Tycho data is weekly basis, and there are 52 weeks in one year). We fixed the total population $P = 10$ million, following the population of the New York State. (the result is not sensitive to the population as long as it is set to sufficiently large value, so we fixed P for all of the states)

Scenarios: In Figure 2.11, different types of report generation processes are illustrated: 1) "flooding effect" scenario, and 2) random reporting.

1) "Flooding effect" scenario: As mentioned in section 2.2.1, "flooding effect" refers to the case where the reported counts during the peak season may not be accurate due to prioritization of clinical care.

Illustration on the "flooding effect" scenario is shown in Figure 2.11 (a). The report duration and the intervals between each report are fixed, except that during the peak seasons, there is no report covering the time period.

2) Random reporting: Random reporting is illustrated in Figure 2.11 (b). In this setting, report duration is still fixed, but the allocation of the report coverage is randomly assigned. This results in overlaps between reports, and long space between reports (long time range with no report coverage). This is a plausible scenario when there are multiple data sources, each of them having different "shifts".



(a) "flooding effect" (b) random reporting

Figure 2.11: Report generation scenarios.

Q1: Effectiveness

In this section, we demonstrate and prove the effectiveness of GB-R on the real-world data.

1) Case of "flooding effect": The effectiveness of GB-R on the real-world Tycho data under "flooding effect" scenario was demonstrated earlier in Figure 2.8: We chose Texas state because the time series pattern in Texas state was more spiky compared to that of New York state, thus more suitable to demonstrate the "flooding effect" scenario. We fixed the report duration to be 5-week, and the "shift" to be 2-week. We left out the peak seasons with no report.

In Figure 2.8 (a) and (b), we have demonstrated that GB-R successfully reconstructs the *spiky peak* patterns by utilizing its domain expert knowledge ensuring the data follows the natural cascade pattern, resulting in reduction of the reconstruction error compared to the competitors.

2) Case of random reporting: Random reporting scenario is the most likely scenario in the real world, so we focus more on the experiments under this setting.

Generalization over diseases and states: In Figure 2.12, bar plots of MSE of the reconstructions by GB-R and the competitor algorithms on different types of diseases (measles, pertussis, hepatitis A, polio, rubella, mumps, and smallpox, all 7 state-level diseases available in level 1 Tycho dataset) in three largest states (California (CA), New York (NY), and Texas (TX)) are shown.

We used first $T = 1,000$ time points of the weekly counts as our unknown true dataset and generated reports from it using random reporting process as illustrated in Figure 2.11 (b). Each disease starts in different time of the year. We fixed the report duration to be 20-week and the total number of reports as $N = 100$.

GB-R wins over competitors in reconstructing all of the diseases except for Hepatitis A, and Smallpox. Even for the losing cases, the amount of loss is small.

Intuition behind GB-R: In this section, we share our intuition on why GB-R outperforms the competitors. In Figure 2.13, a reconstruction of weekly measles count of New York state by H-FUSE-S, and H-FUSE-P is shown in top and bottom, respectively.

In the top plot, H-FUSE-S fails to reconstruct the first and the eighth peaks (no report coverage). For the time region where there is too scarce reports as reference to make a guess on the true data, H-FUSE-S interpolates, or *"smooths out"* the counts of the adjacent time points with

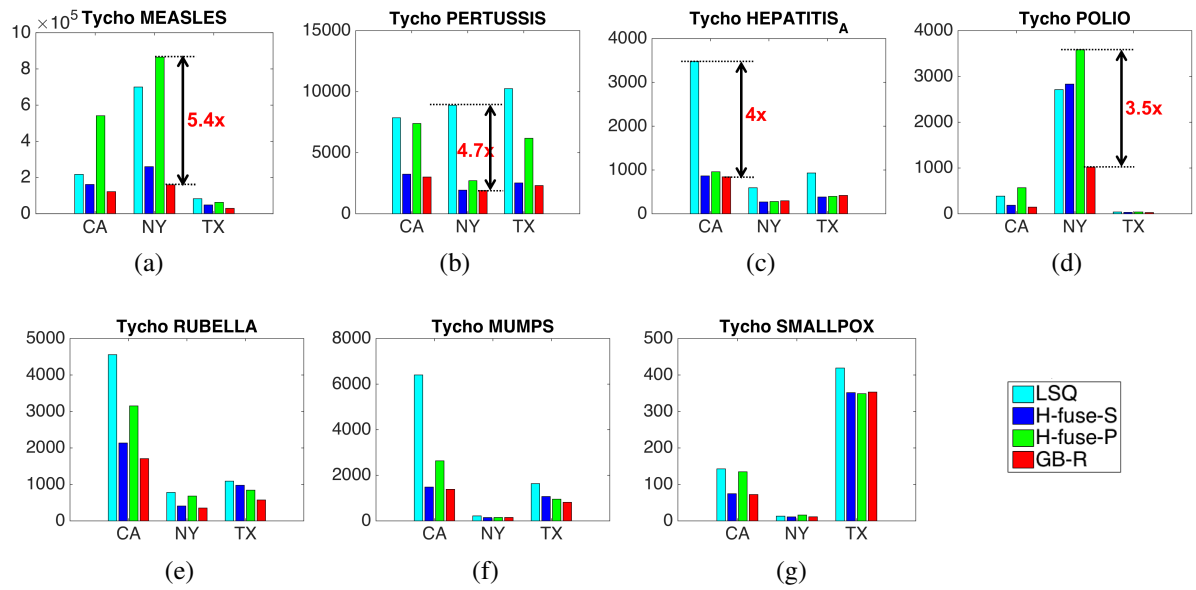


Figure 2.12: **GB-R almost always wins**, often by large margin. Bar plots of MSE of reconstruction of (a) measles, (b) pertussis, (c) hepatitis A, (d) polio, (e) rubella, (f) mumps, and (g) smallpox in three largest states California (CA), New York (NY), and Texas (TX) are shown.

report coverage.

In the bottom plot, H-FUSE-P *overestimates* the peak values for years 1940-1946 due to the strong assumption of periodic pattern that expects repetitive heights for the recurring peaks.

On the other hand, in both plots, GB-R demonstrates good reconstructions; no smoothing out or overestimation problem as H-FUSE-S or H-FUSE-P, thanks to the expert-model, that forces the reconstruction to follow more natural evolving patterns with spiky peaks, spread rate, and recovery rate.

Q2: Scalability

In Figure 2.14, the wall clock time (s) required for GB-R to run on different length of timeticks (T) is plotted along with a line for better understanding. It shows that GB-R scales linearly with respect to the length of the timeticks T in the given data.

Q3: Interpretability

Some of the expert-model parameters values (P_s, δ) learned by GB-R on the weekly counts of measles, polio, and pertussis in California state are shown in Table 2.6.

Phase shift: P_s represents the *peak seasons* of the data. In Table 2.6, GB-R learned different peak seasons for each disease. Our interpretation of the peak seasons of measles, polio, and pertussis matches with the epidemiology:

1) *"Measles increases during the late winter and early spring"* [3]

[3] `http://www.who.int/ith/diseases/measles/en/`

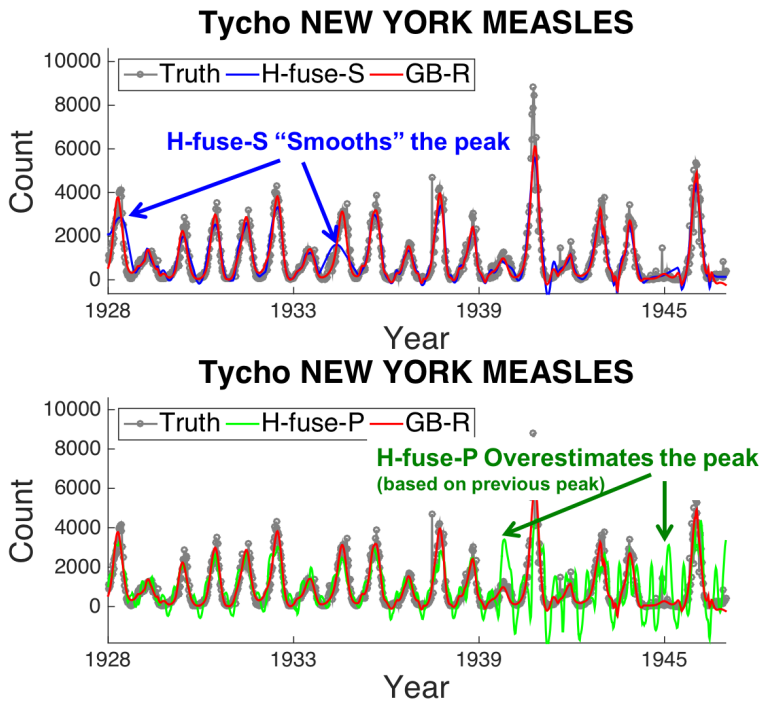


Figure 2.13: **Superiority of GB-R**. Competitors enforcing smoothness or periodicity may miss / over represent the spikes

Table 2.6: expert-model parameters

	Measles	Polio	Pertussis
P_s	52 weeks (peak: Jan)	27 weeks (peak: Jul)	47 weeks (peak: Nov)
δ	0.31	0.31	0.92

2) *"Polio typically peaks in the summer months"* [4]
3) *"Peak season for Pertussis is late summer and early fall."* [5]

Recovery rate: δ corresponds to how sharp is the drop in the spike. Figure 2.15 (top) shows the synthetically generated data with increasing values of δ: larger the δ, sharper the drop.

Figure 2.15 bottom left shows real Tycho data (*gray*) and the fitting by GB-R (*red*). Figure 2.15 bottom right shows the fittings of measles, polio, and pertussis on top of each other for the comparison of the sharpness of the drop. GB-R successfully learned small δ for smooth drop, and large δ for sharp drop that matches with the actual pattern of the disease.

[4] https://www.cdc.gov/vaccines/pubs/pinkbook/polio.html
[5] https://consumer.healthday.com/infectious-disease-information-21/misc-infections-news-411/spread-of-whooping-cough-raises-concern-641758.html

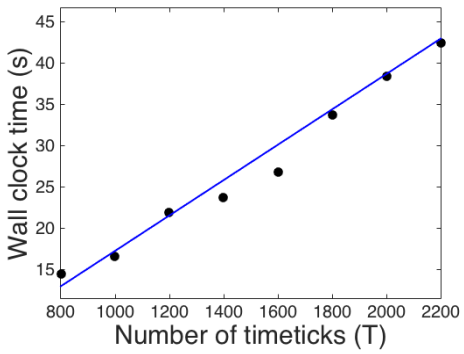


Figure 2.14: **GB-R scales linearly** with respect to the signal length.

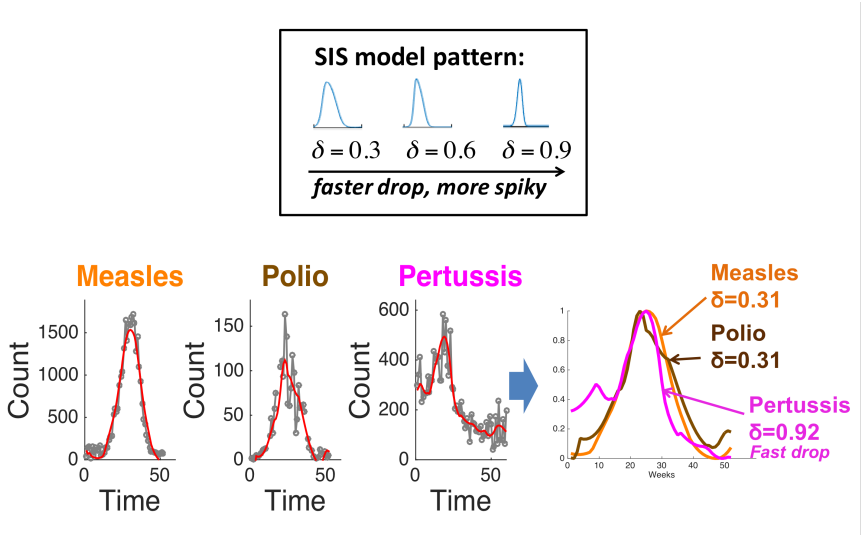


Figure 2.15: **GB-R is interpretable**. GB-R learns the healing rate parameter. Pertussis with larger parameter value heals faster.

2.2.5 Conclusion

In this Section, we proposed GB-R, an algorithm that reconstructs time series counts from aggregated reports by careful infusion of domain knowledge.

1. **Effectiveness**: GB-R outperforms top competitors by **3x-25x** in reconstructing real-world data thanks to its principled infusion of domain knowledge.
2. **Scalability**: GB-R scales linearly on the duration of the output sequence.
3. **Interpretability**: GB-R provides understanding of the time sequence, by estimating the gray-box-model parameters, like the average infection rate (β_0), the peak timing (P_s), the peak size (P_a), etc.

Reproducibility: We provide our source code for at: `http://www.cs.cmu.edu/~hyunahs/code/GB_R.zip`. All our data are publicly available at `https://www.tycho.pitt.edu/`

2.3 ARES: extracting notable patterns [136]

In this section, we discuss ARES, (*Automatic REStoration*), a two-phase signal reconstruction method that can also derive patterns from the target sequences. In the first phase, ARES estimates the sequences using domain knowledge such as smoothness. And then, in the second phase, ARES further refines the estimated sequences by utilizing *annihilating filter technique*, where the idea is to learn a linear shift-invariant operator whose response to the desired sequence is (approximately) zero – yielding a set of null-space equations that the desired signal should satisfy, without the need for the accompanying data. In the process, the annihilating filters help us derive notable patterns in the target sequence.

2.3.1 Introduction

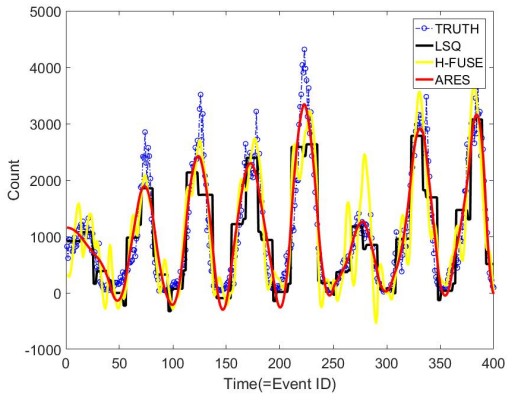


Figure 2.16: **ARES *reconstructs more accurately***: An example of Tycho [100] measles counts

There are numerous historical data sets available from various worldwide groups, such as the Institute for Quantitative Social Science at Harvard, Great Britain Historical GIS at Portsmouth, the International Institute of Social History in Amsterdam, and World-Historical Database at the University of Pittsburgh. In health sciences, the Vaccine Modeling Initiative at the University of Pittsburgh aims to gather and analyze the information from *thousands* of reports on United States epidemiological data, spanning over 100 years. Making sense of such data is particularly important for interdisciplinary research, where a comprehensive picture of the subject requires consolidation/fusion of large amounts of historical data from a variety of disciplines.

Historical data commonly includes reports on time series aggregated at different levels (e.g., weekly and monthly sums of counts of people infected with flu). It is common to have overlapping and even conflicting reports. For example, we may have multiple monthly and annual reports from different authorities about cases of measles in Los Angeles. Meanwhile, the sum of the monthly reports for a particular year might not be the same as the count for that year from an annual report. Besides, some data may be missing (e.g., the reports do not cover the period of 1940-1944 because of World War II). Proper reconstruction of historical events requires reasonably accurate disaggregation of such data.

In this Section, we propose a novel and efficient approach, called ARES (*Automatic REStoration*), which automatically reconstructs historical data in two phases: (1) first, it estimates the sequence

of historical counts utilizing domain knowledge, such as smoothness and periodicity of historical events; (2) then, it uses the estimated sequence to derive notable patterns in the target sequence to refine the reconstructed time series. ARES applies an *annihilating filter technique* that is reminiscent of ideas encountered in spectral estimation and compressed sensing [54, 118]. The idea of annihilating filtering is to learn a linear shift-invariant operator whose response to the desired sequence is (approximately) zero – yielding a set of null-space equations that the desired signal should satisfy, without the need for the accompanying data. ARES further improves the reconstruction accuracy by applying the annihilating filtering at the second phase iteratively.

ARES recovers historical data from aggregated reports with high accuracy and considerably outperforms the state-of-the-art methods. We evaluate ARES on the real epidemiological data from the Tycho project [100]. Figure 2.16 illustrates how ARES compares to the common least squares (LSQ) reconstruction and more advanced H-FUSE method [77]. Figure 2.16(a) plots the New York measles data (in blue, 'Truth'), the least squares reconstruction (in black, 'LSQ'), the reconstruction of the H-FUSE (in yellow, 'H-FUSE'), and the reconstruction of the ARES (in red, 'ARES'). We observe that ARES reconstruction is visibly better than both LSQ and H-FUSE. Moreover, ARES can automatically tune itself for the best performance under specific application constraints (such as report durations and overlaps), as detailed in Section 2.3.3.

The rest of the section is organized as follows: we go through background and related work in section 2.3.2, and explain our proposed method in section 2.3.3. We demonstrate our experimental result in section 2.3.4, elaborate on complexity of our method in section 2.3.5, and conclude our work ARES in section 2.3.6.

2.3.2 Background and Related Work

Information Disaggregation

(We saw most of these materials in section 2.1.2, but we repeat them here for the readers' convenience.)

The problem of historical information disaggregation, which we consider in this section, is a special case of the information fusion dealing with reconstructing objects from multiple observations. In the process of historical information disaggregation, researchers need to recover numbers of historical events from multiple aggregated reports, while handling redundant, and possibly, inconsistent data. Figure 2.17 shows an example of a database with two historical reports on total cases of measles in NYC. Each report covers time interval of 4 weeks with one week overlap. The task of historical information disaggregation would be to reconstruct the most likely weekly numbers of measles cases. The granularity of the reports may differ (e.g,, we may need to reconstruct daily numbers from weekly aggregates). This task can be formalized as finding an (approximate) solution to a *characteristic linear system* [77].

Figure 2.17 shows an example of the *characteristic linear system* for these two NYC measles reports. Each report is represented as a binary row vector in an observation matrix with "one" corresponding to the time units covered by the report. The characteristic system is commonly under-determined (the number of reports is smaller than the number of time-ticks). A disaggregated historical sequence forms an approximate solution to the characteristic system.

In order to reconstruct a historical sequence, we need to find a reasonably accurate solution of

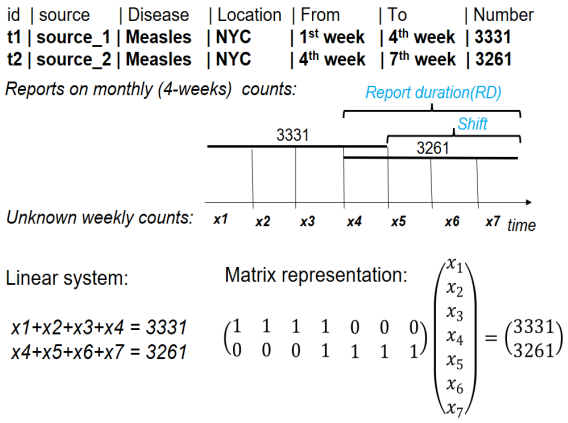


Figure 2.17: **Illustration on the nature of the data and the setting of our problem**.

the characteristic linear system. No matter what kind of reconstruction method we are using, the reconstructed sequence $\vec{x}$ should satisfy the historical reports. More formally, it should minimize the following deviation:

$$\mathcal{F}(\vec{x}) = \sum_{n=1}^{N}(v_n - \sum_{t=1}^{T}\mathbf{O}_{nt}x_t)^2 \tag{2.33}$$

and, in matrix form:

$$\mathcal{F}(\vec{x}) = ||\vec{v} - \mathbf{O}\vec{x}||_2^2 \tag{2.34}$$

where T is the total number of time-ticks in $\vec{x}$, N is the total number of reports, $\vec{v}$ is the value of report, $\mathbf{O}$ is the observation matrix. Ideally, the deviation from the reports should be zero, unless the reports are conflicting.

The 'LSQ' tries to minimize $\mathcal{F}()$, which aims to find the minimum-norm solution ($\min\ ||\vec{x}||_2^2$) that satisfies $\mathcal{F}() = 0$. The problem is usually under-determined and 'LSQ' can find such solution using the *Moore-Penrose pseudo-inverse* [87].

The more advanced H-FUSE method [77] (section 2.1) tries to infuse domain knowledge to improve the reconstruction accuracy. For example, H-FUSE penalizes large differences between the $\vec{x}$ values of adjacent time ticks if we expect the solution sequence $\vec{x}$ to be smooth. Furthermore, if we know that the sequence $\vec{x}$ is periodic, H-FUSE also imposes periodicity constraints. H-Fuse can further improve its performance by combining both smoothness and periodicity constraints.

Smoothness and periodicity constraints are examples of domain knowledge about some simple sequential patterns in the target series. The idea of recognizing and utilizing sequential patterns for sequence reconstruction is actually behind the concept of *Annihilating Filter*, which forms a basis of our proposed approach, and we will explain it in Section 2.3.3.

Related Work

The problem of historical information disaggregation can be stated in a wider context of fusing and making sense of data obtained from a variety of sources, with gaps and overlaps in time and space, and with uncertainty in trust of sources. As an example, considering project Tycho

[100, 126] on large-scale integration of epidemiological data sources spanning more than 100 years. We use Tycho data to evaluate the performance of ARES in Section 2.3.4.

The concept of information fusion has been used in different areas including multi-sensor data fusion [48], human-centered information fusion [49], and information fusion for data integration [14]. Multi-sensor data fusion utilizes multiple sensors to improve signal-to-noise ratio, as well as robustness and reliability of the sensor network in the presence of sensor failures.

Human-centered information fusion enhances fusion techniques by including human observations, as well as web-based information about interactions between humans (e.g. social networks). As an example, in previous work the researchers developed automatic information fusion methods that exploit the collective intelligence of mobile robots and human operators/observers, and efficiently crowd-source the victim detection task [141].

The emergence of the Internet facilitates the access to different types of data sources of varying reliability, which may also be mutually inconsistent due to conflicting data [38]. The challenge of resolving data conflicts and data inconsistencies has been addressed in the task of data integration [14, 139].

Information disaggregation is a key component of historical information fusion. In Section 2.3.2.1 we considered least squares method (LSQ) and more advanced H-FUSE method to recover historical time series from an under-determined linear system. Temporal disaggregation methods have also been studied in time series analysis of mostly economic data (see [20, 107]). Given a low frequency time series (e.g. annual sales, weekly stock market index, etc.), the goal of temporal disaggregation is to produce a high-resolution series (e.g. quarterly sales, daily stock market index, etc.) satisfying temporal aggregation constraints. Temporal disaggregation methods have been used for the cases of non-overlapping aggregated reports and cannot be directly applied to the task of historical information fusion.

Another method, reported in [41], performs information reconstruction/disaggregation from consecutive and non-overlapping summaries (histograms). It is not obvious how this method, called 'HistRecovery' below, could handle overlaps and missing values.

The challenge of information disaggregation has also been explored in domains of image and signal processing, where researchers have been dealing with related ill-posed problems. For example edge detection, surface reconstruction [143], vehicle detection [106], and super-resolution image reconstruction [91] are ill-posed. The annihilating filter technique, used by our ARES approach, is a signal processing method, which we apply to the task of historical information disaggregation.

An *annihilating filter* is a linear shift-invariant operator of finite support, which completely suppresses a certain signal. That is, a filter with impulse response $h(t)$, such that $(x * h)(t) := \sum_\tau x_{t-\tau} h_\tau = 0, \forall t$. The concept has been introduced in spectral analysis, particularly line spectra estimation [54, 118]. It has been subsequently used in other contexts, notably compressed sensing by Vetterli *et al.* [128]. The basic idea is as follows. To recover the fine-scale data, we need additional information – ideally, more measurements. Instead, if we know a linear shift-invariant operator that annihilates the signal of interest, then we can use it to build additional equations that the signal satisfies, without the need for additional data – simply because that data would be zero anyway. In practice, the filter will only approximately annihilate the signal of interest when the latter is not a sum of a few sinusoids, so these virtual equations are only approximate. However, we can use a least squares approach to account for errors in both actual

and 'virtual' measurements. This is the starting point of our approach. Our work is the first work that applies annihilating filters to information disaggregation in the historical data fusion domain.

Table 2.7 contrasts our ARES method against the related state-of-the-art competitors. We present our method in the next section.

Feature / *Algorithm*	Scalability	Overlapping reports	Missing values	Conflicting reports	Derived knowledge	Automatic parameter selection
LSQ	✓	✓				
HistRecovery	✓	?	?			
TDisaggregation						
H-FUSE	✓	✓	✓	✓		
ARES	✔	✔	✔	✔	✔	✔

Table 2.7: **ARES matches all specs**, while competitors miss one or more of the features.

2.3.3 Method

In this section, we explain our ARES method in detail. As mentioned before, smoothness and periodicity constraints are examples of domain knowledge about simple sequential patterns in a time series. ARES applies *Annihilating Filter (AF)* to derive more sophisticated sequential patterns in a historical time series to reconstruct/disaggregate it. AF allows us to discover more notable sequential patterns flexibly.

In the following sub-section, we introduce the calculation of AF and the patterns discovered by AF.

Informal Explanation of Annihilating Filter (AF)

Figure 2.18 shows examples of different AFs and corresponding sequential patterns. The AF length L in the first column is the interval length, which the sequential pattern should work on. Different Ls will result in different types of patterns that the target sequence should follow. Each histogram in the second column corresponds to an AF with length L, where each bar shows an AF coefficient c_l such that $\sum_{l=1}^{L}(c_l \vec{x}_l) = 0$ for any successive L time-ticks of a target sequence. The third column shows this equation for coefficients of each of the considered AFs. For example, for the $L = 2$ with coefficients of -0.6 and 0.6, the equation is $x_1 - x_2 = 0$ for any two successive

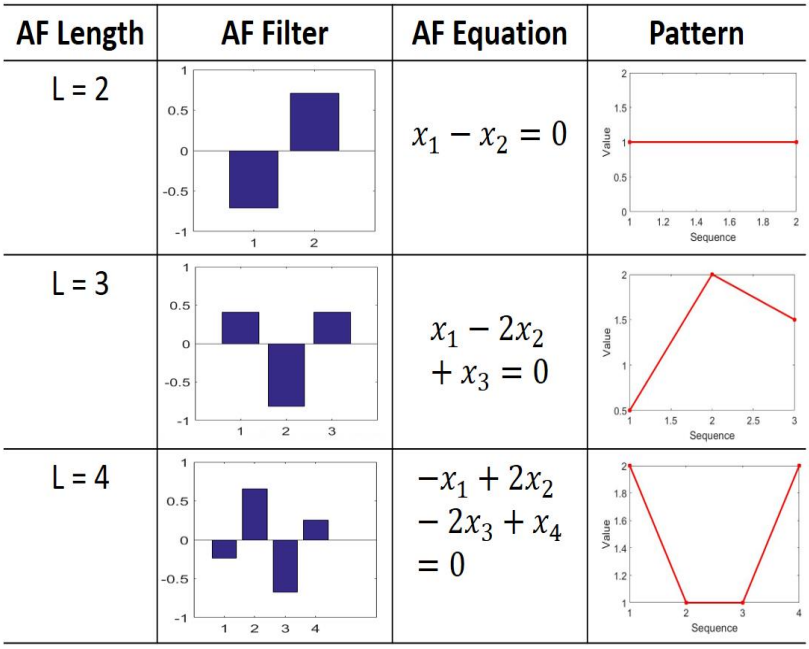


Figure 2.18: **Illustration of patterns for different AF lengths (Ls).** Different Ls result in different types of patterns; the second column shows the bar chart for AF coefficients; the third column shows the AF equation that the target sequence should satisfy; the last column shows an example of a sequence that satisfies this equation.

time-ticks (i.e., the pattern is a flat line and it corresponds to an extreme smoothness constraint). For the $L = 3$ with coefficients of 0.4, -0.8 and 0.4 the equation is $x_1 - 2x_2 + x_3 = 0$ for any three successive time-ticks.

Figure 2.19 shows examples of the AFs reflecting sequential patterns of increasing lengths for three epidemiological time series from the Tycho repository. We observe that all sequences have very similar AFs for $L = 2$ and $L = 3$. For $L = 2$, it almost matches the smoothness constraint, which provides another evidence that smoothness is a reasonable assumption in the process of a sequence reconstruction. For longer filters, the patterns start to deviate, reflecting more specific dynamics of a target sequence.

Generating Annihilating Filter

The algorithm to generate AF (Algorithm 2) is based on finding the eigenvector corresponding to the smallest eigenvalue in the Singular Value Decomposition (SVD) of a square matrix. The square matrix is constructed from the input sequence and the dimension of the matrix is L, as we explain below.

For a given AF length L, the AF intends to find a stable pattern within any L contiguous timeticks. Therefore, the output of the filter can be written in a column vector form as $\overrightarrow{y} = X_{mat}\overrightarrow{h}$, where $\overrightarrow{h}$ is an $L \times 1$ column vector holding the impulse response of the filter, and X_{mat} is $(T - L + 1) \times L$ matrix constructed by all L contiguous timeticks from the input sequence (see

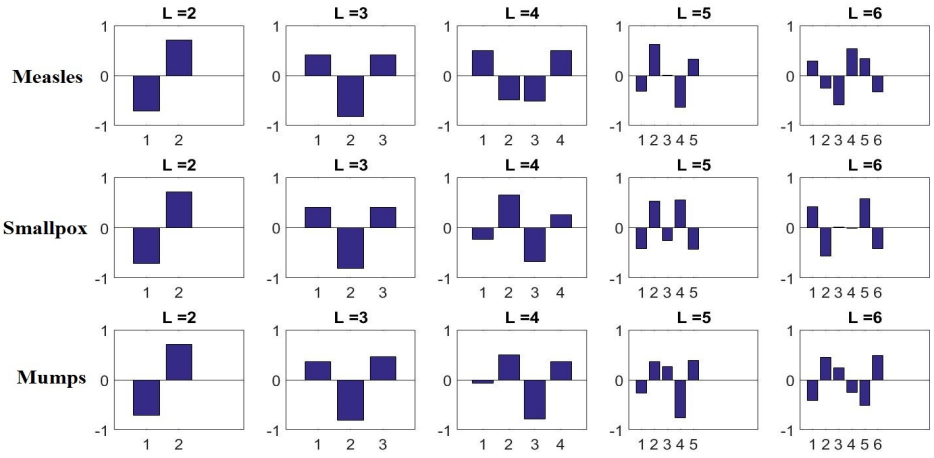
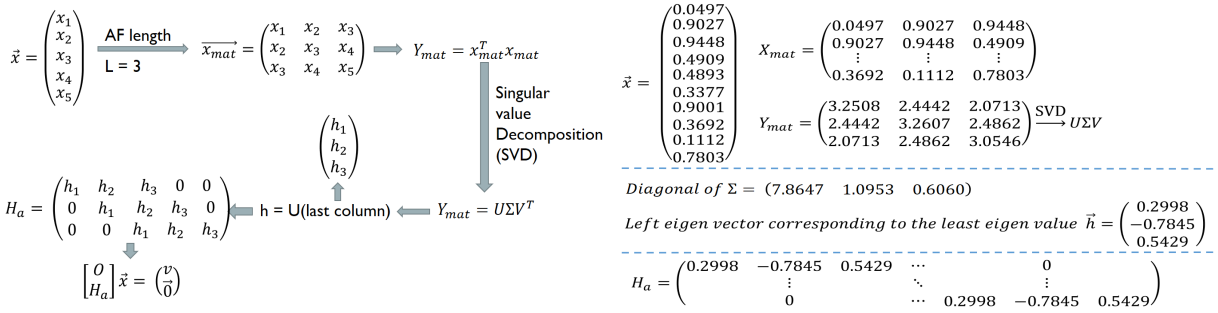


Figure 2.19: **AFs for three epidemiological time series with increasing lengths (from 2 to 6).** All series have very similar AFs for $L = 2$ and $L = 3$, while for longer filters the patterns reflect more specific sequence dynamics.



(a) Major steps of the *AFilter* procedure

(b) An example of calculating AF and corresponding annihilating matrix H_a for a short input sequence

Figure 2.20: ***Illustration of* ARES *AF generation algorithm***

details in Figure 2.20(a)). The norm squared (energy) of the output vector is then $||X_{mat}\overrightarrow{h}||^2 = \overrightarrow{h}^T X_{mat}^T X_{mat}\overrightarrow{h}$. Then the $\overrightarrow{h}$ that minimizes this expression is the eigenvector corresponding to the smallest eigenvalue of $X_{mat}^T X_{mat}$. This can be easily seen because any $\overrightarrow{h}$ can be written as $U\lambda$, where λ is a vector of coefficients, whose norm must be 1 and U contains the eigenvectors of $X_{mat}^T X_{mat}$, which are orthogonal [118]. After $\overrightarrow{h}$ has been determined, the output of the AF can also be written as $\overrightarrow{y} = H_a\overrightarrow{x}$, where H_a is a Toeplitz matrix constructed from $\overrightarrow{h}$. The Toeplitz matrix is a matrix in which each descending diagonal from left to right is constant. Thus H_a is an *annihilating matrix* that we use to define the annihilating constraint below.

The generation procedure of *AF* is illustrated in Figure 2.20. Figure 2.20(a) shows a general flow of the algorithm starting from an input sequence $\vec{x}$ and resulting in construction of an annihilating matrix H_a with $L = 3$. Then the H_a matrix can be combined with the observation matrix O from the original characteristic linear system to determine the system more precisely (bot-

tom part of the Figure 2.20(a)). Figure 2.20(b) shows major steps in generating the annihilating matrix H_a for a simple example of a short input sequence.

Next we will explain how the AF can be used for data reconstruction.

Algorithm 2: Generate Annihilating filter

procedure AFILTER($\vec{x}$)

Step 1: Select an appropriate *AF length (L)*.

Step 2: Build a $(T - L + 1) \times L$ matrix X_{mat} from an input sequence $\vec{x}$ and calculate $Y_{mat} = X_{mat}^T X_{mat}$

Step 3: Perform SVD for Y_{mat} and extract the last column $\overrightarrow{h}$ from the left singular vector matrix.

Step 4: Given $\vec{h}$, construct a Toeplitz $(T - L + 1)$ matrix $\mathbf{H}_a$ starting from the diagonal.

Step 5: Output the annihilating matrix $\mathbf{H}_a$

Data Reconstruction Using AF

Since patterns with different AF length are mutually independent, we can combine appropriate patterns reflected in AFs as additional constraints to improve reconstruction/disaggregation accuracy, as illustrated in Figure 2.21. In this figure, we combine AFs of increasing length in different ranges ($L \in [2, 30]$, $L \in [2, 50]$, $L \in [2, 60]$, $L \in [2, 70]$ and $L \in [2, 90]$) to reconstruct the Tycho New York measles time series. We use a single aggregated report covering the whole sequence (i.e., it reports the total sum of all ticks in the time series). Note, that we intentionally use this case where our characteristic linear system includes only one equation and the reconstruction accuracy depends entirely on the patterns discovered by AFs. This way we can illustrate how the AF patterns alone help to recover the sequence. We observe that as we use more patterns, the reconstruction accuracy improves steadily. Note, that we generate AF coefficients from the target sequence and this is why the reconstruction dynamics is so good. However, our major challenge in ARES is to *discover* AF patterns under an assumption that the target sequence is not available. This is why ARES will apply a two-phase reconstruction strategy, which we propose to generate the AF patterns based on the reconstructed sequence under smoothness and periodicity constrains, and which we explain next.

ARES Model

ARES recovers the original sequence $\vec{x}$ in two phases. In order to generate annihilating filters, ARES requires some knowledge of the original sequence. Since in practice only aggregated reports are available, we propose to resolve this problem by building the filters with the $\vec{x}_{sp}$ reconstructed at the first phase of ARES using smoothness and periodicity constraints The expectation is that smoothness and periodicity constraints combined with a characteristic linear system will

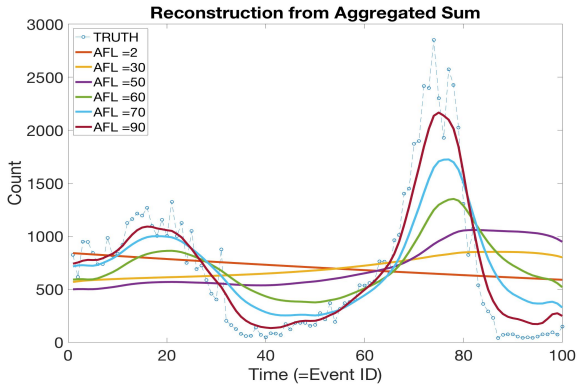


Figure 2.21: **Time series reconstruction using Annihilating Filters.** Reconstructing Tycho New York measles time series from a single report (total sum). Characteristic linear system includes only one equation and the reconstruction accuracy depends entirely on the patterns discovered by AFs. As we combine more patterns, the reconstruction accuracy improves steadily.

refine the information about the target series to an extent that it can be re-used to build annihilating filter representing reasonably accurate sequential patterns in the target series. Such filters can be used to improve reconstruction accuracy.

In the first phase of ARES we use the combined smoothness and periodicity constraint:

$$\mathcal{C}_{sp}(\vec{x}) = \frac{1}{2}\mathcal{C}_s(\vec{x}) + \frac{1}{2}\mathcal{C}_p(\vec{x}),$$

where $\mathcal{C}_s(\vec{x})$ is the smoothness constraint to penalize big jumps between successive timeticks, and $\mathcal{C}_p(\vec{x})$ is the periodicity constraint which is to make the event at timetick t to be close to the one at $t + PD$.

More formally, in the first phase ARES minimizes a total penalty function $\mathcal{L}(\vec{x})$ solving the following optimization problem:

$$\min_{\vec{x}} \mathcal{L}(\vec{x}) = \min_{\vec{x}} \ (\mathcal{F}(\vec{x}) + \mathcal{C}_{sp}(\vec{x})), \tag{2.35}$$

where $\mathcal{F}(\vec{x})$ is defined as in Eq. 2.33.

Next, we introduce the annihilating constraints corresponding to sequential patterns generated by AFs in the second phase. ARES applies such constraints to the characteristic linear system, thus determining the system more precisely. Formally, at the second phase ARES solves the following optimization problem

$$\min_{\vec{x}} \mathcal{L}(\vec{x}) = \min_{\vec{x}} \ (\mathcal{F}(\vec{x}) + \mathcal{C}_a(\vec{x})), \tag{2.36}$$

where $\mathcal{C}_a(\vec{x})$ corresponds to the annihilating constraint that we define below:

- Annihilating constraint $\mathcal{C}_a$: If there is an Annihilating Filter (AF) of length L, then the successive L time-ticks should follow the pattern defined by the AF (i.e., successive L time-ticks multiplied by corresponding AF coefficients should sum up to zero). Formally:

$$\mathcal{C}_a(\vec{x}) = \sum_{t=1}^{T-L+1} (\sum_{l=1}^{L} (c_l x_{l+t}))^2 = ||\mathbf{H}_a \vec{x}||_2^2 \tag{2.37}$$

where $\mathbf{H}_a$ is an $\mathbb{R}^{(T-L+1)\times T}$ *annihilating matrix* whose t^{th} row has c_l in the L successive columns starting from the t^{th} column, respectively.

To sum up, ARES first applies smoothness and periodicity constraints to the aggregated reports to obtain a reconstructed sequence $\vec{x}_{sp}$, then builds a corresponding annihilating constraint $\mathcal{C}_a(\vec{x})$ based on the information in $\vec{x}_{sp}$. Using Eq. 2.36, ARES refines the reconstructed sequence $\vec{x}_{sp}$ to a target sequence $\vec{x}$.

Automatic Parameter Selection

As we observed in Figure 2.21, the AF length L impacts reconstruction accuracy. Moreover, from our experiments we observed that L has different impact on different report configurations (i.e., different report durations and overlaps). Next we propose an automatic method using the *Minimum Description Length (MDL)* principle [103] to select the best AF length L, which provides the best reconstruction for each configuration. We select the length L using the following Selection Criteria (SC):

$$Selection\ criteria : SC(L) = \|\vec{v} - O\widehat{\vec{x}}\|_2^2 + \|H_a\widehat{\vec{x}}\|_2^2 + L \tag{2.38}$$

In Eq.2.38 the first term accounts for data fidelity for the measurements in y; the second term accounts for data fidelity for the 'virtual' measurements, which are approximately zero; and the third term penalizes model complexity – the length of the AF that is used. This is precisely in the spirit of MDL, whose philosophy is to pick the most parsimonious model that describes the data well-enough. In other words, to avoid over-fitting, we penalize L, because if L is too large then we can zero out any signal, and this does not yield useful information.

Iterative ARES

We can extend ARES based on the Alternating Least Squares Optimization (ALS). For two variables, x and h, we can fix one and update the other to minimize the common cost function. We will call this ITERATIVE ARES.

2.3.4 Experiments

In this subsection, we share our experimental results. We will evaluate ARES using real epidemiological time series from project Tycho [100] that collected epidemiological reports spanning more than 100 years.

Experimental setup

In order to generalize our results and observations we apply ARES to six diseases: California smallpox (from Week 501 to Week 900), California hepatitis (from Week 2653 to Week 3051), California mumps (from Week 2756 to Week 3155), California polio (from Week 1659 to Week 2058), California rubella (from Week 2805 to Week 3204), and California pertussis (from Week 1649 to Week 2048). Figure 2.22 plots each of the selected time series. We observe that each

disease has notably different dynamics and different degree of periodicity, which provides a rich test set for ARES evaluation.

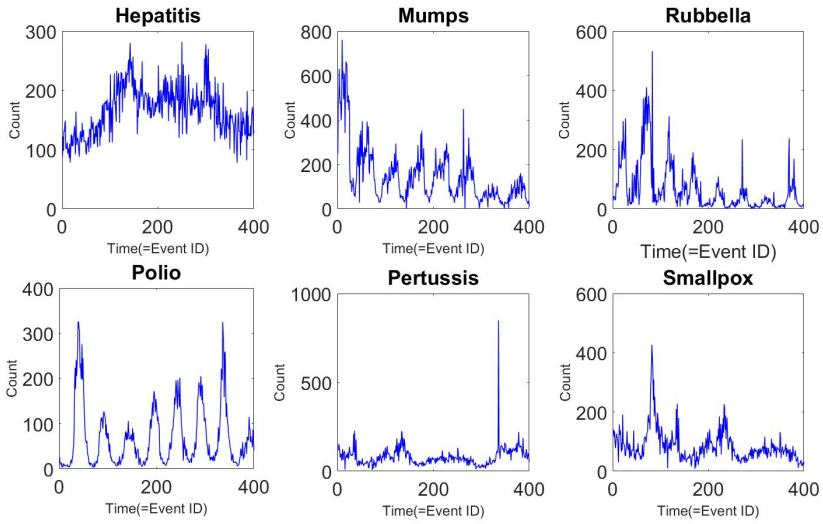


Figure 2.22: **Disease time series used for the experiments.** While most of the diseases have annual periodicity, they have notably different dynamics.

We use Pittsburgh Supercomputing Center Bridges HPC resource[6] to run most of our experiments. Bridges consists of 846 compute nodes (27,408 cores total), each with hardware-enabled coherent shared memory: 4 nodes with 12 TB, 42 nodes with 3 TB, and 800 nodes with 128 GB. A node with 12TB of RAM shares the memory between up to 320 cores while 128GB nodes have 28 cores each. We obtained Bridges allocation through XSEDE [7] at no charge for open data-intensive research.

Effectiveness

To obtain a more general picture of the ARES behavior, we performed experiments under various report configurations. We changed both RD (ranging from 2 to 60) and shift (ranging from 1 to 26). We select those ranges with the awareness that epidemiological data have a prominent yearly periodicity of 52 weeks. If the report shift is exactly 26, then we critically sample this periodicity according to the Shannon-Nyquist sampling theorem [86]: that is, we gather two samples per yearly cycle. If we take less than 2 samples per yearly period, then, because of heavy aliasing, we should not expect to reconstruct the fundamental frequency of the time series, let alone accurately reconstruct the time series. Otherwise, the reconstruction accuracy is reasonably high for all report configurations (RD and Shift).

In Figure 2.23, we show the bar plot of average RMSEs of the reconstructions by different methods. We observe that iterative ARES performs the best, followed by ARES, H-FUSE, and LSQ. Iterative ARES improves reconstruction by $9.6\%, 34\%$ and 60% over ARES, H-FUSE and LSQ, respectively.

[6]https://www.psc.edu/index.php/bridges
[7]https://www.xsede.org/

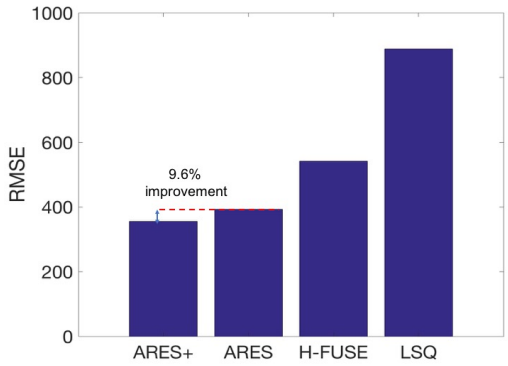


Figure 2.23: **ARES effectively reconstructs**: average RMSEs of reconstructions. Iterative ARES improves reconstruction by $9.6\%, 34\%$ and 60% over ARES, H-FUSE and LSQ, respectively

2.3.5 Complexity

In this section, we analyze the complexity of ARES. Taking into account the two-phase process of ARES, we need to figure out the complexity of each phase. Since both of these two phases aim to solve a sparse linear system, they should be reasonably fast. We estimate the complexity in two steps.

First Phase: $\vec{x}_{sp}$ reconstruction

Lemma 8 *For any report setting, let RD_{max} be the longest report duration, let H be the smooth matrix and let T be the total number of events to recover. Then the total computation time for our method is*

$$O(Tlog(T) + 4RD_{max}^2)$$

Proof 8 *Recall that* H-FUSE *requires computation time of $O(Tlog(T) + 4RD_{max}^2)$: see Lemma 5 in section 2.1.5 for* H-FUSE.

Second Phase: Applying annihilating filter

ARES performs $\vec{x}_{sp}$ reconstruction as the first step, and this takes $O(Tlog(T) + 4RD_{max}^2)$. The second step is applying the annihilation filter of length L. Notice that it results in a banded Toeplitz matrix with bandwidth L, thus this step takes $O(Tlog(T) + 4L^2)$ resulting in

$$O(2Tlog(T) + 4max(L, RD_{max})^2)$$

2.3.6 Conclusion

We proposed ARES, an automatic approach for reconstructing historical counts from possibly overlapping or incomplete aggregated reports. Our approach has the following advanced features:

- ARES efficiently discovers notable patterns in the target time series and utilizes them to improve the reconstruction accuracy.

- ARES applies advanced Annihilating Filters (AF) technique to derive such patterns.
- ARES uses the Minimum Description Length principle to automatically select the AF length that provides the best reconstruction.

Our experiments on the real-world epidemiological time series demonstrated outperforming reconstruction performance of ARES over the competitors.

2.4 HOMERUN: scalable sparse-spectrum reconstruction [3]

In this section, we introduce HOMERUN, a signal reconstruction method that exploits an alternative representation of the sequence and finds the spectrum of the target sequence. More specifically, we formulate the problem as so-called *basis pursuit* using the Discrete Cosine Transform (DCT) as a sparsifying dictionary *and* impose non-negativity and smoothness constraints. HOMERUN utilizes the energy compaction feature of the DCT by finding the sparsest spectral representation of the target sequence that contains the largest (most important) coefficients. We leverage the *Alternating Direction Method of Multipliers* to solve the resulting optimization problem with scalable and memory efficient steps.

2.4.1 Introduction

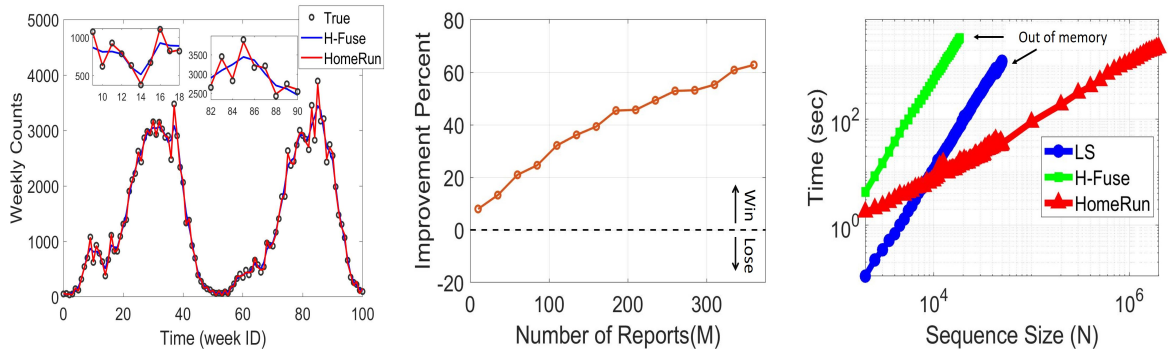
Gathering and analyzing information from multiple historical data sources requires reconstructing the time sequences in finer scale. For example, given multiple *monthly* sums of patient counts, how can we recover the *weekly* patient counts? This is so-called data disaggregation problem [108].

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 the same as in the informal problem formulation 1 in section 2.1.1 - but we repeat here for convenience:

Informal Problem 3 (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 have seen in the previous sections and which we will explain in more details later, this problem is usually under-determined in practice. In cases 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 (H-FUSE) as we have seen in section 2.1. 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. Another advanced approach that does not enforce smoothness or periodicity in the target sequences was introduced earlier in section 2.3 (ARES), where it utilizes annihilating filters to learn notable patterns in the target sequence to refine the reconstructed time series. Also in section 2.2, we introduced another method GB-R where it is applicable if the target sequences are known to follow the SIS epidemics model.



(a) More accurate reconstruction (b) HOMERUN wins (c) Linear scalability

Figure 2.24: **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.

In this section, we introduce 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 [100], the same dataset that we have been using in our earlier works (sections 2.1, 2.2, 2.3). 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 [77] and

LS.

Figure 2.24 shows an example of the results of HOMERUN when reconstructing the weekly counts of measles, given multiple aggregated reports. We observe in Fig. 2.24 (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. 2.24 (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. 2.24 (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**: 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.
- **Generality**: HOMERUN is parameter-free and it adapts to the input signal and automatically detects the prominent periodicities in the data.

This section is structured as follows. We explain the necessary background and the related work in section 2.4.2, and introduce our proposed method in section 2.4.3. Then, we explain our experimental setup in section 2.4.4 and show the experimental results in section 2.4.5. We conclude in section 2.4.6.

2.4.2 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: bold capital letters (e.g., $\mathbf{A}$) denote matrices; bold small letters (e.g., $\mathbf{x}$) denote column vectors; $\mathbf{A}^\dagger$ is the Moore-Penrose pseudo-inverse of a matrix $\mathbf{A}$; $\mathbf{A}^T$ denotes the transpose of $\mathbf{A}$. x_n is the n^{th} element in vector $\mathbf{x}$. $(\mathbf{a})_+$ is the non-negative projection of a vector $\mathbf{a}$, performed by zeroing out the negative elements in $\mathbf{a}$. Table 2.8 summarizes the symbols we use frequently.

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.

Table 2.8: 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
$\mathbf{H}$	$\in \mathbb{R}^{(N-1) \times N}$ smoothness matrix

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 2.25 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.

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}}_{\mathbf{O}} \times \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}}_{\mathbf{y}} \tag{2.39}$$

If $\mathbf{O}$ 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

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

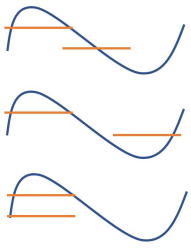


Figure 2.25: Historical data examples: overlap; gap; conflict (from top to bottom).

problem, which led researchers to add domain knowledge penalty terms to the linear system [77] 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 2.4.3). We define the DCT in the next section before proceeding to the proposed methods.

Discrete Cosine Transform (DCT)

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, 90]. 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 $\mathbf{x}$ of length N is as follows [66]:

$$s_k = \sum_{n=0}^{N-1} x_n \underbrace{\alpha(k) cos\left(\frac{\pi k(2n+1)}{2N}\right)}_{\phi(k,n)} = \sum_{n=0}^{N-1} x_n \phi(k,n) \tag{2.40}$$

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

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{N}}, & k = 0, \\ \sqrt{\frac{2}{N}}, & 1 \leq k \leq N-1. \end{cases} \tag{2.41}$$

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

$$x_n = \sum_{k=0}^{N-1} s_k \phi(k,n) \tag{2.42}$$

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:

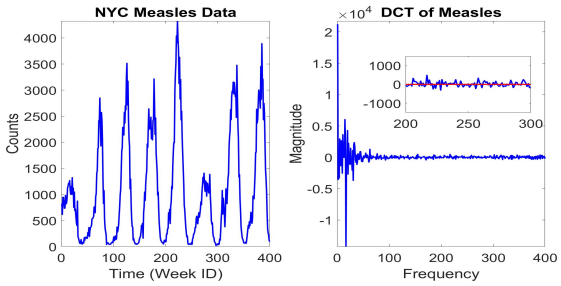


Figure 2.26: NYC measles data in time domain, **x** (left) and its spectrum, **s** (right).

$$\mathbf{D} = \begin{bmatrix} \phi(0,0) & \dots & \phi(0,N-1) \\ \vdots & \ddots & \\ \phi(N-1,0) & \dots & \phi(N-1,N-1) \end{bmatrix} = \begin{bmatrix} \mathbf{d}_0^T \\ \vdots \\ \mathbf{d}_{N-1}^T \end{bmatrix} \tag{2.43}$$

where, as it is clear from (2.40), $\phi(k,n) = \alpha(k)cos(\pi k(2n+1)/2N)$. 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 [90]), i.e., $\mathbf{D}^{-1} = \mathbf{D}^T$. Equations (2.40) and (2.42) can be written as:

$$\mathbf{s} = \mathbf{D}\mathbf{x} \tag{2.44}$$

$$\mathbf{x} = \mathbf{D}^T\mathbf{s} \tag{2.45}$$

Since cosines are *periodic* and *even symmetric*, the DCT transform imposes periodicity in the time domain signal [90]. 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 [105]. 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 [90]. The degree of DCT energy compaction depends on how correlated the original signal 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 2.26 and 2.27, 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 2.28, we show the time sequence of both data sets recovered from this 10% (using Equation (2.45)). It is clear that NYC measles has a better recovery since its DCT is

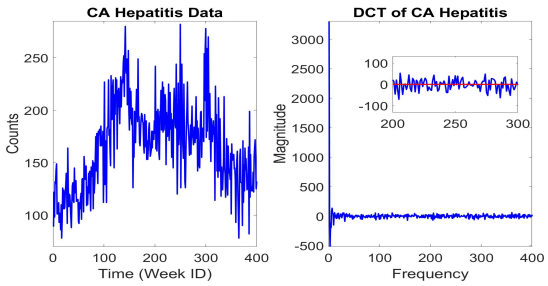


Figure 2.27: CA hepatitis data in time domain, **x** (left) and its spectrum, **s** (right).

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.

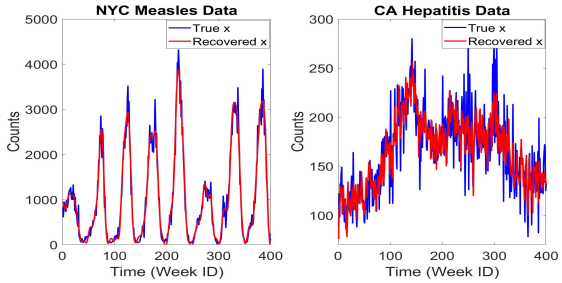


Figure 2.28: Data recovered from the largest 10% coefficients of their DCT – NYC measles (left) and CA hepatitis (right).

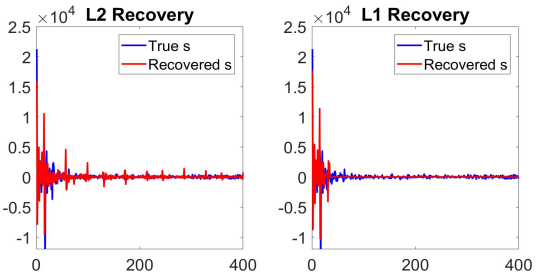


Figure 2.29: Recovery with $L1$ norm (left) and recovery with $L2$ norm by replacing $L1$ by $L2$ in (2.46) (right).

Sparse Signal Recovery

The goal of sparse reconstruction and compressive sensing [24] is to find a *sparse* approximate solution $\mathbf{s}$ 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, $\mathbf{s}$ could be recovered by solving the following convex problem known as BP [29]:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{A}\mathbf{s} = \mathbf{y} \end{aligned} \tag{2.46}$$

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 [39] 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 $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$, 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 [24]. 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., $\mathbf{x} = \mathbf{\Psi}\mathbf{s}$. 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 [25]. In this case, we would recover the coefficient sequence $\mathbf{s}$ with the minimum $L1$ norm that satisfies $\mathbf{A}\mathbf{s} = \mathbf{y}$, where $\mathbf{A} = \mathbf{\Phi}\mathbf{\Psi}$ in problem (2.46).

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 2.29, we illustrate the advantage of using $L1$ norm by showing the solution we get from (2.46) 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 2.26) 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 (2.46) 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

$$\begin{aligned} \min_{\mathbf{s},\mathbf{z}} \quad & f(\mathbf{s}) + g(\mathbf{z}) \\ s.t. \quad & \mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} = \mathbf{c}. \end{aligned} \tag{2.47}$$

by iteratively updating the following blocks

$$\mathbf{s} \leftarrow arg\min_{\mathbf{s}} f(\mathbf{s}) + (\rho/2)\|\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2, \tag{2.48a}$$

$$\mathbf{z} \leftarrow arg\min_{\mathbf{z}} g(\mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2, \tag{2.48b}$$

$$\mathbf{u} \leftarrow \mathbf{u} + (\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c}) \tag{2.48c}$$

where $\mathbf{u}$ is a scaled version of the dual variable corresponding to the equality constraint in (2.47), and $\rho > 0$ is the augmented Lagrangian parameter specified by the user.

Problem (2.46) can be reformulated as follows after introducing the auxiliary variable $\mathbf{z} \in \mathbb{R}^N$:

$$\begin{aligned} \min_{\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{As}=\mathbf{y}\}} + \|\mathbf{z}\|_1 \\ s.t. \quad & \mathbf{s} - \mathbf{z} = \mathbf{0} \end{aligned} \tag{2.49}$$

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

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

We will skip the derivation of the algorithm for brevity – refer to [18] for more comprehensive review of the ADMM algorithm. The solution to (2.46) is provided by the following iterative updates:

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

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

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

Related Work

Disaggregation: The aggregation of the data vector can be seen as representing or summarizing the data vector using linear transform. In [31, 32], 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, 37, 41, 77, 102, 141]

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 [91], or information recovery from noisy or missing data [26].

In section 2.1 [77], we 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 [72]. DCT is widely used especially for image compression [132] due to its energy compaction property, and for image denoising [40, 46]. DCT has been also used in databases community for answering queries in compressed form via DCT transform [55], representing the time series in spectral domain [36], etc.

The principle of finding a sparse signal representation in a basis dictionary has been used in various applications, such as denoising [29] and information recovery from incomplete and/or

noisy measurements [26]. 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 [24]. As an example, wavelet basis has been considered in order to sparsely represent an image to recover it from fewer measurements [25]. In [40], 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 2.9 summarizes our proposed HOMERUN method compared to the related approaches.

Table 2.9: HOMERUN satisfies all properties listed above.

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

2.4.3 Method

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:

$$\min_{\mathbf{x}} \quad \|\mathbf{y} - \mathbf{O}\mathbf{x}\|_2^2 \tag{2.50}$$

where $\mathbf{O}, \mathbf{x}, \mathbf{y}$ have the same definition as in Section 2.4.2. More advanced methods are proposed to infuse domain knowledge, such as smoothness and periodicity, by penalizing (2.50) [77]. 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 2.4.2. 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 [77]. 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.**

HOMERUN-0: The Basic Version

DCT matrix (defined in Equation (2.43)) offers a convenient way to compute the transform and its inverse as follows: $\mathbf{s} = \mathbf{D}\mathbf{x}$ (Equation (2.44)) is the DCT of the time sequence $\mathbf{x}$, and $\mathbf{x} = \mathbf{D}^T\mathbf{s}$ (Equation (2.45)) reconstructs the time sequence from $\mathbf{s}$. The following Insight shows the formulation of our HOMERUN-0 method.

Insight 1 *The historical data disaggregation problem can be formulated in the form of Basis Pursuit as follows:*

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{A}\mathbf{s} = \mathbf{y} \end{aligned} \tag{2.51}$$

Rationale 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:*

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y} \end{aligned} \tag{2.52}$$

Note that (2.52) *is similar to* (2.51) *with* $\mathbf{A} = \mathbf{O}\mathbf{D}^T$.

We solve the above problem using the ADMM algorithm with the iterative updates presented in Section 2.4.2. 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. 2.25: where we have two reports that cover the same time period, from 1st week to 4th week, but have different sums, 300 and 400.), then the linear system $\mathbf{O}\mathbf{x} = \mathbf{y}$ is inconsistent. As a result, the constraints in (2.52) 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 (2.52). 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

$$\overline{\mathbf{y}} = proj_{\mathbf{O}}(\mathbf{y}) = \mathbf{O}(\mathbf{O}^T\mathbf{O})^{-1}\mathbf{O}^T\mathbf{y} \tag{2.53}$$

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 [4]. 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).

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 (2.52). The resulting formulation becomes:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{s} \geq \mathbf{0} \end{aligned} \tag{2.54}$$

Statement 1 *The ADMM algorithm can be adapted to solve the optimization formulation of* HOMERUN-N *in Equation* (2.54).

Proof Problem (2.54) is convex and we reformulate it by introducing the auxiliary variables $\mathbf{r}, \mathbf{z} \in \mathbb{R}^N$ as follows

$$\begin{aligned} \min_{\mathbf{r},\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{z} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{z} = \mathbf{r}, \quad \mathbf{z} = \mathbf{s} \end{aligned} \tag{2.55}$$

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):

$$\mathbf{B} := \begin{bmatrix} \mathbf{O}\mathbf{D}^T \\ \mathbf{D}^T \\ \mathbf{I} \end{bmatrix}; \quad \mathbf{b} := \begin{bmatrix} \mathbf{y} \\ \mathbf{r} \\ \mathbf{s} \end{bmatrix}; \quad \mathbf{u} := \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \end{bmatrix}, \tag{2.56}$$

where $\mathbf{I} \in \mathbb{R}^{N\times N}$ is the identity matrix. By concatenating the constraints as $\mathbf{B}\mathbf{z} - \mathbf{b} = \mathbf{0}$, it is straightforward to see that problem (2.55) above is in the form of the ADMM form defined in Equation (2.47) 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 (2.55):

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}) = \|s\|_1 + \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \frac{\rho}{2}\big(&\|\mathbf{O}\mathbf{D}^T\mathbf{z} - \mathbf{y} + \mathbf{u}_1\|_2^2 \\ &+ \|\mathbf{D}^T\mathbf{z} - \mathbf{r} + \mathbf{u}_2\|_2^2 + \|\mathbf{z} - \mathbf{s} + \mathbf{u}_3\|_2^2\big) \end{aligned} \tag{2.57}$$

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

$$\mathbf{z} \leftarrow arg\min_{\mathbf{z}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \tag{2.58a}$$

$$\begin{cases} \mathbf{s} \leftarrow arg\min_{\mathbf{s}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \\ \mathbf{r} \leftarrow arg\min_{\mathbf{r}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \end{cases} \tag{2.58b}$$

$$\mathbf{u} \leftarrow \mathbf{u} + (\mathbf{B}\mathbf{z} - \mathbf{c}) \tag{2.58c}$$

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 [56] 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 states all these updates using linear algebraic notations. □

Algorithm 3: : HOMERUN-N (2.54)

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 (2.56); compute the pseudo-inverse of $\mathbf{B}$ 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 (2.56) 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, 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 $\mathbf{x}_{\text{HOMERUN-N}} = \mathbf{D}^T\mathbf{s}$.

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:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{s}\|_2^2 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{s} \geq \mathbf{0} \end{aligned} \tag{2.59}$$

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 [77] 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 [77] penalizes the underdetermined 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 (2.59) in a similar manner as HOMERUN-N model in the previous section. Thus, we reformulate (2.59) as follows:

$$\begin{aligned} \min_{\mathbf{r},\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \|\mathbf{s}\|_1 + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{z}\|_2^2 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{z} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{z} = \mathbf{r}, \quad \mathbf{z} = \mathbf{s} \end{aligned} \tag{2.60}$$

where $\mathbf{r}, \mathbf{z} \in \mathbb{R}^N$ are auxiliary variables. The augmented Lagrangian of (2.60) is:

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{r},\mathbf{s},\mathbf{z},\mathbf{u}) = \|s\|_1 + \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{z}\|_2^2 \\ + \frac{\rho}{2}\big(\|\mathbf{O}\mathbf{D}^T\mathbf{z} - \mathbf{y} + \mathbf{u}_1\|_2^2 \\ + \|\mathbf{D}^T\mathbf{z} - \mathbf{r} + \mathbf{u}_2\|_2^2 + \|\mathbf{z} - \mathbf{s} + \mathbf{u}_3\|_2^2\big) \end{aligned} \tag{2.61}$$

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 (2.56), and similarly we define:

$$\mathbf{Q} := \begin{bmatrix} \mathbf{O}\mathbf{D}^T \\ \mathbf{D}^T \\ \mathbf{I} \\ \mathbf{H}\mathbf{D}^T \end{bmatrix} \; ; \; \mathbf{q} := \begin{bmatrix} \mathbf{y} \\ \mathbf{r} \\ \mathbf{s} \\ \mathbf{0} \end{bmatrix} \; ; \; \mathbf{v} := \begin{bmatrix} \mathbf{u} \\ \mathbf{0} \end{bmatrix}, \tag{2.62}$$

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 (2.61) in terms of one variable while fixing the rest. We describe the implementation of HOMERUN in what follows.

Direct Implementation of HOMERUN In Algorithm 4, 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 $\mathbf{x}_{\text{HOMERUN}} = \mathbf{D}^T\mathbf{s}$.

Accelerated Implementation of HOMERUN In this section, we analyze the complexity of Algorithm 4, and derive a fast and memory efficient implementation of HOMERUN.

The steps in Algorithm 4 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

Algorithm 4: : HOMERUN (2.59) (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 (2.56); $\mathbf{q}^k$, $\mathbf{v}^k$ as defined in (2.62); 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 (2.56) using $\mathbf{s}^{k+1}$ and $\mathbf{r}^{k+1}$
- update $\mathbf{q}^{k+1}$ as defined in (2.62) 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 (2.56))
- update $\mathbf{v}^{k+1}$ as defined in (2.62) using $\mathbf{u}^{k+1}$
- Set $k := k + 1$.

Until maximum number of iterations K is reached (K=3000)

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

$$\begin{aligned}
\mathbf{z} &= \mathbf{Q}^\dagger(\mathbf{q} - \mathbf{v}) \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T(\mathbf{q} - \mathbf{v}) \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{D}\underbrace{\big(\mathbf{O}^T(\mathbf{y} - \mathbf{u}_1) + (\mathbf{r} - \mathbf{u}_2) + \overbrace{\mathbf{D}^T(\mathbf{s} - \mathbf{u}_3)}^{\mathbf{g}\in\mathbb{R}^N}\big)}_{\mathbf{p}\in\mathbb{R}^N} \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{D}\mathbf{p} \qquad (2.63) \\
&= \big(\mathbf{D}\mathbf{O}^T\mathbf{O}\mathbf{D}^T + \mathbf{D}\mathbf{D}^T + \mathbf{I} + \mathbf{D}\mathbf{H}^T\mathbf{H}\mathbf{D}^T\big)^{-1}\mathbf{D}\mathbf{p} \\
&= \big(\mathbf{D}(\mathbf{O}^T\mathbf{O} + 2\mathbf{I} + \mathbf{H}^T\mathbf{H})\mathbf{D}^T\big)^{-1}\mathbf{D}\mathbf{p} \\
&= \mathbf{D}(\underbrace{\mathbf{O}^T\mathbf{O} + 2\mathbf{I} + \mathbf{H}^T\mathbf{H}}_{\mathbf{Z}\in\mathbb{R}^{\mathbf{N}\times\mathbf{N}}})^{-1}\underbrace{\mathbf{D}^T\mathbf{D}}_{\mathbf{I}}\mathbf{p} \\
&= \mathbf{D}\mathbf{Z}^{-1}\mathbf{p}
\end{aligned}$$

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

$$\underbrace{\mathbf{D}^T\mathbf{z}}_{\mathbf{t}\in\mathbb{R}^N} = \mathbf{Z}^{-1}\mathbf{p} \qquad (2.64)$$

where the second equality in (2.63) is by the pseudo-inverse equation for $\mathbf{Q}$ (since it is tall with linearly independent columns due to $\mathbf{I}$; Eq. (2.62)); 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 (2.64) (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 5.

Lemma 9 *For any report setting in the historical disaggregation problem, if $M \leq N$, then the total computational complexity of* HOMERUN *(Algorithm 5) is*

$$\mathcal{O}(b^2N + NlogN) \tag{2.65}$$

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 9 *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 5 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:

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

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 2.4.5.

2.4.4 Experiments

In this section, we explain the set up of the experiments performed to evaluate the proposed method: datasets, baselines and metrics, and description of the input settings and configuration.

Data Sets

In order to study the performance of HOMERUN, we apply it to the data from project Tycho [100], 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 2.4.2), we explore the performance using six more data sets with different behavior. We pick the

Algorithm 5: : HOMERUN (2.59) (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)

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 2.30 (NYC measles and CA hepatitis are shown earlier in Figures 2.26 and 2.27). 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.

Baselines and Evaluation Metrics

We compare the performance of our method against two baselines, LS in (2.50) and H-FUSE [77], 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:

$$\min_{\mathbf{x}} \quad \|\mathbf{y} - \mathbf{O}\mathbf{x}\|_2^2 + \|\mathbf{H}\mathbf{x}\|_2^2 \tag{2.66}$$

where $\mathbf{H}$ is the smoothness matrix defined in Section 2.4.3.

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

$$RED = \frac{\text{RMSE}(baseline) - \text{RMSE}(proposed)}{\max(\text{RMSE}(baseline), \text{RMSE}(proposed))} \tag{2.67}$$

We use RED with the result figures in Section 2.4.5, ranging between -1 and 1. Clearly, positive RED means that the proposed method improves the baseline and vice versa.

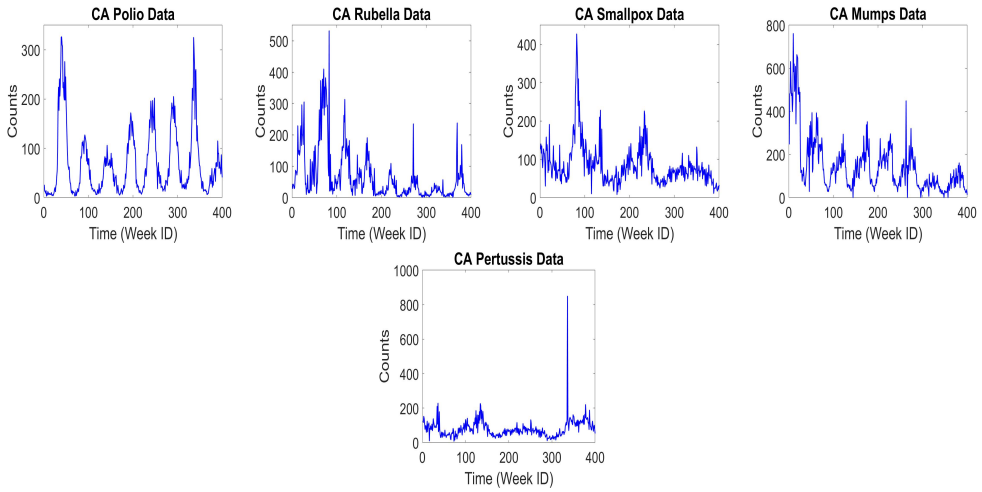


Figure 2.30: Disease Time Sequences

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 2.4.2, 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*.

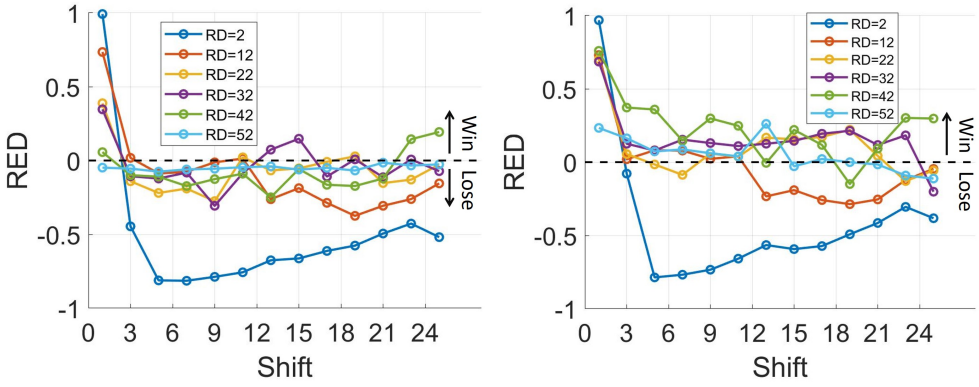
For most experiments, we generate reports with the same RD and $shift$ values and show the results on a wide range (RD spans from 2 to 52 weeks (1 year) with increment of 10 and the $shift$ span from 1 to 25 with increment of 2). Specifically, the first report (y_1) includes the weeks from 1 to RD; y_2 includes weeks from ($shift + 1$) to ($shift + 1 + RD$) and so on – refer to the example in Section 2.4.2. 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.

2.4.5 Results

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.

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 2.4.3. Presenting results of the earlier versions of HOMERUN demonstrates the benefit of the added constraints (non-negativity and smoothness).



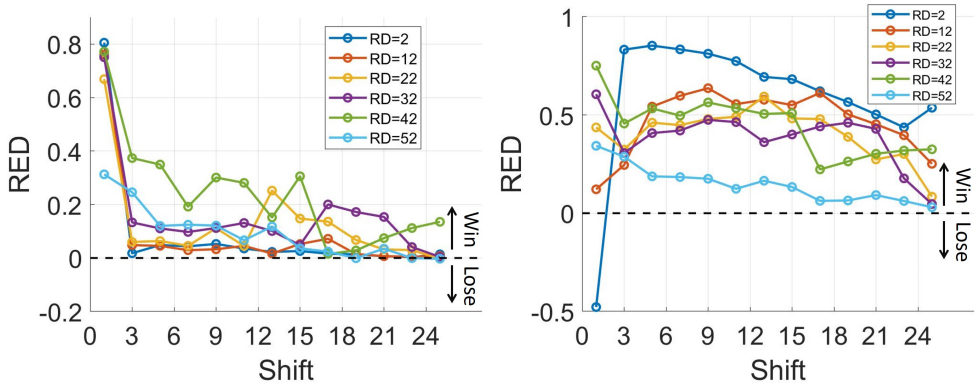
(a) HOMERUN-0 (b) HOMERUN-N

Figure 2.31: Comparing HOMERUN-0 and HOMERUN-N versions against the baseline H-FUSE with NYC measles data.

Performance of HOMERUN-0 We compare the reconstruction accuracy of HOMERUN-0 with the baseline H-FUSE using NYC measles data in Figure 2.31 (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 NYC measles data is shown in Figure 2.31 (b). Adding non-negativity constraint to HOMERUN-0 improves the accuracy significantly as it is clear from comparing Fig. 2.31 (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 2.4.4. 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 outper-



(a) Against H-FUSE (b) Against LS

Figure 2.32: **HOMERUN wins**. Performance of HOMERUN versus the baselines H-FUSE and LS using NYC measles data.

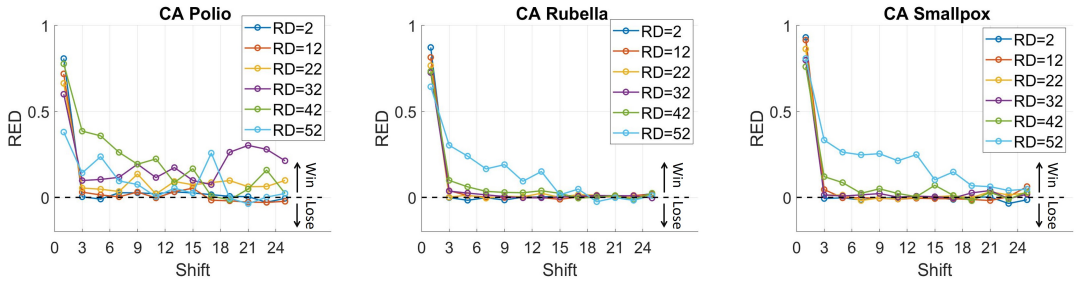
forms the competing baseline methods. Figure 2.32 (a) shows comparison between HOMERUN and the baseline H-FUSE with NYC measles data, HOMERUN is always superior with significant improvement. Note here that generally speaking, greater improvement happens with larger RD, which makes the problem more difficult as we have fewer observations (reports). Figure 2.32 (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 2.10, showing the RMSE at various RD and $shift$ values (a subset of the input settings in Fig. 2.32 (a) and (b), but similar pattern with other RD values) – again, note that the improvement is higher as RD increases.

Table 2.10: RMSE of HOMERUN and the baselines (H-FUSE and LS) using NYC measles data.

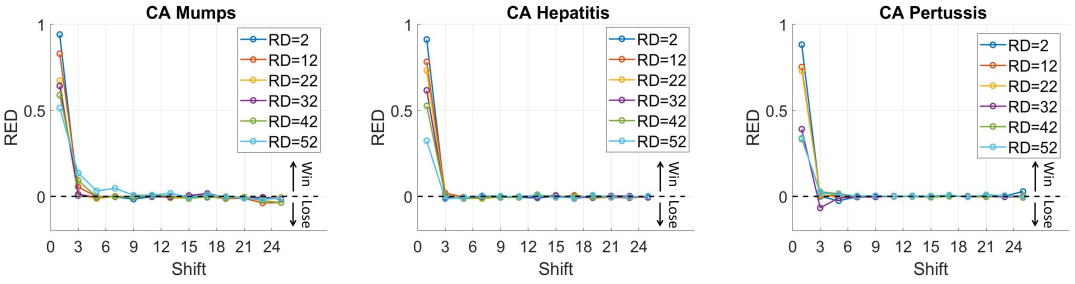
	$RD = 2$					$RD = 22$					$RD = 42$				
$shift$	1	7	13	19	25	1	7	13	19	25	1	7	13	19	25
HOMERUN	20.30	**205.96**	**415.79**	**609.03**	**644.38**	**42.49**	**191.10**	**227.03**	**400.07**	894.11	**57.28**	**186.79**	**305.27**	**565.85**	**514.48**
H-FUSE	104.23	215.38	425.56	617.27	653.45	128.38	200.22	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

We compare HOMERUN with H-FUSE (since it is the best baseline) using the other six data sets in Figure 2.33. 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 2.11 highlights the results comparing HOMERUN with the baseline H-FUSE in Fig. 2.32 (a) and Fig. 2.33. 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%.



(a) CA Polio (b) CA Rubella (c) CA Smallpox



(d) CA Mumps (e) CA Hepatitis (f) CA Pertussis

Figure 2.33: **HOMERUN consistently wins**. Performance of HOMERUN vs. the baseline H-FUSE with different data sets (positive=win and negative=lose).

Table 2.11: 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

So far, each single point in the result plots (or Table 2.10) corresponds to solving the problem given aggregated reports that have the same duration/resolution (i.e., same RD). In Figure 2.34, 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.

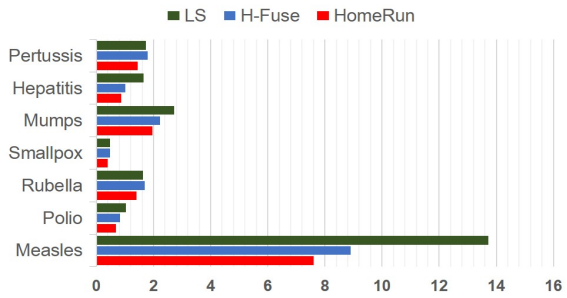


Figure 2.34: **HOMERUN wins**. RMSE of HOMERUN compared to H-FUSE and LS with reports having different RD.

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 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. 2.32 (a), Table 2.10, and top three data in Fig. 2.33. 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. (2.59)). 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. 2.33.

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

$$RED_{logical} = \begin{cases} 1 & RED > threshold \\ -1 & RED < -threshold \\ 0 & else \end{cases} \tag{2.68}$$

where we empirically set the threshold to 0.015. In Figure 2.35, 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 2.35 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.

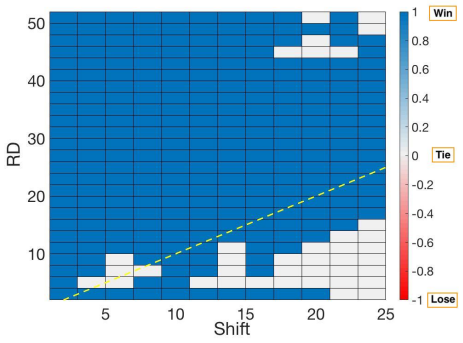


Figure 2.35: HOMERUN almost always wins and sometimes ties. HOMERUN performance against the baseline H-FUSE using NYC measles data.

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. 2.32 (a) and polio in Fig. 2.33 (a)). With the less periodic sequences (e.g., hepatitis in Fig. 2.33 (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. 2.26) and hepatitis (Fig. 2.27) in their time domain, and their performance (Fig. 2.32 (a) and 2.33 (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. 2.31 (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.

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 5 scales linearly with the length of the time sequence (see Figure 2.24 (c)). As clear from the comparison in Figure 2.24 (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 4 versus Algorithm 5). Moreover, HOMERUN is very efficient in terms of memory and can handle very large sequences.

2.4.6 Conclusion

In this section, 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 our work are summarized as follows:

1. Formulation: 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 $\mathcal{O}((2RD_{max} - 1)^2 N + NlogN)$.
4. Generality: 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.

2.5 Discussions

In this section, we discuss how our methods can be extended in the future works and provide a comparison of the methods to help users understand and use our methods appropriately.

2.5.1 How would this work for Poisson distributed data?

The Poisson distribution is a discrete probability distribution that is commonly used to model counts of occurrences such as number of patients arriving at emergency rooms, etc. Thus it makes more sense for modeling number of patients than continuous models such as normal distribution.

The probability mass function of a discrete random variable $\mathbf{X}$ having a Poisson distribution is given as follows:

$$Pr(X = k) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{2.69}$$

for Poisson parameter $\lambda > 0$, and $k = 0, 1, 2, \cdots$, where e is Euler's number and $k!$ is the factorial of k. The mean and variance of a Poisson distribution is given as $E(X) = Var(X) = \lambda$.

Lemma 10 *We can assume the number of patients Y to have a Poisson distribution. Then the objective function becomes:*

$$\sum_{i=1}^{n} y_i \left(\sum_{t=1}^{T} x_t A_{it} \right) - e^{(\sum_{t=1}^{T} x_t A_{it})} \tag{2.70}$$

where y_i is the i^{th} report, x_t is the true data or the number of patients in finer granularity that we are interested in reconstructing, A is the mixing matrix.

Proof 10 *In our problem setting, we would like to model the number of patients Y to have a Poisson distribution:*

$$Y \sim Poisson(\lambda).$$

Note that our information fusion problem can be formulated as a system of linear equations (Figure 2.3, equation 2.2). This makes it possible for us to utilize the Poisson regression formulation or the Generalized Liner Models (GLM) for Poisson distribution. (This is a well-known problem and we repeat the derivation below for the completeness [69].) Assuming that the logarithmic of the expected value of Y is represented as a linear combination of predictor variables: A and x or the mixing matrix and the target patient count in finer granularity in our problem setting:

$$\log(E(y_i)) = \lambda = x_1 A_{i1} + x_2 A_{i2} + \cdots + x_T A_{iT}. \tag{2.71}$$

Maximizing the likelihood of the data, we get:

$$\max_x \mathcal{L}(x|y) = \prod_{i=1}^{n} \frac{\lambda^{y_i} e^{-\lambda}}{y_i!}.$$

Maximizing the log likelihood:

$$\begin{aligned}\max_x \log \mathcal{L}(x|y) &= \sum_{i=1}^{n} y_i \log(\lambda) - \lambda - \log(y_i!) \\ &= \sum_{i=1}^{n} y_i \log(\lambda) - \lambda \\ &= \sum_{i=1}^{n} y_i(x_1 A_{i1} + \cdots + x_T A_{iT}) - e^(x_1 A_{i1} + \cdots + x_T A_{iT}) \\ &= \sum_{i=1}^{n} y_i \left(\sum_{t=1}^{T} x_t A_{it} \right) - e^{(\sum_{t=1}^{T} x_t A_{it})} \end{aligned} \tag{2.72}$$

We have our objective function in equation 2.70. There is no closed form solution, but we can use gradient descent to find x that maximizes this objective function. Note that this part is for solving reconstruction problem only from the reports, and if we want to add regularizers, we can repeat the steps in our previous works.

2.5.2 Possible extension of the work

We have introduced various methods that we proposed in solving the historical reconstruction problem. There are several possible interesting extensions to this line of work:

Extend to spatial domain: The time sequences we have worked on are 1-dimensional, the number of patient counts at a certain location. But since we do have data available for multiple different locations, it would be interesting to extend our method to consider space or the location as well. We would need to consider the spatial distances between the location points, and also determine the spread rate over the distance. In addition, we would have to apply suitable regularizers such as smoothness across the spatial domain. This extension would be interesting to see how the breakout at a location may dynamically affect the other locations.

Other applications - power grid systems: One possible application for our methods introduced in this section is power grid systems. On power grids, there are several different power meters that measure the time sequences of voltages, and currents over time. Often times, these meters do not have the same sampling rate - one may have coarser granularity than the other. It would be interesting to see how our method can be used to solve for the reconstruction of the power measurements from different meters with different resolutions. One thing to note would be that we would need a different set of regularizers suitable for this domain. We could start with the most naive ones like smoothness and periodicity as we did in H-FUSE, and develop the method further more with domain knowledge from the power domain.

2.5.3 Comparison of the methods

In this section, we compare the methods and suggest when to use which method.

- H-FUSE: This is the *simplest* signal reconstruction method. If the time sequences of your interest evolve *smoothly* over time, with a known *period*, then the most naive method you can try is H-FUSE.

- GB-R: If the target sequences follow the *epidemics model (*SIS*)*, then GB-R, which incorporates *nonlinear domain knowledge from epidemiology* is the better choice.
- ARES: If the target sequences are not necessarily epidemiological data, and you would like to *discover notable linear patterns* in your target sequences, we suggest using ARES.
- HOMERUN: If you know that the target sequences should be non-negative and are sure that the sequences consist of *multiple periods* (or wish to represent your target sequences using sparse dictionaries), then HOMERUN is the right choice.

In Table below (repeated from Table 2.1), we summarize a few main properties of the methods.

Comparison of H-FUSE, GB-R, ARES, HOMERUN (this Table is repeated from Table 2.1 in section 2 for convenience.)

Distinctions	LSQ	H-FUSE (2.1)	GB-R (2.2)	ARES (2.3)	HOMERUN (2.4)
Smoothness reconstruction		✓	✓	✓	✓
Periodicity reconstruction		✓	✓	✓	✓
Epidemics model (SIS)			✓		
Auto-detect linear patterns				✓	
Auto-detect multiple periods (quasi-periodic)					✓

Chapter 3

Brain data

3.1 BRAINZOOM: leveraging different brain scan modalities [42]

In this section, we introduce BRAINZOOM, a signal reconstruction method that cross-leverages multi-modal brain signals. Our experiments using a popular realistic brain signal simulator to generate fMRI and MEG demonstrate that high spatio-temporal resolution brain imaging is possible from these two modalities. The experiments also suggest that smoothness seems to be the best prior, among several we tried.

3.1.1 Introduction

Research in neuroimaging have resulted in the development of several techniques, viz., Electroencephalograms (EEG), functional Magnetic Imaging (fMRI) [89], Magnetoencephalograms (MEG) [30], etc. These uni-modal techniques use different mechanisms to measure human brain activity. While EEG measures electrical activity of the brain, MEG measures magnetic fields generated by electric currents in the brain, and fMRI measures metabolic response due to neural activation through a blood oxygen-level dependent (BOLD) signal. Due to the diversity in the sources of these measurements, different neuroimaging techniques often have complementary strengths. For instance, while EEG and MEG have low spatial but high temporal resolution, fMRI posses just the opposite – high spatial but low temporal resolution.

Combining two or more of such complementary signals to overcome limitations posed by individual modalities has resulted in the development of several multimodal neuroimaging techniques — see [13] for a recent review. They have been used for source localization and for learning commonalities across multiple modalities, with many of these methods targeted at reducing the dimension of the brain image data, or studying specific hypotheses about relationships between imaging modalities.

However, our interest in this work lies in reconstructing *super-resolution* images of activity across the brain – that is capturing the *union* of the information derived from each modality, not their intersection. While super-resolution algorithms have been applied in uni-modal settings [127], they have not been widely explored for multi-modal neuroimaging. We fill this gap by

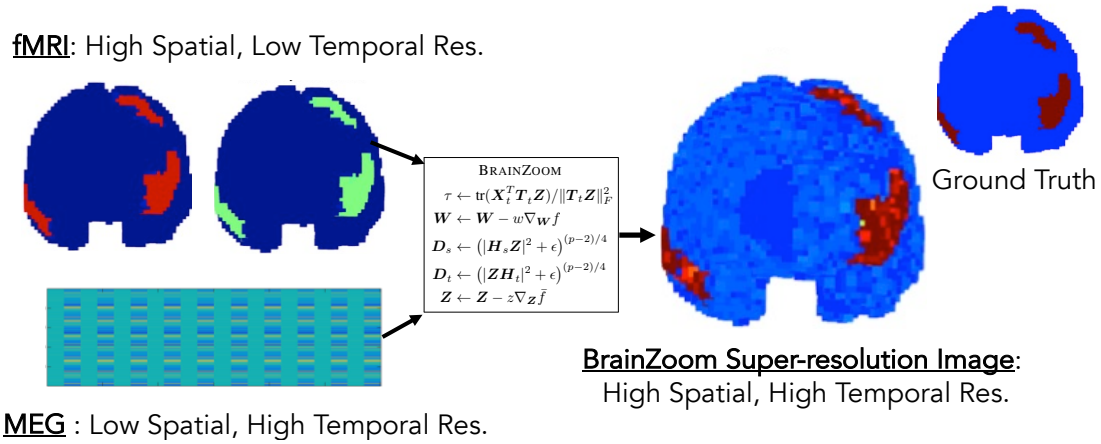


Figure 3.1: **BRAINZOOM effectively reconstructs**. BRAINZOOM effectively combines different brain measurement modalities with complementary strengths with respect to temporal and spatial resolution into a *super-resolution, whole-brain image*.

introducing BRAINZOOM, a novel method for super-resolution reconstruction of brain activity by combining different modalities of brain imaging. A snapshot of BRAINZOOM's capabilities is shown in Figure 3.1. Our long term goal is to complement the missing information of each modality, to ultimately construct the richest possible spatial-temporal summary of the underlying neural activity.

For the experiments in this work, we employ BRAINZOOM to produce super-resolution brain images by combining MEG/EEG and fMRI. However, we note that BRAINZOOM is more widely applicable and may be used to create super-resolution images from other modalities as well. We make the following contributions:

- **Novel Problem Formulation**: In Section 3.1.3 we formulate and solve the problem of generating a super-resolution brain image by combining different modalities (e.g., fMRI and MEG/EEG measurements). To the best of our knowledge, this is the first work to rigorously formulate this task as we have, as an optimization problem with this specific optimization objective, which takes into account the non-linearity in the fMRI measurements. Our formulation is also flexible in that it can easily incorporate a variety of priors that can help enhance the quality of reconstruction – so that one can seamlessly try out different hypotheses and pick out the one that gives the best performance.
- **Efficient Algorithm**: BRAINZOOM, introduced in Section 3.1.3, uses a novel modified alternating optimization method to solve the optimization problem mentioned above with high scalability. Through variable splitting, BRAINZOOM is able to handle nonlinear measurements. We also provide rigorous convergence analysis of the proposed method.
- **Evaluation**: Since the ground truth, high-resolution brain activity is not known, we cannot test the performance of our method on real brain data. Instead, in Section 3.1.4 we use simulated brain signals in order to test the ability of our method to recover the true solution. We generate synthetic brain activity using the Virtual Brain (TVB) [73] simulator. TVB is an open-source platform that creates brain activity simulations using biologically realistic assumptions, relying on well-known neuroscience models. With TVB, we can simulate

Property	*Localization*	*CCA, ICA*	*BRAINZOOM*
Hi-resolution imaging	✗	✗	✓
Scalability	✓	✓	✓
Quadratic modeling	✗	✗	✓
Convergence proof	N/A	✓	✓
Auto-scaling	✗	✓	✓
Generality	✗	✗	✓

Table 3.1: Comparison between BRAINZOOM and other related methods along various properties. We observe that BRAINZOOM overcomes limitations of competing methods.

synthetic brain signals with similar patterns as real brain data, as well as generate multi-modality observations such as fMRI, MEG, or EEG, at different sampling rates.

3.1.2 Background and Related Work

fMRI and MEG: The exact relationship between neural activity, observed fMRI, and observed MEG signals is not fully understood. It is known that fMRI measures the ratio of oxygenated to deoxygenated hemoglobin in the blood, at a spatial resolution of approximately 1 mm, and it is widely accepted that this oxygen fluctuation is related to the convolution of local neural activity with an impulse response that lasts for 8- 10 seconds. It is also believed that fMRI corresponds more closely to local field potentials in the brain than to actual spiking [78]. In contrast, the MEG device measures magnetic fields emanating from the brain, at 1 msec temporal resolution. These magnetic fields are generated from neural currents, and are especially strong when many neurons are spatially aligned and fire synchronously. MEG signals also depend, unlike fMRI, on the direction of the neural currents generating these magnetic fields. There have been several studies of the exact relationship between MEG and fMRI measures of neural activity. For example, Kujala et al. [70] provide evidence that fMRI correlates with source-localized MEG activity in many regions of the brain, that the exact nature of this correlation varies somewhat across the brain, and that fMRI typically correlates best with the high frequency components of MEG.
Multi-modal Methods: In the literature, there are two main classes of methods for analyzing multi-modal brain imaging data: those that aim to localize a few dominant 'point' sources of brain activity using inverse imaging or high-resolution methods; and those that target the recovery of a few dominant 'latent' components. The latter include popular multi-modal fusion techniques that rely on factor analysis tools, such as Independent Component Analysis (ICA) which aims to recover statistically independent latent sources of brain activation; and Canonical Correlation Analysis (CCA) which aims to unravel sources of co-variation in multiple 'views' of the brain stemming from the different modalities. Variants of PCA [27], ICA [23], and CCA [33] have been studied for multi-modal fusion of various brain imaging modalities. See

[12, 13, 34, 47, 76, 119, 124] for additional background on existing localization and common component extraction methods.

The main difference between our approach and the aforementioned two broad lines of work is that we aim to reconstruct a high spatio-temporal resolution image of the entire brain – not just a few point or dominant sources / components.

Super-resolution in Image Processing: Super-resolution via fusing multi-modal data is also an active research area in image processing. Specifically, fusion of remotely sensed hyper-spectral and multi-spectral images is of great interest [79]. Both hyper- and multi-spectral images are special images whose pixels span many frequency bands and can provide spectral information of the materials contained in the pixels – which finds many applications in geoscience, mineral detection, and food safety. Hyper-spectral images have very high spectral resolution but very coarse spatial resolution, and the situation of the multi-spectral images is exactly the opposite. Much effort has been spent on fusing these two types images that captured on the sites, to create super-resolution data in both space and frequency. The task is very similar to ours, except that the transformation from the virtual super-resolution image (i.e., $\boldsymbol{Z}$) to hyper- and multi-spectral images are both linear (to be precise, sub-sampling) [111, 135]. But in our problem, the transformation from $\boldsymbol{Z}$ to fMRI is nonlinear – which poses a much more challenging optimization problem.

3.1.3 Method

In this section, we first present assumptions behind our modeling of entire brain super-resolution image construction from multi-modal neuroimaging data, and then formulate it as an optimization problem. Subsequently, we discuss various priors we can add to the optimization objective. Then we propose BRAINZOOM, a highly scalable and flexible algorithmic framework that solves the above optimization problem, while provably converging to a stationary point at least sub-linearly, despite the fact that the problem is non-convex.

Notations & Modeling Assumptions

Viewing the brain as a 3D object, we can imagine its volume partitioned into many small 3D areas called voxels (equivalent to pixels in a 2D image). We are interested to know the intensity of the neural activity at each of these voxels, during a particular time frame. We start by denoting the complete brain activity as a matrix $\boldsymbol{Z}$, where each column of $\boldsymbol{Z}$ is the (vectorized) brain activity at all voxels at a particular time tick, and each row of $\boldsymbol{Z}$ corresponds to the brain activity at a particular voxel, at all recorded time ticks. However, we are not able to observe $\boldsymbol{Z}$ directly, and resort to fMRI and MEG/EEG images as inputs. The goal of this work is to reconstruct the high-resolution brain activity matrix $\boldsymbol{Z}$ in *both* spatial and temporal domain, given the MEG and fMRI measurements.

Given the different phenomena observed by MEG and fMRI, what shall we define as the units of our latent neural activity Z? In this work, we assume Z is a measure of the intensity of neural activity that is linearly related to the MEG or EEG observations. Let $\boldsymbol{X}_t$ denote the high temporal-resolution image given by MEG or EEG. It is related to $\boldsymbol{Z}$ through a linear operator T_t

that operates on the spatial domain, possibly corrupted with noise.

$$\boldsymbol{X}_t = T_t \boldsymbol{Z} \tag{3.1}$$

Notice that the temporal resolution of $\boldsymbol{Z}$ is the same as that of the given MEG/EEG measurements, since there is no way we can go beyond them in this dimension.

To capture the fact that fMRI is insensitive to the direction of the activity that MEG observes, we model fMRI as dependent on a linear function of the *square* of the Z activity. Although this is at best an approximation of the unknown, complex relationship between neural spiking, local field potentials, neural currents, and observed fMRI and MEG, this model does capture some basic properties, and it enables us to frame a well-posed problem of solving for high spatial and temporal resolution measure of neural activity. The success of our method in a simulator [73] of brain activity suggests that this approximation is reasonable. Let $\boldsymbol{X}_s$ be the high spatial resolution image captured by fMRI. We model the dependence between $\boldsymbol{X}_s$ and Z through a linear operator T_s as shown below, again possibly corrupted with noise.

$$\boldsymbol{X}_s = \sigma |\boldsymbol{Z}|^2 T_s, \tag{3.2}$$

Here, $\sigma|\boldsymbol{Z}|^2$ is the element-wise square of $\boldsymbol{Z}$ with an unknown scaling factor σ. We note that he spatial resolution of $\boldsymbol{Z}$ is the same as that of the given fMRI images.

Reconstructing the high-resolution brain activity matrix $\boldsymbol{Z}$ in *both* the spatial and temporal domains, given the fMRI measurements $\boldsymbol{X}_s$ and MEG/EEG measurements $\boldsymbol{X}_t$, and their corresponding sensing matrices T_s and T_t, is a highly ill-posed inverse problem, since the number of measurements we are given is much smaller than the number of unknowns we want to estimate. One has to impose additional constraints on the neural activity to make the problem well-defined, which will be discussed in detail in the next sections. Furthermore, considering the relationship between the underlying neural activity $\boldsymbol{Z}$ and the fMRI measurements as a *nonlinear* transformation makes the problem even harder.

Problem formulation

According to the physics of multi-modal brain imaging measurements described in equations (3.1) and (3.2), the problem of infering the super-resolution description of neural activity Z can be formulated naively as the following optimization problem,

$$\min_{\boldsymbol{Z},\sigma} \ \|\boldsymbol{X}_t - T_t\boldsymbol{Z}\|_F^2 + \|\boldsymbol{X}_s - \sigma|\boldsymbol{Z}|^2 T_s\|_F^2 + r(\boldsymbol{Z}), \tag{3.3}$$

where $r(\boldsymbol{Z})$ is some regularization function we impose on $\boldsymbol{Z}$ to make the estimation problem well-defined. Before we dive into different assumptions on the brain activities, let us first focus on the second term that tries to fit the fMRI data, which is a sixth-order polynomial with respect to σ and $\boldsymbol{Z}$ together, and makes the problem highly non-convex and seemingly difficult to tackle.

In terms of σ, we notice that it is just an unknown scaling factor, and we can equivalently put it into the first term in (3.3)

$$\min_{\boldsymbol{Z},\tau} \ \|\boldsymbol{X}_t - \tau T_t\boldsymbol{Z}\|_F^2 + \|\boldsymbol{X}_s - |\boldsymbol{Z}|^2 T_s\|_F^2 + r(\boldsymbol{Z}),$$

resulting the optimal solution $\boldsymbol{Z}$ to be just a scaled version of that of (3.3), which is usually inconsequential in practice. With respect to the term that involves $|\boldsymbol{Z}|^2$, we invoke the *variable splitting* trick by introducing another variable $\boldsymbol{W}$, and at the same time try to minimize the

difference between $\boldsymbol{Z}$ and $\boldsymbol{W}$ in the following way

$$\min_{\boldsymbol{Z},\boldsymbol{W},\tau} \quad \|\boldsymbol{X}_t - \tau T_t \boldsymbol{Z}\|_F^2 + \|\boldsymbol{X}_s - (\boldsymbol{Z} * \boldsymbol{W}) T_s\|_F^2 + \mu \|\boldsymbol{Z} - \boldsymbol{W}\|_F^2 + r(\boldsymbol{Z}), \tag{3.4}$$

where $*$ denotes Hadamard (element-wise) multiplication. Although the resulting formulation (3.4) is still non-convex, the idea is to alternatingly update $\boldsymbol{Z}$, $\boldsymbol{W}$, and τ, so that the cost decreases monotonically. With careful design of the algorithm, we can even claim stronger convergence results, which will be shown (Theorem 1).

There exist a lot of commonly used priors that can be used to reconstruct high-resolution brain signals in our formulation (3.4). In this work we consider the following possibilities:

- minimum energy: $r(\boldsymbol{Z}) = \rho \|\boldsymbol{Z}\|_F^2$;
- sparsity: $r(\boldsymbol{Z}) = \rho \|\boldsymbol{Z}\|_1$, the sum of absolute values;
- low rank: $r(\boldsymbol{Z}) = \rho \|\boldsymbol{Z}\|_*$, the sum of singular values;
- smoothness: $r(\boldsymbol{Z}) = \rho(\|\boldsymbol{H}_s \boldsymbol{Z}\|_F^2 + \|\boldsymbol{Z} \boldsymbol{H}_t\|_F^2)$,
  where matrices $\boldsymbol{H}_t$ and $\boldsymbol{H}_s$ have the form [17]

$$\begin{bmatrix} -1 & 2 & -1 & 0 & \dots \\ 0 & -1 & 2 & -1 & \dots \\ \vdots & & \ddots & & \\ & & -1 & 2 & -1 \end{bmatrix}, \tag{3.5}$$

  with appropriate sizes.
- total variation: $r(\boldsymbol{Z}) = \rho(\|\boldsymbol{H}_s \boldsymbol{Z}\|_{p,\epsilon}^p + \|\boldsymbol{Z} \boldsymbol{H}_t\|_{p,\epsilon}^p)$, similar to the smoothness regularization, but in this case the matrices $\boldsymbol{H}_s$ and $\boldsymbol{H}_t$ have the form

$$\begin{bmatrix} 1 & -1 & 0 & & \dots \\ 0 & 1 & -1 & & \dots \\ \vdots & & \ddots & & \\ & & & 1 & -1 \end{bmatrix}, \tag{3.6}$$

  with appropriate sizes and $\|\cdot\|_{p,\epsilon}^p$ denotes the ℓ_p quasi-norm defined as

$$\|\boldsymbol{A}\|_{p,\epsilon}^p = \sum_{i,j} (|a_{ij}|^2 + \epsilon)^{p/2}, \;\; 0 < p \leq 2.$$

  For $p = 2$, this norm reduces to the normal ℓ_2 norm that is used in the smoothness regularization; for $0 < p \leq 1$, this regularization encourages the solution to be piece-wise constant, which is something we want to promote in this case.

In the rest of this section we will design an algorithm based on the latter two regularizations, which have similar forms but only differ in the definitions of the $\boldsymbol{H}$ matrices, for the following reasons:

1. Because fMRI measures convolutions of $|\boldsymbol{Z}|^2$, there is an inherent sign ambiguity in the entries of $\boldsymbol{Z}$ that are not sampled by the T_t matrix using the first three regularizations, whereas by enforcing elements that are spatially and temporally close to each other to have similar values, this inherent sign ambiguity can be resolved. Therefore, before real

validations, our conjecture is that the latter two regularizations seem to be better models.

2. Computationally, the first three regularizations can be handled by very simple *proximity operators* [96], thus it will be easy to modify the forth-coming algorithm to accommodate them. The smoothness and total-variation regularization, on the other hand, are more complicated to deal with—this is partly why we choose to use the ℓ_p quasi-norm rather than the more commonly used ℓ_1 norm to promote the effect that we want in $\boldsymbol{Z}$.

Towards this end, we formalize the problem formulation as

$$\begin{aligned}\min_{\boldsymbol{Z},\boldsymbol{W},\tau} \quad & \|\boldsymbol{X}_t - \tau T_t \boldsymbol{Z}\|_F^2 + \|\boldsymbol{X}_s - (\boldsymbol{Z} * \boldsymbol{W}) T_s\|_F^2 \\ & + \mu \|\boldsymbol{Z} - \boldsymbol{W}\|_F^2 + \rho(\|\boldsymbol{H}_s \boldsymbol{Z}\|_{p,\epsilon}^p + \|\boldsymbol{Z}\boldsymbol{H}_t\|_{p,\epsilon}^p),\end{aligned} \tag{3.7}$$

and we denote the cost function in (3.7) as $f(\boldsymbol{Z}, \boldsymbol{W}, \tau)$.

Proposed method

Based on the formulation, we propose BRAINZOOM, a highly scalable and flexible algorithmic framework for the brain super-resolution problem based on alternating optimization.

Let us start with the update of τ, which is independent of $\boldsymbol{W}$, and by fixing $\boldsymbol{Z}$, the conditionally optimal update of τ is

$$\tau \leftarrow \text{tr}(\boldsymbol{X}_t^T T_t \boldsymbol{Z}) / \|T_t \boldsymbol{Z}\|_F^2. \tag{3.8}$$

As for the update of $\boldsymbol{W}$, to avoid heavy computations, we propose to perform just one simple gradient step, where the gradient with respect to $\boldsymbol{W}$ can be calculated as

$$\begin{aligned}\nabla_{\boldsymbol{W}} f = \boldsymbol{Z} * \left[(\boldsymbol{Z} * \boldsymbol{W}) T_s T_s^T\right] - \boldsymbol{Z} * (\boldsymbol{X}_s T_s^T) \\ + \mu(\boldsymbol{W} - \boldsymbol{Z}).\end{aligned} \tag{3.9}$$

To ensure decrease of the cost function value, we need to make sure that the step-size is less than one over the Lipschitz constant of $\boldsymbol{W}$ conditioned on $\boldsymbol{Z}$. Our selection is

$$w = 1 \Big/ \left(\lambda_{\max}(T_s T_s^T) \max(|\boldsymbol{Z}|^2) + \mu\right), \tag{3.10}$$

where $\lambda_{\max}(\cdot)$ computes the largest eigenvalue of the argument matrix, and in Appendix A.1 we prove that it ensures decrease. Notice that $\lambda_{\max}(T_s T_s^T)$ only needs to be computed once, can then saved for subsequent iterations.

Now we turn to the update of $\boldsymbol{Z}$, and we again propose to go just a similar gradient step to avoid heavy computations. However, the regularization terms $\|\boldsymbol{H}_s \boldsymbol{Z}\|_{p,\epsilon}^p + \|\boldsymbol{Z}\boldsymbol{H}_t\|_{p,\epsilon}^p$ are non-differentiable. To compensate this, we first invoke the following property for the ℓ_p quasi-norm:

$$\|\boldsymbol{A}\|_{p,\epsilon}^p \leq \sum_{i,j} \frac{p}{2} \left(|\tilde{a}_{ij}|^2 + \epsilon\right)^{(p-2)/2} |a_{ij}|^2 + \text{constant},$$

for all $\tilde{\boldsymbol{A}}$, and the equality holds when $\boldsymbol{A} = \tilde{\boldsymbol{A}}$. Using this property, we define $\bar{f}(\boldsymbol{Z}; \boldsymbol{W}, \tau)$ as

$$\begin{aligned}\bar{f}(\boldsymbol{Z}; \boldsymbol{W}, \tau) = & \|\boldsymbol{X}_t - \tau T_t \boldsymbol{Z}\|_F^2 + \|\boldsymbol{X}_s - (\boldsymbol{Z} * \boldsymbol{W}) T_s\|_F^2 \\ & + \rho(\|\boldsymbol{D}_s * (\boldsymbol{H}_s \boldsymbol{Z})\|_F^2 + \|\boldsymbol{D}_t * (\boldsymbol{Z}\boldsymbol{H}_t)\|_F^2) \\ & + \mu \|\boldsymbol{Z} - \boldsymbol{W}\|_F^2,\end{aligned} \tag{3.11}$$

where

$$\begin{aligned} \boldsymbol{D}_s &= \left(|\boldsymbol{H}_s\tilde{\boldsymbol{Z}}|^2+\epsilon\right)^{(p-2)/4}, \\ \boldsymbol{D}_t &= \left(|\tilde{\boldsymbol{Z}}\boldsymbol{H}_t|^2+\epsilon\right)^{(p-2)/4}, \end{aligned} \tag{3.12}$$

and $\tilde{\boldsymbol{Z}}$ is from the previous iteration. Notice that when $p=2$, i.e., in the smoothness case, the $\boldsymbol{D}$ matrices are simply the all one matrices, which makes $\bar{f}=f$. In the total variation case when $0<p\leq 1$, function $\bar{f}(\boldsymbol{Z};\boldsymbol{W},\tau)$ is a smooth upperbound function for $f(\boldsymbol{Z},\boldsymbol{W},\tau)$ with respect to $\boldsymbol{Z}$ while fixing $\boldsymbol{W}$ and τ. Then we can apply a similar gradient descent step with gradient calculated as

$$\begin{aligned} \nabla_{\boldsymbol{Z}}\bar{f} =& \tau^2 T_t^T T_t\boldsymbol{Z} - \tau T_t^T\boldsymbol{X}_t + \mu(\boldsymbol{Z}-\boldsymbol{W}) \\ &+ \boldsymbol{W}*\left[(\boldsymbol{Z}*\boldsymbol{W})T_sT_s^T\right] - \boldsymbol{W}*(\boldsymbol{X}_sT_s^T) \\ &+ \boldsymbol{H}_s^T\left[\boldsymbol{D}_s^2*(\boldsymbol{H}_s\boldsymbol{Z})\right] + \left[\boldsymbol{D}_t^2*(\boldsymbol{Z}\boldsymbol{H}_t)\right]\boldsymbol{H}_t^T, \end{aligned} \tag{3.13}$$

and step size chosen as

$$\begin{aligned} z = 1\Big/\Big(&\tau^2\lambda_{\max}(T_t^TT_t)+\lambda_{\max}(T_sT_s^T)\max(|\boldsymbol{W}|^2) \\ &+\mu+\rho(\max(\boldsymbol{D}_s^2)c_H^2+\max(\boldsymbol{D}_t^2)c_H^2)\Big), \end{aligned} \tag{3.14}$$

where c_H is a sharp upperbound on the maximum singular value of the $\boldsymbol{H}$ matrices

$$c_H = \begin{cases} 4, & \text{for smoothness (3.5)}, \\ 2, & \text{for total variation (3.6)}. \end{cases} \tag{3.15}$$

Again, in Appendix A.1 we prove that this choice of step size guarantees decrease of the cost function.

Summarizing (3.8)-(3.14), we propose BRAINZOOM as the following iterative update rule

$$\boxed{\begin{aligned} &\text{BRAINZOOM} \\ &\tau \leftarrow \text{tr}(\boldsymbol{X}_t^TT_t\boldsymbol{Z})/\|T_t\boldsymbol{Z}\|_F^2 \\ &\boldsymbol{W} \leftarrow \boldsymbol{W} - w\nabla_{\boldsymbol{W}}f \\ &\boldsymbol{D}_s \leftarrow \left(|\boldsymbol{H}_s\boldsymbol{Z}|^2+\epsilon\right)^{(p-2)/4} \\ &\boldsymbol{D}_t \leftarrow \left(|\boldsymbol{Z}\boldsymbol{H}_t|^2+\epsilon\right)^{(p-2)/4} \\ &\boldsymbol{Z} \leftarrow \boldsymbol{Z} - z\nabla_{\boldsymbol{Z}}\bar{f} \end{aligned}} \tag{3.16}$$

Convergence Properties

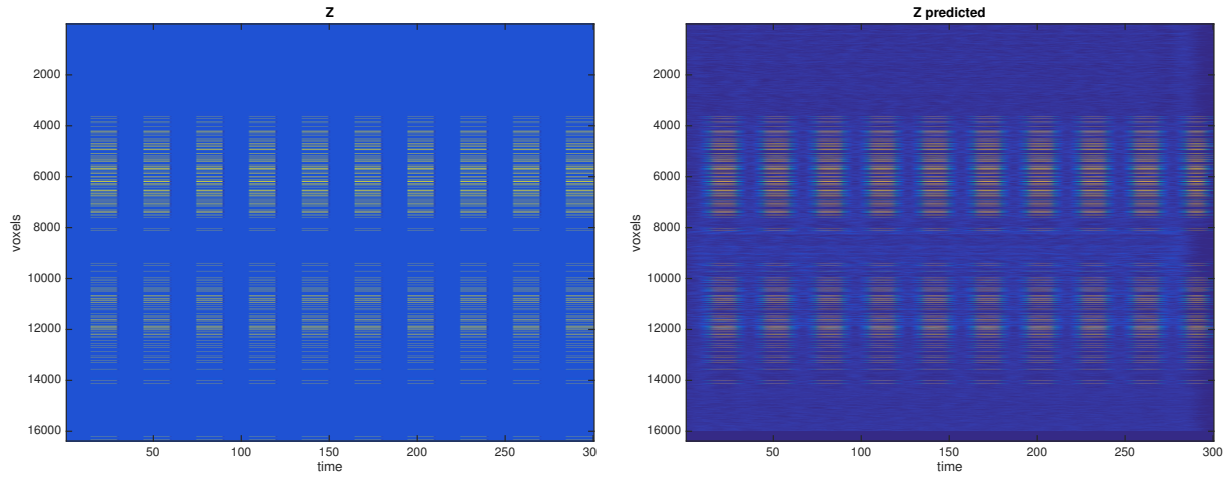
BRAINZOOM has very nice convergence properties, as we show here.

Theorem 1 *Let $\{(\boldsymbol{Z}^{(r)},\boldsymbol{W}^{(r)},\tau^{(r)})\}_r$ be the solution sequence produced by the proposed* BRAINZOOM (3.16)*. We have*

1) Every limit point of $\{(\boldsymbol{Z}^{(r)},\boldsymbol{W}^{(r)},\tau^{(r)})\}_r$ is a stationary point of Problem (3.7)*.*

2) The optimality gap between $\{(\boldsymbol{Z}^{(r)},\boldsymbol{W}^{(r)},\tau^{(r)})\}_r$ and a stationary point is at most $\mathcal{O}(1/r)$; i.e., the algorithm approaches a stationary point at least sub-linearly.

The proof of Theorem 1 is relegated to Appendix A.2.



(a) Ground-truth

(b) Reconstruction by BRAINZOOM

Figure 3.2: **BRAINZOOM reconstructs well:** (a) Ground-truth $\boldsymbol{Z}$, and (b) reconstructed $\boldsymbol{Z}$.

Remark. The proposed BRAINZOOM is a flexible combination between alternating optimization and gradient descent. A clear advantage is that all the variations that people have been using to modify the algorithm to accommodate constraints and non-linearities, by simply putting an additional projection or proximal step. This includes the first three regularizations that we discussed in Section 3.1.3, namely, minimum energy, sparsity, and low rank. Moreover, if we have the prior knowledge that the underlying signal $\boldsymbol{Z}$ is non-negative, we could also add a simple projection step to zero out the negative entries. All the convergence results can be similarly applied.

3.1.4 Experiments

We evaluate BRAINZOOM on simulated brain data. Since in real neuroscience data we do not have access to the ground truth super-resolution brain activity, we generate realistic brain signals using an open source framework, the Virtual Brain (TVB) [73], which is able to simulate brain networks with biologically realistic connectivity. TVB implements several well-known neuroscience models, and can produce brain signals with similar patterns as real brain data, as well as generate multi-modal observations such as fMRI, MEG, or EEG, at different sampling rates.

Data preparation: In the following few lines, we describe details of the simulation. First, we select a type of stimulus, and apply it to the brain at a location and time of choice. In this experiment, we use a 3s pulse stimulus, with 3s delay. TVB records the response of the brain to this stimulus, at a very high frequency, with an integration step size of 0.0625ms. From this response we generate our $\boldsymbol{Z}$ as the temporal average of this raw data, sampled at 200ms.

We show $\boldsymbol{Z}$ in matrix form in Figure 3.2, where the horizontal axis represents time (we get 300 samples in time, over a total of 60s), and the vertical axis represents 16384 voxels (vectorized). Next, TVB can create multi-modality observations of this underlying signal. We sample MEG data at 200ms, and fMRI data at 1000ms. The MEG data is produced by TVB by projecting the underlying signal to the surface of the brain, at the predefined location of 248 MEG

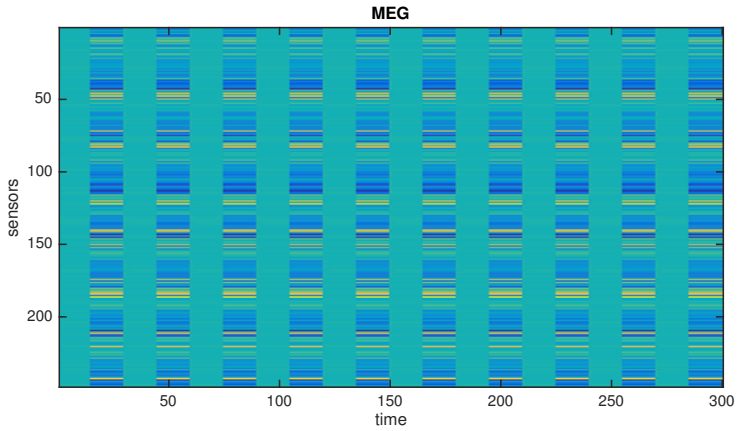


Figure 3.3: MEG for 248 sensors $\times$ 300 time points, across 60s of simulation.

sensors. This projection is also encoded in our T_t matrix. The fMRI data is obtained by convolving the brain signal with a hemodynamic response function (hrf) of choice. Here, we chose the Gamma hrf that is given by $f(t) = (\frac{t}{\tau})^{(n-1)} * \frac{exp(-\frac{t}{\tau})}{\tau(n-1)!}$ with the default parameters suggested by TVB. We also encode this hrf in our T_s matrix. Note that TVB applies this convolution directly on underlying signal $\boldsymbol{Z}$, resulting in a simple linear transformation. Since in our method we can model a more complex relationship between fMRI and $\boldsymbol{Z}$ that TVB does not capture, and we want to test if our algorithm can recover the underlying signal under this condition, we modify generative process of the fMRI measurements to capture this non-linearity – i.e., we set the fMRI signal to be the convolution of $|Z|^2$ and the hrf described above.

Results: Using the fMRI, MEG, T_s and T_t thus constructed, we apply our algorithm to recover the super-resolution brain $\boldsymbol{Z}$. We show $\boldsymbol{Z}$, MEG, and fMRI as space $\times$ time matrices in Figures 3.3 and 3.4, respectively. Figure 3.2 shows our reconstructed $\boldsymbol{Z}$, using the smoothing regularizer. One can see that the reconstructed $\boldsymbol{Z}$ matches the ground truth very well, although the recovering problem is a challenging under-determined non-linear inverse problem. This suggests that the proposed algorithm combined with the smoothness regularization is effective in fusing MEG and fMRI data.

We also map the above space$\times$time matrices back to brain manifolds at different time points and show some interesting results in Figure 3.5. Here, we present two fMRI images as in the right column, which are recorded at time points 38.0s and 40.0s, respectively. Before 38.0s, there was no activity in the brain (cf. the upper-left subfigure). At time point 38.0s, a stimulus is presented at three regions of the brain, and the activity spans a duration starting from 38.0s across 40.0s. Our recovered $\boldsymbol{Z}$'s at 37.9s and 39.9s – when fMRI are not recorded – are presented in the middle column. One can see that the recovered $\boldsymbol{Z}$'s match the true $\boldsymbol{Z}$'s very well. This indicates that our approach correctly interpolates the time points by using the information provided by MEG, thereby achieving temporal super-resolution.

We also show the results obtained by using the minimal energy regularization in Figure 3.6. It can be seen that the interpolation between 39.0s and 40.0s is not as clear as we have seen in Figure 3.5. In addition, at the time point prior to the stimulus being present, there is an obvious artifact (bright spot) near the right edge of the estimated brain. This suggests that under the considered signal model, smoothness is a more promising regularizer.

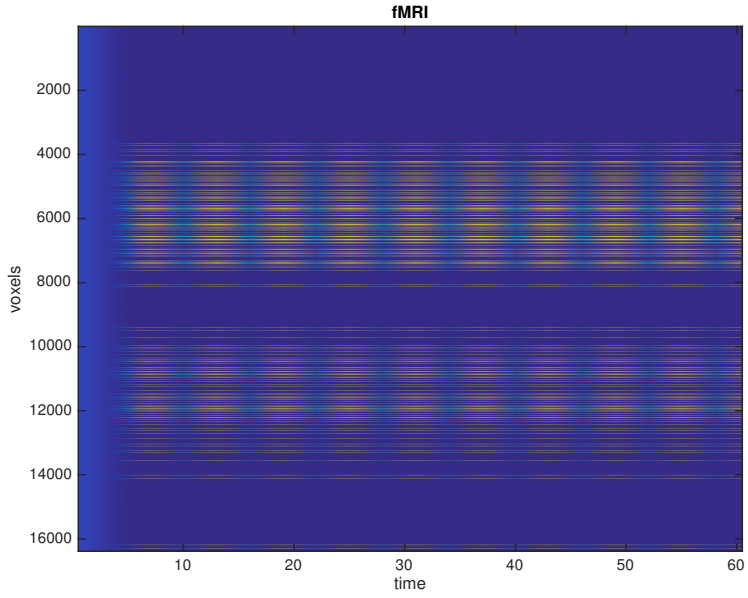


Figure 3.4: fMRI for 16384 voxels $\times 60$ time points, across 60s of simulation.

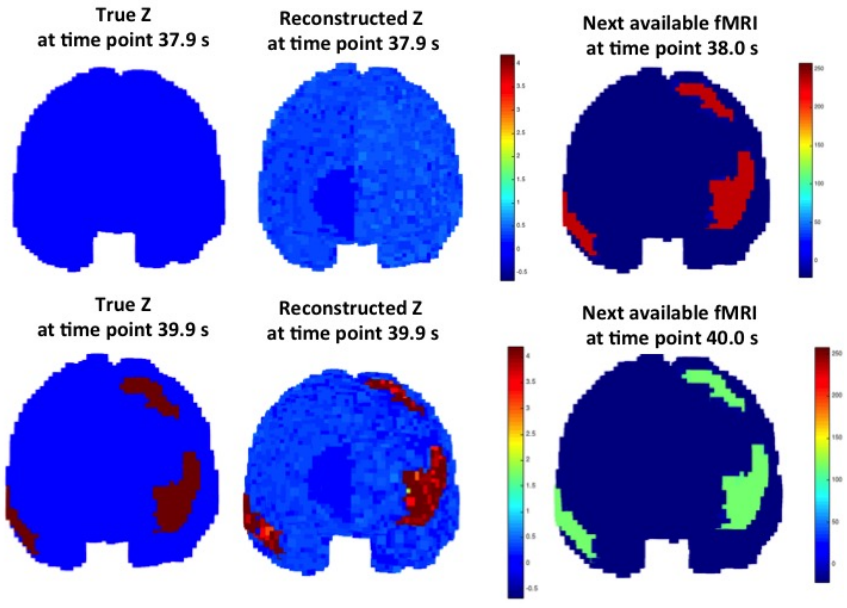


Figure 3.5: **Smoothness regularizer recovers well.** True brain activity, predicted brain activity and fMRI at two time points: when the stimulus is present (top row), and when it is absent (bottom row). Reconstructed $\boldsymbol{Z}$ is by using the smoothness regularizer and $\lambda = 1,000$.

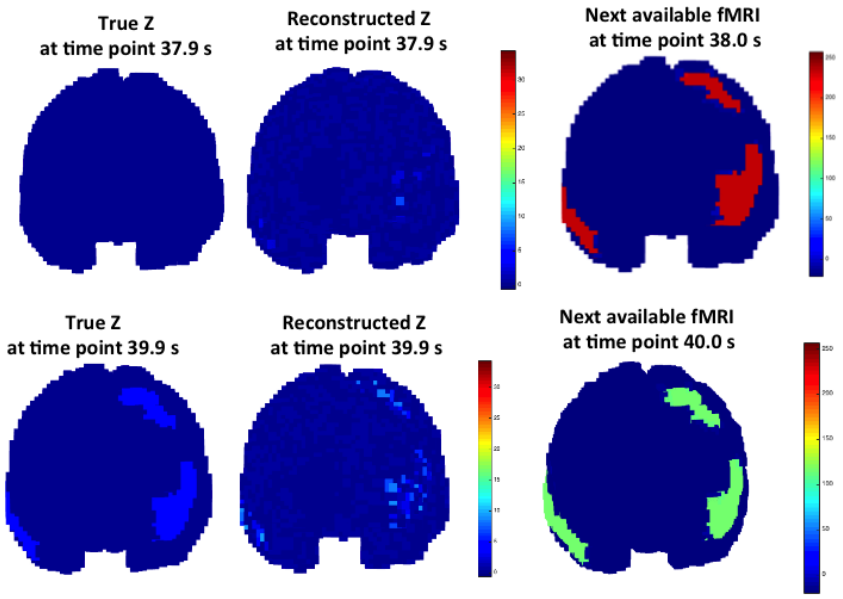


Figure 3.6: **Minimal energy regularizer fails to recover.** True brain activity, predicted brain activity and fMRI at two time points: when the stimulus is present (top row), and when it is absent (bottom row). Reconstructed $\boldsymbol{Z}$ is by using the minimal energy regularizer.

3.1.5 Conclusions

Estimating high spatial-temporal neural activity in the brain from multiple imperfect imaging methods is a key to studying and understanding brain function. In this work we introduce BRAINZOOM, a principled algorithmic tool to address the brain activity super-resolution problem.

Our contributions are as follows:

- **Novel Problem Formulation**: We formulate the problem of recovering a super-resolution image from multi-modal brain signals. To the best of our knowledge, this work is the first to directly address the reconstruction of a *super-resolution brain image for the whole brain*.
- **Efficient Algorithm**: We propose BRAINZOOM, an algorithmic framework that
  - handles non-linearity
  - learns the unknown scaling between multi-modal signals
  - incorporates a large number of regularizations
  - is efficient
  - provably converges
- **Evaluation**: Our experimental results on the realistic simulated data matches with intuition. Both the space$\times$time plots of the estimated super-resolution matrices and the brain-manifold illustration show that our recovered space-time super-resolution brain matches with the ground-truth very well.

3.1.6 Extension of the work - epilepsy

One natural follow-up question for our BRAINZOOM work would be: 'Can this work be extended to model various brain disorders, e.g. epilepsy?'. The short answer is 'maybe, with appropriate regularizers'. Our objective function is to *reconstruct* the brain signals in high resolution in both space and time, and to achieve this goal, we are enforcing smoothness across each dimension. But BRAINZOOM is capable of incorporating various constraints as we discussed in section 3.1.3. Brain disorders such as epilepsy are known to exhibit sharp high frequency brain signals compared to those of normal condition. Since BRAINZOOM of the final choice of the regularizer is enforcing smoothness, BRAINZOOM as it is would not be suitable to model high frequency signals. However, it would be interesting to try out several different regularizers to see what would be the most appropriate one for solving for epilepsy brain signals. One suggestion would be to enforce sparseness in the frequency domain so that we are using only a few frequency signals to represent the brain signal, of which would be for normal signals and a few for epilepsy signals.

Part II

Part 2: Signal mining

In Part I, we showed how to reconstruct time sequences from multiple sources in larger granularity. We showed methods and applications for 1-dimensional signals (historical epidemiology data in Chapter 2) as well as high-dimensional signals (brain scans in Chapter 3). In the following Part II, we show how to mine given signals, i.e. detecting anomalies. We will present our methods and applications for lower-dimensionality signals in Chapter 4 (power grid signals), and higher-dimensionality signals in Chapter 5 (aircraft signals).

Chapter 4

Power grid data

In this Chapter, we describe the methods that we have proposed to solve the signal mining problem in power grid domain.

We introduce POWERCAST (section 4.1), and STREAMCAST (section 4.2). The methods are introduced in increasing order of complexity. POWERCAST is applicable when we have a history of power consumption data, and STREAMCAST further improves it by incorporating weather components and allowing for streams of sequences.

4.1 POWERCAST: tensor decomposition for multi-step forecasting [115]

In this section, we introduce POWERCAST, a tensor decomposition based signal mining method for forecasting the electrical power demand, by carefully incorporating domain knowledge (physics model called BIG model). POWERCAST utilizes domain expert knowledge which allows for an *intuitive interpretation* of the power load. With the help of the domain knowledge, POWERCAST is able to provide accurate prediction in multi-step ahead forecasting and extrapolations under various *what-if* scenarios.

4.1.1 Introduction

The goal of the smart electrical grid is to manage the demand and supply of electricity while maintaining both efficiency and reliability. Indeed, [5] finds that improving the reliability of the U.S. grid could provide savings of around $\$49$ *billion* per year and provide a 12% to 18% reduction in emissions, while improving efficiency could save an additional $\$20$ *billion*. Toward this goal, monitoring systems have been put in place, including Phasor Measurement Units (PMUs) and newer high-precision micro-PMUs [117]. Using these data sources to accurately model load behavior in the grid, as well as to forecast future power load, is important in protecting the grid from failure and for maintaining reliability.

Our main goal is to understand how a specific service area consumes power (a university campus, a small factory, a village or neighborhood), by studying its past behavior. Once we have a good model for the power-consumption behavior, then we can do forecasting (*how much power will our campus/factory need tomorrow*), spot anomalies (when our forecast is too far from what actually happened), and answer "what-if" scenarios, like *how much power will we need, during spring-break on campus*; or *during a heat-wave, in our neighborhood*.

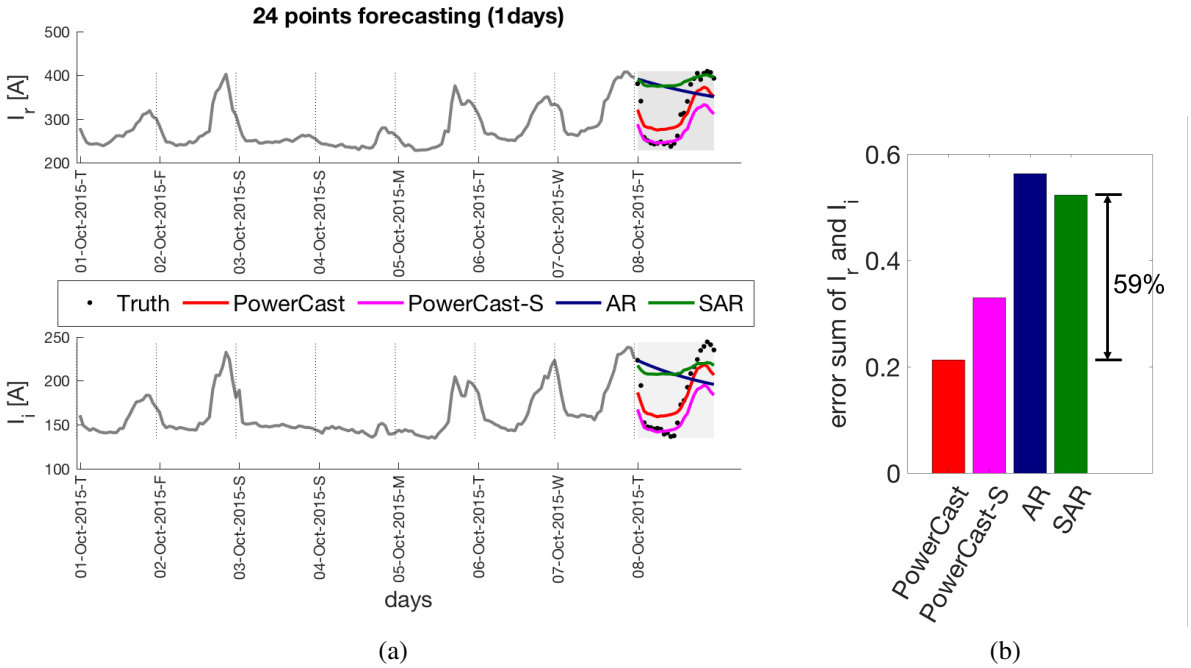
We focus on these two problems: forecasting, and 'what-if' scenarios. The informal definitions, are as follows. Note that alternating current (AC) (I) and voltage (V) are modeled as complex numbers.

Informal Problem 1 (Multi-step forecasting on power grid)

- ***Given:*** *real and imaginary current ($I_r(t), I_i(t)$) and voltage ($V_r(t), V_i(t)$) of previous N time points ($t = 1, \cdots, N$),*
- ***Forecast:*** *the electric current demand, for N_f steps in the future (i.e., guess $I_r(t), I_i(t)$, for $t = N+1, \cdots, N+N_f$)*

Throughout this work, we assume that the voltage in the future, is given ($V_r(t), V_i(t)$ for $t = N+1, \cdots, N+N_f$). This is realistic: except for rare brown-outs, power-plants try hard to provide near-constant voltage to consumers.

In Figure 4.1, 24 step (1-day) forecasting result on Lawrence Berkeley National Laboratory (LBNL) Open μPMU project data [117] is shown. In (a), POWERCAST (*red*) and POWERCAST-S (*pink*) provides more accurate forecasting, following closely to the truth (*black* dots) while the competitors (AR: *blue*, SAR:*green*) fail. We observe that POWERCAST is able to forecast the the current demand with an accurate daily pattern while AR and SAR fail to consider the daily



(a) (b)

Figure 4.1: **POWERCAST forecasts accurately.** (a) POWERCAST (*red*) and POWERCAST-S (*pink*) forecasts 24 steps (1-day) on I_r (top row), and I_i (bottom row) more accurately compared to competitors (AR: *blue*; SAR: *green*; ground truth *black circles*). (b) RMSE comparison of the methods

pattern. In (b), a quantitative comparison on the forecasting accuracy is shown. POWERCAST forecasts with **59%** reduction in error, compared to the competitors.

An additional benefit of our domain-aware approach is that it can handle the *what-if* extrapolation problem:

Informal Problem 2 ('what-if' extrapolation)

- **Given** *the historical data, as above (real and imaginary current $I_r(t)$, $I_i(t)$, and voltage $V_r(t)$, $V_i(t)$, $t = 1, \ldots, N$*
- **Guess** *what will be the power demand in the future (currents $I_r(t)$, $I_i(t)$, $t > N$), if, say, the student population doubles on our campus (or our factory cuts production in half, etc).*

Handling what-if scenarios is beyond the reach of black-box methods (ARIMA etc), exactly because it demands domain knowledge, which we carefully infuse, using the established "BIG" model.

Our contributions are as follows:

1. **Infusion of domain knowledge**: Our method is domain-aware: among the many aggregated load models (BIG, "PQ", "ZIP"), we chose the first, because it allows for an accurate and *intuitive interpretation* of the power load, being derived from physics-based first principles.
2. **Forecasting/What-if**: The proposed POWERCAST (a) leads to *more accurate* forecasts (up to **59%** lower error) compared to textbook, black-box methods, and (b) it can answer *what-if* scenarios that black-box methods can not.

3. **Anomaly detection**: POWERCAST can spot, and even explain, anomalies in the given time sequences. (see section 4.1.4, Figure 4.6)

Reproducibility: Our code is open-sourced at: `www.cs.cmu.edu/~hyunahs/code/PowerCast.zip`; the LBNL dataset is at: `powerdata.lbl.gov/`.

The structure of this section is typical: We give the background and related works (section 4.1.2), our proposed method (section 4.1.3), experiments (section 4.1.4), and conclusions (section 4.1.5).

4.1.2 Background and Related Work

Related works

Load model for aggregated electrical demand: the BIG model Electrical load modeling for power system analysis has been traditionally done using the constant power *constant PQ* and *ZIP* models [92]. However, industry experience has shown that these models incorrectly characterize load behavior [82]. Recent advances in steady-state power system simulations [19, 93] have introduced the use of physics based state variables, i.e. currents and voltages, and shown that load behavior at a given time instance can be accurately described by a linear relationship between current and voltage. From circuit theory, this can be represented by a parallel or series combination of reactance (B) and conductance (G). This model captures both voltage magnitude and angle information, in contrast to existing traditional load models [92]. Further adding a current source (I) results in the BIG load model, which accurately captures load sensitivities to voltage variations over a period of time [61]. The complete description of the BIG load model is given in section 4.1.3.

Table 4.1: **POWERCAST captures all of the listed properties**. *AR++* = ARIMA, seasonal ARIMA etc. *LDS* = Linear Dynamical Systems. '*Pattern discovery*' = concept / latent-variable discovery.

Properties	AR++[10, 16, 137]	Kalman/LDS[60, 63]	PARAFAC++[68, 104]	HMM++[74, 130]	AutoCyclone [121]	POWERCAST
Pattern (=concept) discovery			✓	✓	✓	✔
Forecasting	✓	✓		✓		✔
Seasonal patterns	✓				✓	✔
Domain knowledge inclusion						✔
What-if scenarios						✔

Time series forecasting Classical methods for time-series forecasting include the family of autoregression (AR)-based methods, including ARMA, exponential smoothing [21], ARIMA [16], ARMAX [137] and ARFIMA [10] models. Seasonal ARIMA [16] incorporates pre-determined, constant periods allowing the model to capture seasonal patterns. More recent generalizations include TBATS [35] which allows for more complex seasonality patterns. Other methods for time series forecasting include Kalman filtering [63], Linear Dynamical Systems (LDS) [60], Hidden Markov Models (HMMs) [74], wavelet-based methods such as AWSOM [94] and non-linear dynamical systems such as RegimeCast [83].

In the area of modeling of aggregated load in the power grid, autoregression is an extremely common approach due to its simplicity and interpretability [22, 58, 113, 114]. [97] uses ARMA models and exponential smoothing for short-term load forecasting. [88] uses a two-step procedure of seasonality removal, followed by ARMA with hyperbolic noise.

Tensor-based time series analysis Tensor decomposition methods, including PARAFAC decomposition, Tucker decomposition [67, 68], multilinear principal components analysis [80], and Bayesian tensor analysis [123] are powerful tools to understand latent factors of a target dataset. Recent work such as Marble [52] and Rubik [131] applies tensor factorization for concept analysis with domain knowledge. In terms of time series applications, tensors have been used for modeling multiple coevolving time series for epidemiology [85], community discovery [75], and concept discovery [64]. [84] uses a tensor-based approach for forecasting based on complex sequences of timestamped events. AutoCyclone [121] models seasonality in tensor datasets by 'folding' the data into a higher-order tensor. TensorCast [7] effectively forecasts time-evolving networks by incorporating multiple data sources in coupled tensors.

However, all of the aforementioned methods are typically not based on electrical load models derived from first principles, as is the case for the BIG load model, which the herein proposed method is based on, thus allowing for more accurate forecasts and interpretable models. In addition, our approach allows for *what-if* scenarios, e.g. variation of electrical load due to change in number of people in the building.

Background

BIG model The BIG equivalent circuit model [62] is a linear representation of the current using the voltage data follows:

$$I_r(t) = G(t)V_r(t) - B(t)V_i(t) + \alpha_r(t) \tag{4.1}$$

$$I_i(t) = B(t)V_r(t) + G(t)V_i(t) + \alpha_i(t) \tag{4.2}$$

By modeling the given time sequences I_r, I_i, V_r, V_i using the BIG model, we learn the BIG parameters G, B, α_r, α_i that provides us with an intuitive interpretation of the aggregated conductance (G) and susceptance (B) of the power grid as it varies over time. In POWERCAST, we convert the given time sequences to the BIG domain (G, B, α_r, α_i) and further proceed with pattern analysis and forecasting. As we will demonstrate in our analysis with experimental results in section 4.1.4, working in the BIG domain provides more stable and interpretable analysis of the power systems: understanding the power system (section 4.1.4), accurate forecasting and exploration of various what-if scenarios (section 4.1.4), and anomaly detection (section 4.1.4).

4.1.3 Method

In this work, our interest is in: 1) including the domain knowledge of the intrinsic behavior of the campus (G, B), 2) capture the latent periodic patterns in the sequences, and 3) do forecasting.

The core parts of POWERCAST algorithm is: 1) to convert the given time sequences into the BIG domain, and 2) forecasting in tensor structure. By converting the time sequences into the BIG domain, we try to understand the power system (G, B value range, dynamics, ratio, etc) that generated the sequences I_r, I_i, V_r, V_i that we observe, which enables us to do forecasting under *what-if* scenarios of change in the power systems on campus. Working with tensors tells us the long-term, and daily patterns. By performing forecasting based on the learned patterns from the past, POWERCAST provides more stable forecasting result (section 4.1.4). In this section, we will explain how we convert the time sequences into the BIG domain and how we analyze tensors to do forecasting in more details.

A table of symbols that is used throughout this section is shown in Table 4.2.

Table 4.2: Symbols and definitions

Symbols	Definitions
I_r, I_i, V_r, V_i	given time sequences. Real and imaginary part of complex current and voltage.
N	total number of timeticks in the given time sequences. $N = N_d \times N_{dt}$
N_d	number of days in the given time sequences
N_{dt}	number of time points for each day
G, B, α_r, α_i	BIG parameters. (conductance, susceptance, offset to I_r and I_i)
$\mathcal{X}$	a tensor of given time sequences I_r,I_i,V_r,V_i. $\mathcal{X} \in \mathbb{R}^{N_d \times N_{dt} \times 4}$
$\mathcal{B}_a$	a tensor of BIG parameters G, B, α_r, α_i. $\mathcal{B}_a \in \mathbb{R}^{N_d \times N_{dt} \times 4}$
$\mathcal{B}_b$	a tensor of BIG parameters G, B, α_r, α_i, after extension. $\mathcal{B}_b \in \mathbb{R}^{(N_d+N_{fd}) \times N_{dt} \times 4}$
N_{fd}	number of days for forecasting
N_f	number of time steps for forecasting. $N_f = N_{fd} \times N_{dt}$
P_{ar}	auto-regression parameter
R	rank for tensor decomposition
N_w, σ	Parameters for Gaussian filter. (window size, standard deviation of Gaussian)

POWERCAST algorithm Pseudocode of POWERCAST is described in Algorithm 6, where each function will be explained in more detail. In Figure 4.2, a flowchart of the Algorithm 6 is illustrated.

Step 1: Seq2Tensor (Tensor construction): Given four time sequences $I_r(1:N), I_i(1:N), V_r(1:N), V_i(1:N)$, we construct a tensor $\mathcal{X} \in \mathbb{R}^{N_d \times N_{dt} \times 4}$ by cutting the sequences into

Algorithm 6: POWERCAST.

Data: $I_r(1:N), I_i(1:N), V_r(1:N+N_f), V_i(1:N+N_f)$
Result: $\hat{I}_r(N+1:N+N_f), \hat{I}_i(N+1:N+N_f)$
Construct a tensor from given sequences:;
$[\mathcal{X}]$ = Seq2Tensor$(I_r(1:N), I_i(1:N), V_r(1:N), V_i(1:N))$;
Covert the data to BIG domain (fit the data to BIG model):;
$[\mathcal{B}_a]$ = X2B$(\mathcal{X})$;
Extend the tensor for forecasting:;
$[\mathcal{B}_b]$ = Bextension$(\mathcal{B}_a)$;
Recover the data from BIG domain:;
$[\hat{I}_r(N+1:N+N_f), \hat{I}_i(N+1:N+N_f)]$ =
B2Seq$(\mathcal{B}_b, V_r(N+1:N+N_f), V_i(N+1:N+N_f))$;

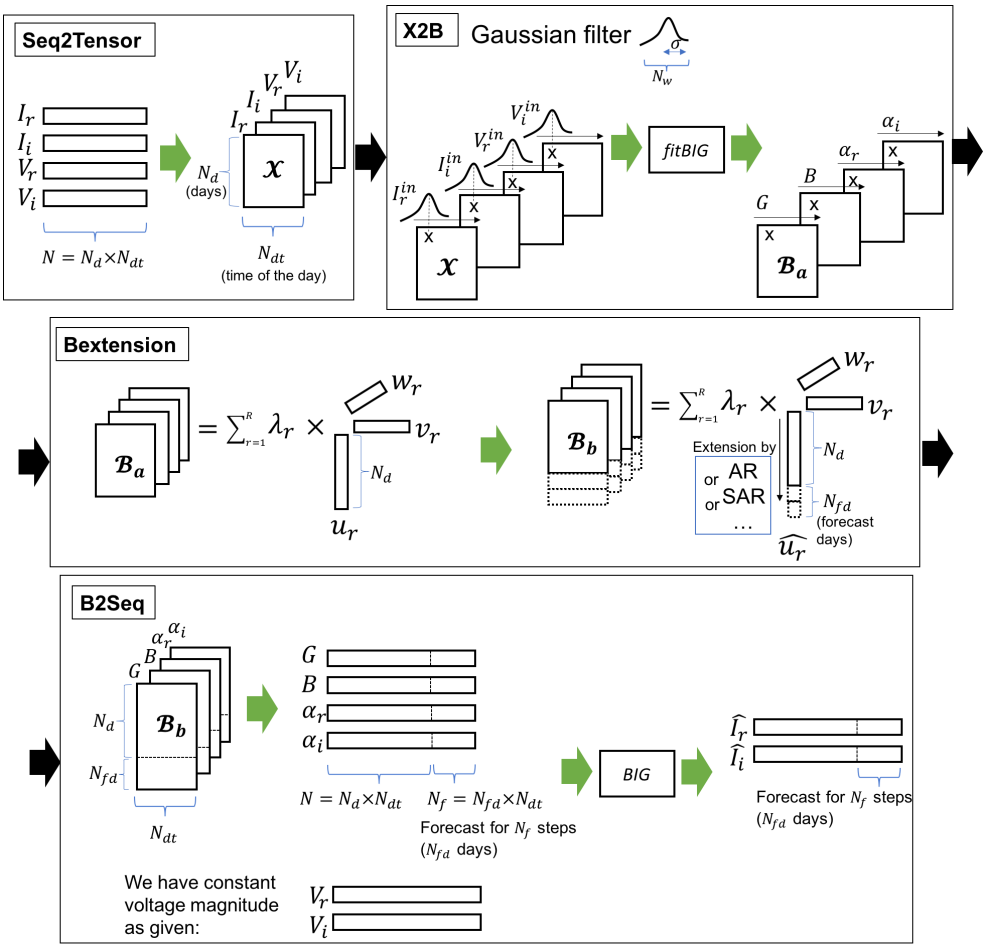


Figure 4.2: **Flowchart of POWERCAST.**

daily unit as described in Figure 4.2 "*Seq2Tensor*." The first, second, and third mode of the tensor corresponds to *days*, *hour of the day*, and *given time sequences*, respectively. For example, if the time sequences are hourly samples (24 points per day) for 7 days, then the dimensions of the first

and second modes become 7 (days) and 24 (samples per day) respectively.

Step 2: X2B (Convert the data into BIG domain): Then we convert the data tensor ($\mathcal{X}$) to BIG domain ($\mathcal{B}_a$) by fitting the data to BIG model as described in "*X2B*" in Figure 4.2. For fitting the BIG parameters, as we cannot fit the BIG parameters to a single point, we consider multiple points before and after the current timetick by apply a moving Gaussian filter to a window size of N_w, standard deviation σ to the data.

Function *fitBIG* takes in $[I_r^{in}, I_i^{in}, V_r^{in}, V_i^{in}]$ and fits the BIG models in equations (4.1) and (4.2) to the given window of data, using weighted least squares fitting, using this Gaussian filter as weights. Then we construct a new tensor $\mathcal{B}_a$ consisting of BIG parameters.

We next combine these fitted values into a tensor in the BIG domain whose first, second, and third mode of the tensor corresponds to *days*, *hour of the day*, and *BIG parameters*, respectively.

Step 3: Bextension (Extend the tensor for forecasting): After constructing a tensor from the given data in the BIG domain, we decompose the tensor using canonical polyadic alternating least squares (CP_ALS) algorithm [67, 68]: $\mathcal{B}_a = \sum_{r=1}^{R} \lambda_r \times \mathbf{u}_r \times \mathbf{v}_r \times \mathbf{w}_r$. Here, $\mathbf{u}, \mathbf{v}, \mathbf{w}$ corresponds to the hidden variables across multiple days, within a day, and among the BIG parameters. We will refer to these hidden variables as *long-term-concepts, daily-concepts, users-profile-concepts*, respectively.

After learning the tensor components, we extend the first mode of the tensor ($\mathbf{u}$, corresponding to *days*) for N_{fd} more points for forecasting - we conduct auto-regression on the $\mathbf{u}$ vector and forecast/extend it. We learn the AR parameters for P_{ar}^{th} order auto-regression $AR(P_{ar})$ using least squares on $\mathbf{u}$. Then we use AR parameters for the extension of the $\mathbf{u}$ vector to get an extended vector $\hat{\mathbf{u}}$: $\hat{\mathbf{u}}_r(N_d + f) = c_0 + \sum_{p=1}^{P_{ar}} c_p \hat{\mathbf{u}}_r(N_d + f - p)$

From the extended components $\hat{\mathbf{u}}$, we can reconstruct an *extended* tensor, $\mathcal{B}_b = \sum_{r=1}^{R} \lambda_r \times \hat{\mathbf{u}}_r \times \mathbf{v}_r \times \mathbf{w}_r$ with next N_{fd} days worth of data. A detailed steps are illustrated in Figure 4.2 "*Bextension*" part.

Step 4: B2Seq (Recover the data from BIG domain): Now that we have a extended tensor $\mathcal{B}_b$ on BIG domain, we need to recover the N_{fd} days of forecast data back into the original data domain. A detailed description of steps is illustrated in Figure 4.2 "*B2Seq*" part. We matricize the tensor into four matrices and reshape each matrix into a vector in row-wise manner. We apply the BIG equations (4.1), and (4.2) to the assumed voltage data and forecast corresponding $I_r(t), I_i(t)$ for the next $t = N + 1, \cdots, N + N_f$ steps.

POWERCAST-S POWERCAST-S is a variation of POWERCAST. In function **Bextension**, we can use any type of forecasting function in the place of *AR*.

We tried applying seasonal periodicity on the *long-term-concepts* ($\mathbf{u}$ vector), but on our dataset (which does not span over long enough period of time to capture the seasonal pattern) POWERCAST-S did not work well. Thus we recommend users to use plain POWERCAST unless: 1) you have a long enough history of the time sequences to capture the weekly pattern, and 2) you strongly believe that there is a seasonality in the data (weekly, monthly, seasonal, etc).

4.1.4 Experiments

In this section we conduct experiments on the real world data to answer the following research questions: **Q1: Interpretability**, **Q2: Forecasting**, and **Q3: Anomaly detection**.

Dataset description: We next give a brief explanation of the datasets we used for experiments. All datasets are 5-tuples of the form (V_r, V_i, I_r, I_i, t).

- **CMU data:** The voltage and current measurements were recorded for the Carnegie Mellon University (CMU) campus for 23 days, from July 29, 2016 to August 20, 2016. The time sequences were sampled every hour ($N_{dt} = 24$).
- **LBNL data:** This is from the Lawrence Berkeley National Laboratory (LBNL) Open μPMU project[1] [117]. It spans 3 months, with sampling rate of 120Hz; but we used only the interval (October 1, 2015 to October 8, 2015) and we down-sampled it to hourly samples ($N_{dt} = 24$). Moreover, we de-noised it using moving averages.

Experimental setup: For the parameters, we used $P_{ar} = 1$ for $AR(P_{ar})$ for extension of the $\mathbf{u}$ vector, $R = 2$ for the tensor decomposition, and $N_w = 5$, $\sigma = 0.5$ for the moving Gaussian filter. We used tensor tool box from [9] [8].

Baseline methods: As baseline methods, we conducted experiments using auto-regression (AR) and seasonal auto-regression (SAR) for comparison with our POWERCAST. For the baseline experiments, we assume that we do not have the domain knowledge on the power systems such as BIG model, etc. Thus we run AR and SAR methods on the given time sequences of I_r, I_i directly. The autoregression order for AR and SAR are determined via AIC criterion.

Q1: Interpretability

In this section we analyze the hidden variables in CMU data, and interpret the result using our domain knowledge.

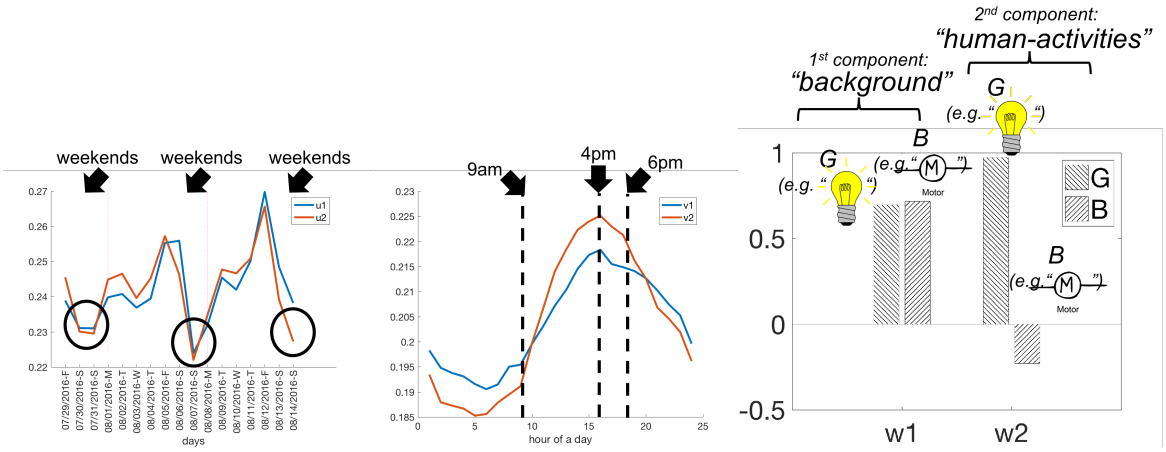
As a reminder, G can be interpreted as a component of electrical load that contributes to real power consumption (for e.g. *light bulbs*), while B can be interpreted as a component of electric load that contributes to reactive power (for e.g. lagging reactive power (+Q) is absorbed by the *motors* where leading reactive power (-Q) is supplied by the capacitor) In our analysis below, we include images of "*light bulbs*" and "*motors*" to represent one example of the sources of G and B for more intuitive understanding.

Observations 1 (Weekly pattern in Figure 4.3 (a))

- The *long-term-concepts* ($\mathbf{u}$ vector) shows weekly periodicity accounting for the activities across the days (drops during weekends)
- Both of the two components capture similar weekly pattern

Observations 2 (Daily pattern in Figure 4.3 (b))

[1] http://powerdata.lbl.gov/



(a) *long-term-concepts* (b) *daily-concepts* (c) *users-profile-concepts*

Figure 4.3: (a) **Weekly periodicity** in *long-term-concepts* ($\mathbf{u}$), (b) **Daily pattern** in *daily-concepts* ($\mathbf{v}$), (c) ***users-profile-concepts*** ($\mathbf{w}$), on CMU data

- The *daily-concepts* ($\mathbf{v}$ vector) captures daily activity patterns throughout the day, lowest during the night (midnight to 9am), and grows from 9am, when peaking start coming to campus, peaking at 4pm.
- The first component, $\mathbf{v}_1$, shows smoother varying pattern throughout the day. We can think of this component as the representation of the"*background*" activities, as it is insensitive to the dynamics of the human activities throughout the day.
- The second component, $\mathbf{v}_2$, shows more dynamic changes with 1) deeper drop during the night, 2) faster growth in early stage (logarithmic), and 3) rapid decay after the peak. This pattern closely follows the dynamics of the expected human activities throughout the day. We can think of this component as the representation of the "*human-activity*" factor.

Observations 3 (G and B factor analysis in Figure 4.3 (c))

- The *users-profile-concepts* ($\mathbf{w}$ vector) explains how much G and B account for each of the component.
- The first component - hidden variable (left of Figure 4.3(a)) accounts for both G (e.g"*light bulbs*") and B (e.g."*motors*") in a close to 50-50 ratio. Combined with our interpretation in the *daily-concepts* for plot (b), we can interpret the first component as the *background* component that explains the background activities of the G and B (e.g. *light bulbs* and *motors*) factors that are operated independently of the dynamics of the major human factors. (basic facilities such as air conditioning units, *light bulbs*, etc, that runs throughout the day to maintain the minimum required conditions in the buildings)
- The second component (right of Figure 4.3(a)) accounts dominantly for G (e.g."*light bulbs*"). Aligned with our interpretation of the second component as accounting for the "*human-activity*" factor, this tells us that G (e.g. *light bulbs*) is more dependent on the dynamics of the daily activities compared to B. This is reasonable since addition of one more person on campus may result in turning on 10 G-related facilities (e.g. *light bulbs*) while it merely affects B-related facilities (e.g. *motors*- air-conditioner, heaters, etc) that

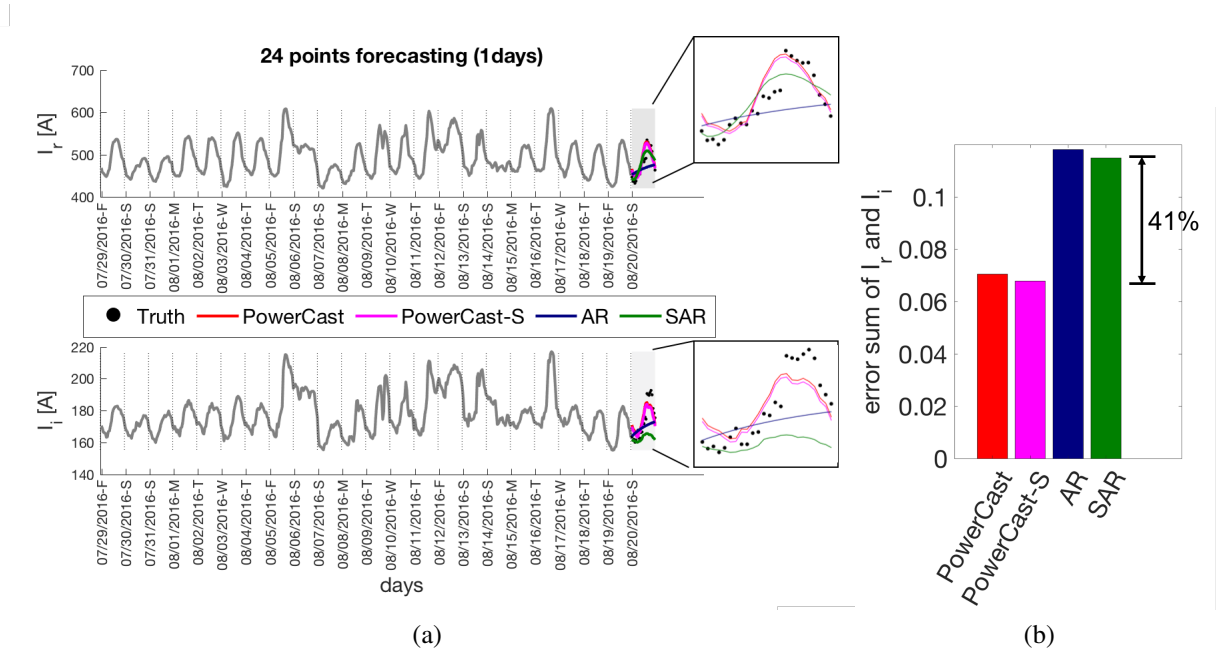
are operated to maintain the background conditions regardless of the human factors.

Q2: Forecasting

In this section we demonstrate forecasting results by POWERCAST including various *what-if* scenarios.

Multi-step forecasting We start by showing multi-step ahead forecasting by POWERCAST in comparison with other competitors on two different real world datasets.

- **CMU data**: Comparisons on the forecasting by POWERCAST and competitors are shown for $N_f = 24$ steps ($N_{fd} = 1$-day) (Figure 4.4).



(a) (b)

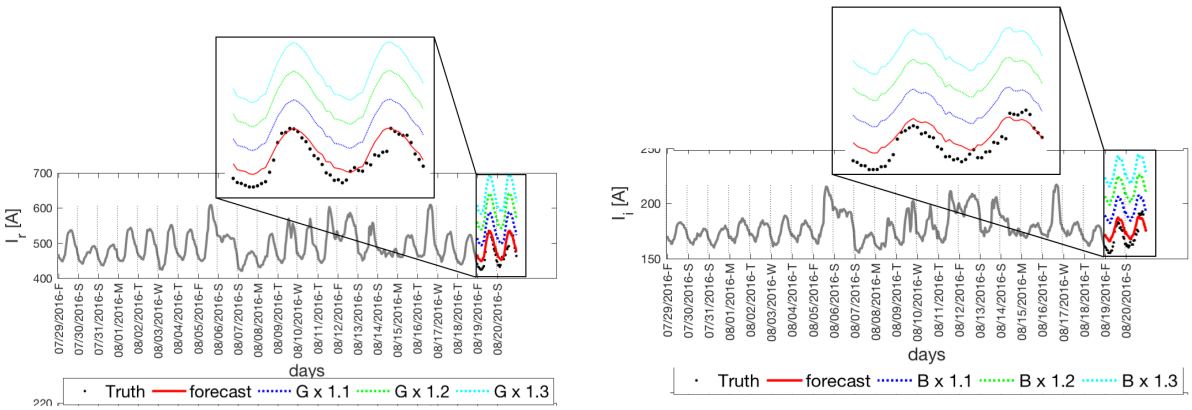
Figure 4.4: **POWERCAST forecasts 24 steps (1-day) accurately on CMU data** (a) POWERCAST (*red*) and POWERCAST-S (*pink*) forecasts 24 steps (I_r on the top row, and I_i on the bottom row) more accurately compared to the competitors (AR: *blue* and SAR: *green*). (b) RMSE comparison of the methods. POWERCAST achieves **41%** error reduction.

In (a), we observe that POWERCAST and POWERCAST-S are able to demonstrate the daily periodic pattern while competitors fails to correctly demonstrate daily periodicity. In (b), a quantitative comparison is given by normalized root mean square error (RMSE) sum of I_r, I_i forecast results in Amperes unit. The RMSE is computed as follows: $RMSE(x, \hat{x}) = \sqrt{\frac{\sum_{t=1}^{N_f}(x(t)-\hat{x}(t))^2)}{\sum_{t=1}^{N_f}(x(t))^2}}$

- **LBNL data**:
  Figure 4.1 shows a similar quantitative comparison on the LBNL dataset, where POWERCAST also outperforms its competitors.

What-if scenario If we know that the number of people on campus will increase by 10% tomorrow due to a big event, how much more power do we expect? Naive forecasting methods cannot tell us the future demand under various scenarios.

In this section, we illustrate how POWERCAST can handle forecasting under various *what-if* scenarios by adjusting G, B accordingly. We created three scenarios, assuming that we will have $\{10, 20, 30\}\%$ more activities on campus for next 2 days (thus increased values for G and B); and performed forecasting of the power demand under our scenarios. In Figure 4.5, 48 step (2-day) ahead forecasting on CMU data under our scenarios are shown for (a) real current demand and (b) imaginary current demand. We see that I_r and I_i increase, according to the "BIG" model (see (a) and (b) respectively), allowing us to plan ahead. (Red, blue, green, and cyan, correspond to $\{0, 10, 20, 30\}\%$ increases).



(a) (b)

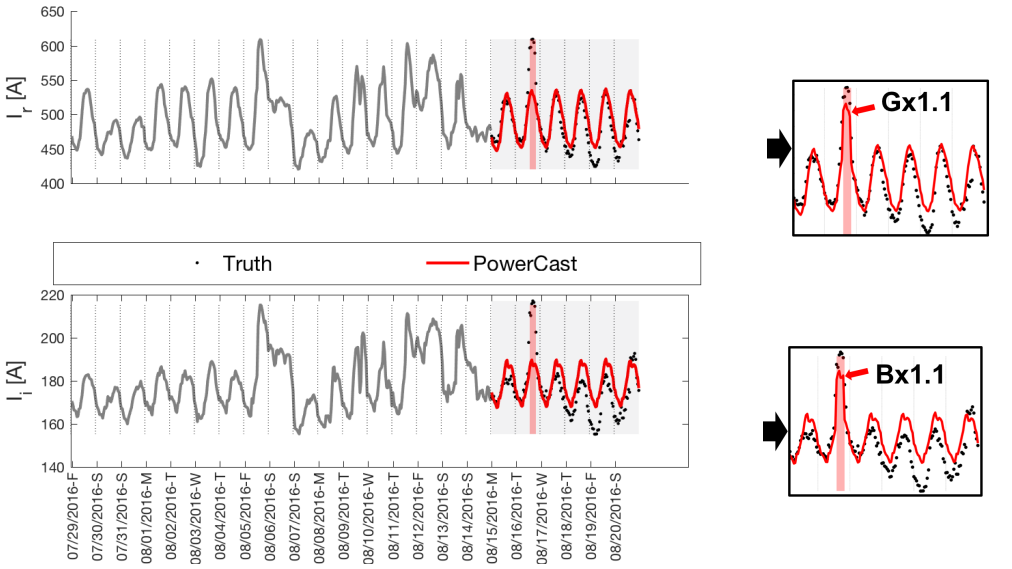
Figure 4.5: **POWERCAST can handle forecasting under various what-if scenarios** on demands for (a) I_r and (b) I_i on CMU data

Q3: Anomaly detection

In Figure 4.6 (a) $N_f = 144$ step forecasting ($N_{fd} = 6$ days) on CMU data is shown in solid red along with the actual reading in black circles. We report the highest deviation from our POWERCAST forecast, with a red vertical line. This happened on Tuesday, August 16, which was in the middle of the 3-day new graduate student orientation, probably ≈1,000 people.

The natural follow up question is: Was the extra population the reason for this anomaly? The answer seems 'no', for two reasons: extra people would mainly boost I_r, that is 'G' (e.g. "*light bulbs*"), as our "*human-activity*" component showed in Figure 4.3, but we have spikes in both I_r and I_i; the second reason is that there was no spike during the other two days of orientation.

The only explanation is that something boosted the "*background*" component of Figure 4.3 (e.g., "*motors*", like elevators, heaters, air-conditioners), resulting in roughly equal boost to both I_r and I_i. Does reality corroborate the conjecture? The answer is '*yes*': the temperature on Aug. 16 spiked, to a scorching $88°$F ($\approx 30.5°C$), while the surrounding days was closer to $80°F$. This is exactly what we show in (b): POWERCAST shows that for a 10% increase in the component



(a) Anomaly detection

(b) Possible Explanation

Figure 4.6: **POWERCAST spots anomalies**: (a) August 16, 2017 (spotted, and marked with red stripe) was during the new graduate student orientation. (b) POWERCAST shows that a 10% increase in activities ('B' and 'G'), could explain this spike.

"*background*" (air-conditioners, etc), we get a 10% increase in both G (e.g. "*light bulbs*") and B (e.g. "*motors*"); the resulting answers correspond to the red line in (b), which is very close to the ground-truth black circles.

In short, POWERCAST can not only spot anomalies, but also give hints about their cause.

Scalability

In Figure 4.7, the wall clock time for POWERCAST to run on a dataset of different number of timeticks (N) is plotted in black solid dots, along with a linear line in blue. The time was measured for 24 step (1-day) forecast given past time sequences of length $N = 240 - 528$ timeticks ($N_d = 10 - 22$ days) on CMU data.

4.1.5 Conclusions

We proposed POWERCAST, a novel domain-aware method to mine, understand, and forecast the power demand of an enterprise (university, factory, neighborhood). Our contributions are as follows:

1. **Infusion of domain knowledge**: We carefully picked a successful electric-power model, "BIG", which is derived from first-principles, and allows for *intuitive interpretation* of the power system of interest.
2. **Forecasting/What-if**: Our domain-aware approach, coupled with the BIG model provides (a) *accurate* prediction in multi-step ahead forecasting with up to 59% lower error, and

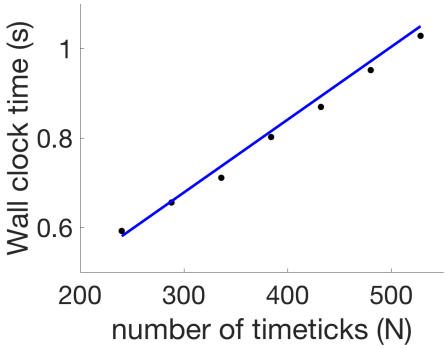


Figure 4.7: **POWERCAST scales linearly** with respect to the number of timeticks (N)

(b) answers *what-if* scenarios, like "what will be our power demands when 80% of our students leave campus for March-break"

3. **Anomaly detection**: POWERCAST can spot anomalies and explain as we show in section 4.1.4, Figure 4.6.

Reproducibility: Our code is open-sourced at: `www.cs.cmu.edu/~hyunahs/code/PowerCast.zip`, and the LBNL dataset is at: `powerdata.lbl.gov`

4.2 STREAMCAST: online mining of power grid time sequences [53]

In this section, we introduce STREAMCAST for mining power grid time sequences. STREAMCAST allows for online mining, making it possible to handle streams of time sequences. And by incorporating weather components into the algorithm, STREAMCAST improves forecasting performance over POWERCAST.

4.2.1 Introduction

The smart electrical grid is a set of technologies designed to improve the efficiency and security of power delivery. Estimates [5] suggest that reducing outages in the U.S. grid could save $\$49$ billion per year, reduce emissions by 12 to 18%, while improving efficiency could save an additional $\$20.4$ billion per year. A key part of achieving this goal is to use monitoring data to accurately model the behavior of the grid and forecast future load requirements, especially under extreme conditions such as changes in temperature, number of appliances in the grid, or voltage patterns. This allows the grid to remain reliable even under adverse conditions.

A major challenge is scalability - power systems data can be both high-volume and received in real time. This motivates us to develop fast methods that work in this online (or streaming) setting. Rather than operating on an entire dataset at once, online algorithms allow input that arrives over time as a continuous stream of data points. When each new data point is received, the algorithm updates itself - for our algorithm, each update requires constant time, and bounded memory (i.e. it does not need to remember the entire history of the time series).

Hence, our goal is an online algorithm for modelling and forecasting power consumption of a location. The input is a stream over time of real and imaginary voltage and current values:

Informal Problem 3 (Online Model Estimation)

- ***Given:*** *A continuous stream of values of real and imaginary current $(I_r(t), I_i(t))$, voltage $(V_r(t), V_i(t))$, and temperature $T(t)$, for $t = 1, 2, \cdots$*

- ***Estimate:*** *Time-varying parameters of a physics-based model of electrical load behavior that accurately explains the observed values.*

Our model is based on the circuit theoretic BIG model [62], which characterizes load in the electric grid by modeling its voltage sensitivities. Importantly, the use of physics-based models together with current and voltage state variables improves interpretability. For instance, real power consuming loads such as light bulbs can be interpreted as the contribution of the conductance (G) parameter, while susceptance (B), the reactive power component, represents the contributions of motors or capacitors in the grid.

The fitted model is used to forecast future values:

Informal Problem 4 (Multi-step Forecasting)

- ***Given:*** *values of current $(I_r(t), I_i(t))$, voltage $(V_r(t), V_i(t))$, and temperature $T(t)$ for $t = 1, \cdots, N$, and given temperature forecasts for the next N_f time steps (i.e. for $t = N + 1, \cdots, N + N_f$),*

- ***Forecast:*** *voltage and current for N_f time steps in the future; i.e. $(V_r(t), V_i(t))$ and $(I_r(t), I_i(t))$ for $t = N + 1, \cdots, N + N_f$.*

Due to our physics-based model, our algorithm can handle **what-if scenarios** in which the temperatures or voltages change, which is useful for future planning.

Informal Problem 5 (What-if Scenarios)

- ***Given:*** *current, voltage and temperature data, as above,*
- ***Forecast:*** *future values of voltage and current, under the condition that, e.g., temperature increases by $10°C$, and voltage levels increase by 5%.*

Our contributions are as follows:

1. **Domain knowledge infusion:** we propose a novel, *Temporal BIG* model that extends the physics-based BIG model, allowing it to capture trends, seasonality, and temperature effects.
2. **Forecasting:** our STREAMCAST algorithm forecasts multiple steps ahead and outperforms baselines in accuracy by 27% or more. STREAMCAST is online, requiring linear time and bounded memory.
3. **What-if scenarios and anomaly detection:** our approach accurately handles scenarios in which the voltage levels, temperature, or number of appliances change. We also use it to detect anomalies in a real dataset.

Reproducibility: our code is publicly available at `www.andrew.cmu.edu/user/bhooi/power.tar`.

4.2.2 Background and Related Work

Related Work

The BIG model for electrical load As we mentioned in section 4.1.2, the common approach for power grid modeling is *PQ* model [92]. However, a recent work BIG [62] showed that it can successfully describe the load behavior at a given time by a linear relationship between current and voltage.

Time series forecasting As we mentioned in section 4.1.2, commonly used textbook time-series forecasting methods include autoregression (AR)-based methods, including ARMA, ARIMA [16], seasonal ARIMA [16], vector autoregression (VAR) [50], and exponential smoothing (ETS) models [134], including Holt-Winters [134].

Contrast with existing literature Other than our earlier work POWERCAST [115] (4.1), all methods above do not use physics-based electrical models, and do not consider what-if scenarios, while our approach does both. Compared to POWERCAST [115], our approach in STREAMCAST (which is completely different from our tensor-based approach) additionally allows for weather data, is an online algorithm, and produces confidence intervals.

Table 4.3: **STREAMCAST captures the listed properties**. AR++ refers to ARIMA, seasonal ARIMA etc.

Properties	AR++[16]	Kalman/LDS[63]	Weather-based [109, 142]	HMM++[74]	PowerCast [115] (4.1)	STREAMCAST
Forecasting	✓	✓	✓	✓	✓	✔
Seasonal patterns	✓	?	✓		✓	✔
Physics-based model					✓	✔
Weather-based			✓			✔
Online algorithm		✓				✔
Confidence intervals	✓	✓		✓		✔
What-if scenarios					✓	✔

Background

BIG model (Here we repeat the description in section 4.1.2 for equations 4.1, 4.2 for the readers' convenience.) The BIG model [62] models the current as a linear function of the voltage, parameterized by the BIG parameters: susceptance (B), conductance (G), and a current offset (α_r, α_i). G can be interpreted as the component contributing to real power consuming loads (e.g. due to light-bulbs), while B can be interpreted as the contribution of the reactive power component (e.g. due to motors or capacitors):

$$\begin{aligned} I_r(t) &= G \cdot V_r(t) - B \cdot V_i(t) + \alpha_r + \text{noise} \\ I_i(t) &= B \cdot V_r(t) + G \cdot V_i(t) + \alpha_i + \text{noise} \end{aligned} \tag{4.3}$$

Holt-Winters model The Holt-Winters model [134] models a univariate time series $x(t)$ with seasonal structure. The key idea is to model $x(t)$ as the sum of a nonseasonal or *level* component $l(t)$ and a *seasonal* component $s(t)$. The level component changes according to smooth trends, while the seasonal component is approximately periodic with period m (e.g. $m = 24$ for hourly data with daily seasonality). Smooth trends over time are modelled by a linear trend $b(t)$, representing the rate of change of $l(t)$. Full details can be found in [134].

4.2.3 Method

Proposed Model

Table 4.4 shows the symbols used in this section.

Table 4.4: Symbols and definitions

Symbols	Definitions
N	Number of time ticks in time sequences
I_r, I_i, V_r, V_i	Real and imaginary current and voltage
B, G, α_r, α_i	BIG parameters (susceptance, conductance, offset to I_r and I_i)
$\theta(t)$	Parameter vector $(B, G, \alpha_r, \alpha_i)$ at time t
$y(t), X(t)$	BIG in linear model form; see Eq. (4.5)
$\theta_L(t)$	Nonseasonal part of $\theta(t)$
$\theta_S(t)$	Seasonal part of $\theta(t)$
$\theta_T(t)$	Trend in $\theta_L(t)$
$\theta_W(t)$	Weather part of $\theta(t)$
w	Weather coefficients
T_0	Temperature threshold
N_{init}	Initialization period

Proposed Dynamic BIG Model As in section 4.1.2, we use BIG model. Consider the static BIG model in Eq. (4.3). Its parameters B, G, α_r, α_r can be interpreted as types of load; but in practice, these should change over time as appliances are switched on and off, or usage levels change. How do we add temporal structure to this model? A natural step is to replace B, G, α_r, α_r by time series $B(t), G(t), \alpha_r(t), \alpha_r(t)$. Eq. (4.3) becomes:

$$\begin{aligned} I_r(t) &= G(t) \cdot V_r(t) - B(t) \cdot V_i(t) + \alpha_r(t) + \text{noise} \\ I_i(t) &= B(t) \cdot V_r(t) + G(t) \cdot V_i(t) + \alpha_i(t) + \text{noise} \end{aligned} \tag{4.4}$$

For notational simplicity, rewrite this equivalently as:

$$\underbrace{\begin{pmatrix} I_r(t) \\ I_i(t) \end{pmatrix}}_{y(t)} = \underbrace{\begin{pmatrix} V_r(t) & -V_i(t) & 1 & 0 \\ V_i(t) & V_r(t) & 0 & 1 \end{pmatrix}}_{X(t)} \underbrace{\begin{pmatrix} G(t) \\ B(t) \\ \alpha_r(t) \\ \alpha_i(t) \end{pmatrix}}_{\theta(t)} + \text{noise} \tag{4.5}$$

Now, changes in usage (e.g. appliances switching on and off) correspond to changes in $\theta(t)$. What temporal patterns do we need to capture? Intuitively, some appliances follow daily seasonality (e.g. lights used during working hours), while others follow slow-moving trends, e.g. a gradual increase in load due to population growth. Hence, like in Holt-Winters, we decompose $\theta(t)$ into a nonseasonal **level** part $\theta_L(t)$, and a **seasonal** part $\theta_S(t)$; hence, $\theta(t) = \theta_L(t) + \theta_S(t)$. Here $\theta_L(t)$ changes according to smooth trends, while $\theta_S(t)$ has seasonal patterns, with period m.

To model smooth trends (e.g. population growth), we additionally define a **trend** term $\theta_T(t)$,

which approximates the change in $\theta_L(t)$ from one time tick to the next; i.e. $\theta_L(t) \approx \theta_L(t-1) + \theta_T(t)$. Then, our assumptions under this model are that:

A1: $y(t) \approx X(t)(\theta_L(t) + \theta_S(t))$ (Low noise; see (4.5))

A2: $\theta_L(t) \approx \theta_L(t-1) + \theta_T(t)$ (Level moves wrt. trend)

A3: $\theta_T(t) \approx \theta_T(t-1)$ (Trends change smoothly)

A4: $\theta_S(t) \approx \theta_S(t-m)$ (Seasonality)

We will formalize these as soft constraints in our optimization objective in Section 4.2.3.

Dynamic BIG with Temperature Model Let $T(t)$ denote the temperature at time t. When temperature increases above a threshold, electricity demand tends to increase due to the use of air conditioning. To capture this, we introduce **weather coefficients** w and a **temperature threshold** T_0: temperature over the threshold linearly adds to a weather component $\theta_W(t)$. Hence, we replace assumption A1 with:

A1': $y(t) \approx X(t)(\theta_L(t) + \theta_S(t) + \theta_W(t))$, where
$\theta_W(t) = w \cdot \max(0, T(t) - T_0)$

The $\max$ function ensures that only temperatures above the threshold contribute to the weather component.

Proposed Optimization Objective

We now define our optimization objective based on our assumptions A1' to A4. All norms are L2 norms, for later computational simplicity.

$$\begin{aligned}
\mathbf{L} &= \mathbf{L}_1 + \lambda L_2 + \mu \mathbf{L}_3 + \nu \mathbf{L}_4, \text{ where:} \\
\mathbf{L}_1 &= \sum_{t=1}^{N} \|y(t) - X(t)(\theta_L(t) + \theta_S(t) + \theta_W(t))\|^2 \\
\mathbf{L}_2 &= \sum_{t=2}^{N} \|\theta_L(t) - \theta_L(t-1) - \theta_T(t)\|^2 \\
\mathbf{L}_3 &= \sum_{t=1}^{N} \|\theta_T(t) - \theta_T(t-1)\|^2 \\
\mathbf{L}_4 &= \sum_{t=m+1}^{N} \|\theta_S(t) - \theta_S(t-m)\|^2
\end{aligned} \tag{4.6}$$

Here $\lambda, \mu, \nu > 0$ are hyperparameters (which we end up tuning indirectly rather than directly; see Section 4.2.3 'Proposed STREAMCAST Algorithm '). The problem to solve is then:

$$\underset{\theta_L, \theta_T, \theta_S, w, T_0}{\text{minimize}} \ \mathbf{L} \tag{4.7}$$

Proposed STREAMCAST Algorithm

How do we approximately minimize $\mathbf{L}$ in an efficient and online manner?

Overview In this section, we outline our proposed STREAMCAST. STREAMCAST has two steps: 1) an offline step TEMPFIT, which takes a subset of N_{init} data points, and fits w and T_0; 2) an online 'extension' stage STREAMFIT: upon receiving each new data point at time t, this updates $\theta_L, \theta_T, \theta_S$ according to Eq. (4.9).

Algorithm 7: STREAMCAST

input: Streams V_r, V_i, I_r, I_i
$w, T_0 \leftarrow$ TEMPFIT$(V_r(1\colon N_{\text{init}}), V_i(1\colon N_{\text{init}})$;
$\quad I_r(1\colon N_{\text{init}}), I_i(1\colon N_{\text{init}}))$;
while *input at time t is received:* **do**
  Update $\theta_L(t), \theta_T(t), \theta_S(t)$ using STREAMFIT;

We will describe STREAMFIT, and TEMPFIT in the following paragraphs.

Streaming Optimization (STREAMFIT) STREAMFIT estimates θ_L, θ_T and θ_S for fixed temperature parameters w, T_0, by minimizing our objective, Eq. (4.7). Note that Eq. (4.7) could in theory be minimized using least squares; however, this is not online, and is also much too slow as it contains $O(N)$ unknowns for N time points, which would then take around $O(N^3)$ time. We would like a linear-time algorithm with constant update time per data point.

Our approach is to split up the objective over t, and take gradient update steps with respect to the term for $t = 1, 2, \cdots$ successively. At time t, we take a gradient step with respect to only the terms of $\mathbf{L}$ corresponding to time t. This allows the fitted parameters to 'track' the true values over time as we perform gradient updates. Meanwhile, each update is highly efficient as it only involves a single term of the objective function. Assume that we have fit $\theta_L, \theta_T, \theta_S$ up to time $t-1$ and are fitting them at time t. The component of $\mathbf{L}$ for time t is:

$$\begin{aligned}\mathbf{L}(t) = &\|y(t) - X(t)(\theta_L(t) + \theta_S(t) + \theta_W(t))\|^2 \\ &+ \lambda\|\theta_L(t) - \theta_L(t-1) - \theta_T(t)\|^2 \\ &+ \mu\|\theta_T(t) - \theta_T(t-1)\|^2 + \nu\|\theta_S(t) - \theta_S(t-m)\|^2\end{aligned} \tag{4.8}$$

To do gradient descent at time t, we start from the previously learned values and take a gradient step. Hence, we start by assuming the previous trend ($\theta_T(t) = \theta_T(t-1)$) and seasonality ($\theta_S(t) = \theta_S(t-m)$), while for the level we follow a 'default extrapolation' of starting at the previous level and following the previous trend ($\theta_L(t) = \theta_L(t-1) + \theta_T(t-1)$).

Next, we want to move in the gradient direction with respect to minimizing $\mathbf{L}$. Computing the gradients of (4.8) is straightforward and results in the update equations: letting $\hat{y}(t) = X(t)(\theta_L(t-1) + \theta_T(t-1) + \theta_S(t-m) + \theta_W(t))$,

$$\begin{aligned}\theta_L(t) &= \underbrace{\theta_L(t-1) + \theta_T(t-1)}_{\text{default extrapolation}} + \alpha \underbrace{X(t)^T(y(t) - \hat{y}(t))}_{\text{gradient update}} \\ \theta_T(t) &= \underbrace{\theta_T(t-1)}_{\text{previous trend}} + \beta \underbrace{(\theta_L(t) - \theta_L(t-1) - \theta_T(t-1))}_{\text{gradient update}} \\ \theta_S(t) &= \underbrace{\theta_S(t-m)}_{\text{previous seasonality}} + \gamma \underbrace{X(t)^T(y(t) - \hat{y}(t))}_{\text{gradient update}}\end{aligned} \tag{4.9}$$

$\alpha, \beta, \gamma > 0$ are 'learning-rate' tuning parameters that replace λ, μ, ν. We will consider how to set these, and how to initialize θ_L, θ_T and θ_S, in the following paragraph.

Temperature Model Optimization (TEMPFIT) In TEMPFIT, we optimize the temperature coefficient w and threshold T_0 using an alternating approach. First, fixing w and T_0, we solve for θ_L, θ_T and θ_S to minimize $\mathbf{L}$. We then do the reverse (fixing θ_L, θ_T and θ_S and solving for w and T_0), until convergence, as shown in Algorithm 8. The former step of solving for θ_L, θ_T and θ_S given w and T_0 is done using STREAMFIT.

Algorithm 8: TEMPFIT

input: V_r, V_i, I_r, I_i
while *not converged* **do**
  Solve $\theta_L, \theta_T, \theta_S$ using Eq. (4.9) (STREAMFIT);
  Solve w, T_0 by minimizing Eq. (4.10);

It remains to solve for w and T_0 for fixed $\theta_L, \theta_T, \theta_S$. Define $r(t) = y(t) - X(t)(\theta_L(t) + \theta_S(t))$. Then, since changing w and T_0 only affects the $\mathbf{L}_1$ loss term, minimizing $\mathbf{L}$ is equivalent to minimizing:

$$\mathbf{L}' = \sum_{t=1}^{N} \|r(t) - X(t) \cdot w \cdot \max(0, T(t) - T_0)\|^2 \tag{4.10}$$

Minimizing over w is a straightforward least squares problem. Minimizing over T_0 is less straightforward since $\max(0, T(t) - T_0)$ is nonlinear in T_0. However, temperature is typically given to 0 or 1 decimal of precision (roughly the precision of most thermometers), and varies within a fairly narrow range, e.g. $0°C$ to $35°C$. Hence, it suffices to try all thresholds between $\min_t T(t)$ and $\max_t T(t)$ in intervals of $0.1°C$, and select among these to minimize $\mathbf{L}'$.

To complete our algorithm, we need to explain how to choose α, β, γ, and initial values for $\theta_L(t), \theta_T(t), \theta_S(t)$. We select the former using nonlinear optimization (Levenberg-Marquadt algorithm [81]) of $\mathbf{L}$. For the latter, it can be verified that the objective $\mathbf{L}$ is a quadratic in these initial values, so we solve for them using least squares. We do these in Line 2 of TEMPFIT, then fix these values subsequently for the rest of the algorithm (Line 2-4 of STREAMCAST).

Forecasting Step (FORECAST) Given fitted model parameters up to time N, how do we forecast future currents $\hat{I}_r(t), \hat{I}_i(t)$, for $t = N+1, \cdots, N+N_f$?

We first forecast voltages $\hat{V}_r(t), \hat{V}_i(t)$, $t = N+1, \cdots, N+N_f$ using standard univariate Holt-Winters. Then, for $k = 1, 2, \cdots$ we forecast $\theta(N+k)$ as the sum of last estimated level $\theta_L(N)$, trend $\theta_T(N)$, last estimated seasonality at the corresponding position $\theta_S(N - m + 1 + ((k-1) \bmod m))$, where $\bmod$ is the modulo function, and temperature component $w \cdot \max(0, T(N+k) - T_0)$. Finally, we forecast $\hat{I}_r(t), \hat{I}_i(t)$ using the BIG model, Eq. (4.4).

Extensions **Anomaly Detection** To detect anomalies, we compute an anomaly score at each time. Intuitively, the larger the error between the fitted and actual values, the more anomalous

a point is. The fitted values $\hat{I}_r(t), \hat{I}_i(t)$ are obtained by plugging the learned parameters into the BIG equations, Eq. (4.4). The real and imaginary errors are then $E_r(t) = I_r(t) - \hat{I}_r(t)$, and $E_i(t) = I_i(t) - \hat{I}_i(t)$. The anomalousness at time t is then the sum of real and imaginary errors at time t, each in units of inter-quartile ranges (IQRs[2]):

$$\text{anomalousness}(t) = \frac{E_r(t)}{\text{IQR}(E_r(1\colon N))} + \frac{E_i(t)}{\text{IQR}(E_i(1\colon N))}$$

While any threshold may then be used, we can follow common practice of designating deviation of $\geq 2 \times IQR$ as outliers, resulting in a threshold of 4 for our anomalousness score. In Section 4.2.4 we show a clear anomaly found in the LBNL dataset.

Confidence Intervals Confidence intervals allow us to provide lower and upper bounds that contain future values, with e.g. 95% confidence. We use the past distribution of residuals, $I_r(t) - \hat{I}_r(t)$ and $I_i(t) - \hat{I}_i(t)$, as an estimate of residuals in the future. Thus, we sample a 'possible future' by repeatedly sampling from this distribution of past residuals, and treating the sampled values as residuals at time $N + 1$, repeating this process for $N + 2$, and so on. We sample 1000 possible futures in this way. To generate $(1 - \alpha)$ confidence intervals, we use the empirical $\alpha/2$ and $(1 - \alpha/2)$-quantiles of these possible futures. The results are shown in Figure 4.13.

4.2.4 Experiments

We design experiments to answer the questions:

- **Q1. Forecasting accuracy:** how accurately does STREAMCAST forecast, based on real data?
- **Q2. Scalability:** how does the algorithm scale with the data size?
- **Q3. What-if scenarios:** does STREAMCAST give accurate results under what-if scenarios, and detect real anomalies in real data?

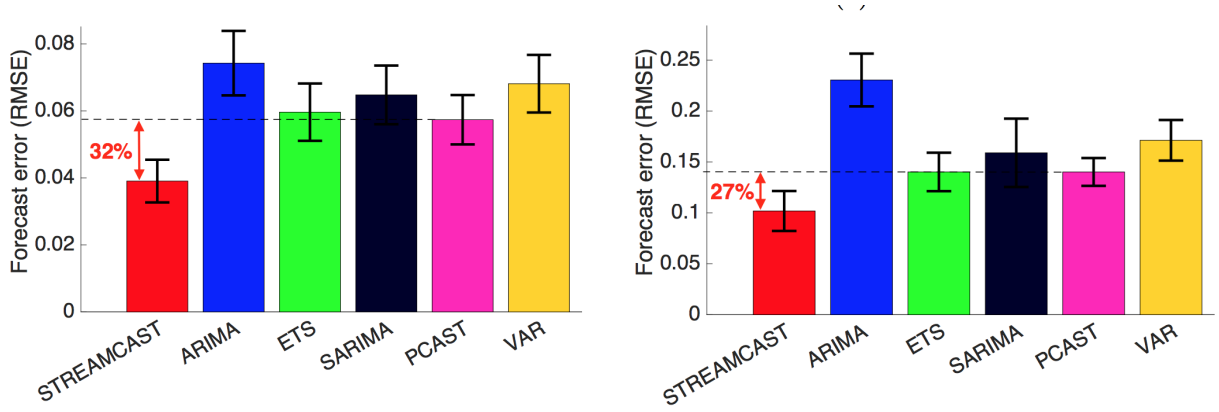
Our code and links to datasets are publicly available at `www.andrew.cmu.edu/user/bhooi/power.tar`. Experiments were done on a 2.4 GHz Intel Core i5 Macbook Pro, 16 GB RAM running OS X 10.11.2. We set N_{init} to $10m$ (i.e. 10 days) in our experiments.

Data

We use the following two datasets:

- **CMU data:** ($N = 648$) hourly voltage and current for the Carnegie Mellon University (CMU) campus for 23 days, from July 29, 2016 to August 20, 2016. Voltage angle is unavailable for this data, so here V_r is the voltage magnitude and $V_i = 0$.
- **LBNL data:** ($N = 3168$) from the Lawrence Berkeley National Laboratory (LBNL) Open μPMU project [117], from October 1, 2015 to October 11, 2015. The data is originally at 120Hz, but we downsample it to one sample every 5 minutes (where each new data point is the mean of the raw data within those 5 minutes).

[2]The IQR is the difference between 75% and 25% quartiles, used as a more robust measure of spread compared to standard deviation.[125]



(a) CMU data

(b) LBNL data

Figure 4.8: **StreamCast forecasts accurately:** it has at least 27% lower forecasting error than baselines. Error bars show 1 standard deviation.

Q1. Forecasting accuracy

Baselines our baselines are ARIMA [16], Holt-Winters (ETS) [21], seasonal ARIMA [16], PowerCast (PCAST) [115] (a recent tensor-based power grid forecasting approach), and vector autoregression (VAR), which uses temperature data and the voltage time sequences as input. Following standard practice, the ARIMA, SARIMA, and VAR orders are selected using AIC (Akaike information criterion) [57], and the Holt-Winters hyperparameters are selected using nonlinear optimization. For PowerCast, we follow the original work in setting $N_w = 5, \sigma = 0.5$.

Experimental setup In each trial, an algorithm is given the data for the first N days and forecasts I_r and I_i for each time point of the $(N+1)$th day, where we average each algorithm's accuracy over $N = 11, 12, \cdots, 20$ for CMU and $N = 4, 5, \cdots, 8$ for LBNL. The forecasts are compared with the true values using normalized RMSE: $\text{RMSE}(x, \hat{x}) = \sqrt{\|x - \hat{x}\|_2^2 / \|x\|_2^2}$, where $x = [I_r \;\; I_i]$ contains I_r and I_i values stacked into a single vector.

Results Figure 4.8 shows that StreamCast outperforms the baselines in both datasets, with at least 27% lower RMSE. The tensor-decomposition based PCAST, which is also physics-based, is the second-best performer, suggesting that physics-based models may be beneficial for forecasting.

Q2. Scalability

We run StreamCast on a version of our CMU dataset, duplicated repeatedly so as to produce larger datasets. We run StreamCast on time series of sizes as plotted on the x-axis of Figure 4.9, from around 1 million to around 40 million. The plot is parallel to the diagonal, indicating linear growth.

StreamCast takes less than 4 minutes for the trial of size 40 million, making it scalable to large datasets.

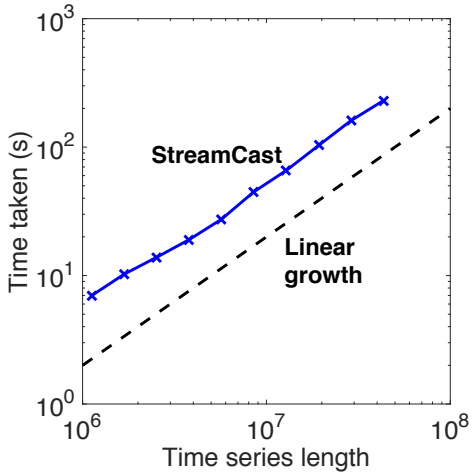


Figure 4.9: **STREAMCAST is fast and scales linearly:** growth parallel to the diagonal indicates linear growth.

Q3. What-if scenarios

Changing temperature and number of appliances: how can we forecast under the scenario that temperature increases by $10°C$, and number of appliances increases by 20%? Such scenarios are useful for future planning, but standard forecasting methods cannot handle them. The BIG parameter G represents the contribution of the conductive load component (e.g. light-bulbs) and B as the contribution of the reactive load component (e.g. motors), so we examine the results of increasing either G, B, or temperature, and plot how the forecasts change under each scenario.

The results are shown in Figure 4.10. The left plots are for increasing G; the center plots for changing B, and the right plots for increasing temperature. Upper plots are for I_r while lower plots are for I_i. In each plot, the colored lines correspond to different amounts of increase: e.g. $1.1 \times G$ means that the amount of reactive load on campus increased by 10%. The results show that: 1) when G is increased, only I_r increases, but not I_i; 2) when B is increased, only I_i increases, but not I_r; 3) when temperature increases, both I_r and I_i increase. All three results are intuitive, given that in the CMU dataset we have $V_i = 0$; it can be verified from (4.4) that in this case I_r should be influenced by changes in G, but not by changes in B.

Changing voltage An important goal in practice is to ensure that the system can make accurate predictions under changes to voltage levels. To test our model, we use an electrical system simulator, SUGAR [92]. The simulator uses physics-based models for different electrical equipment: we specify 10 motors and additional load with maximum resistance 50Ω, varying sinusoidally with daily periodicity. SUGAR generates realistic time series currents for an electrical system given specified input voltages V_r, V_i. In order to have a realistic voltage series, our input voltages are the voltage series from our CMU dataset. We obtain currents (I_r, I_i) as outputs from this simulation.

To test STREAMCAST, we fit it on (V_r, V_i, I_r, I_i), but then evaluate its RMSE on a different (V_r', V_i', I_r', I_i') in which V_r and V_i are defined in one of two ways: 1) **Increase:** $V_r' = 1.05 \times V_r$, $V_i' = 1.05 \times V_i$. 2) **Decrease:** $V_r' = 0.95 \times V_r$, $V_i' = 0.95 \times V_i$. We then obtain I_r' and I_i' from

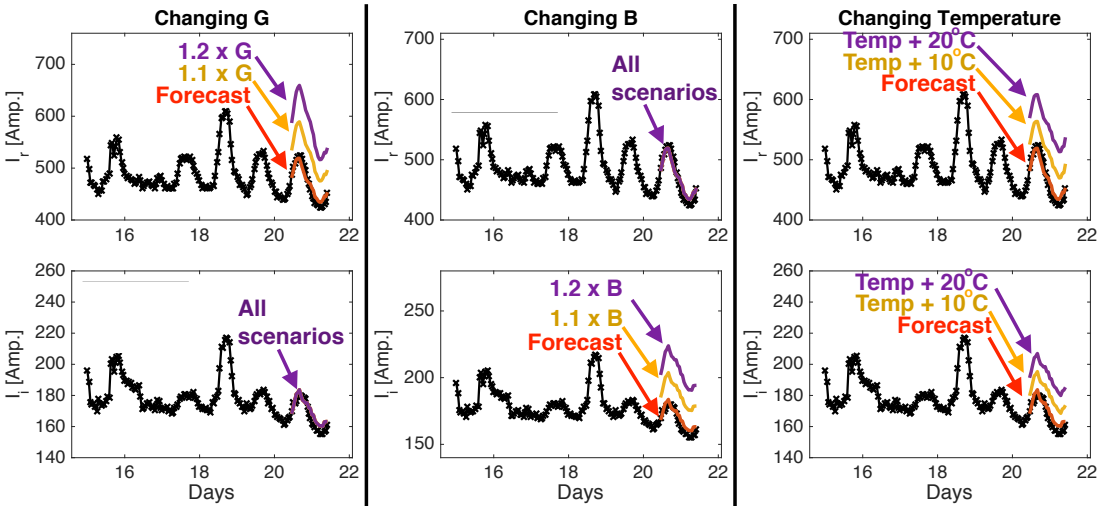


Figure 4.10: **STREAMCAST can handle forecasting under what-if scenarios:** the plots show the results of 1) increasing G; 2) increasing B; and 3) increasing temperature.

the same electrical simulation under voltages V_r' and V_i'.

Standard forecasting methods would not work as baselines, since an electrical model is needed to accurately predict what happens to I when V changes. Hence, we use the following baselines: 1) PQ: this is the same as our STREAMCAST approach, but substituting the BIG model with the more common PQ electrical model [92]. 2) PowerCast [115] as in the previous section; 3) Window k: this fits the *static* BIG model (Section 4.2.2) to short time windows of size k, where $k = 2, 4, 8, 16$.

Figure 4.11 shows the fit of STREAMCAST, PQ and Window4 against the true values (the remaining methods are not plotted for visibility, but their RMSE is in Table 4.5), and Table 4.5 shows the RMSE of each method against the true values. STREAMCAST outperforms the baselines by a clear margin. Because PQ is a constant-power model, increases in voltage magnitude necessarily lead to the model predicting a decrease in current of a similar magnitude. However, in practice, both voltage and current can change in the same direction, in which case the PQ model makes the wrong qualitative predictions. The Window$k$ baselines tend to overfit to near-term behavior due to their use of windows; this explains the high variance over time in Figure 4.11.

Anomaly Detection Section 4.2.3 explains how to detect anomalies under our method; Figure 4.12 shows the results on the LBNL dataset, where our method outputs a plot ('anomalousness') of the anomaly score over time. There is a large spike in anomalousness around day 2; note that the anomaly is not at all visible from only the current time series. In the voltage time series, however, we find a 2-second period of quick oscillations between negative and positive values exactly at the time of the anomaly. Follow-up analysis reveals that the oscillation is likely an error in the measuring device: the complex angle of voltage is synchronized with the rest of the system via a GPS signal, and in case of loss of this GPS signal, we may observe oscillations such as those in Figure 4.12, which explains why the voltage magnitude remains stable while the angle oscillates.

Test case	StreamCast	PQ	PowerCast	Window2	Window4	Window8	Window16
Increase	**0.19**	7.13	5.26	11.18	4.94	4.95	4.72
Decrease	**0.27**	7.69	5.66	9.46	5.78	5.56	4.58

Table 4.5: **StreamCast is accurate even under different voltage levels:** here the methods are tested on what-if scenarios with different voltage levels. Bold underline shows the best performer. Error values are given as **percentage** RMSE (i.e. RMSE $\times 100$).

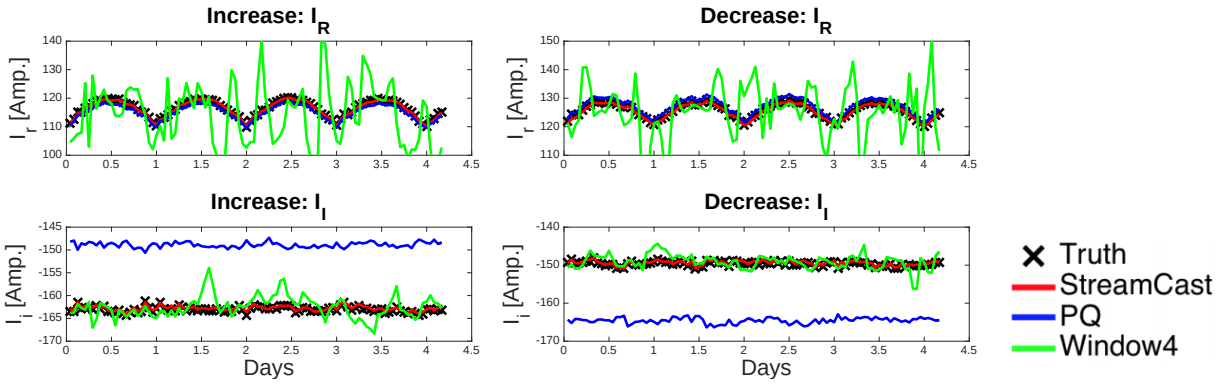


Figure 4.11: **StreamCast accurately responds to changes in voltage:** forecasts of I_r and I_i by each method vs. true values, when voltage was increased or decreased by 5%. StreamCast fits the data more accurately then baselines. RMSE results (with additional baseline methods) are in Table 4.5.

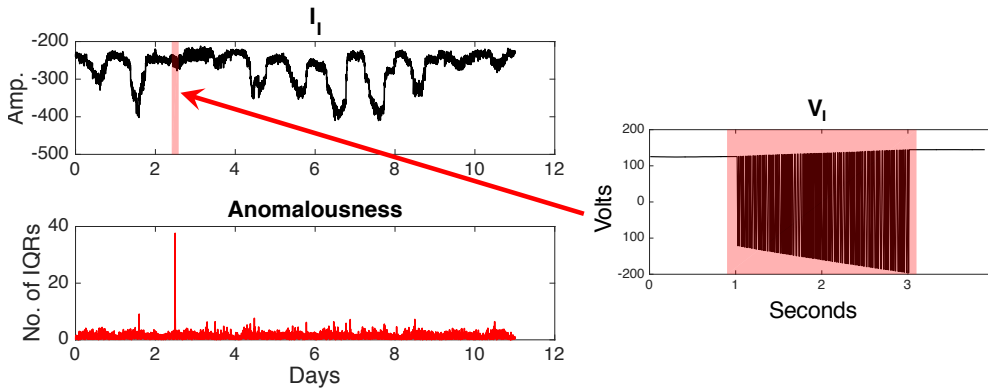


Figure 4.12: **STREAMCAST detects an anomaly in the LBNL dataset**. It corresponds to a 2-second period where voltage rapidly oscillates between negative and positive values.

Confidence Intervals Confidence intervals are useful for obtaining ranges of predictions: e.g. when monitoring the grid. Section 4.2.3 explains how we compute them. 95% and 99% confidence intervals on our CMU dataset are shown in Figure 4.13.

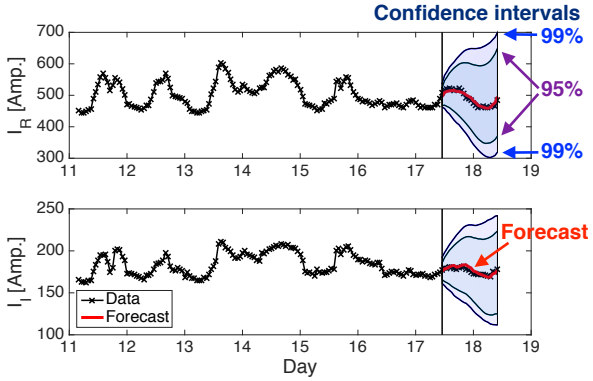


Figure 4.13: **STREAMCAST provides confidence intervals**.

4.2.5 Conclusions

Our contributions are as follows:

1. **Domain knowledge infusion:** we propose a novel, *Temporal BIG* model that extends the physics-based BIG model, allowing it to capture changes over time, trends, seasonality, and temperature effects.
2. **Forecasting:** our STREAMCAST algorithm forecasts multiple steps ahead and outperforms baselines in accuracy. STREAMCAST is online, requiring linear time and bounded memory.
3. **What-if scenarios and anomaly detection:** our approach accurately handles scenarios in which the voltage levels, temperature, or number of appliances change. We also use it to detect anomalies in a real dataset. Finally, STREAMCAST provides confidence intervals for its forecasts, to assist in planning for various scenarios.

Reproducibility: our code is publicly available at `www.andrew.cmu.edu/user/bhooi/power.tar`.

Chapter 5

Aircraft data

5.1 A-MINE: coupled matrix-tensor factorization

How can we exploit side information, in addition to the multiple, co-evolving time sequences? In this section, we introduce A-MINE, a coupled matrix-tensor factorization based signal mining method for detecting anomalous subgroups of sensors with the aid of the side information on the alerts.

5.1.1 Introduction

How can we tell if some sensors display anomalous behaviors in aircraft systems? Can we map the sensors to the part of the aircraft?

Let us assume that sensor signals are recorded during every flight. So if we denote the number of sensors, number of timeticks, and the number of flights as n_s, n_t, n_f, then we have sensor readings in a tensor format $\mathcal{S} \in \mathbb{R}^{n_s \times n_t \times n_f}$. Additional to the sensor readings, let us assume that we are given alerts information from each flight, i.e. warnings or alerts triggered during the flight, which may be caused by the sensor anomalities. Our goal is to detect a subgroup of anomalous sensors with the help of the alert information. To summarize, our problem formulation is as follows:

Informal Problem Formulation:
- *Given:* Sensor recordings and the alerts counts for each flights
- *Detect:* anomalous subgroup of sensors

In order to tackle the mining tasks, we first need to understand the nature of the data, and design an algorithm. After exploring the dataset, we came up with some significant insights we can use to solve our problem. These are listed below:

Insights:
1. Although sensor values may differ depending on the flight condition, sensor-sensor correlation relationship does not change throughout the flight under normally operating condition.
2. Assume most of the sensors are normal, only a small subset of sensors behave anomaly.
3. Anomalous sensors can be held accountable for the types of alerts triggered for a flight.

We discuss our approach in section 5.1.3 in greater detail.

5.1.2 Background

Tensor Factorizations

Tensors are multidimensional arrays that generalize the concept of matrices. As a consequence, they are a popular choice in various applications including representing time-evolving relations, such as Facebook interactions [95], sensor networks [120], EEG data for detecting the origin of epilepsy seizures [1], fMRI analysis [129], image classification [122], or heterogeneous graph analysis [110]. When properly applied, tensor factorizations identify the underlying low-dimensional latent structure of the data. The latent factors are then used to identify anomalies, to estimate missing values or to understand how the data was generated in the first place. The PARAFAC [51] (also called CP) decomposition is one of the most popular among the many tensor factorizations flavors [67], as it factorizes a tensor into a sum of rank-1 tensors. In three modes, the problem is usually framed as finding factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ that minimize the squared error between $\mathcal{X}$ and the reconstructed tensor: $\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \left\| \mathcal{X} - \sum_f \mathbf{a_f} \circ \mathbf{b_f} \circ \mathbf{c_f} \right\|_F^2$.

Coupled Factorizations

We are often interested in analyzing real-world tensors when additional information is available from distinct sources. For example, in an aircraft sensor data, we might have additional flight information available such as whether a flight was normal/abnormal, weather conditions, routes taken, etc, which we wish to incorporate when analyzing the sensor signals for multiple flights.

Coupled Matrix-Tensor Factorizations is a natural extension to the standard tensor factorization formulation. For instance, the factorization of a third-order tensor $\mathcal{X}$ coupled with a matrix $\mathbf{M}$ on its first mode can be obtained by minimizing

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \left\| \mathcal{X} - \hat{\mathcal{X}} \right\|_F^2 + \lambda \left\| \mathbf{M} - \hat{\mathbf{M}} \right\|_F^2 \tag{5.1}$$

where λ is a parameter representing the strength of the coupling for this task, i.e., how important $\mathbf{M}$ is to improve the prediction. An illustration of the coupled matrix-tensor factorization is shown in Figure 5.1, where the modes are color-coded. In this simple example, the first mode of the tensor $\mathcal{X}$ in red is coupled with the second mode of the additional information matrix $\mathbf{M}$ in red as well.

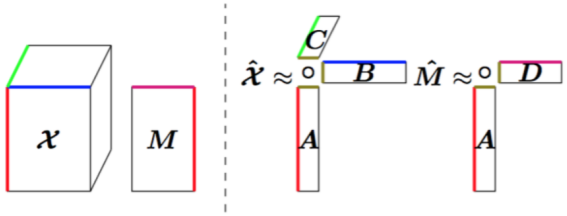


Figure 5.1: A coupled matrix-tensor factorization example

Many techniques have been proposed to solve this non-negative optimization problem, such as projected Stochastic Gradient Descent (SGD) [11] (i.e., additive update rules) and multiplicative update rules. Most of this work extends Lee and Seung's multiplicative matrix updates formulae [71] for matrices, notably the simple extension for tensors [133] and the many coupled extensions, e.g. Generalized Tensor Factorization [112, 138].

5.1.3 Method

One of the insights we have on the aircraft sensor systems data is that the absolute values of the sensor readings change from flight to flight depending on the flight condition. For example, altitude sensor readings would be different for flights taking different paths or routes. Thus we cannot simply use the raw sensor readings as the input to our algorithm. From our first insight, even though the absolute values may vary across different flights, the sensor-sensor correlation relationships do not change across the flights. Thus, we reform our input data into correlation matrices. Let $\mathcal{X} \in \mathbb{R}^{n_s \times n_s \times n_f}$ be a tensor of sensor-sensor correlation matrices of n_s continuous sensors for n_f flights.

In addition to the sensor readings, we also have the alerts triggered for each flight. Let us denote the number of alert categories or the 'ATA chapters' as n_a, then we have the message matrix for the flights, $\mathbf{M} \in \mathbb{R}^{n_f \times n_a}$. M_{ij} denotes the number of alerts triggered for the $j-th$ alert type during the $i-th$ flight.

The matrix $\mathbf{M}$ and the tensor $\mathcal{X}$ share a mode, the flight mode. So we propose to factorize $\mathbf{M}$ and the tensor $\mathcal{X}$ using coupled tensor factorization algorithm.

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \frac{1}{2}||\mathcal{X} - \mathcal{C} - \mathcal{D}||_F^2 + \frac{1}{2}||\mathbf{M} - \mathbf{C}\mathbf{D}^T||_F^2 \tag{5.2}$$

where $\mathcal{C}, \mathcal{D}$ are the common and difference tensors: $\mathcal{C} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{I}_r$, and $\mathcal{D} = \sum_{r=1}^{R} \mathbf{b}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$. Here, $\mathcal{C}$ is a tensor that captures normal sensor-sensor correlations, and $\mathcal{D}$ is a tensor that captures subtle anomalous sensor-sensor correlations that can be factorized with $\mathbf{M}$ in coupled manner: $\mathbf{M} = \sum_{r=1}^{R} \mathbf{c}_r \circ \mathbf{d}_r$, sharing the $\mathbf{c}_r$ factors or the latent factors for the flight mode.

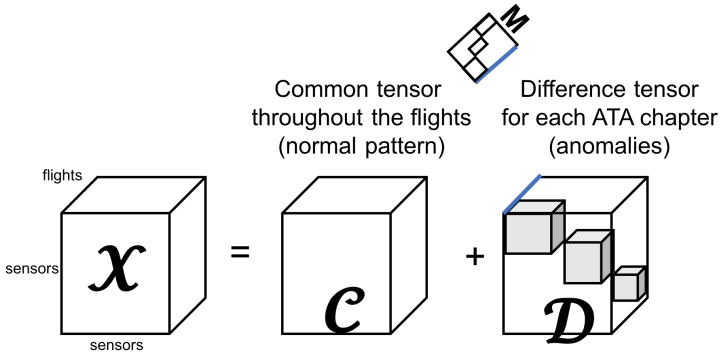


Figure 5.2: Tensor-matrix formulation

We are assuming that majority of normal sensor-sensor correlation would be captured by the common tensor $\mathcal{C}$ and subtle subgroups of anomalous sensors that break away from the normal

sensor-sensor correlation would be captured by the difference tensor $\mathcal{D}$, which is factorized in a coupled manner with M, the alerts information matrix.

In Figure 5.2, an illustration of matrix-tensor factorization is shown. The high level idea is to separate the data tensor $\mathcal{X}$ into two separate tensors that captures normal, and abnormal activities respectively, where the abnormal tensor is to be coupled with the additional matrix of alerts information. Here, the shared mode is denoted in light blue color.

5.1.4 Experiments

Synthetic data experiments

To check if our method seems reasonable, we conducted a simple synthetic data experiment. We assumed that there are five parts in the aircraft system where each part consist of 10 sensors with high correlations, making 50 sensors in total. For each group of sensor, we designed the sensor pattern to look distinct from one another, and added random noise to generate 10 sensor readings for each group. In total, we generate a sensor reading of 100 time ticks, for 120 flights.

In Figure 5.3, we show the result of what our method captured.

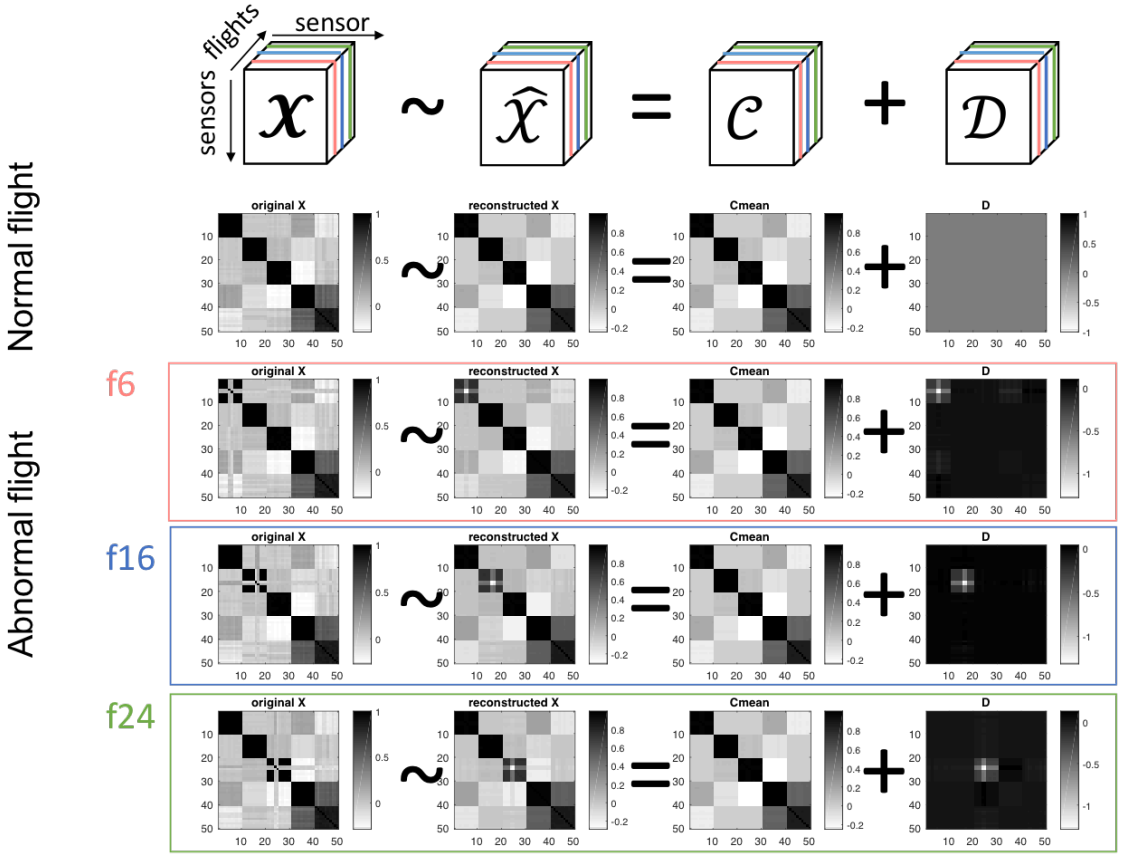


Figure 5.3: Visualization of the result

In the top row, we remind what our method does: we have $\mathcal{X}$ data tensor, and we are assuming it is a sum of two tensors: Common and difference tensors in $\mathcal{C}$ and $\mathcal{D}$.

We generated 3 anomalous flights in pink, blue, and green slices (flights).

In the second row, we show a slice for the normal flight, say $f1$. This is what the observed correlation looks like, and it learned that the dominant common pattern throughout the flight is 5 groups of sensors with strong correlation, and nothing captured in $\mathcal{D}$ slice (because it is a normal flight).

Third row in pink box shows the pink slice ($f6$ slice) of above tensors: We see that the data given to us had a subgroup of sensors that had zeroed-out correlation with other sensors in the group. And this subgroup of sensors with abnormal correlation was captured in $\mathcal{D}$ slice.

And same goes for the blue slice, $f16$ and the green slice $f24$. $\mathcal{D}$ slices captured the subgroup of sensors that were corrupted as shown in the slices of the data tensor $\mathcal{X}$.

Real-world data experiments

In this section, we share experimental results on the real-world dataset. We apply our method on NASA dataset[1] that is publicly available. NASA dataset consists of records of sensor signals for multiple flights. There are 180 sensors, sampled at varying sampling rate (0.25-16). Among 180 sensors, we select continuous sensors, which comprises of 94 sensors. Then we normalize the lengths of signals to 1,000 time points by re-sampling the signals, by *decimation* down-sampling technique, i.e. if we are targeting for a time sequence of length 10 given 100, we keep only every 10th sample. At the end of the pre-processing, we have 94 sensors recorded for 1,000 time points, for 50 flights.

We injected anomalies as follows. Total number of alerts were set to two, where each alert was triggered for two randomly selected consecutive flights. For each alert, one sensor was randomly selected to be perturbed to mimic anomalous behavior by replacing the signal with white noise. We repeat and try ten different anomaly injections and averaged the performance over ten trials.

There are mainly three components in our method: 1) working in the correlation space instead of raw sensor signals, 2) separating common and difference tensors, and 3) applying coupled matrix-tensor factorization for extracting sensor subgroups responsible for the alerts. The most naive method one could try in solving this problem would be using the sensor signals itself instead of working in the correlation space, without assuming common and difference tensors separately. Here, the modes of the input tensor would be (*sensors, time, flights*). We will denote this method as 'CMTF'. From our experiments, this method did not prove to be a good choice for solving our problem. One could think of 'A-MINE-0' where we still feed the sensor signals as given to us (not correlations, but a tensor of modes (sensors, time, flights)) but learning two separate tensors of common and difference tensors. (This was the first model that we tried to solve this problem, which did not quite work out well, which led us to improve even further, to come up with our final proposed method, A-MINE.) Finally, our proposed method A-MINE would be where we work on the correlation space of sensor signals and separate it into two tensors: common and difference tensors.

[1] `https://c3.nasa.gov/dashlink/resources/664/`

In Figure 5.4, ROC curve for A-MINE-0 and A-MINE is shown. We varied the threshold for detecting anomalies in the slices of difference tensor to compute the ROC curves. We repeated 10 experiments and averaged the curves.

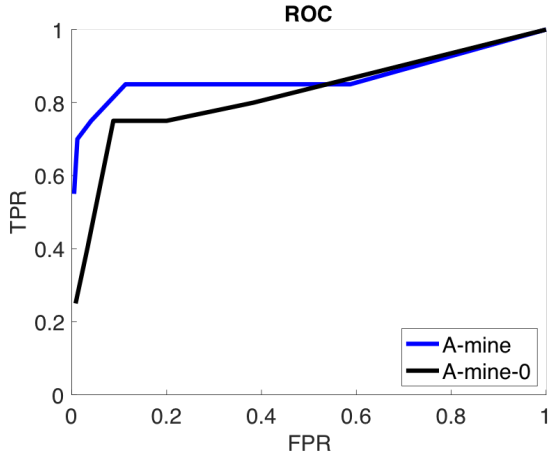


Figure 5.4: ROC curve for A-MINE-0 and A-MINE

The AUCs (Area Under the Curve) for A-MINE-0 and A-MINE were 0.82 and 0.87, respectively. From this experiment, we could confirm that our intuition on A-MINE algorithm to more effectively solve the problem was correct: working in the correlation space of the sensors instead of the raw signals.

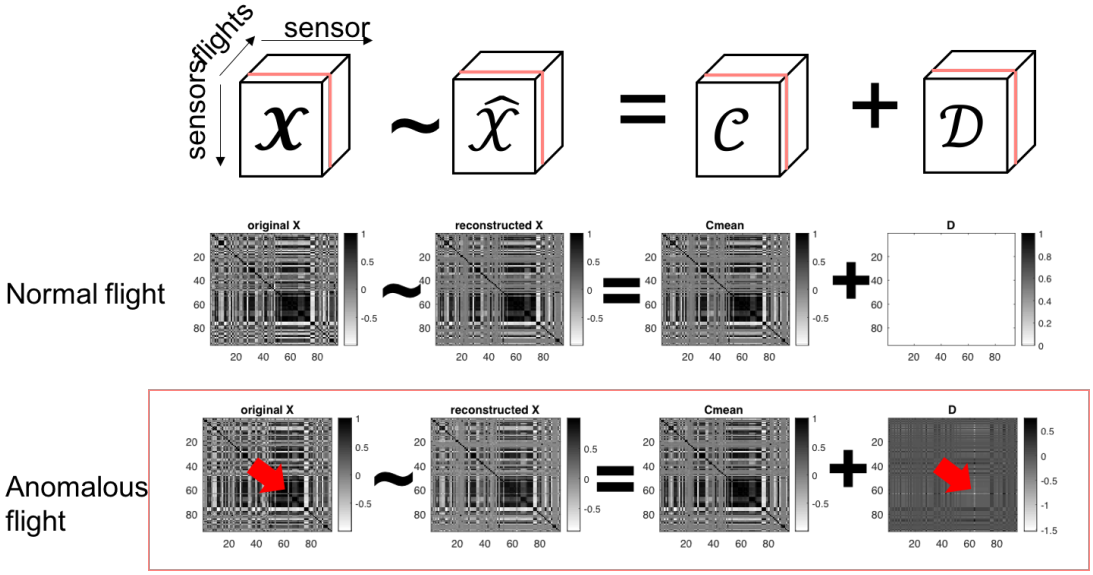


Figure 5.5: Examples from NASA dataset. Each row shows - slices from $\mathcal{X}, \hat{\mathcal{X}}, \mathcal{C}, \mathcal{D}$.in order. First row is an illustration, second row is an example of a normal flight, and the third row is an example of an anomalous flight.

In Figure 5.5, we show some examples of result on NASA dataset. In the second row, we show slices from tensors $\mathcal{X}, \hat{\mathcal{X}}, \mathcal{C}, \mathcal{D}$ for a normal flight. We see that $\mathcal{D}$ slice did not capture any

sensors. In the third row, we show the slices for an anomalous flight. This anomalous flight had sensor 63 perturbed. In the first column of the third row, we can see that in the slice $\mathcal{X}$, there is a broken correlation for sensor 63, where there shows white lines crossing through the a large group sensors (indicated with a red arrow). Our proposed method successfully captured this sensor 63 in the $\mathcal{D}$ slice (white line striking through the sensor 63, indicated with a red arrow) in the last column of the third row.

5.1.5 Conclusion

In this section, we described the aircraft mining problem that we are interested in. We shared our insights on the aircraft sensor signals data, and proposed an algorithm to learn normal sensor-sensor correlations and detect subtle anomalous sensor subgroups. Our proposed method consists of three parts: 1) working in correlation space, 2) learning two separate tensors, common and difference tensors, and 3) utilizing coupled matrix-tensor factorization. From our experimental results on NASA dataset, we showed that our proposed method improved anomaly detection performance compared to naively applying coupled matrix-tensor factorization with or without learning two separate tensors.

Chapter 6

Conclusion and Future Directions

6.1 Conclusion

In this thesis, we discussed our works on two different tasks: 1) signal reconstruction and 2) signal mining in various applications.

- **Signal reconstruction**:
  1. *Historical epidemiological records*: Reconstructing finer granularity of patient counts from multiple sources of reports in different granularities.
  2. *Brain scans*: Reconstructing finer brain scan images from two different modalities of different resolution, i.e. MEG and fMRI.
- **Signal mining**:
  1. *Power grid*: Learning latent features utilizing the domain knowledge, and forecasting and detecting anomalies in the power consumption data.
  2. *Aircraft systems*: Detecting anomalies in a large-scale aircraft sensor system corresponding to specific alerts.

In Chapters I and II, we have demonstrated our works on signal reconstruction and mining. We have proposed novel problem formulations and algorithms to tackle the problems, and share our insights on the real world data we are dealing with. From our experimental results, we showcase that our insights and proposed algorithms perform well, yielding some meaningful and interesting results for further analysis.

6.2 Future directions

Infusing domain knowledge

In our signal reconstruction works, we have been working on the historical epidemiology and brain domains. As we have shown, signal reconstruction requires additional information on the data, i.e. domain knowledge. While it is crucial to introduce appropriate domain knowledge for the task, it is often not straight-forward to incorporate it into the machine learning techniques. It would facilitate applications of our algorithms if we provide more principled way of incorporating domain knowledge into the algorithms.

Handling high dimensionality

As we have experienced in our previous works, signal mining usually deals with large datasets with high dimensions. High dimensionality causes machine learning algorithms to behave in the unexpected manners, resulting in bad performances. Therefore, it is of great interest to come up with effective algorithms that can efficiently handle high dimensional datasets. We expect that extension of our algorithms appropriate for high dimensional data would help them broaden their applications.

Visualization and summarization

Datasets we deal with usually consist of high dimensionality and long-length time sequences. Because of its size and also complexity, it is often very difficult to understand the data just by looking at it. Also, when analyzing experimental results, it is sometimes hard to interpret the result other than the numbers. Therefore, it is important that we make algorithms that can do sense-making. We expect that effective visualization and summarization of data would benefit the users' easy understanding of the data.

Appendix A

BRAINZOOM supplementary materials

Here proofs were done by one of our collaborators, Xiao Fu[1]. We place the proofs here to make the thesis self-contained.

A.1 Step size choices.

In this appendix we show that the step sizes chosen for BRAINZOOM leads to monotonic decrease of the cost function of (3.7). The idea is to simply show that the selected step sizes are smaller than the reciprocal of the respective Liptchitz constants of the conditional functions while fixing the other variables. We show that for the step size of $\boldsymbol{Z}$, and that of $\boldsymbol{W}$ follow similar arguments.

The Liptchitz constant is the supremum of the spectral norm of the Hessian matrix of a function—for quadratic functions, it boils down to the spectral norm of Hessian. To derive it, let us first write the $\boldsymbol{Z}$-subproblem 3.11 in the vectorized form

$$\begin{aligned}\bar{f}(z;\boldsymbol{w},\tau) = &\|x_t - \tau\bar{T}_t z\|^2 + \|x_s - \bar{T}_s \text{Diag}(\boldsymbol{w})z\|^2 \\ &+\rho\left(\|\text{Diag}(\boldsymbol{d}_s)\bar{\boldsymbol{H}}_s z\|^2 + \|\text{Diag}(\boldsymbol{d}_t)\bar{\boldsymbol{H}}_t z\|^2\right) \\ &+\mu\|z-\boldsymbol{w}\|^2,\end{aligned} \tag{A.1}$$

where the lowercase letters are vectorized versions of the matrices denoted by the uppercase letters, and

$$\bar{T}_t = \boldsymbol{I}\otimes T_t, \qquad \bar{T}_s = T_s \otimes \boldsymbol{I},$$
$$\bar{\boldsymbol{H}}_s = \boldsymbol{I}\otimes \boldsymbol{H}_s, \qquad \bar{\boldsymbol{H}}_t = \boldsymbol{H}_t \otimes \boldsymbol{I}.$$

The Hessian matrix can easily be calculated as

$$\begin{aligned}\nabla_z^2 \bar{f} = &\tau^2 \bar{T}_t^T\bar{T}_t + \text{Diag}(\boldsymbol{w})\bar{T}_s^T\bar{T}_s\text{Diag}(\boldsymbol{w}) \\ &+\rho\bar{\boldsymbol{H}}_s^T\text{Diag}(\boldsymbol{d}_s^2)\bar{\boldsymbol{H}}_s + \rho\bar{\boldsymbol{H}}_t^T\text{Diag}(\boldsymbol{d}_t^2)\bar{\boldsymbol{H}}_t + \mu\boldsymbol{I},\end{aligned} \tag{A.2}$$

a summation of several individual matrices. An upperbound on the spectral norm of $\nabla_z^2\bar{f}$ is the sum of the individual spectral norms. Furthermore, we invoke the following properties on the

[1] http://people.oregonstate.edu/ fuxia/

matrix spectral norm:

$$\|\boldsymbol{A}\boldsymbol{B}\| \leq \|\boldsymbol{A}\|\|\boldsymbol{B}\|,$$
$$\|\boldsymbol{A} \otimes \boldsymbol{B}\| = \|\boldsymbol{A}\|\|\boldsymbol{B}\|.$$

Then we have that

$$\|\nabla_z^2 \bar{f}\| \leq \tau^2 \lambda_{\max}(T_t^T T_t) + \max(\boldsymbol{w}^2)\lambda_{\max}(T_t^T T_t)$$
$$+\rho\left(\max(\boldsymbol{d}_s^2)\|\boldsymbol{H}_s\|^2 + \max(\boldsymbol{d}_t^2)\|\boldsymbol{H}_t\|^2\right) + \mu.$$

Finally, we show that $\|\boldsymbol{H}_s\|$ and $\|\boldsymbol{H}_t\|$ are upperbounded by c_H defined in (3.15), by taking the total variation regularization as an example, assuming its size is $(n-1)\times n$.

The definition of matrix spectral norm is

$$\|\boldsymbol{H}\| = \max_{\|\boldsymbol{u}\|=1} \|\boldsymbol{H}\boldsymbol{u}\|.$$

Assume $\hat{\boldsymbol{H}}$ is obtain by adding one more row into $\boldsymbol{H}$, then we trivially have $\|\boldsymbol{H}\| \leq \|\hat{\boldsymbol{H}}\|$. Consider $\hat{\boldsymbol{H}}$ to be the following *circulant matrix*

$$\hat{\boldsymbol{H}} = \begin{bmatrix} 1 & -1 & 0 & & \dots \\ 0 & 1 & -1 & & \dots \\ \vdots & & \ddots & & \\ & & & 1 & -1 \\ -1 & 0 & \dots & 0 & 1 \end{bmatrix},$$

it can be diagonalized by the n-point discrete Fourier transform (DFT) matrix $\boldsymbol{\Phi}$

$$\hat{\boldsymbol{H}} = \boldsymbol{\Phi}\text{Diag}(\hat{\boldsymbol{h}})\boldsymbol{\Phi}^*,$$

where $\hat{\boldsymbol{h}}$ is the DFT of first row of $\hat{\boldsymbol{H}}$. Because $\boldsymbol{\Phi}$ has orthogonal columns, by rotating the elements of $\hat{\boldsymbol{h}}$ to be non-negative real, this becomes the singular value decomposition of $\hat{\boldsymbol{H}}$, and the largest absolute value of $\hat{\boldsymbol{h}}$ is the spectral norm of $\|\hat{\boldsymbol{H}}\|$. By the definition of DFT,

$$\hat{\boldsymbol{h}} = \mathbf{1} + [\ 1\ e^{-i/n}\ e^{-i2/n}\ ...\ e^{-i(n-1)/n}\ ]^T,$$

therefore we trivially have $\max(|\hat{\boldsymbol{h}}|) \leq 2$, thus

$$\|\boldsymbol{H}\| \leq 2$$

Similarly, for the smoothness regularization, $\|\boldsymbol{H}\| \leq 4$.

A.2 Proof of Theorem 1

Two claims were made in Theorem 1. Here we separate them into two propositions, and prove them individually.

Proposition 1 *Let* $\{(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)})\}_r$ *be the solution sequence produced by the proposed* BRAINZOOM (3.16)*, then every limit point of* $\{(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)})\}_r$ *is a stationary point of Problem* (3.7)*.*

Proof We prove that BRAINZOOM described in (3.16) falls into the framework of successive upper-bound minimization (BSUM) [101]. As a result, every limit point is a stationary point,

according to [101, Theorem 2]. To do so, we re-write the algorithm as

$$\begin{aligned}\tau^{(r+1)} &\leftarrow \arg\min_{\tau} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau),\\ \boldsymbol{W}^{(r+1)} &\leftarrow \arg\min_{\boldsymbol{W}} u_w(\boldsymbol{W}; \boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}),\\ \boldsymbol{Z}^{(r+1)} &\leftarrow \arg\min_{\boldsymbol{Z}} u_z(\boldsymbol{Z}; \boldsymbol{W}^{(r+1)}, \boldsymbol{Z}^{(r)}, \tau^{(r+1)}),\end{aligned}$$

where the $\arg\min$'s are *uniquely* defined, function u_z is an auxiliary function that satisfies that $\forall \tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}$,

$$\begin{aligned}u_z(\boldsymbol{Z}; \tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}) &\geq f(\boldsymbol{Z}, \tilde{\boldsymbol{W}}, \tilde{\tau}), \forall \boldsymbol{Z}\\ u_z(\tilde{\boldsymbol{Z}}; \tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}) &= f(\tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}),\\ \nabla_{\boldsymbol{Z}} u_z(\tilde{\boldsymbol{Z}}; \tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}) &= \nabla_{\boldsymbol{Z}} f(\tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}),\end{aligned}$$

and similarly for u_w.

For the update of τ, it is a scalar least-squares problem, and the minimizer is unique as long as $\|T_t \boldsymbol{Z}\|_F^2 \neq 0$, which can be guaranteed as long as the regularization parameter ρ is not too big.

As for the auxiliary function with respect to $\boldsymbol{Z}$, we define it as

$$u_z(\boldsymbol{Z}; \tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}) = \bar{f}(\tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}) + \left\langle \nabla_{\boldsymbol{Z}} \bar{f}(\tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}), \boldsymbol{Z} - \tilde{\boldsymbol{Z}} \right\rangle + \frac{1}{2z}\|\boldsymbol{Z} - \tilde{\boldsymbol{Z}}\|_F^2.$$

As we have shown in Appendix A.1 'Step-size choices' section, $1/z$ is larger than the Lipschitz constant of $\bar{f}$ when fixing $\boldsymbol{W}$ and τ, therefore

$$u_z(\boldsymbol{Z}; \tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{W}}, \tilde{\tau}) \geq \bar{f}(\boldsymbol{Z}; \tilde{\boldsymbol{W}}, \tilde{\tau}) \geq f(\boldsymbol{Z}, \tilde{\boldsymbol{W}}, \tilde{\tau}).$$

It is also easy to see that the function value and gradient also coincides with that of f with respect to $\tilde{\boldsymbol{Z}}$. Furthermore, u_z is strongly convex, implying the minimizer is unique.

Similarly, and with simpler derivations, we have that u_w also satisfies the sharp upperbound requirements and the minimizer is unique. Invoking [101, Theorem 2], every limit point of BRAINZOOM is a stationary point. □

Proposition 2 *In addition to Proposition 1, the optimality gap between $\{(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)})\}_r$ and a stationary point is at most $\mathcal{O}(1/r)$; i.e., the algorithm approaches a stationary point at least sub-linearly.*

Proof For the τ-subproblem, because

$$\tau^{(r+1)} = \text{tr}(\boldsymbol{X}_t^T T_t \boldsymbol{Z}^{(r)})/\|T_t \boldsymbol{Z}^{(r)}\|_F^2,$$

we have that

$$\begin{aligned}&f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)}) - f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}) \qquad \text{(A.3)}\\ &= \|\boldsymbol{X}_t - \tau^{(r)} T_t \boldsymbol{Z}^{(r)}\|_F^2 - \|\boldsymbol{X}_t - \tau^{(r+1)} T_t \boldsymbol{Z}^{(r)}\|_F^2\\ &= \frac{1}{\|T_t \boldsymbol{Z}^{(r)}\|_F^2} \left(\partial_\tau f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)})\right)^2\\ &= t^{(r)} \left(\partial_\tau f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)})\right)^2,\end{aligned}$$

where we define $t^{(r)} = 1/\|T_t \boldsymbol{Z}^{(r)}\|_F^2$

For the $\boldsymbol{W}$-subproblem, we have the following inequality:

$$\begin{aligned} & f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)}) \\ \leq\ & f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}) + \frac{L_w^{(r)}}{2}\|\boldsymbol{W}^{(r)} - \boldsymbol{W}^{(r+1)}\|_F^2 \\ & + \left\langle \nabla_{\boldsymbol{Z}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}), \boldsymbol{W}^{(r+1)} - \boldsymbol{W}^{(r)} \right\rangle. \end{aligned} \tag{A.4}$$

We also notice that

$$\begin{aligned} \boldsymbol{W}^{(r+1)} = \arg\min_{\boldsymbol{W}}\ & \frac{1}{2w^{(r)}}\|\boldsymbol{W} - \boldsymbol{W}^{(r)}\|_F^2 \\ & + \left\langle \nabla_{\boldsymbol{W}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}), \boldsymbol{W} - \boldsymbol{W}^{(r)} \right\rangle, \end{aligned} \tag{A.5}$$

which means

$$\begin{aligned} & \left\langle \nabla_{\boldsymbol{W}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}), \boldsymbol{W}^{(r+1)} - \boldsymbol{W}^{(r)} \right\rangle \\ & + \frac{1}{2w^{(r)}}\|\boldsymbol{W}^{(r+1)} - \boldsymbol{W}^{(r)}\|_F^2 \leq 0, \end{aligned} \tag{A.6}$$

therefore

$$\begin{aligned} & f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}) - f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)}) \\ & \geq \left(\frac{1}{2w^{(r)}} - \frac{L_w^{(r)}}{2} \right) \|\boldsymbol{W}^{(r+1)} - \boldsymbol{W}^{(r)}\|_F^2. \end{aligned} \tag{A.7}$$

On the other hand, since $\boldsymbol{W}^{(r+1)}$ is the minimizer of (A.5), by the first order optimality condition, we have

$$\nabla_{\boldsymbol{W}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}) - \frac{1}{w^{(r)}}(\boldsymbol{W}^{(r+1)} - \boldsymbol{W}^{(r)}) = 0.$$

In sum,

$$\begin{aligned} & f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)}) - f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)}) \\ \geq\ & \left(\frac{w^{(r)}}{2} - \frac{L_w^{(r)}}{2{w^{(r)}}^2} \right) \|\nabla_{\boldsymbol{W}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)})\|_F^2. \end{aligned} \tag{A.8}$$

Similarly for $\boldsymbol{Z}$, we can show that

$$\begin{aligned} & f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)}) - f(\boldsymbol{Z}^{(r+1)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)}) \\ \geq\ & \left(\frac{z^{(r)}}{2} - \frac{L_z^{(r)}}{2{z^{(r)}}^2} \right) \|\nabla_{\boldsymbol{Z}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)})\|_F^2. \end{aligned} \tag{A.9}$$

Combining (A.3), (A.8), and (A.9), we obtain

$$\begin{aligned} & f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)}) - f(\boldsymbol{Z}^{(r+1)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)}) \\ & \geq t^{(r)} \left(\partial_\tau f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)}) \right)^2 \\ & + \left(\frac{w^{(r)}}{2} - \frac{L_w^{(r)}}{2{w^{(r)}}^2} \right) \|\nabla_{\boldsymbol{W}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)})\|_F^2 \\ & + \left(\frac{z^{(r)}}{2} - \frac{L_z^{(r)}}{2{z^{(r)}}^2} \right) \|\nabla_{\boldsymbol{Z}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)})\|_F^2. \end{aligned} \tag{A.10}$$

To show the convergence rate, let us define

$$\begin{aligned}\phi^{(r)} = &\left(\partial_\tau f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)})\right)^2 \\ &+ \|\nabla_{\boldsymbol{W}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)})\|_F^2 \\ &+ \|\nabla_{\boldsymbol{Z}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)})\|_F^2\end{aligned}$$

One can see that

$$\begin{aligned}\phi^{(r)} \to 0 \qquad &\Longrightarrow \\ &\left(\partial_\tau f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)})\right)^2 \to 0 \\ &\|\nabla_{\boldsymbol{W}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)})\|_F^2 \to 0 \\ &\|\nabla_{\boldsymbol{Z}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)})\|_F^2 \to 0\end{aligned}$$

which implies a stationary point is attained. Let us assume that the first time $\phi^{(r)} < \varepsilon$ requires T iterations. Then, summing up (A.10) over $r = 1, ..., T$, we have

$$\begin{aligned}&f(\boldsymbol{Z}^{(1)}, \boldsymbol{W}^{(1)}, \tau^{(1)}) - f(\boldsymbol{Z}^{(T)}, \boldsymbol{W}^{(T)}, \tau^{(T)}) \\ &\geq \sum_{r=1}^{T} t^{(r)} \left(\partial_\tau f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r)})\right)^2 \\ &+ \sum_{r=1}^{T} \left(\frac{w^{(r)}}{2} - \frac{L_w^{(r)}}{2{w^{(r)}}^2}\right) \|\nabla_{\boldsymbol{W}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r)}, \tau^{(r+1)})\|_F^2 \\ &+ \sum_{r=1}^{T} \left(\frac{z^{(r)}}{2} - \frac{L_z^{(r)}}{2{z^{(r)}}^2}\right) \|\nabla_{\boldsymbol{Z}} f(\boldsymbol{Z}^{(r)}, \boldsymbol{W}^{(r+1)}, \tau^{(r+1)})\|_F^2. \\ &\geq \sum_{r=1}^{T} c\phi^{(r)}\end{aligned} \tag{A.11}$$

where

$$c = \min_r \left\{ t^{(r)}, \left(\frac{w^{(r)}}{2} - \frac{L_w^{(r)}}{2{w^{(r)}}^2}\right), \left(\frac{z^{(r)}}{2} - \frac{L_z^{(r)}}{2{z^{(r)}}^2}\right) \right\}.$$

The above implies

$$\frac{f(\boldsymbol{Z}^{(1)}, \boldsymbol{W}^{(1)}, \tau^{(1)}) - f(\boldsymbol{Z}^{(T)}, \boldsymbol{W}^{(T)}, \tau^{(T)})}{T} \geq c\phi^{(r)},$$

and so

$$\phi^{(r)} \leq \frac{1}{T}\left(\frac{f(\boldsymbol{Z}^{(1)}, \boldsymbol{W}^{(1)}, \tau^{(1)}) - f(\boldsymbol{Z}^{\star}, \boldsymbol{W}^{\star}, \tau^{\star})}{c}\right),$$

where $\boldsymbol{Z}^{\star}, \boldsymbol{W}^{\star}, \tau^{\star}$ denote a global optimal solution of Problem (3.7). This completes the proof. □

Bibliography

[1] Evrim Acar, Canan Aykut-Bingol, Haluk Bingol, Rasmus Bro, and Bulent Yener. Multiway analysis of epilepsy tensors. *Bioinformatics*, 23(13):i10–i18, 2007. 5.1.2

[2] Nasir Ahmed, T_ Natarajan, and Kamisetty R Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974. 2.4.2

[3] Faisal M Almutairi, Fan Yang, Hyun Ah Song, Christos Faloutsos, Nicholas Sidiropoulos, and Vladimir Zadorozhny. Homerun: scalable sparse-spectrum reconstruction of aggregated historical data. *Proceedings of the VLDB Endowment*, 11(11):1496–1508, 2018. (document), 2.4

[4] Edoardo Amaldi and Viggo Kann. The complexity and approximability of finding maximum feasible subsystems of linear relations. *Theoretical computer science*, 147(1-2): 181–210, 1995. 2.4.3

[5] S Massoud Amin. U.s. grid gets less reliable [the data]. *IEEE Spectrum*, 48(1):80–80, 2011. 4.1.1, 4.2.1

[6] Roy M. Anderson and Robert M. May. *Infectious diseases of humans: Dynamics and control*. Oxford Press, 2002. 2.2.2

[7] Miguel Araujo, Pedro Ribeiro, Hyun Ah Song, and Christos Faloutsos. Tensorcast: forecasting and mining with coupled tensors. *Knowledge and Information Systems*, 59(3): 497–522, 2019. 4.1.2

[8] Brett W. Bader and Tamara G. Kolda. Algorithm 862: MATLAB tensor classes for fast algorithm prototyping. *ACM Transactions on Mathematical Software*, 32(4):635–653, December 2006. doi: 10.1145/1186785.1186794. 4.1.4

[9] Brett W. Bader, Tamara G. Kolda, et al. Matlab tensor toolbox version 2.6. Available online, February 2015. URL `http://www.sandia.gov/~tgkolda/TensorToolbox/`. 4.1.4

[10] Jan Beran. *Statistics for long-memory processes*, volume 61. CRC press, 1994. 4.1, 4.1.2

[11] Alex Beutel, Partha Pratim Talukdar, Abhimanu Kumar, Christos Faloutsos, Evangelos E Papalexakis, and Eric P Xing. Flexifact: Scalable flexible factorization of coupled tensors on hadoop. In *SDM*, pages 109–117. SIAM, 2014. 5.1.2

[12] Felix Biessmann, Frank C Meinecke, Arthur Gretton, Alexander Rauch, Gregor Rainer, Nikos K Logothetis, and Klaus-Robert Müller. Temporal kernel cca and its application in multimodal neuronal data analysis. *Machine Learning*, 79(1-2):5–27, 2010. 3.1.2

[13] Felix Biessmann, Sergey Plis, Frank C Meinecke, Tom Eichele, and Klaus-Robert Muller. Analysis of multimodal neuroimaging data. *IEEE Reviews in Biomedical Engineering*, 4: 26–58, 2011. 3.1.1, 3.1.2

[14] J. Bleiholder and F. Naumann. Data fusion. *ACM Computing Surveys*, 41(1), 2008. 2.1.1, 2.3.2

[15] Jens Bleiholder and Felix Naumann. Data fusion. *ACM Computing Surveys (CSUR)*, 41 (1):1–41, January 2009. ISSN 0360-0300. 2.4.2

[16] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015. 4.1, 4.1.2, 4.2.2, 4.3, 4.2.4

[17] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004. 3.1.3

[18] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011. 2.4.2

[19] David M Bromberg, Marko Jereminov, Xin Li, Gabriela Hug, and Larry Pileggi. An equivalent circuit formulation of the power flow problem with current and voltage state variables. In *PowerTech, 2015 IEEE Eindhoven*, pages 1–6, 2015. 4.1.2

[20] I. Brown. An empirical comparison of benchmarking methods for economic stock time series. *US Census Bureau*, 2012. 2.1.2, 2.3.2

[21] Robert Goodell Brown. *Statistical forecasting for inventory control*. McGraw/Hill, 1959. 4.1.2, 4.2.4

[22] D Bunn and E Dillon Farmer. Comparative models for electrical load forecasting. 1985. 4.1.2

[23] Vince D Calhoun, Tulay Adalı, Kent A Kiehl, Robert Astur, James J Pekar, and Godfrey D Pearlson. A method for multitask fmri data fusion applied to schizophrenia. *Human brain mapping*, 27(7):598–610, 2006. 3.1.2

[24] Emmanuel J Candès. Compressive sampling. In *Proceedings of the international congress of mathematicians*, volume 3, pages 1433–1452. Madrid, Spain, 2006. 2.4.2, 2.4.2, 2.4.2

[25] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008. 2.4.2, 2.4.2

[26] Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006. 2.4.2

[27] Arvind Caprihan, Godfrey D Pearlson, and Vincent D Calhoun. Application of principal component analysis to distinguish patients with schizophrenia from healthy controls based on fractional anisotropy measurements. *Neuroimage*, 42(2):675–682, 2008. 3.1.2

[28] B. Chen. An empirical comparison of methods for temporal distribution and interpolation at the national accounts. *Bureau of Economic Analysis*, 2007. 2.1.2

[29] Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition

by basis pursuit. *SIAM review*, 43(1):129–159, 2001. 2.4.2, 2.4.2

[30] David Cohen et al. Magnetoencephalography: evidence of magnetic fields produced by alpha-rhythm currents. *Science*, 161(3843):784–786, 1968. 3.1.1

[31] Edith Cohen and Haim Kaplan. Bottom-k sketches: Better and more efficient estimation of aggregates. In *ACM SIGMETRICS Performance Evaluation Review*, volume 35, pages 353–354. ACM, 2007. 2.4.2

[32] Graham Cormode, Minos Garofalakis, Peter J Haas, and Chris Jermaine. Synopses for massive data: Samples, histograms, wavelets, sketches. *Foundations and Trends in Databases*, 4(1–3):1–294, 2012. 2.4.2

[33] Nicolle M Correa, Yi-Ou Li, Tulay Adali, and Vince D Calhoun. Canonical correlation analysis for feature-based fusion of biomedical imaging modalities and its application to detection of associative networks in schizophrenia. *IEEE journal of selected topics in signal processing*, 2(6):998–1007, 2008. 3.1.2

[34] Sven Dähne, Felix Biessmann, Frank C Meinecke, Jan Mehnert, Siamac Fazli, and Klaus-Robert Müller. Integration of multivariate data streams with bandpower signals. *IEEE Transactions on Multimedia*, 15(5):1001–1013, 2013. 3.1.2

[35] Alysha M De Livera, Rob J Hyndman, and Ralph D Snyder. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association*, 106(496):1513–1527, 2011. 4.1.2

[36] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *PVLDB*, 1(2):1542–1552, 2008. 2.4.2

[37] Xin Luna Dong and Felix Naumann. Data fusion: resolving data conflicts for integration. *PVLDB*, 2(2):1654–1655, 2009. 2.4.2

[38] Xin Luna Dong and Divesh Srivastava. Big data integration. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 1245–1248. IEEE, 2013. 2.3.2

[39] David L Donoho and Michael Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via $l1$ minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003. 2.4.2

[40] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006. 2.4.2

[41] Christos Faloutsos, H. V. Jagadish, and Nikolaos Sidiropoulos. Recovering information from summary data. In *VLDB'97, August 25-29, 1997, Athens, Greece*, pages 36–45, 1997. 2.1.2, **??**, 3, 2.2.2, 2.4, 2.3.2, 2.4.2

[42] Xiao Fu, Kejun Huang, Otilia Stretcu, Hyun Ah Song, Evangelos Papalexakis, Partha Talukdar, Tom Mitchell, Nicholas Sidiropoulos, Christos Faloutsos, and Barnabas Poczos. Brainzoom: High resolution reconstruction from multi-modal brain signals. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 216–227. SIAM, 2017. (document), 2.2.2, 2.4, 3.1

[43] Silvia Gazzola and James G. Nagy. Generalized arnoldi–tikhonov method for sparse reconstruction. *SIAM Journal on Scientific Computing*, 36(2):B225–B247, 2014. doi: 10.1137/130917673. URL http://dx.doi.org/10.1137/130917673. 2.1.2, 2.2.2

[44] G.Chamberlin. Temporal disaggregation. *Economic and Labour Market Review*, 2010. 2.1.2

[45] Gene H. Golub, Per Christian Hansen, and Dianne P. O'Leary. Tikhonov regularization and total least squares. *SIAM Journal on Matrix Analysis and Applications*, 21(1): 185–194, 1999. doi: 10.1137/S0895479897326432. URL http://dx.doi.org/10.1137/S0895479897326432. 2.1.2, 2.2.2

[46] Onur G Guleryuz. Weighted overcomplete denoising. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 2, pages 1992–1996. IEEE, 2003. 2.4.2

[47] Yaroslav O Halchenko, Stephen Jose Hanson, and Barak A Pearlmutter. Multimodal integration: fmri, mri, eeg, meg. *Advanced image processing in magnetic resonance imaging*, pages 223–265, 2005. 3.1.2

[48] D. Hall. *Mathematical techniques in multi-sensor data fusion*. Artech House, 2004. 2.1.1, 2.3.2

[49] D. Hall and J. Jordan. *Human-centered information fusion*. Artech House, 2010. 2.1.1, 2.3.2

[50] James Douglas Hamilton. *Time series analysis*, volume 2. Princeton university press Princeton, 1994. 4.2.2

[51] Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multi-modal factor analysis. 1970. 5.1.2

[52] Joyce C Ho, Joydeep Ghosh, and Jimeng Sun. Marble: high-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization. In *ACM SIGKDD*, pages 115–124, 2014. 4.1.2

[53] Bryan Hooi, Hyun Ah Song, Amritanshu Pandey, Marko Jereminov, Larry Pileggi, and Christos Faloutsos. Streamcast: Fast and online mining of power grid time sequences. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 531–539. SIAM, 2018. (document), 4.2

[54] Ali Hormati and Martin Vetterli. Annihilating filter-based decoding in the compressed sensing framework. In *Optical Engineering+ Applications*, pages 670121–670121. International Society for Optics and Photonics, 2007. 2.3.1, 2.3.2

[55] Ming-Jyh Hsieh, Wei-Guang Teng, Ming-Syan Chen, and Philip S Yu. Dawn: an efficient framework of dct for data with error estimation. *The VLDB JournalâĂŤThe International Journal on Very Large Data Bases*, 17(4):683–702, 2008. 2.4.2

[56] Kejun Huang, Nicholas D Sidiropoulos, and Athanasios P Liavas. A flexible and efficient algorithmic framework for constrained matrix and tensor factorization. *IEEE Transactions on Signal Processing*, 64(19):5052–5065, 2016. 2.4.3

[57] Clifford M Hurvich and Chih-Ling Tsai. A corrected akaike information criterion for vector autoregressive model selection. *Journal of time series analysis*, 14(3):271–279, 1993. 4.2.4

[58] Rob J Hyndman and Shu Fan. Density forecasting for long-term peak electricity demand. *IEEE Transactions on Power Systems*, 25(2):1142–1153, 2010. 4.1.2

[59] A. Jain. Fast inversion of banded toeplitz matrices by circular decompositions. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(2):121–126, Apr 1978. ISSN 0096-3518. doi: 10.1109/TASSP.1978.1163064. 5

[60] Ankur Jain, Edward Y Chang, and Yuan-Fang Wang. Adaptive stream resource management using kalman filters. In *ACM SIGMOD*, pages 11–22, 2004. 4.1, 4.1.2

[61] Marko Jereminov, David M Bromberg, Xin Li, Gabriela Hug, and Larry Pileggi. Improving robustness and modeling generality for power flow analysis. In *2016 IEEE/PES Transmission and Distribution Conference and Exposition (T&D)*, pages 1–5. IEEE, 2016. 4.1.2

[62] Marko Jereminov, Amritanshu Pandey, Hyun Ah Song, Bryan Hooi, Christos Faloutsos, and Larry Pileggi. Linear load model for robust power system analysis. In *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pages 1–6. IEEE, 2017. 4.1.2, 4.2.1, 4.2.2, 4.2.2

[63] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960. 4.1, 4.1.2, 4.3

[64] U Kang, Evangelos Papalexakis, Abhay Harpale, and Christos Faloutsos. Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In *ACM SIGKDD*, pages 316–324, 2012. 4.1.2

[65] David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 24 - 27, 2003*, pages 137–146, 2003. doi: 10.1145/956750.956769. URL `http://doi.acm.org/10.1145/956750.956769`. 2.2.2

[66] Syed Ali Khayam. The discrete cosine transform (dct): Theory and application. department of electrical and computing engineering, 2003. 2.4.2

[67] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009. 4.1.2, 4.1.3, 5.1.2

[68] Tamara G Kolda and Jimeng Sun. Scalable tensor decompositions for multi-aspect data mining. In *ICDM*, pages 363–372, 2008. 4.1, 4.1.2, 4.1.3

[69] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009. 10

[70] Jan Kujala, Gustavo Sudre, Johanna Vartiainen, Mia Liljestrom, Tom Mitchell, and Riitta Salmelin. Multivariate analysis of correlation between electrophysiological and hemodynamic responses during cognitive processing. *NeuroImage*, 92:207–216, 2014. 3.1.2

[71] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization.

In *Advances in neural information processing systems*, pages 556–562, 2001. 5.1.2

[72] Ju-Hong Lee, Deok-Hwan Kim, and Chin-Wan Chung. Multi-dimensional selectivity estimation using compressed histogram information. In *ACM SIGMOD Record*, volume 28, pages 205–214. ACM, 1999. 2.4.2

[73] Paula Sanz Leon, Stuart A Knock, M Marmaduke Woodman, Lia Domide, Jochen Mersmann, Anthony R McIntosh, and Viktor Jirsa. The virtual brain: A simulator of primate brain network dynamics. *Information-based methods for neuroimaging: analyzing structure, function and dynamics*, page 8, 2015. 3.1.1, 3.1.3, 3.1.4

[74] Julie Letchner, Christopher Re, Magdalena Balazinska, and Matthai Philipose. Access methods for markovian streams. In *ICDE*, pages 246–257, 2009. 4.1, 4.1.2, 4.3

[75] Yu-Ru Lin, Jimeng Sun, Hari Sundaram, Aisling Kelliher, Paul Castro, and Ravi Konuru. Community discovery via metagraph factorization. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(3):17, 2011. 4.1.2

[76] Zhongming Liu, Lei Ding, and Bin He. Integration of eeg/meg with mri and fmri in functional neuroimaging. *IEEE engineering in medicine and biology magazine*, 25(4):46, 2006. 3.1.2

[77] Zongge Liu, Hyun Ah Song, Vladimir Zadorozhny, Christos Faloutsos, and Nicholas Sidiropoulos. H-fuse: Efficient fusion of aggregated historical data. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 786–794. SIAM, 2017. (document), 2.1, 2.2.1, 2.2.2, 2.2.2, 2.4, 2.3.1, 2.3.2, 2.3.2, 2.4.1, 2.4.2, 2.4.2, 2.4.3, 2.4.3, 2.4.4

[78] Nikos K Logothetis, Jon Pauls, Mark Augath, Torsten Trinath, and Axel Oeltermann. Neurophysiological investigation of the basis of the fmri signal. *Nature*, 412(6843):150–157, 2001. 3.1.2

[79] Laetitia Loncan, Luis B de Almeida, Jose M Bioucas-Dias, Xavier Briottet, Jocelyn Chanussot, Nicolas Dobigeon, Sophie Fabre, Wenzhi Liao, Giorgio A Licciardi, Miguel Simoes, et al. Hyperspectral pansharpening: a review. *IEEE Geoscience and remote sensing magazine*, 3(3):27–46, 2015. 1, 3.1.2

[80] Haiping Lu, Konstantinos N Plataniotis, and Anastasios N Venetsanopoulos. Multilinear principal component analysis of tensor objects for recognition. In *ICPR*, volume 2, pages 776–779, 2006. 4.1.2

[81] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963. 4.2.3

[82] Jose R Marti, Hamed Ahmadi, and Lincol Bashualdo. Linear power-flow formulation based on a voltage-dependent load model. *IEEE Transactions on Power Delivery*, 28(3): 1682–1690, 2013. 4.1.2

[83] Yasuko Matsubara and Yasushi Sakurai. Regime shifts in streams: Real-time forecasting of co-evolving time sequences. In *ACM SIGKDD*, pages 1045–1054, 2016. 4.1.2

[84] Yasuko Matsubara, Yasushi Sakurai, Christos Faloutsos, Tomoharu Iwata, and Masatoshi Yoshikawa. Fast mining and forecasting of complex time-stamped events. In *ACM*

SIGKDD, pages 271–279, 2012. 4.1.2

[85] Yasuko Matsubara, Yasushi Sakurai, Willem G Van Panhuis, and Christos Faloutsos. Funnel: automatic mining of spatially coevolving epidemics. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 105–114. ACM, 2014. 2.2.2, 4.1.2

[86] F.P. Miller, A.F. Vandome, and J. McBrewster. *Nyquist-Shannon Sampling Theorem: Aliasing, Sine Wave, Signal Processing, Nyquist Rate, Nyquist Frequency, Sampling Rate, Shannon-Hartley Theorem, Whittaker-Shannon Interpolation Formula, Reconstruction from Zero Crossings*. Alphascript Publishing, 2010. ISBN 9786130045814. URL `https://books.google.com/books?id=lPvtQQAACAAJ`. 2.3.4

[87] M Zuhair Nashed. *Generalized Inverses and Applications: Proceedings of an Advanced Seminar Sponsored by the Mathematics Research Center, the University of Wisconsin Madison, October 8-10, 1973*. Number 32. Elsevier, 2014. 2.3.2

[88] Joanna Nowicka-Zagrajek and Rafał Weron. Modeling electricity loads in california: Arma models with hyperbolic noise. *Signal Processing*, 82(12):1903–1915, 2002. 4.1.2

[89] Thom F Oostendorp and Adriaan Van Oosterom. Source parameter estimation in inhomogeneous volume conductors of arbitrary shape. *IEEE Transactions on Biomedical Engineering*, 36(3):382–391, 1989. 3.1.1

[90] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009. ISBN 0131988425, 9780131988422. 2.4.2, 2.4.2, 2.4.2

[91] Antigoni Panagiotopoulou and Vassilis Anastassopoulos. Super-resolution image reconstruction techniques: Trade-offs between the data-fidelity and regularization terms. *Information Fusion*, 13(3):185–195, 2012. 2.1.2, 2.3.2, 2.4.2

[92] Amritanshu Pandey, Marko Jereminov, Xin Li, Gabriela Hug, and Larry Pileggi. Aggregated load and generation equivalent circuit models with semi-empirical data fitting. In *Green Energy and Systems Conference (IGSEC), 2016 IEEE*, pages 1–6, 2016. 4.1.2, 4.2.2, 4.2.4

[93] Amritanshu Pandey, Marko Jereminov, Gabriela Hug, and Larry Pileggi. Improving power flow robustness via circuit simulation methods. In *PES General Meeting*, 2017. 4.1.2

[94] Spiros Papadimitriou, Anthony Brockwell, and Christos Faloutsos. Adaptive, hands-off stream mining. In *VLDB*, pages 560–571, 2003. 4.1.2

[95] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. Parcube: Sparse parallelizable tensor decompositions. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 521–536. Springer, 2012. 5.1.2

[96] Neal Parikh and Stephen P. Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014. 2

[97] JH Park, YM Park, and KY Lee. Composite modeling for adaptive short-term load forecasting. *IEEE Transactions on Power Systems*, 6(2):450–457, 1991. 4.1.2

[98] Vorapoj Patanavijit and Somchai Jitapunkul. A lorentzian stochastic estimation for a ro-

bust iterative multiframe super-resolution reconstruction with lorentzian-tikhonov regularization. *EURASIP Journal on Advances in Signal Processing*, 2007(1):1–21, 2007. 2.1.2

[99] B Aditya Prakash, Deepayan Chakrabarti, Nicholas C Valler, Michalis Faloutsos, and Christos Faloutsos. Threshold conditions for arbitrary cascade models on arbitrary networks. *Knowledge and information systems*, 33(3):549–575, 2012. 2.2.2

[100] Tycho Project:. https://www.tycho.pitt.edu. (document), 2.1, 2.1.1, 2.1.2, 2.1.4, 2.2.1, 2.2.2, 2.2.4, 2.16, 2.3.1, 2.3.2, 2.3.4, 2.4.1, 2.4.4

[101] Meisam Razaviyayn, Mingyi Hong, and Zhi-Quan Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23(2):1126–1153, 2013. A.2

[102] Theodoros Rekatsinas, Manas Joglekar, Hector Garcia-Molina, Aditya Parameswaran, and Christopher Ré. Slimfast: Guaranteed results for data fusion and source reliability. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1399–1414. ACM, 2017. 2.4.2

[103] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978. 2.3.3

[104] Mark Rogers, Lei Li, and Stuart J Russell. Multilinear dynamical systems for tensor time series. In *NIPS*, pages 2634–2642, 2013. 4.1

[105] Subhasis Saha. Image compression – from dct to wavelets: A review. *Crossroads*, 6 (3):12–21, March 2000. ISSN 1528-4972. doi: 10.1145/331624.331630. URL `http://doi.acm.org/10.1145/331624.331630`. 2.4.2

[106] R Alberto Salinas, Christopher Richardson, Mongi A Abidi, and Ralph C Gonzalez. Data fusion: Color edge detection and surface reconstruction through regularization. *IEEE Transactions on Industrial Electronics*, 43(3):355–363, 1996. 2.1.2, 2.3.2

[107] C. Sax and P. Steiner. Temporal disaggregation of time series. *The R Journal*, 41(5), 2013. 2.1.2, 2.3.2

[108] Christoph Sax and Peter Steiner. Temporal disaggregation of time series. *The R Journal*, 5(2):80–87, 2003. 2.4.1

[109] A Selakov, D Cvijetinovic, L Milovic, S Mellon, and D Bekut. Hybrid pso–svm method for short-term load forecasting during periods with significant temperature variations in city of burbank. *Applied Soft Computing*, 16:80–88, 2014. 4.3

[110] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):17–37, 2017. 5.1.2

[111] Miguel Simoes, Jose Bioucas-Dias, Luis B Almeida, and Jocelyn Chanussot. A convex formulation for hyperspectral image superresolution via subspace-based regularization. *IEEE Transactions on Geoscience and Remote Sensing*, 53(6):3373–3388, 2015. 3.1.2

[112] Umut Simsekli, Beyza Ermis, A Taylan Cemgil, and Evrim Acar. Optimal weight learning for coupled tensor factorization with mixed divergences. In *21st European Signal*

Processing Conference (EUSIPCO 2013), pages 1–5. IEEE, 2013. 5.1.2

[113] Michael Smith. Modeling and short-term forecasting of new south wales electricity system load. *Journal of Business & Economic Statistics*, 18(4):465–478, 2000. 4.1.2

[114] SA Soliman, S Persaud, K El-Nagar, and ME El-Hawary. Application of least absolute value parameter estimation based on linear programming to short-term load forecasting. *International Journal of Electrical Power & Energy Systems*, 19(3):209–216, 1997. 4.1.2

[115] Hyun Ah Song, Bryan Hooi, Marko Jereminov, Amritanshu Pandey, Larry Pileggi, and Christos Faloutsos. Powercast: Mining and forecasting power grid sequences. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 606–621. Springer, 2017. (document), 4.1, 4.2.2, 4.3, 4.2.4, 4.2.4

[116] Hyun Ah Song, Fan Yang, Zongge Liu, Wilbert Van Panhuis, Nicholas Sidiropoulos, Christos Faloutsos, and Vladimir Zadorozhny. Gb-r: A fast and effective gray-box reconstruction of cascade time-series. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 494–501. IEEE, 2017. (document), 2.2

[117] Emma M. Stewart, Anna Liao, and Ciaran Roberts. Open upmu: A real world reference distribution micro-phasor measurement unit data set for research and application development. 10/2016 2016. 4.1.1, 4.1.1, 4.1.4, 4.2.4

[118] Petre Stoica and Randolph L Moses. *Introduction to spectral analysis*, volume 1. Prentice hall Upper Saddle River, 1997. 2.3.1, 2.3.2, 2.3.3

[119] Jing Sui, Tulay Adali, Qingbao Yu, Jiayu Chen, and Vince D Calhoun. A review of multivariate methods for multimodal fusion of brain imaging data. *Journal of neuroscience methods*, 204(1):68–81, 2012. 3.1.2

[120] Jimeng Sun, Dacheng Tao, and Christos Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 374–383. ACM, 2006. 5.1.2

[121] Tsubasa Takahashi, Bryan Hooi, and Christos Faloutsos. Autocyclone: Automatic mining of cyclic online activities with robust tensor factorization. In *WWW*, pages 213–221, 2017. 4.1, 4.1.2

[122] Dacheng Tao, Steve Maybank, Weiming Hu, and Xuelong Li. Stable third-order tensor representation for color image classification. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 641–644. IEEE Computer Society, 2005. 5.1.2

[123] Dacheng Tao, Mingli Song, Xuelong Li, Jialie Shen, Jimeng Sun, Xindong Wu, Christos Faloutsos, and Stephen J Maybank. Bayesian tensor approach for 3-d face modeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(10):1397–1410, 2008. 4.1.2

[124] Kamil Uludag and Alard Roebroeck. General overview on the merits of multimodal neuroimaging data fusion. *NeuroImage*, 102:3–10, 2014. 3.1.2

[125] Graham Upton and Ian Cook. *Understanding statistics*. Oxford University Press, 1996. 2

[126] W. van Panhuis, J. Grefenstette, S. Jung, N. Chok, A. Cross, H. Eng, B. Lee,

V. Zadorozhny, S. Brown, D. Cummings, and D. Burke. Contagious diseases in the united states from 1888 to the present. *The New England Journal of Medicine*, 369(22), 2013. 2.1.2, 2.2.2, 2.2.4, 2.3.2

[127] Eric Van Reeth, Ivan WK Tham, Cher Heng Tan, and Chueh Loo Poh. Super-resolution in magnetic resonance imaging: A review. *Concepts in Magnetic Resonance Part A*, 40 (6):306–325, 2012. 3.1.1

[128] Martin Vetterli, Pina Marziliano, and Thierry Blu. Sampling signals with finite rate of innovation. *IEEE transactions on Signal Processing*, 50(6):1417–1428, 2002. 2.3.2

[129] Peter B Walker, Sean Gilpin, Sidney Fooshee, and Ian Davidson. Constrained tensor decomposition via guidance: Increased inter and intra-group reliability in fmri analyses. In *International Conference on Augmented Cognition*, pages 361–369. Springer, 2015. 5.1.2

[130] Peng Wang, Haixun Wang, and Wei Wang. Finding semantics in time series. In *ACM SIGMOD*, pages 385–396, 2011. 4.1

[131] Yichen Wang, Robert Chen, Joydeep Ghosh, Joshua C Denny, Abel Kho, You Chen, Bradley A Malin, and Jimeng Sun. Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *ACM SIGKDD*, pages 1265–1274, 2015. 4.1.2

[132] Andrew B Watson. Image compression using the discrete cosine transform. *Mathematica journal*, 4(1):81, 1994. 2.4.2

[133] Max Welling and Markus Weber. Positive tensor factorization. *Pattern Recognition Letters*, 22(12):1255–1261, 2001. 5.1.2

[134] Peter R Winters. Forecasting sales by exponentially weighted moving averages. *Management science*, 6(3):324–342, 1960. 4.2.2, 4.2.2

[135] Eliot Wycoff, Tsung-Han Chan, Kui Jia, Wing-Kin Ma, and Yi Ma. A non-negative sparse promoting algorithm for high resolution hyperspectral imaging. In *2013 IEEE ICASSP*, pages 1409–1413, 2013. 3.1.2

[136] Fan Yang, Hyun Ah Song, Zongge Liu, Christos Faloutsos, Vladimir Zadorozhny, and Nicholas Sidiropoulos. Ares: Automatic disaggregation of historical data. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 65–76. IEEE, 2018. (document), 2.3

[137] Hong-Tzer Yang and Chao-Ming Huang. A new short-term load forecasting approach using self-organizing fuzzy armax models. *IEEE Transactions on Power Systems*, 13(1): 217–225, 1998. 4.1, 4.1.2

[138] Yusuf Kenan Yilmaz. *Generalized tensor factorization*. PhD thesis, Citeseer, 2012. 5.1.2

[139] V. Zadorozhny and G. Grant. A systematic approach to reliability assessment in integrated databases. *Journal of Intelligent Information Systems*, 46(3), 2016. 2.3.2

[140] V. Zadorozhny and Y.-F. Hsu. Conflict-aware fusion of historical data. In *Proc. of the 5th International Conference on Scalable Uncertainty Management (SUM11)*, 2013. 2.1.2, 2.2.1, 2.2.2, 2.2.4

[141] Vladimir Zadorozhny and Michael Lewis. Information fusion for usar operations based on crowdsourcing. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 1450–1457. IEEE, 2013. 2.3.2, 2.4.2

[142] Pei Zhang, Xiaoyu Wu, Xiaojun Wang, and Sheng Bi. Short-term load forecasting based on big data technologies. *CSEE Journal of Power and Energy Systems*, 1(3):59–67, 2015. 4.3

[143] Ying Zhu, Dorin Comaniciu, Martin Pellkofer, and Thorsten Koehler. Reliable detection of overtaking vehicles using robust information fusion. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):401–414, 2006. 2.1.2, 2.3.2