Spatio Temporal Signal Processing

Harnessing public transit for effective spatiotemporal sampling, estimation and prediction.

Doctoral Thesis

by

CHARUL

ROLL NUMBER :- MT16089

Electronics and Communication

Indraprastha Institute of Information Technology

Delhi

ADVISOR
Dr. PRAVESH BIYANI

YEAR OF SUBMISSION
2023

Certificate

This is to certify that the thesis titled **"Spatio Temporal Signal Processing: Harnessing public transit for effective spatiotemporal sampling, estimation and prediction."** being submitted by **Charul**, Roll Number **MT16089**, to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of Doctor of Philosophy, is an original research work carried out by her under my supervision. In my opinion, the thesis has reached the standard fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree or diploma.

September, 2023

Dr. Pravesh Biyani

Deptt. of Electronics and Communication Engineering,

Indraprastha Institute of Information Technology Delhi-110020, India.

Declaration

This is certified that the thesis entitled **"Spatio Temporal Signal Processing: Harnessing public transit for effective spatiotemporal sampling, estimation and prediction."** being submitted by me to the Indraprastha Institute of Information Technology Delhi, for the award of degree of Doctor of Philosophy, is a bonafide work carried out by me. This research work has been carried out under the supervision of **Dr. Pravesh Biyani**. The study pertaining to this thesis has not been submitted in part or in full, to any other University or Institution for the award of any other degree.

September, 2023

Charul

PhD Student,

Deptt. of Electronics and Communication Engineering,

Indraprastha Institute of Information Technology Delhi-110020, India.

Acknowledgements

I want to take this opportunity to express my heartfelt gratitude to the individuals who have been the pillars of support throughout my academic journey. Their support, guidance, and belief in my abilities have made this achievement possible.

Firstly, I am grateful to my advisor, Dr. Pravesh Biyani. His expertise, encouragement, and insightful feedback have truly shaped the direction of this thesis. Under their guidance, I have acquired the necessary technical skills and learned the art of formulating precise research inquiries and fostering genuine enthusiasm for my research. I would also like to thank the members of my thesis committee, Dr. Saket Anand and Dr. Tavpritesh, for their insightful feedback, which has greatly enriched the quality of this work. I am also thankful for the opportunity to collaborate with Dr. Ketan Rajawat, whose expertise played a pivotal role in the research presented in this thesis.

I am grateful to my parents, my first teachers, for their support and encouragement. My mother has shaped me into a better human being, while my father has imparted life's invaluable lessons. I am forever indebted to them for their faith in my journey. I am grateful to my brother, Ripul, my perpetual source of happiness, for his positive influence in my life. To my husband, Aayush Tyagi, his patience, understanding, and encouragement have been my anchor. His belief in me has been a driving force.

I would also like to extend my deepest gratitude to Dr. Dhananjay and Karan Dhingra for their exceptional technical guidance and invaluable life advice. They have always been my unwavering support during crucial times. Additionally, I extend my thanks to Kshitij and Rajan for their assistance during project deployment discussions, accompanying me to the bus depot. Working with both of them has been an absolute delight. I gained a wealth of knowledge from my initial labmates, and credit is due to my advisor, Dr. Pravesh Biyani, for assigning the lab to me. Milan was the perfect guide during that period, and I learned a

lot from his expertise. I learned dedication from Sneihil Gopal, sincerity from Anil, and enjoying work from Tanya. Anupriya embodies a delightful balance, always bringing fun but also serving as a mentor when needed. I also want to thank my labmates - Aanchal Mongia, Shikha Singh, Anand Singh, Megha Gupta Gaur, Pooja Gupta and Shalini Sharma for all the chit-chats and support.

Lastly, I extend my acknowledgement to the Indraprastha Institute of Information Technology, Delhi (IIIT-D), for offering outstanding infrastructure and a conducive research environment. A special thank you to Priti ma'am and Ashutosh sir for their exceptional efforts in ensuring seamless administrative processes.

Abstract

Public transportation can be a potential source of generating a tremendous amount of data as a part of its daily operation. GPS (Global Positioning System) installed system can be used to track the position of buses and thereby collect a massive stream of traffic speed/ETA (Estimated Time of Arrival) data. An alternate approach called drive-by sensing where sensors can be installed on moving vehicles is a way of collecting highly-granular space/time datasets that can be merged with public transportation (buses) to provide a cost-effective solution. This approach can be used to sense a wide range of phenomena, including traffic speed, air pollution, road lighting, street surface quality, unsafe pedestrian movement, record parking violations, traffic congestion, and crowd flows. Our work mainly focuses on traffic speed and air quality data sensing. The data sampled using sensor sources contain missing values due to sensor malfunctioning or the irregularity in the sensor measurements. The missing data percentage further shoots up in case of drive-by sensing data collection.

In this work, we explored three problems spatiotemporal sampling, estimation and prediction for effective and reliable public transportation data acquisition and analysis. First, we propose a Robust Variational Bayesian Subspace Filtering framework for missing data estimation and outlier removal. We also propose an Extreme Matrix completion for missing data estimation using Variational Bayesian Filtering with Subspace information for a higher percentage of missing data. We showed that incorporating the previous subspace information can reduce the sampling complexity of the data; therefore, it can be a potential algorithm to estimate the data in case of moving sensors. Second, we propose Regressive Facility Location, a sampling algorithm to pick sets of paths (using vehicles) that perform representative sampling in space and time. Third, we propose a deep learning-based

generative model that predicts the ETA information of buses for a trip and updates it as the trip progresses based on the real-time information.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

> Without big data, you are blind and deaf and in the middle of a freeway
>
> *Geoffrey Moore*

The transportation network accounts for about 64% of global oil consumption, 27% of all energy use, and 23% of the world's energy-related carbon dioxide emissions [4]. Therefore, sustainable transportation is a direction in progressing towards sustainable development, fulfilling the social and economic requirements of the people while respecting the needs of future generations. Sustainable transportation aims to support society's mobility needs that are least damageable to the environment. Transportation built around public transit is a way to attain sustainable mobility that can provide economic accessibility along with lower emissions and congestion on the roads. Many countries have targeted increasing public transit usage to achieve balanced, sustainable mobility goals. For example, Dubai set a target to increase the public transport modal share to 30% by 2030, Malaysia set an increase in the public transport modal share to 40% by 2030, Copenhagen's vision is that at least 1/3 of all driven traffic in the city should be made by bicycle 1/3 by public transport and not more than 1/3 by car [5]. Promoting public transit to users is crucially dependent on its reliability. One of the primary metrics that measure the reliability of public

transit is the predictability of its vehicles in reaching specific pre-determined locations. A predictable and reliable public transportation attracts more users, thereby increasing the economic viability of the transit as well as reducing congestion on the road, a huge urban challenge, especially in the developing world. Establishing reliability in public transit is built around gathering, analyzing and providing data to users so that they can optimize their journey and authorities to plan and operate efficient transit networks. In this thesis, we work on delivering solutions to reliable transportation by estimating and predicting network traffic state, which is a challenging problem, especially in developing countries due to randomness in road traffic, non-homogeneous infrastructure and the presence of multimodal transport .

A critical aspect of sustainable mobility is sustainability in terms of its environmental impacts. Promoting public transit can help in the reduction of private vehicles, which in turn can reduce the environmental impact. Further, a transportation network can be used to monitor the environmental impact, e.g. monitoring of air quality of a region. Air quality has drawn attention in recent years because of its effect on the health of the people, which includes cardiovascular diseases, lung cancer etc. It is crucial to assess the level of pollution in a region to check if it meets the air quality standards. The air quality measurements are required to make strategies for emission control, verify strategies for emission control, and provide the air quality index data (AQI) to the general public and authorities for decision-making. In this thesis, we work towards providing an effective strategy for sensing and providing AQI data using a transportation network.

Overall, this thesis provides a sustainable solution for effective sensing using public transit and delivers reliable real-time and future data to the user. The data obtained using a transportation network includes spatiotemporal dimensions and exhibits correlation in the spatial and temporal dimensions. We discuss the spatiotemporal data and the associated correlation in the next section. We then discuss the spatiotemporal mobility data collection

Fig. 1.1 Traffic Data

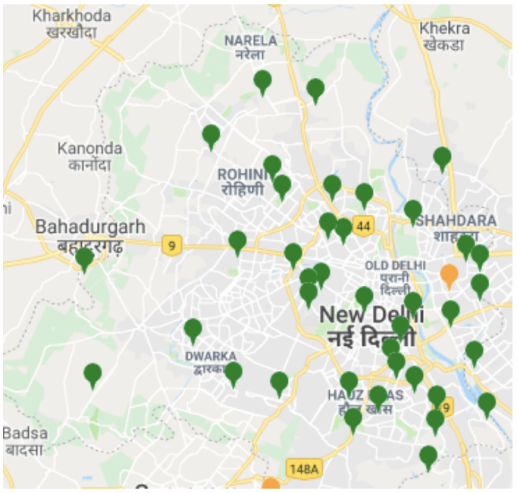


Fig. 1.2 Air Quality Data

using public transit and underline the motivating factors leading to the research pursued in this thesis. Lastly, we briefly explain the objectives and contributions of this thesis. In the end, we provide an overview of follow-up chapters.

1.1 Spatiotemporal Data

A large amount of spatiotemporal data is collected in diverse domains like transportation, remote sensing, climate science, social sciences, neuroscience etc. Spatiotemporal data samples are not independent and identically distributed, instead follow a structure and are correlated in space and time. Spatially associated patterns are often observed in the data measured across different locations. Such spatial patterns change over time, and therefore the temporal factor of the signal needs to be examined along with the spatial aspect. In this section, we discuss the spatial, temporal and spatiotemporal dimensions of a dataset.

Spatial dimension Points, and links can be used to denote the spatial dimension of an observation as shown in Fig. 1.1-1.2. Point is the most commonly used spatial dimension as shown in Fig. 1.2, where the observation corresponding to a particular point l will be the data reading for that point l. Link can be used to denote a road segment as shown in

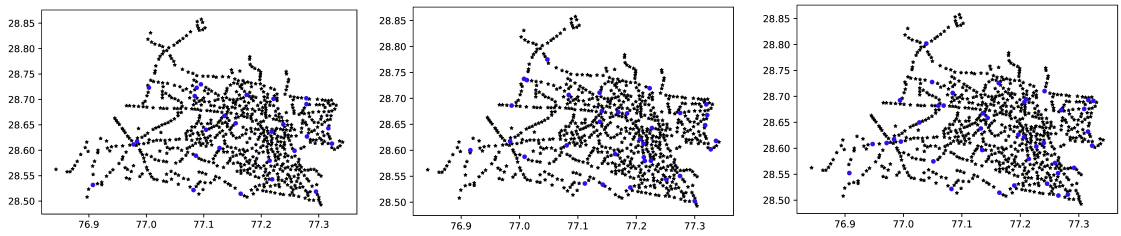


Fig. 1.3 Different time stamps for a spatiotemporal signal. All the locations are denoted by black. Blue denotes the sampled observations at a given time stamp.

Fig. 1.1, where the traffic data (e.g. speed) on a road segment l can be a proxy of data for that road segment l. For L spatial observations, we can refer to the point or the links from $l = 1$ to L. Tobler's first law of geography states that "Everything is related to everything else, but near things are more related than distant things" [6]. This means that there exists a spatial dependence in the data, and data is spatially distributed in a non-random manner. In this thesis, we exploit the correlation in spatial observations.

Temporal dimemsion Spatial observations vary in time. However, there is a correlation in the spatial observations over time. For every spatial observation l, we can record the observations temporally. The data arriving can be sampled for different time stamps or averaged for different time intervals. The spatial observation evolves slowly with time $t = 1$ to T, and we will exploit this temporal correlation in this thesis.

Spatiotemporal Dimension Sensing the spatiotemporal data using a static sensor can be regarded as a sequence of temporal dimension data for all the spatial dimensions. Sensing the spatiotemporal data using a moving vehicle where a vehicle follows a trajectory can be regarded as a sequence that varies in spatiotemporal dimensions. An example of such trajectory data is shown in Fig 1.3, where for different time instances, we observe spatial data at different locations. In both cases, there is a spatiotemporal correlation in the data. In this thesis, we will exploit this spatiotemporal correlation in the data.

1.2 Mobility Data Acquisition using Public Transportation

Public transit, e.g. buses, can be used as GPS (Global Positioning System) probe vehicles that relay back the vehicle's position along with the sensed data to a server at different time intervals to collect spatiotemporal data. We discuss two ways of acquiring mobility data in this section. One is the transit data that is used by the transit agency to get real-time bus data. Another is the drive-by sensing, where the moving sensor paradigm can be used to obtain high coverage spatiotemporal data map without needing an expensive setup of hundreds of static monitors [7, 8]. Sensors can be installed on public transit for real-time environmental monitoring of cities.

- Transit data

  Rapid population growth in many major cities results in a strain on the existing public transportation infrastructure. Collecting the mobility data and providing information to the public can be beneficial for public comfort. This also provides a doorstep for better resource management, planning etc. Therefore, public transportation operators are motivated to publish their transportation network and schedule data in electronic form, accessible to the public via a unified API (Application programming interface) known as the transit data. The data format is GTFS (General Transit Feed Specification) which has become an industry standard for transit data. One such open transit data is available for Delhi, India [9], which provides static and real-time bus transit data. Static data refers to the information about the schedule, routes, and stops provided by bus agencies. Real-Time Data provides the live GPS coordinates and registration number of each bus running at that particular moment. We will explore the Delhi bus transit data [9] in this thesis.

- Selective Drive-by sensing

A large number of static monitors are required to sample different locations in a geographical area, thereby making it a costly option. Dense spatiotemporal coverage can be achieved using only a fraction of static sensors by deploying them on the moving vehicles, known as the drive-by sensing paradigm. We do not need static monitors at all times and in all locations; one may "move" these monitors to sample at other locations. Since one monitor can be used to sample many locations during different timestamps, the actual number of sensors needed is just a fraction of the total number of nodes. Vehicle-based drive-by sensing emerged as a paradigm to create high granular spatiotemporal datasets [10]. Drive-by sensing can be employed using dedicated vehicles [11]. However, this increases the cost of the data collection. Therefore, we explore the drive-by sensing paradigm using public transit (buses) in this thesis.

1.3 Research Overview

With the advent of smartphones, public transportation services, as well as private on-demand transportation companies, are increasingly relying on the availability of real-time spatiotemporal traffic maps for resource allocation and logistics. Further, authorities can take effective strategies against air pollution and climate change based on spatiotemporal air quality data. The availability of reliable real-time spatiotemporal data can attract more users and help them make more informed decisions. So it is crucial to provide a spatiotemporal data map of an area, where a user can avail the data for any location and for any time period. Building an anytime-anywhere real-time dense data map for a city requires first sensing/sampling the spatiotemporal data using GPS enabled probe vehicles that upload the observations and corresponding location tags at sporadic times to the server, as shown in Fig 1.4.

The spatiotemporal data acquired using the transit as discussed in section 1.2 is incomplete and is often ridden with outliers. Further, the problem of incomplete data shoots up when employing drive-by sensing. Therefore, the next task to provide dense spatiotemporal maps is to estimate spatiotemporal data at locations and times where no measurements are available in the presence of outliers. Overall, providing reliable real-time spatiotemporal dense data maps requires jointly solving the problem of spatiotemporal sampling and spatiotemporal estimation. Finally, predicting the future spatiotemporal data can further improve the user experience; for example, predicting traffic in the near future is necessary to calculate expected-time-of-arrival (ETA), fastest route, and other related quality service metrics for road users. The future traffic prediction problem becomes particularly challenging in regions with diverse modes of transport, such as in India, where ETA calculations must account for the multimodal nature of traffic. This thesis delivers solutions to collect and provide accurate spatiotemporal data to the users. As shown in figure 1.4, firstly, we can employ spatiotemporal sampling framework to collect the data by selected GPS vehicles and develop a model to estimate and predict the spatiotemporal data. The next section will elaborate on the spatiotemporal sampling, estimation and prediction problems, and our thesis contribution.

1.4 Thesis Contribution

In this dissertation, we investigate the following problems

- How can we estimate the missing data by exploiting the spatiotemporal structure of the data (Spatiotemporal Estimation).
- How can we select the set of vehicles that provide an effective spatiotemporal sampling (Spatiotemporal Sampling).
- How can we predict the future spatiotemporal data (Spatiotemporal Prediction)

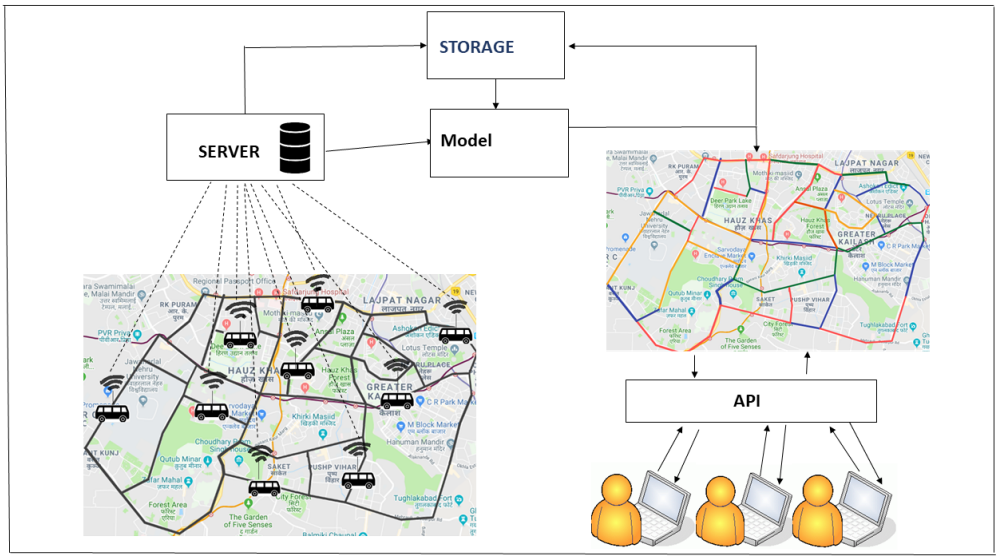


Fig. 1.4 Setup for dense maps. First, use spatiotemporal sampling to collect the data by selected GPS vehicles and send it to the server. Then use a spatiotemporal estimation and prediction model.

1.4.1 Spatiotemporal Estimation

Real-world data collected from various sensors are often incomplete and noisy, with the possibility of outliers. Consequently, such data naturally demands filling the missing entries and removal/segregation of outliers to perform tasks in a meaningful manner. Examples of applications that benefit from imputing missing data include road traffic estimation and prediction, air quality measurement across the city network, and electricity load prediction and estimation. The collected data in these applications can be represented in the form of a spatiotemporal matrix with missing entries that need to be estimated by exploiting the spatiotemporal structure. The inference of the missing entries is further affected by the presence of outliers in the sampled data. Therefore, the estimation of the spatiotemporal data in the presence of outliers is another problem that needs attention. A more recent way of collecting the sensor data is employing drive-by sensing. The drive-by sensing scheme uses relatively fewer sensors resulting in high data "gaps" in both spatial and temporal

dimensions. This motivates the problem of extreme spatiotemporal estimation, where the percentage of data sampled may be as low as 10%.

Our Contribution

We explored the traffic and air quality datasets in our work [1–3, 12, 13]. The algorithm proposed in this work can be applied to the static sensor as well as drive-by sensing missing data problems as well. To extrapolate the missing entries, we exploit the underlying low rank and slowly time-varying structure of the spatiotemporal traffic/air quality data. Jointly exploiting the spatiotemporal and periodic structure, which is generally not captured by classical matrix completion approaches, can improve the imputation performance of sensor data in such real-world conditions. We propose the Variational Bayesian Subspace Filtering (VBSF), a low-rank subspace filtering approach for data imputation. We consider low-rank matrices whose underlying subspace evolves according to a state-space model. As incomplete columns of the data matrix arrive sequentially over time, the low-rank components, as well as the state-space model, are learned in an online fashion using the variational Bayes formalism. Moreover, we extended the probabilistic model to the Robust Variational Bayesian Subspace Filtering (RVBSF) case that is flexible enough to allow imputation while handling outliers. We further capture the periodic evolution in the data by a penalty function on the slowly varying subspace using Variational Bayesian Filtering with Subspace Information (VBFSI). This improves the performance for high missing data, referred to as extreme matrix completion in this work.

1.4.2 Spatiotemporal Sampling

Fixed static sensing is a primary way to monitor environmental data like air quality in cities. However, to obtain a dense spatial coverage, a large number of static monitors are required, thereby making it a costly option. Dense spatiotemporal coverage can be

achieved using only a fraction of static sensors by deploying them on the moving vehicles, known as the drive-by sensing paradigm. There are two steps to create a dense data map for an area. The first is to sample the best representative sampling set of the city in space and time, and the second is the extrapolation or imputation of the data based on the sampled set. The second problem is discussed already in section 1.4.1 as the inherent temporal and spatial redundancy available in the spatiotemporal data can be used to impute the remaining (incomplete) data by utilizing the matrix completion techniques and obtaining a dense data map in a fraction of the corresponding fixed sensing cost. However, the accuracy of imputation is dependent on the extent to which the moving sensors capture the inherent structure of the spatiotemporal matrix. There has been limited research on the spatiotemporal sampling problem where the challenge is to pick those sets of paths (using vehicles) that perform representative sampling in space and time.

Our Contribution

Most works [14–17] in the current literature for vehicle subset selection focus on maximizing the spatiotemporal coverage by maximizing the number of samples for different locations and time stamps, which is not an effective representative sampling strategy and results in noisy imputations. We present a novel regressive facility location-based drive-by sensing, an efficient vehicle selection framework that incorporates the smoothness in neighbouring locations and autoregressive time correlation while selecting the optimal set of vehicles for effective spatiotemporal sampling. We show that the proposed drive-by sensing problem is submodular, thereby lending itself to a greedy algorithm but with performance guarantees. We address the problem of the choice of efficient drive-by-sensing routes for Delhi. In our work, we investigate the usage of public transit buses for drive-by-sensing. We pick those routes using the proposed regressive facility location that exploits the spatiotemporal structure in the air quality data resulting in an effective imputation and

therefore achieving the required dense air quality map. We illustrate that the proposed method samples the representative spatiotemporal data in turn reducing the extrapolation error. Our approach, therefore, has the potential to provide cost-effective dense air quality maps.

1.4.3 Spatiotemporal Prediction

The next problem we focus on is predicting future spatiotemporal data by exploiting the historic spatiotemporal patterns. This thesis explores the use-case of predicting the accurate spatiotemporal ETA data. From the point of view of the transit operators, predictability is key to maintaining its efficiency. From the passenger's perspective, predictability is generally measured by the accurate determination of the expected time of arrival (ETA) of a vehicle on a route. Predicting the accurate expected time of arrival (ETA) information is crucial in maintaining the quality of service of any public transit.

Our Contribution

We present a simple but robust model for spatiotemporal ETA prediction for a bus route that only relies on the historical data of the particular route. We propose a system that generates ETA information for a trip and updates it as the trip progresses based on the real-time information. We train a deep learning based generative model that learns the probability distribution of ETA data across trips and conditional on the current trip information updates the ETA information on the go. Our plug and play model not only captures the non-linearity of the task well but that any transit agency can use without needing any other external data source.

1.5 Thesis Organization

In chapter 2, we present Variational Bayesian Subspace Filtering, a Bayesian model that fits sequential multivariate measurements arising from a low-dimensional time varying subspace. The proposed model learns the underlying subspace and its state-transition matrix. We use this approach to predict real time missing data. In chapter 3, we present a Robust Variational Bayesian Subspace Filtering approach to predict the missing data in the presence of outliers. We also predict the location and value of the outliers in the data. In chapter 4, we present a Variational bayesian approach for extreme matrix completion for high percentage of missing data. We exploit the subspace information of previous days to reduce the sample complexity and improve the estimation performance. In chapter 5, we present regressive facility location based drive by sensing, an efficient vehicle selection framework for effective spatiotemporal sampling. In Chapter 6, we present a deep learning based generative model that learns the probability distribution of spatiotemporal ETA data across trips and conditional on the current trip information predicts the ETA information on the go. Conclusion and future research outline is discussed in Chapter 7.

1.6 Publications of this Thesis

- Paliwal, Charul, and Pravesh Biyani. "To each route its own ETA: A generative modeling framework for ETA prediction." In 2019 IEEE Intelligent Transportation Systems Conference (ITSC), pp. 3076-3081. IEEE, 2019.

- Paliwal, Charul, Uttkarsha Bhatt, Pravesh Biyani, and Ketan Rajawat. "Traffic estimation and prediction via online variational bayesian subspace filtering." IEEE Transactions on Intelligent Transportation Systems 23, no. 5 (2021): 4674-4684.

- C. Paliwal and P. Biyani, Variational Bayesian Filtering for spatiotemporal matrix completion, in NeurIPS Workshop on Gaussian Processes, Spatiotemporal Modeling, and Decision-making Systems, 2022.

- Paliwal, Charul, Pravesh Biyani, and Ketan Rajawat. "Variational Bayesian Filtering with Subspace Information for Extreme Spatio-Temporal Matrix Completion." arXiv preprint arXiv:2201.08307 (2022).

- Paliwal, Charul, and Pravesh Biyani. "Dense Air Quality Maps Using Regressive Facility Location Based Drive By Sensing." arXiv preprint arXiv:2201.09739 (2022).

Chapter 2

Variational Bayesian Subspace Filtering

2.1 Overview

Sensor measurements are often incomplete, noisy due to malfunctions or intermittent errors. Imputation of the missing entries is a critical first step that must be carried out prior to any data analytics. Examples of applications that benefit from such a pre-processing step include estimation/prediction of city-wide road traffic, regional air quality, electricity consumption in power distribution networks and foreground-background separation in videos. For most of these applications, the measurements can be arranged as a matrix, some of whose entries may be missing. Pertinent approaches model the measurements as arising from a low-dimensional subspace whose recovery allows us to reject the noise and impute the missing entries [18–23].

Many real-world applications, including the aforementioned ones, involve time-varying data that arrives in a sequential manner and must be processed as such. As a result, the data matrices arising in such applications comprise of low-dimensional subspaces that evolve over time. State-of-the-art approaches for processing time-varying subspaces can mostly be classified into approaches based on tensor completion [24] and regularized matrix completion [25]. Matrix or tensor completion is applied to data collected as a whole

to impute the missing entries. In contrast, many real-world applications including the ones mentioned above follow a common theme of inherently dynamic, consisting of sequentially arriving data that must be dealt in an online manner. Also, offline algorithms based on matrix completion often involve tuning parameters with very limited possibility of altering them in an online fashion. On the other hand, Bayesian approaches offer a higher degree of flexibility in modeling the data vis-a-vis the matrix completion based approaches. As a result, Bayesian formulation can be potentially more effective in the above mentioned scenarios. In this chapter, we discuss the problem of spatiotemporal estimation, where we estimate the missing data by exploiting the spatiotemporal structure of the sensor data by using a Bayesian approach.

The organisation of this chapter is as follows. We begin in section 2.2 with Variational Bayesian Subspace Filtering (VBSF), where we introduce the approach to estimate the missing data with temporal evolution and motivate the proposed approach to two main applications of traffic prediction and air quality estimation. We then provide the literature review in section 2.3. We discuss the proposed VBSF framework in section 2.4. We provide details of the data and the experiments in section 2.5. Finally, in section 2.6, we conclude the chapter.

2.2 Variational Bayesian Subspace Filtering

This chapter considers low-rank robust subspace filtering approach for matrix imputation. Unlike approaches for subspace tracking, we capture the temporal evolution in the data, a structure available in many applications like traffic, air quality and electricity data etc., through a state space model as shown in Fig. 2.1. As incomplete columns of the data matrix arrive sequentially over time, the low rank components as well as the state-space model are learned in an online fashion using the variational Bayes formalism. In particular, component distributions are chosen to allow automatic relevance determination (ARD)

and unlike the matrix or tensor completion works, the algorithm parameters such as rank, noise powers, and state noise powers need not be specified or tuned. A low-complexity forward-backward algorithm is also proposed that allows the updates to be carried out efficiently. Enhancements to the proposed algorithm, capable of learning time-varying state-transition matrices, operating with a fixed lag are also detailed. Our approach is general and we demonstrate its efficacy on various settings. In particular, we discuss the traffic estimation problem and air quality estimation and show that the variational Bayesian approach can be used to impute road traffic densities and air quality observations from only a few observations. The superior performance of our algorithm vis-a-vis other state of the art subspace tracking and online matrix factorisation algorithms may be attributed to the proposed state space model as well as the flexibility in the data modeling provided by the variational Bayesian approach. In summary, we present VBSF algorithm and demonstrate its ability to perform data modeling, imputation and temporal prediction in an online setting wherein the key algorithmic parameters are automatically tuned.

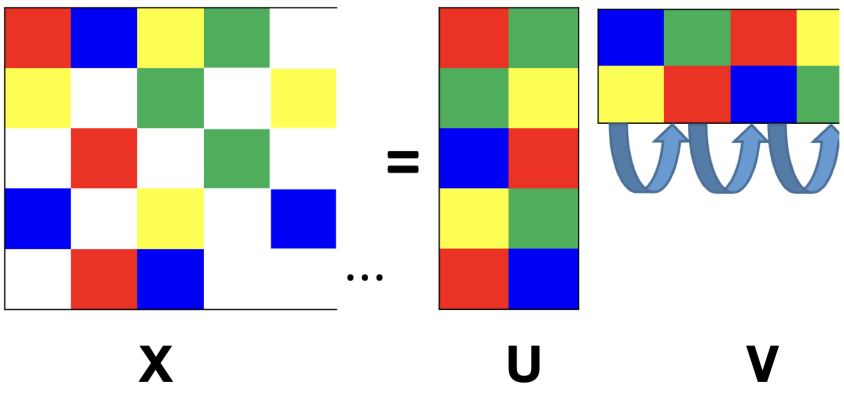


Fig. 2.1 Online Variational Bayesian Filtering

2.2.1 Applications

Traffic Estimation

Traffic estimation is the central component of any urban traffic congestion management system [26]. Comprehensive traffic density maps not only aid in the discovery of traffic flow patterns but are also invaluable for city planners. With the advent of smartphones, public transportation services as well as private on-demand transportation companies, are increasingly relying on the availability of real-time traffic maps for resource allocation and logistics [27]. Such providers rely on GPS enabled probe vehicles and possibly crowd-sourced agents that upload speed measurements and corresponding location tags at sporadic times. The traffic estimation problem entails estimating traffic densities at locations and times where no measurements are available. A class of pertinent approaches have sought to visualize the traffic data as an incomplete matrix or tensor and exploited this correlation to fill-in the missing entries [28–31]. These techniques fail to model the temporal evolution in the data.

Air Quality Estimation

The burning of fossil fuels and crops, as well as industrial emissions, are one of the most significant contributory factors of climate change ([32]). Measuring the local air quality (AQ) is a reliable proxy of measuring the emissions in a given location. Apart from impacting climate, poor AQ is also a major cause of cardiovascular diseases and lung cancer, etc. ([33, 34]). The measurement of AQ parameters in a locality is the first crucial step towards the localization of credible pollution sources like local industrial units, transportation, or even crop burning. Due to the environmental disturbance, communication error or sensor fault, it is inevitable that data may be lost during the collection process [35]. Air quality data exhibits both spatial and temporal correlation thereby generating redundancy (not high rank nature). The presence of correlation may be attributed to

the contribution of common pollution sources in the adjacent geographies. The spatial correlation is unsurprising as there exist finite pollution sources and comparatively many more nodes in a city network. Further, in a given location, the pollution data varies slowly in time. By appropriately modeling the spatio-temporal structure present in the data, one can extrapolate/impute the missing data in an accurate fashion.

2.3 Related work

Variational Bayesian approaches for matrix completion and robust principal component analysis are well known [20, 21, 31, 36–41]. One of the first works considered the measured matrix to be expressible as a product of low-rank matrices, associated with appropriate ARD priors [20] while faster algorithms for similar settings were proposed in [36, 37]. More recently, other approaches towards modeling the measured matrices have also been proposed [38], [39]. Moreover, variational Bayesian approaches have also been applied to road traffic estimation; see e.g. [31]. However, these approaches do not explicitly model the evolution of the underlying subspace. Likewise, none of the existing variational Bayesian approaches for low rank matrix completion model the evolution of the subspace [20, 39, 41]. In contrast to these, the state-space modeling in our work is inspired from [40], where the low-complexity updates were first proposed in the context of linear dynamical models. The VBSF algorithm in the current work extends and generalizes that in [40] to incorporate low-rank structure and outliers.

On a related note, temporal evolution of the additive noise is modeled in [21] using a forgetting factor. Different from [21] however, we use a state-space model to capture the evolution of the underlying subspace. An online Bayesian matrix factorization model is also proposed in [42] wherein the time-stamps are directly incorporated as features. In contrast, the present model is more specific and suited to a slowly time-varying system.

Several non-Bayesian algorithms have been proposed to address the online subspace estimation problem from incomplete observations[19, 42–44]. GROUSE [19] is one of the early approaches that uses an update on the Grassmannian manifold to estimate the subspace.

2.4 Proposed Framework

Data for m locations are collected into the matrix $\mathbf{X} \in \mathbb{R}^{m \times t}$, where t denotes the number of time instances over which measurements are made. For instance, if h measurements are made per day, we have that $t = dh$ after the end of d days. More generally, $\mathbf{X}$ is an incomplete and growing matrix whose columns arrive sequentially over time. Specifically, for each column $\mathbf{x}_\tau$ with $1 \leq \tau \leq t$, only entries from the index set $\Omega_\tau \subset \{1, \ldots, m\}$ are observed. Our goal is to achieve *imputation* which yields $\{\hat{y}_{i\tau}\}_{i \notin \Omega_\tau}$ for $1 \leq \tau \leq t$.

2.4.1 Hierarchical Bayesian Model

We begin with detailing a generative model for the matrix $\mathbf{X}$. The proposed model will capture not only the rank deficient nature of $\mathbf{X}$ [45] but also the temporal correlation between successive columns of $\mathbf{X}$ [46]. Recall that the standard low-rank parametrization of the full matrix $\mathbf{X}$ takes the form $\mathbf{X} = \mathbf{UV}$ where $\mathbf{U} \in \mathbb{R}^{m \times r}$ and $\mathbf{V} \in \mathbb{R}^{r \times t}$ where r is the rank of matrix $\mathbf{X}$. Classical non-negative matrix completion approaches seek to obtain such a factorization. In such algorithms, the choice of r is critical to avoiding underfitting or overfitting.

Within the Bayesian setting however, the measurements are modeled as arising from a distribution with unknown hyper-parameters, while various components or parameters are assigned different prior distributions. The Bayesian framework allows the use of ARD,

wherein associating appropriate priors to the model parameters leads to the pruning of the redundant features [45].

Adopting a Gaussian noise model, the entries of $\mathbf{X}$ are generated to capture the low rankness in the data. The likelihood distribution is obtained as

$$p(\mathbf{x}_{i\tau} \mid \mathbf{u}_{i\cdot}, \mathbf{v}_\tau, \beta) = \mathcal{N}(\mathbf{x}_{i\tau} \mid \mathbf{v}_\tau^T \mathbf{u}_{i\cdot}, \beta^{-1}) \qquad i \in \Omega_\tau \tag{2.1}$$

for all $\tau \geq 1$, where $\mathbf{U} \in \mathbb{R}^{m\times r}$, $\mathbf{V} \in \mathbb{R}^{r\times t}$, and $\beta \in \mathbb{R}_{++}$ are the (hidden) problem parameters. Unlike the deterministic setting however, the rank hyper-parameter r is not critical to the imputation or prediction accuracy but is only required to be chosen according to computational considerations. The temporal evolution of the entries of $\mathbf{X}$ is modeled by making the columns of $\mathbf{V}$ adhere to the following first-order autoregressive model:

$$\mathbf{v}_\tau = \mathbf{F}\mathbf{v}_{\tau-1} + noise \tag{2.2}$$

In the Bayesian setting, we model the autoregressive model as

$$p(\mathbf{v}_\tau \mid \mathbf{F}, \mathbf{v}_{\tau-1}) = \mathcal{N}(\mathbf{v}_\tau \mid \mathbf{F}\mathbf{v}_{\tau-1}, \mathbf{I}_r) \qquad 2 \leq \tau \leq t \tag{2.3}$$

for $\tau \geq 2$, where $\mathbf{F} \in \mathbb{R}^{r\times r}$ is again a problem parameter. Here, $\mathbf{F}$ captures the temporal structure of the underlying subspace and is learned from the data itself. The scaling ambiguity present in matrix factorization allows the transition matrix $\mathbf{F}$ to capture both slow and fast variations in $\mathbf{v}_\tau$ without the need to explicitly model the state noise variance. It follows from (2.3) that the conditional distribution of $\mathbf{v}_\tau$ given $\mathbf{F}$ is given by

$$p(\mathbf{V} \mid \mathbf{F}) = \mathcal{N}(\mathbf{v}_1; \mu_1, \Lambda_1) \prod_{\tau=2}^{t} \mathcal{N}(\mathbf{v}_\tau \mid \mathbf{F}\mathbf{v}_{\tau-1}, \mathbf{I}_r). \tag{2.4}$$

Observe that the model complexity depends on the rank r, which is also the number of columns in $\mathbf{U}$ and $\mathbf{F}$. In order to ensure the value of r is learned in a data-driven fashion, the columns of $\mathbf{U}$ and $\mathbf{F}$ are assigned multivariate Gaussian priors with column-specific precisions, i.e.,

$$p(\mathbf{U} \mid \gamma) = \prod_{i=1}^{r} \mathcal{N}(\mathbf{u}_i \mid 0, \gamma_i^{-1}\mathbf{I}_m) \tag{2.5}$$

$$p(\mathbf{F} \mid \upsilon) = \prod_{i=1}^{r} \mathcal{N}(\mathbf{f}_i \mid 0, \upsilon_i^{-1}\mathbf{I}_r) \tag{2.6}$$

where the precisions γ and υ are problem parameters. It can be seen that if any of γ_i or υ_i are large, the corresponding columns will be close to zero and consequently irrelevant. Indeed, the priors in (2.5)-(2.6) aid in automatic relevance determination since the subsequent optimization process may drive some of the precisions to infinity, yielding a low-rank factorization.

Finally, the three precision variables are selected to have Jeffrey's priors [45]

$$p(\beta) = \frac{1}{\beta}, \qquad p(\gamma_i) = \frac{1}{\gamma_i}, \qquad p(\upsilon_i) = \frac{1}{\upsilon_i} \tag{2.7}$$

for $1 \leq i \leq r$. Let $\mathbf{x}_\Omega$ denote the collection of measurements $\{y_{i\tau}\}_{i\in\Omega_\tau,\tau=1}^{t}$. Collecting the hidden variables into $\mathscr{H} := \{\mathbf{U}, \mathbf{V}, \mathbf{F}, \beta, \gamma, \upsilon\}$, the joint distribution of $\{\mathbf{x}_\Omega, \mathscr{H}\}$ can be written as

$$p(\mathbf{x}_\Omega, \mathscr{H}) = p(\mathbf{x}_\Omega|\mathbf{U}, \mathbf{V}, \beta)p(\mathbf{U}|\gamma)p(\mathbf{V}|\mathbf{F})p(\mathbf{F}|\upsilon)p(\beta)p(\upsilon)p(\gamma)$$

The full hierarchical Bayesian model adopted here is summarized in Fig. 2.2.

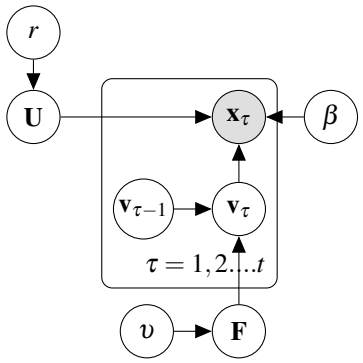


Fig. 2.2 Hierarchical Bayesian Model for Matrix Completion

2.4.2 Variational Bayesian Inference

Having specified the generative model for the data, the goal is to determine the posterior distribution $p(\mathscr{H}|\mathbf{x}_\Omega)$ given the likelihood probability of $\mathbf{x}_\Omega$ and the prior distribution of $\mathscr{H}$. However, the posterior distribution $p(\mathscr{H}|\mathbf{x}_\Omega)$ is intractable as $p(\mathbf{x}_\Omega)$ needs to be marginalized over all parameters ($\mathscr{H}$). Therefore, we utilize the mean-field approximation, wherein the posterior distribution factorizes as:

$$p(\mathscr{H} \mid \mathbf{x}_\Omega) = q_{\mathbf{U}}(\mathbf{U}) q_{\mathbf{V}}(\mathbf{V}) q_{\mathbf{F}}(\mathbf{F}) q_\upsilon(\upsilon) q_\beta(\beta) q_\gamma(\gamma). \tag{2.8}$$

In other words, the posterior is now restricted to a family of distributions that adhere to (2.8). The factors $q_{\mathbf{U}}$, $q_{\mathbf{V}}$, $q_{\mathbf{F}}$, q_υ, q_β, and q_γ can be determined by minimizing the Kullback–Leibler divergence of $p(\mathscr{H}|\mathbf{x}_\Omega)$ from $q(\mathscr{H})$, usually via an alternating minimization approach [47]. Indeed, thanks to the choice of conjugate priors for the parameters, it can

be shown that the individual factors in (2.8) take the following forms [40]:

$$q_{\mathbf{V}}(\mathbf{V}) = \mathscr{N}(\mathrm{vec}\,(\mathbf{V}) \mid \mu^{\mathbf{V}}, \Xi^{\mathbf{V}}) \tag{2.9a}$$

$$q_{\mathbf{u}_{i\cdot}} = \mathscr{N}(\mathbf{u}_{i\cdot} \mid \mu_i^{\mathbf{U}}, \Xi_i^{\mathbf{U}}) \tag{2.9b}$$

$$q_{\mathbf{f}_{i\cdot}} = \mathscr{N}(\mathbf{f}_{i\cdot} \mid \mu_i^{\mathbf{F}}, \Xi_i^{\mathbf{F}}) \tag{2.9c}$$

$$q_{\beta}(\beta) = \mathrm{Ga}(\beta; a^{\beta}, b^{\beta}) \tag{2.9d}$$

$$q_{\gamma_i}(\gamma_i) = \mathrm{Ga}(\gamma_i; a_i^{\gamma}, b_i^{\gamma}) \tag{2.9e}$$

$$q_{\upsilon_i}(\upsilon_i) = \mathrm{Ga}(\upsilon_i; a_i^{\upsilon}, b_i^{\upsilon}) \tag{2.9f}$$

where, $\mu^{\mathbf{V}} \in \mathbb{R}^{rt}$, $\Xi^{\mathbf{V}} \in \mathbb{R}^{rt \times rt}$, $\mu_i^{\mathbf{U}} \in \mathbb{R}^{r}$, $\Xi_i^{\mathbf{U}} \in \mathbb{R}^{r \times r}$, $\mu_i^{\mathbf{F}} \in \mathbb{R}^{r}$, $\Xi_i^{\mathbf{F}} \in \mathbb{R}^{r \times r}$, and a^{β}, b^{β}, a_i^{γ}, b_i^{γ}, a_i^{υ}, $b_i^{\upsilon} \in \mathbb{R}_{++}$.

We use EM Algorithm [48] to derive the posterior distribution in (2.9). EM algorithm treats $\mathscr{H}_h := \{\mathbf{U}, \mathbf{V}, \mathbf{F}\}$ as hidden variables (with posterior pdf $q_h(\mathscr{H}_h) := q_{\mathbf{V}}(\mathbf{V}) q_{\mathbf{U}}(\mathbf{U}) q_{\mathbf{F}}(\mathbf{F})$) and uses maximum a posteriori (MAP) estimates for the precision variables $\mathscr{H}_p := \{\upsilon, \gamma, \beta\}$.

2.4.3 Update Equations

Each iteration of EM algorithm simply involves updating the variables $\{\mu^{\mathbf{V}}, \Xi^{\mathbf{V}}, \{\mu_i^{\mathbf{U}}\}, \{\Xi_i^{\mathbf{U}}\}, \{\mu_i^{\mathbf{F}}\}, \{\Xi_i^{\mathbf{F}}\}, \hat{\upsilon}, \hat{\gamma}, \hat{\beta}\}$ in a cyclic manner. Let us denote $\omega_\tau := |\Omega_\tau|$ and let $\omega := \sum_\tau \omega_\tau$ be the total number of observations made. Then, the updates for hyperparameters $\{\upsilon, \gamma\}$ take the following form

$$\hat{\upsilon}_i = \frac{m-2}{\sum_{k=1}^{m} \left([\mu_k^{\mathbf{F}}]_i^2 + [\Xi_k^{\mathbf{F}}]_{ii} \right)} \tag{2.10a}$$

$$\hat{\gamma}_i = \frac{m-2}{\sum_{k=1}^{m} \left([\mu_k^{\mathbf{U}}]_i^2 + [\Sigma_k^{\mathbf{U}}]_{ii} \right)}. \tag{2.10b}$$

Since $\mathbf{v}_\tau$ denotes the τ-th column of $\mathbf{V}^T$, its posterior distribution may be written as $q_{\mathbf{v}_\tau}(\mathbf{v}_\tau) = \mathcal{N}(\mathbf{v}_\tau \mid \mu_\tau^{\mathbf{V}}, \Xi_\tau^{\mathbf{V}})$, where $\mu_\tau^{\mathbf{V}}$ and $\Xi_\tau^{\mathbf{V}}$ comprise of the corresponding elements of $\mu^{\mathbf{V}}$ and $\Xi^{\mathbf{V}}$, respectively. Also, define the posterior covariance matrices as

$$\Sigma_{\tau,\iota}^{\mathbf{V}} := \mu_\tau^{\mathbf{V}}(\mu_\iota^{\mathbf{V}})^T + \Xi_{\tau,\iota}^{\mathbf{V}} \tag{2.11}$$

$$\Sigma_i^{\mathbf{F}} := \mu_i^{\mathbf{F}}(\mu_i^{\mathbf{F}})^T + \Xi_i^{\mathbf{F}} \tag{2.12}$$

$$\Sigma_i^{\mathbf{U}} := \mu_i^{\mathbf{U}}(\mu_i^{\mathbf{U}})^T + \Xi_i^{\mathbf{U}}. \tag{2.13}$$

Therefore, the update for $\hat{\beta}$ becomes

$$\hat{\beta} = \frac{\omega - 2}{\sum_{\tau=1}^{t}\sum_{i\in\Omega_\tau}\left[y_{i\tau}^2 - 2y_{i\tau}(\mu_i^{\mathbf{U}})^T\mu_\tau^{\mathbf{V}} + \operatorname{tr}\left(\Sigma_i^{\mathbf{U}}\Sigma_{\tau,\tau}^{\mathbf{V}}\right)\right]}. \tag{2.14}$$

Next, the updates for the factors $\mathbf{F}$,$\mathbf{U}$ and $\mathbf{V}$ take the following form

$$\mu_i^{\mathbf{F}} = [\Xi_i^{\mathbf{F}}\Sigma_{\tau,\tau-1}^{\mathbf{V}}]_{\cdot i} \tag{2.15a}$$

$$\Xi_i^{\mathbf{F}} = \left(\operatorname{Diag}(\hat{\upsilon}) + \sum_{\tau=1}^{t-1}\Sigma_{\tau,\tau-1}^{\mathbf{V}}\right)^{-1} \tag{2.15b}$$

$$\mu_i^{\mathbf{U}} = \hat{\beta}\Xi_i^{\mathbf{U}}\sum_{\tau\in\Omega_i'}\mu_\tau^{\mathbf{V}}y_{i\tau} \tag{2.15c}$$

$$\Xi_i^{\mathbf{U}} = \left(\hat{\gamma}_i\mathbf{I}_r + \hat{\beta}\sum_{\tau\in\Omega_i'}\Sigma_{\tau,\tau}^{\mathbf{V}}\right)^{-1} \tag{2.15d}$$

$$\mu^{\mathbf{V}} = \Xi^{\mathbf{V}} \begin{bmatrix} \hat{\beta} \sum_{i \in \Omega_1} y_{i1} \mu_i^{\mathbf{U}} + \Lambda_1^{-1} \mu_1 \\ \hat{\beta} \sum_{i \in \Omega_2} y_{i2} \mu_i^{\mathbf{U}} \\ \vdots \\ \hat{\beta} \sum_{i \in \Omega_t} y_{it} \mu_i^{\mathbf{U}} \end{bmatrix}. \tag{2.16}$$

(2.17)

$$\begin{aligned} \left[\Xi^{\mathbf{V}}\right]^{-1} &= \hat{\beta} \mathrm{Diag}\left(\Xi_{(1)}^{\mathbf{U}}, \ldots, \Xi_{(t)}^{\mathbf{U}}\right) + \\ &+ \begin{bmatrix} \Lambda_1^{-1} & -\hat{\mathbf{F}} & \ldots & 0 \\ -\hat{\mathbf{F}} & \mathbf{I}_r + \Sigma^{\mathbf{F}} & -\hat{\mathbf{F}} & \ldots \\ \vdots & \vdots & & \vdots \\ \ldots & 0 & -\hat{\mathbf{F}} & \mathbf{I}_r \end{bmatrix}. \end{aligned} \tag{2.18}$$

where $\Omega_i' := \{\tau \mid i \in \Omega_\tau\}$, $\hat{\mathbf{F}} := \mathbb{E}[\mathbf{F} \mid \mathbf{x}_\Omega]$ as the matrix whose i-row is given by $(\mu_i^{\mathbf{F}})^T$, and $\Sigma^{\mathbf{F}} := \sum_{i=1}^{r} \Sigma_i^{\mathbf{F}}$ It is remarked that although the $rt \times rt$ matrix $[\Xi^{\mathbf{V}}]^{-1}$ is block-tridiagonal, the matrix $\Xi^{\mathbf{V}}$ is dense, and direct inversion would be prohibitively costly. Moreover, the classical Rauch-Tung-Striebel (RTS) smoother cannot be directly applied since evaluating the conditional expectations under $q(\mathbf{V})$ is difficult and not amenable to the Matrix Inversion Lemma [40, 48]. The RTS correction method is a smoothing technique widely used in Gaussian Linear models. It cannot be applied in Variational Bayesian parameter learning, since they are derived for the point-parameters model. Interestingly, observe that the updates in (2.14) and (2.15) depend only on diagonal and super-diagonal blocks of $\Xi^{\mathbf{V}}$, namely $\Xi_{\tau,\tau}^{\mathbf{V}}$ and $\Xi_{\tau,\tau-1}^{\mathbf{V}}$, respectively. The next subsection details a low-complexity algorithm for carrying out the updates for these blocks as well as for $\mu^{\mathbf{V}}$.

2.4.4 Low-complexity updates via LDL-decomposition

Thanks to the block-tridiagonal structure of $[\Xi^{\mathbf{V}}]^{-1}$, it is possible to use the LDL decomposition to carry out the updates in an efficient manner. Decomposing $[\Xi^{\mathbf{V}}]^{-1} = \mathscr{L}\mathbf{D}\mathscr{L}^T$, the key idea is that left multiplication with $\Xi^{\mathbf{V}}$ is equivalent to left multiplication with $\mathscr{L}^{-T}\mathbf{D}^{-1}\mathscr{L}^{-1}$. Towards this end, we utilize the algorithm from [40], that comprises of two phases: the forward pass that carries out the multiplication with $\mathbf{D}^{-1}\mathscr{L}^{-1}$ and the backward pass that implements the multiplication with $\mathscr{L}^{-T}$. Let us define for $2 \leq \tau \leq t$,

$$\Psi_\tau := \hat{\beta} \sum_{i \in \Omega_\tau} \Sigma^{\mathbf{U}}_{(i)} + \mathbf{I}_r + \mathbb{1}_{\tau \neq t} \sum_{i=1}^{r} \Sigma^{\mathbf{F}}_i \tag{2.19}$$

$$\mathbf{v}_\tau := \hat{\beta} \sum_{i \in \Omega_\tau} y_{i\tau} \mu^{\mathbf{U}}_i. \tag{2.20}$$

The forward pass outputs intermediate variables $\breve{\Xi}^{\mathbf{V}}_{\tau,\tau}$, $\breve{\Xi}^{\mathbf{V}}_{\tau,\tau+1}$, and $\breve{\mu}_\tau$, that are subsequently used in the backward pass. The updates take the following form:

1. Initialize $\hat{\Xi}^{\mathbf{V}}_{1,1} = \Lambda_1$ and $\hat{\mu}^{\mathbf{V}}_1 = \mu_1 + \hat{\beta} \sum_{i \in \Omega_\tau} y_{i\tau} \Lambda_1 \mu^{\mathbf{U}}_i$

2. For $\tau = 1, \ldots, t-1$

$$\breve{\Xi}^{\mathbf{V}}_{\tau,\tau+1} = -\hat{\Xi}^{\mathbf{V}}_{\tau,\tau} \hat{\mathbf{F}} \tag{2.21a}$$

$$\breve{\Xi}^{\mathbf{V}}_{\tau+1,\tau+1} = (\Psi_{\tau+1} - (\breve{\Xi}^{\mathbf{V}}_{\tau,\tau+1})^T \Psi^{\mathbf{V}}_{\tau,\tau+1})^{-1} \tag{2.21b}$$

$$\breve{\mu}^{\mathbf{V}}_{\tau+1} = \breve{\Xi}^{\mathbf{V}}_{\tau+1,\tau+1} (\mathbf{v}_{\tau+1} - (\breve{\Xi}^{\mathbf{V}}_{\tau,\tau+1})^T \breve{\mu}^{\mathbf{V}}_\tau) \tag{2.21c}$$

3. For $\tau = t-1, \ldots, 1$

$$\Xi^{\mathbf{V}}_{\tau,\tau+1} = -\breve{\Xi}^{\mathbf{V}}_{\tau,\tau+1} \Xi^{\mathbf{V}}_{\tau+1,\tau+1} \tag{2.21d}$$

$$\Xi^{\mathbf{V}}_{\tau,\tau} = \breve{\Xi}^{\mathbf{V}}_{\tau,\tau} - \hat{\Xi}^{\mathbf{V}}_{\tau,\tau+1} (\Xi^{\mathbf{V}}_{\tau,\tau+1})^T \tag{2.21e}$$

$$\mu^{\mathbf{V}}_\tau = \breve{\mu}^{\mathbf{V}}_\tau - \breve{\Xi}^{\mathbf{V}}_{\tau,\tau+1} \mu^{\mathbf{V}}_{\tau+1} \tag{2.21f}$$

4. Output $\{\Xi^{\mathbf{V}}_{\tau,\tau+1}, \Xi^{\mathbf{V}}_{\tau,\tau}, \mu^{\mathbf{V}}_{\tau}\}_{\tau=2}^{t}$

Note that while $\Xi^{\mathbf{V}}_{i,j} \neq 0$ for $|i-j| > 1$, these blocks are neither calculated in the forward and backward passes nor required in any of the variational updates.

Finally, the predictive distribution $p(y_{i\tau} \mid \mathbf{x}_\Omega)$ for $\tau \notin \Omega_i$ or $\tau \geq t+1$ is still not tractable in the present case. Instead, we simply use point estimates for estimating the missing entries. Specifically, for $\tau \notin \Omega_i$, the missing entries are imputed as

$$y_{i\tau} = (\mu^{\mathbf{V}}_{\tau})^T \mu^{\mathbf{U}}_i. \tag{2.22}$$

Likewise for $\tau \geq t+1$, the prediction becomes

$$y_{i\tau} = (\hat{\mathbf{F}}^{\tau-t} \mu^{\mathbf{V}}_t)^T \mu^{\mathbf{U}}_i. \tag{2.23}$$

It can be seen that as compared to the updates in (2.16)-(2.18) that incur a complexity of $\mathcal{O}(t^3)$, the complexity incurred due to (2.21) is only $\mathcal{O}(t)$. Overall, the different parameters are updated cyclically until convergence for each $t = 1, 2, \ldots$.

Remarks on the Convergence of VBSF

The VB framework used in the present work is a special case of a more general mean field approximation approach. The convergence of the VB algorithm is well-known; see e.g. [49], [50]. Intuitively, the variational approximation renders the evidence lower bound convex in individual factors, and thus amenable to coordinate ascent iterations. Since the lower bound is also differentiable with respect to each factor, the coordinate ascent iterations converge to a stationary point; see [51] for a more general result. However, convergence to the global optimum is not guaranteed.

Algorithm 1: Variational Bayesian Subspace Filtering

Initialize $\gamma, \beta, \mathbf{v}$, $sub = 1$, Ω_τ, $\Omega'_i, \Xi^{\mathbf{U}}, \mu^{\mathbf{U}}, \Xi^{\mathbf{V}}, \mu^{\mathbf{V}}, \Xi^{\mathbf{F}}_{diag}, \mu^{\mathbf{F}} \Lambda_1, \mu_1$,
$\hat{\mathbf{X}} = \mu^A (\mu^{\mathbf{V}})^T$
while $X_{conv} < 10^{-5}$ **do**
    $\mathbf{X}_{old} = \hat{\mathbf{X}}$
    $\Gamma = diag(\gamma)$
    if $sub == 1$ **then**
        Update using (2.21)
        $sub = 2$
        Update using (2.10a), (2.11), (2.15a), (2.15b) $\forall\ 1 \leq i \leq r$
    else if $sub == 2$ **then**
        Update using (2.13), (2.15c), (2.15d), (2.10b) $\forall 1 \leq i \leq m$
        $sub = 1$
    end
    $\hat{\mathbf{X}} = \mu^A (\mu^{\mathbf{V}})^T$
    Update using (2.14)
    $X_{conv} = \frac{\|\mathbf{X} - \mathbf{X}_{old}\|_F}{\|\mathbf{X}_{old}\|_F}$
end
return $(\hat{\mathbf{X}}, \Xi^{\mathbf{U}}, \mu^{\mathbf{U}}, \Xi^{\mathbf{V}}, \mu^{\mathbf{V}}, \Xi^{\mathbf{F}}_{diag}, \mu^{\mathbf{F}})$

2.4.5 Fixed-lag tracking

Algorithm 1 can be viewed as an offline algorithm that must be run for every t. In practical settings, it may be impractical to remember and process the entire history of measurements at each t. Moreover, given data at time t, estimates may only be required for entries at time $t - \Delta$ for some $\Delta < h$. Towards this end, we consider a sliding window of measurements. Since $\mathbf{U}_t$ and $\mathbf{F}_t$ may be seen as transition matrices for the latent states and between latent state and observations, we initialize the next sliding-window with inferred approximate distributions on the transition matrices of the current window. For instance, within the context of traffic density prediction, the inferred approximate distribution for a day may be used as a prior for the coming days. That is, the distributions for $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{F}$ for a day and sliding window can be initialized with the approximate distributions obtained from the previous month's data.

2.5 Experimentation

2.5.1 Performance Index

To measure the effectiveness of our algorithm and for the comparison with other relevant algorithms, we use mean relative error (MRE) as the performance index. For any time instance τ, the MRE denoted by MRE_τ is defined as:

$$\text{MRE}_\tau = \frac{1}{z}\sum_{k=1}^{z}\frac{\| \hat{\mathbf{x}}_{\tau,k} - \mathbf{x}_{\tau,k} \|_2}{\| \mathbf{x}_{\tau,k} \|_2}. \tag{2.24}$$

where $\mathbf{x}_{\tau,k}$ and $\hat{\mathbf{x}}_{\tau,k}$ are the ground truth and estimated data for k^{th} day and τ^{th} time instance and $\| \hat{\mathbf{x}}_{\tau,k} - \mathbf{x}_{\tau,k} \|_2$ denotes the L2 norm error in the ground truth and estimated data. Since the value for the known data (sampled entries) may be modified post-estimation, we compute the MRE over the whole column for a given time instance. For calculating the overall accuracy of prediction for a day, we calculate MRE averaged over z days. The value of z is taken as 50 for weekdays (Mon-Fri) and 10 for the weekends (Sat-Sun).

2.5.2 Real Time Traffic Estimation

We now discuss the performance of the proposed VBSF algorithm for real-time traffic estimation. To evaluate the VBSF algorithm, we use the partial road network of Delhi with an area of 200 square km consisting of $m = 519$ edges. We collect the traffic data in the form of the average speed of vehicles on a particular segment using the Google map APIs for nearly 3 months across 519 edges. Taking advantage of the slow varying nature of the speed in the network edges, we sample the traffic data at the rate of one sample every $t_s = 15$ minutes. Unlike the complete data available from the API, real-world data may have missing entries. For instance, over the smaller area shown in Fig. 2.4, speed

Fig. 2.3 Region where traffic data is collected

Fig. 2.4 Map with red as missing and blue as known traffic entries

measurements may be available on the blue edges but not on the red ones. Finally, we evaluate our algorithm for real-time traffic estimation.

In order to evaluate the VBSF algorithm, an incomplete data set is created by randomly sampling a fraction p of the measurements. In our evaluations, we consider three different cases with 0.75, 0.5 and 0.25 fractions of present data. We select previous $h = 30$ time intervals for the traffic dataset. We compare our algorithm with other methods that potentially solve the current traffic estimation problem in the missing data scenario. The algorithms are

- Low rank tensor completion (LRTC) [24].
- Grassmannian Rank-One Update Subspace Estimation (GROUSE) [19].
- Bayesian augmented tensor factorization(BATF) [52].

We now discuss simulation results for the current traffic estimation based on the current and past missing data using the VBSF algorithm. For a typical day, Fig. 2.5a shows the heatmap of the actual traffic data. The x-axis of each heatmap represents time instances while the y-axis represents different edges (519 edges). Each pixel of a heatmap indicates the speed, where higher speed is represented by a lighter colour. Figures 2.5b, 2.5e and 2.5h are heatmaps with missing entries of varying degrees. The corresponding completed matrices using VBSF algorithm are shown in Figs. 2.5c, 2.5f, and 2.5i. Since the proposed VBSF is an online method that completes one column at a time given the incomplete data from previous columns, the corresponding heatmaps are also generated in an online fashion. In other words, in the spirit of the online methodology, the window of $h+1$ incomplete columns is used to complete the last column followed by moving the window by one column. Finally, all the completed columns form a matrix represented in these heatmaps. Unsurprisingly, the heatmaps show that the performance of VBSF improves as the size of missing data decreases.

The MRE values for real time traffic estimation using VBSF for weekends are shown in Fig. 2.6a. It is observed that the prediction error is higher during the peak traffic time (in the evening) vis-a-vis non-peak time intervals. This may be due to a greater variance in traffic during the peak time intervals. However, the difference between the MRE values for 50% and 25% missing data case is only about 0.15 in the worst case. Equivalently, the average error of estimation of speed is only around 2 km/hr during the peak-time when the average speed is 15 km/hr even with 75% missing data. Similarly, for non-peak hours, even though the observed speed are higher (around 30-40 km/hr), the MRE values for

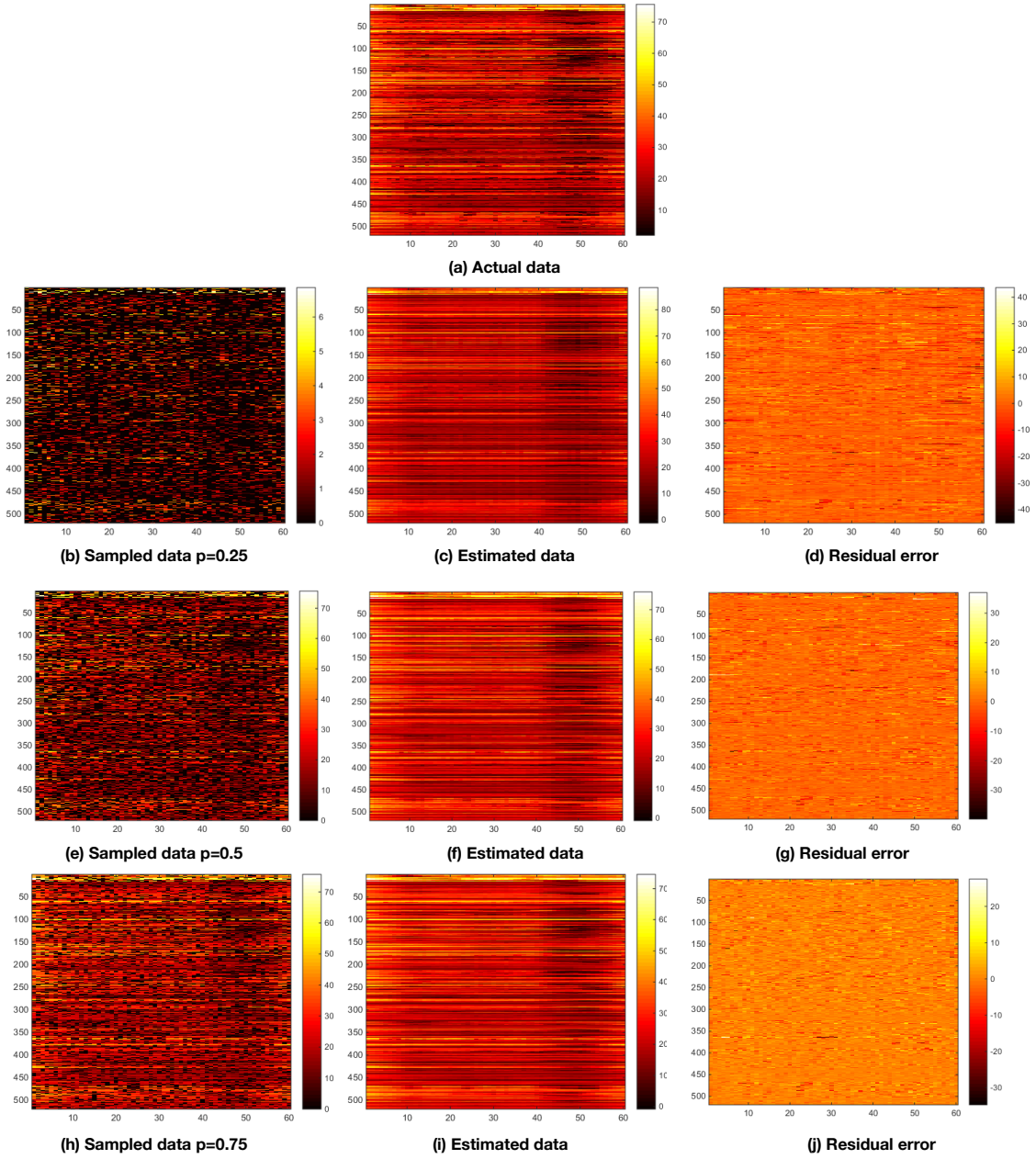


Fig. 2.5 Estimation of traffic speed data for different percentage of missing entries (a) Actual Traffic data , (b) Traffic data for $p = 0.25$, (c) Estimated Traffic for $p = 0.25$, (d) Residual error for estimation for $p = 0.25$, (e) Traffic data for $p = 0.5$, (f) Estimated Traffic with for $p = 0.5$, (g) Residual error for estimation for $p = 0.5$, (h) Traffic data for $p = 0.75$, (i) Estimated Traffic for $p = 0.75$, (j) Residual error for estimation for $p = 0.75$

$p = 50\%$ and $p = 25\%$ is around 0.1, which in other words indicate an average error of 3-4 km/hr in the estimation of speed.

The performance of the proposed VBSF algorithm is compared with that of Mean, (LRTC) [24], (GROUSE) [19] and (BATF) [52]. We used a grid search based approach for rank initialization in GROUSE and choose the rank that gives the least error. Table 2.1 presents the overall results along with the standard deviation of MRE over z days. Further, Figs. 2.7a and 2.7b show the comparison of our algorithm for different percentage of missing traffic data. It is observed that for high sampling rate of traffic data (p=75%), the LRTC and VBSF obtain similar performance. And for low sampling rate of traffic data (p=25%), the BATF and VBSF obtain similar performance. Also, for all the cases, VBSF performs better than GROUSE. This difference in performance can be attributed to the fact that the VBSF framework captures the temporal dependencies as well as the latent factors in the traffic matrix better than other methods. In terms of running time, VBSF is faster than LRTC, BATF and is comparable to GROUSE as shown in Table 2.2.

	$p = 0.25$ MRE	$p = 0.50$ MRE	$p = 0.75$ MRE
VBSF	0.1439 ± 0.022	0.11277 ± 0.0167	0.09336 ± 0.0106
GROUSE	0.372 ± 0.024	0.3446 ± 0.0233	0.3085 ± 0.0245
LRTC	0.1921 ± 0.041	0.1418 ± 0.0284	0.09578 ± 0.0195
BATF	0.1352 ± 0.026	0.12067 ± 0.0194	0.10142 ± 0.0176
Mean	0.1696 ± 0.0118	0.1692 ± 0.0115	0.1693 ± 0.0116

Table 2.1 Performance comparison for real-time traffic estimation

2.5.3 Air Quality Estimation

We motivate the problem of air quality estimation as: For a given day d the AQ data is represented in the form of a matrix $\mathbf{X} \in \mathbb{R}^{m \times t}$ where m and t are the numbers of spatial

[1]Experiments are conducted to evaluate average running time per column on Matlab using PC: Intel i5-6200U CPU 2.4 GHz.

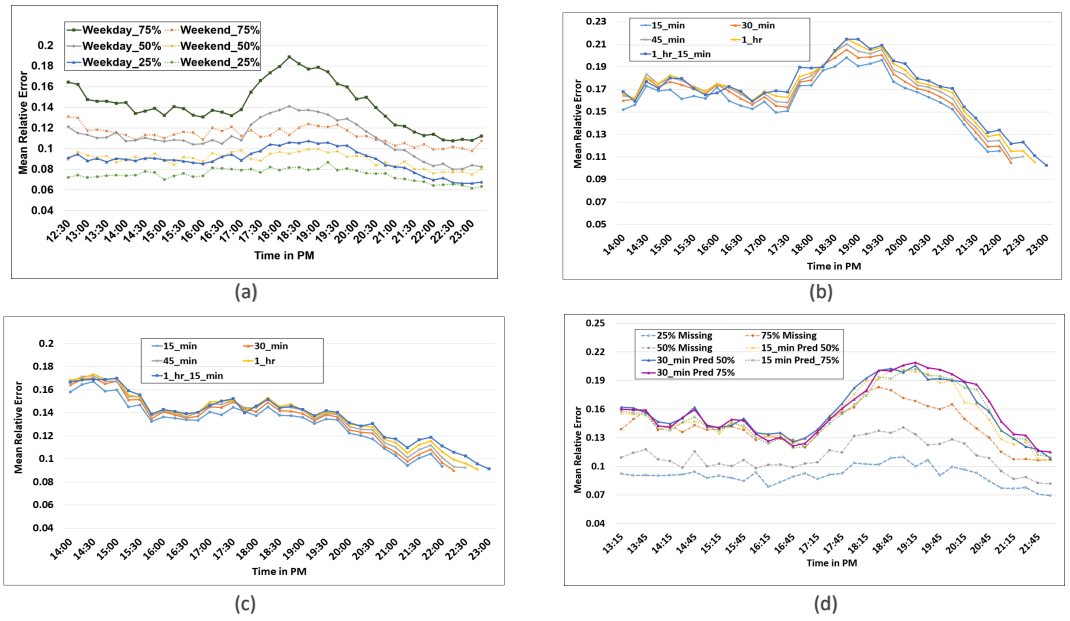


Fig. 2.6 Real-time traffic estimation and prediction for different missing entries (a) Real-time traffic estimation for different missing entries, (c) Weekday prediction 50% missing entries, (d) Weekend prediction 50% missing entries, (d) Overall prediction

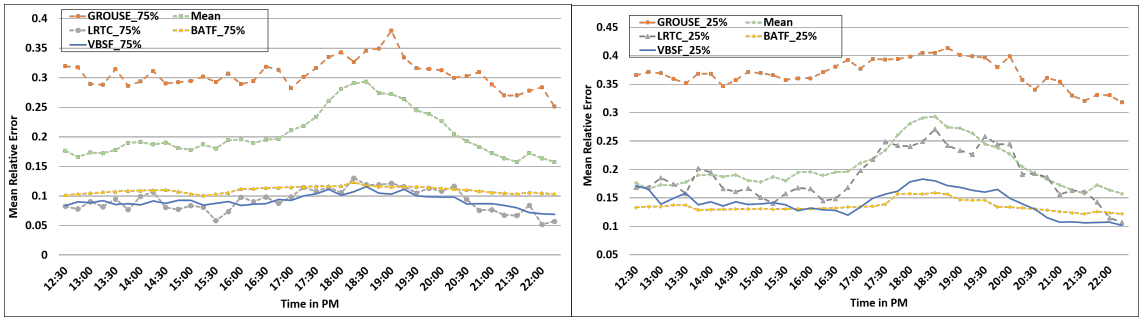


Fig. 2.7 a)Real-time traffic estimation for $p = 0.75$, b) Real-time traffic estimation for $p = 0.25$.

	$p = 0.25$ time(sec)	$p = 0.50$ time(sec)	$p = 0.75$ time(sec)
VBSF	7.001	8.685	9.675
GROUSE	7.935	8.5324	9.23960
BATF	9.388	9.5911	9.94917
LRTC	29.2	43.2	62.3

Table 2.2 Comparison of running time for different algorithms[1]

location and time slots respectively. In this work, we first randomly sample p percentage of real-world AQ data collected for Beijing and Guangzhou in China([1–3]). We then exploit the structure in the AQ data using VBSF and predict the missing data and report the results in Table. 2.3 and Fig. 2.9. We observe that even with a fraction of sampled data, we can complete the remaining matrix with reasonable accuracy.

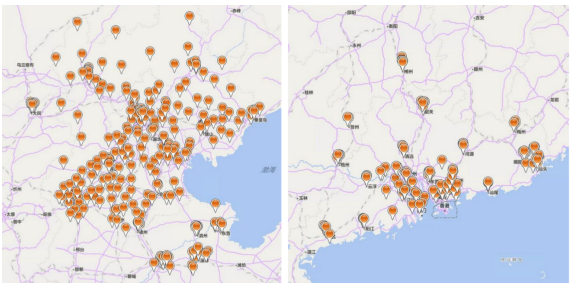


Fig. 2.8 China data ([1–3])

	$p = 0.25$ MRE	$p = 0.50$ MRE	$p = 0.75$ MRE
VBSF	0.308	0.231	0.2
GROUSE	0.509	0.446	0.418
LRTC	0.322	0.236	0.201
BATF	0.301	0.242	0.217

Table 2.3 Performance comparison for Air Quality Estimation

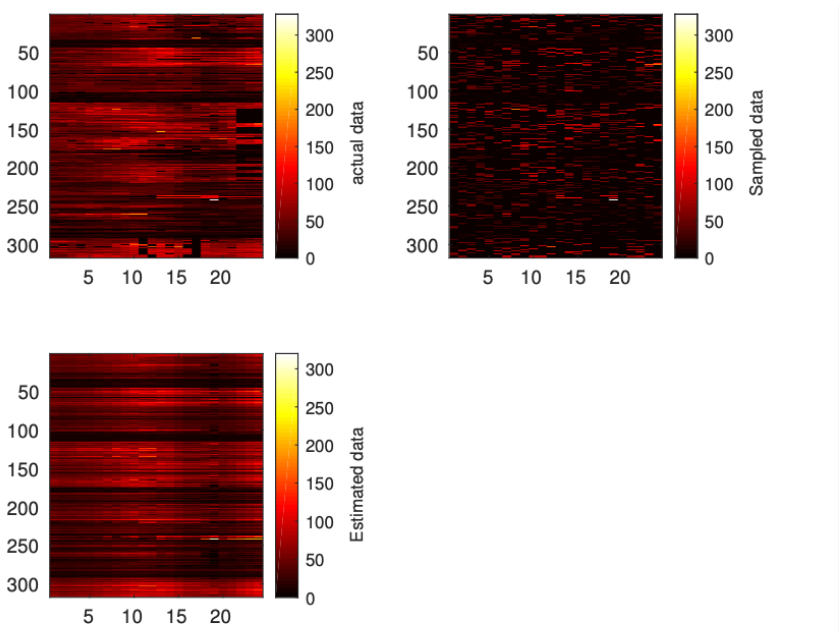


Fig. 2.9 Air quality estimation for 50% sampling

2.6 Summary

In this chapter we model the problem of spatiotemporal estimation where multivariate data arrives sequentially that resides in a time-varying low-dimensional subspace. The temporal evolution of the underlying low-rank subspace is characterized via a state-space model and low-complexity variational Bayesian subspace filtering algorithms are proposed for matrix completion. Simulation experiments quantify that the suggested model can be deployed to estimate the missing traffic data and missing air quality data with a reasonable accuracy even with a fraction of random measurements in the network. Extensive simulations on both the data sets (Traffic and Air Quality) demonstrate that the suggested model and the accompanying algorithms seem to capture the temporal evolution of the data well as compared to the current state-of-the-art matrix completion and the online subspace estimation algorithms.

Chapter 3

Robust Variational Bayesian Subspace Filtering

3.1 Overview

Sensor measurements are often depleted due to malfunction or intermittent errors that are often called as the outliers. In the context of mobility data, traffic densities are inferred from speed measurements, and they are often ridden with outliers, e.g., corresponding to random velocity changes unrelated to traffic due to sensor malfunctions, communication errors, and impulse noise. Also, for air quality observations, high AQI peaks can be observed due to the sudden smoke or malfunction of the sensor. An example of such outliers for AQI readings is shown in Fig. 3.1. In the previous chapter, we discussed the problem of estimating the missing entries of the spatio-temporal matrix. Estimating the spatio-temporal matrix in the presence of outlier data can result in noisy estimates. In this chapter, we will propose a Robust Variational Bayesian Subspace Filtering that improves the performance of spatio-temporal estimation in the presence of outlier data. The robust subspace filtering problem is more difficult as the removal of such outliers entails estimating their magnitudes as well as locations. And unlike the missing entries, the

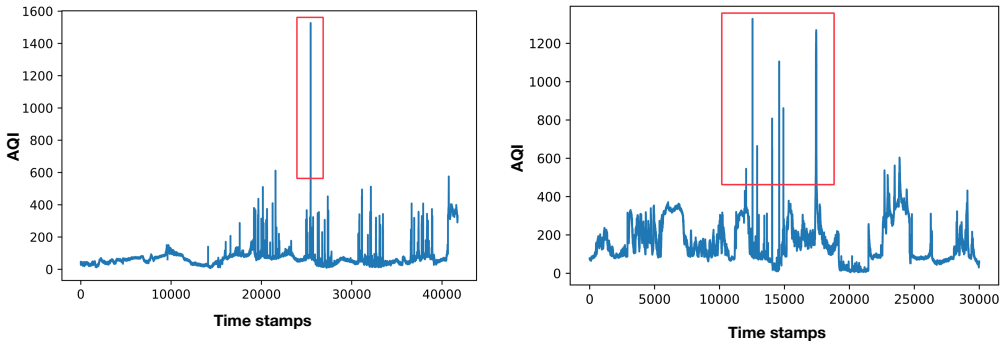


Fig. 3.1 Outliers in the sensed AQI data

location of these outliers is not known. For example, two cases are shown for the outlier in Fig. 3.1. For the first scenario, the location of the outlier is evident, but for the second scenario, the locations of the outlier are not evident. If we apply a filter based on the magnitude in the second scenario, actual AQI data at time stamp 24000 will be considered as outlier.

The organisation of this chapter is as follows. We begin in section 3.2 with Robust Variational Bayesian Subspace Filtering (RVBSF), where we introduce the approach to estimate the missing data in the presence of outlier data. We then provide the literature review in section 3.3. We discuss the proposed RVBSF framework in section 3.4. We provide details of the data and the experiments in section 3.5. Finally, in section 3.6, we conclude the chapter.

3.2 Robust Variational Bayesian Subspace Filtering

Performance of VBSF and other matrix estimation frameworks discussed in Sec. 2.4 degrades in the influence of anomalies or outliers. For example, if an outlier is sampled as shown in Fig 3.2(b), then the sampled outlier, affects the estimation as shown in Fig

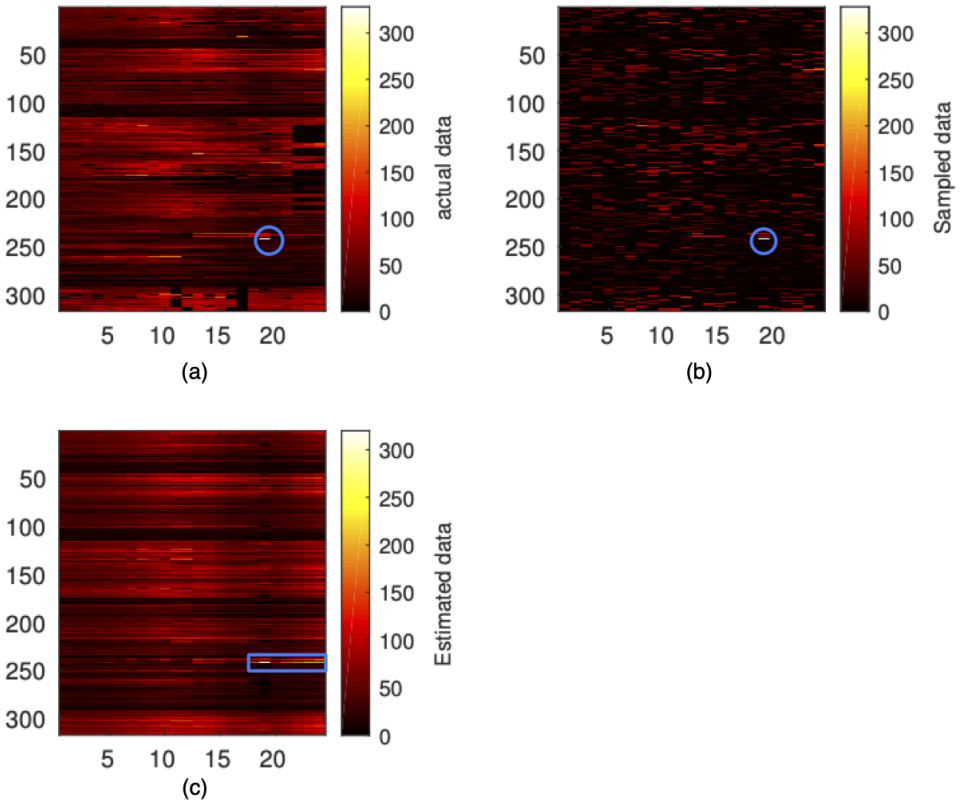


Fig. 3.2 Effect of an outlier in Estimation. (a) actual data with an outlier (b) outlier is sampled (c) effect of an outlier in the estimated data

3.2(c). Therefore we propose a Robust version of VBSF in this chapter to improve the performance of estimation in case of an outlier as well as to detect the outliers in the observations. The RVBSF (Robust Variational Bayesian Subspace Filtering) recovers the low-rank matrix **L** from highly corrupted measurements as Y= L+E. Unlike the noise discussed in Sec. 2.4, the entries in **E** are sparse and have a larger magnitude. These sparse entries are the anomalies in the measurements. Robust low-rank estimation, also denoted as Robust Principal Component Analysis (RPCA) in literature, is applied to various applications in the field of image and video processing; specifically, the foreground-background separation problem [53, 54]. However, this is not explored in the case of traffic or air quality observations. In this chapter, a robust version of the VBSF algorithm is proposed for outlier removal and data cleansing.

3.3 Related Work

Several non-Bayesian algorithms have been proposed to address the online subspace estimation problem from incomplete observations[19, 42–44]. GROUSE [19] is one of the early approaches that uses an update on the Grassmannian manifold to estimate the subspace. The robust variant of GROUSE, namely GRASTA , handles outliers by by incorporating the l_1 norm cost function [43]. OP-RPCA[42] is a robust subspace estimation technique that uses alternating minimization to compute the outliers and the underlying subspace. A number of online subspace tracking algorithms, such as ROSETA [44], have since been proposed. The proposed approach is compared with some of these algorithms in Sec. 3.5.

3.4 Proposed Framework

In this section we consider the robust version of the variational Bayesian subspace filtering problem in Sec. 2.4. Within this context, in addition to the missing entries in $\mathbf{X}$, some entries of $\mathbf{X}$ are also contaminated with outliers. Unlike the missing entries however, the location of these outliers is not known. In the traffic prediction problem, such entries arise due to sensor malfunctions, communication errors, and impulse noise. The robust subspace filtering problem is more difficult as the removal of such outliers entails estimating their magnitudes as well as locations.

Within the deterministic robust PCA framework, the traffic matrix is modeled as taking the form $\mathbf{X} = \mathbf{UV} + \mathbf{E}$ where $\mathbf{U} \in \mathbb{R}^{m\times r}$, $\mathbf{V} \in \mathbb{R}^{r\times t}$ are low-rank matrices as before. Additionally, we also need to estimate the sparse outlier matrix $\mathbf{E} \in \mathbb{R}^{m\times t}$. As before, both r and the level of sparsity in $\mathbf{E}$ are tuning parameters that must generally be carefully selected.

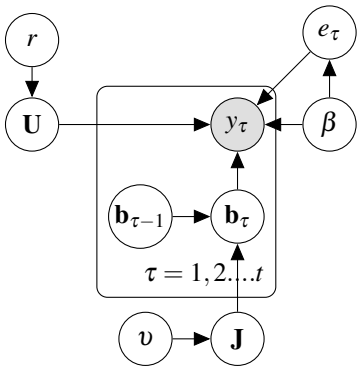


Fig. 3.3 Robust Hierarchical Bayesian Model for Matrix Completion

Here, we put forth the variational Bayesian subspace filtering algorithm that makes use of ARD priors to prune the redundant features. Consider the measurement matrix $\mathbf{X}$, whose entries are generated from the following pdf:

$$p(y_{i\tau} \mid \mathbf{u}_{i\cdot}, \mathbf{v}_\tau, e_{i\tau}, \beta) = \mathcal{N}(y_{i\tau} \mid \mathbf{v}_\tau^T \mathbf{u}_{i\cdot} + e_{i\tau}, \beta^{-1}) \qquad i \in \Omega_\tau \tag{3.1}$$

for all $\tau \geq 1$, and apart from the matrices $\mathbf{U}$ and $\mathbf{V}$ defined earlier, we also have $\{e_{i\tau}\}_{\tau=1, i\in\Omega_\tau}^{t}$ as the additional (hidden) problem parameter that captures the outliers. The generative models for $\mathbf{U}$ and $\mathbf{V}$ are the same as before, i.e.,

$$p(\mathbf{V} \mid \mathbf{F}) = \mathcal{N}(\mathbf{v}_1; \mu_1, \Lambda_1) \prod_{\tau=2}^{t} \mathcal{N}(\mathbf{v}_\tau \mid \mathbf{F}\mathbf{v}_{\tau-1}, \mathbf{I}_r) \tag{3.2a}$$

$$p(\mathbf{U} \mid \gamma) = \prod_{i=1}^{r} \mathcal{N}(\mathbf{u}_i \mid 0, \gamma_i^{-1}\mathbf{I}) \tag{3.2b}$$

$$p(\mathbf{F} \mid \upsilon) = \prod_{i=1}^{r} \mathcal{N}(\mathbf{f}_i \mid 0, \upsilon_i^{-1}\mathbf{I}) \tag{3.2c}$$

for $\tau \geq 2$, and γ and υ are problem parameters. Additionally, we also associate an ARD prior to the outliers, i.e.,

$$p(e_{i\tau}) = \mathcal{N}(e_{i\tau} \mid 0, \alpha_{i\tau}^{-1}) \qquad i \in \Omega_\tau \tag{3.3}$$

for $1 \leq \tau \leq t$, where the precision $\alpha_{i\tau}$ is a hidden variable, that would be driven to infinity whenever e_{ij} is zero. It is remarked that the prior for $e_{i\tau}$ is only specified for the measurements, i.e., for $i \in \Omega_\tau$ and no predictions are made for the outliers. As before, we associate Jeffery's prior to the precisions β, $\{\gamma_i\}$, $\{\upsilon_i\}$, and $\{\alpha_{i\tau}\}$.

$$p(\beta) = \frac{1}{\beta}, \qquad p(\gamma_i) = \frac{1}{\gamma_i}, \qquad p(\upsilon_i) = \frac{1}{\upsilon_i}, \qquad p(\alpha_{i\tau}) = \frac{1}{\alpha_{i\tau}}. \tag{3.4}$$

Let the vectors $\mathbf{e} \in \mathbb{R}^\omega$ and $\alpha \in \mathbb{R}^\omega$ collect the variables $\{e_{i\tau}\}$ and $\{\alpha_{i\tau}\}$, respectively. Likewise, defining all the hidden variables as $\mathscr{H} := \{\mathbf{U}, \mathbf{V}, \mathbf{F}, \mathbf{e}, \beta, \gamma, \upsilon\}$, the joint distribution of $\{\mathbf{x}_\Omega, \mathscr{H}\}$ can be written as

$$\begin{aligned}
&p(\mathbf{x}_\Omega, \mathscr{H}) \\
&= p(\mathbf{x}_\Omega|\mathbf{U}, \mathbf{V}, \beta)p(\mathbf{U}|\gamma)p(\mathbf{V}|\mathbf{F})p(\mathbf{F}|\upsilon)p(\mathbf{e}|\alpha)p(\beta)p(\upsilon)p(\gamma) \\
&= \prod_{\tau=1}^{t} \prod_{i \in \Omega_\tau} \mathscr{N}(y_{i\tau} \mid \mathbf{v}_\tau^T \mathbf{u}_{i\cdot}, \beta^{-1}) \mathscr{N}(e_{i\tau} \mid 0, \alpha_{i\tau}^{-1}) \frac{1}{\alpha_{i\tau}} \\
&\qquad \times \prod_{i=1}^{r} \left[\mathscr{N}(\mathbf{u}_i \mid 0, \gamma_i^{-1}\mathbf{I}) \mathscr{N}(\mathbf{f}_i \mid 0, \upsilon_i^{-1}\mathbf{I}) \right] \\
&\times \mathscr{N}(\mathbf{v}_1; \mu_1, \Lambda_1) \prod_{\tau=2}^{t} \mathscr{N}(\mathbf{v}_\tau \mid \mathbf{F}\mathbf{v}_{\tau-1}, \mathbf{I}) \frac{1}{\beta} \prod_{i=1}^{r} \frac{1}{\gamma_i \upsilon_i}.
\end{aligned} \tag{3.5}$$

The full hierarchical Bayesian model adopted here is summarized in figure 3.3(b).

3.4.1 Variational Bayesian Inference

Utilizing the mean field approximation, the posterior distribution $p(\mathscr{H} \mid \mathbf{x}_\Omega)$ factorizes as

$$\begin{aligned}
p(\mathscr{H} \mid \mathbf{x}_\Omega) &\approx q(\mathscr{H}) \\
&= q_{\mathbf{U}}(\mathbf{U}) q_{\mathbf{V}}(\mathbf{V}) q_{\mathbf{F}}(\mathbf{F}) q_{\mathbf{e}}(\mathbf{e}) q_\upsilon(\upsilon) q_\beta(\beta) q_\gamma(\gamma).
\end{aligned} \tag{3.6}$$

where the individual factors take the same forms as in (2.9), in addition to

$$q_{\mathbf{e}}(\mathbf{e}) = \prod_{\tau=1}^{t} \prod_{i \in \Omega_\tau} \mathcal{N}(e_{i\tau} | \mu_e^{i\tau}, \Xi_e^{i\tau}). \tag{3.7}$$

As before, the variational inference problem can be solved by updating the variables $\{\mu^{\mathbf{V}}, \Xi^{\mathbf{V}}, \{\mu_i^{\mathbf{U}}\}, \{\Xi_i^{\mathbf{U}}\}, \{\mu_i^{\mathbf{F}}\}, \{\Xi_i^{\mathbf{F}}\}, \{\mu_e^{i\tau}\}, \{\Xi_e^{i\tau}\}, a^\beta, b^\beta, \{a_i^\gamma\}, \{b_i^\gamma\}, \{a_i^\upsilon\}, \{b_i^\upsilon\}\}$ in a cyclic manner. However, a more compact form for the updates may be derived as follows.

3.4.2 Update Equations

Specifically, the updates for $\{\hat{\upsilon}_i, \hat{\gamma}_i\}$ remain the same as in (2.10). However, the update for $\hat{\beta}$ takes the form:

$$\hat{\beta} = \frac{\omega}{\sum_{\tau=1}^{t} \sum_{i \in \Omega_\tau} \nu_{i\tau}} \tag{3.8}$$

$$\begin{aligned} \nu_{i\tau} :=& y_{i\tau}^2 - 2(y_{i\tau} - \mu_e^{i\tau})(\mu_i^{\mathbf{U}})^T \mu_\tau^{\mathbf{V}} - 2y_{i\tau}\mu_e^{i\tau} \\ &+ (\mu_e^{i\tau})^2 + \Xi_e^{i\tau} + \operatorname{tr}\left(\Sigma_i^{\mathbf{U}} \Sigma_{\tau,\tau}^{\mathbf{V}}\right). \end{aligned} \tag{3.9}$$

Further, the parameters $\mu_e^{i\tau}$ and $\Xi_e^{i\tau}$ are updated as

$$\Xi_e^{i\tau} = \frac{1}{\hat{\beta} + (\mu_e^{i\tau})^2 + \Xi_e^{i\tau}} \tag{3.10a}$$

$$\mu_e^{i\tau} = \hat{\beta}\Xi_e^{i\tau}(y_{i\tau} - (\mu_i^{\mathbf{U}})^T \mu_\tau^{\mathbf{V}}). \tag{3.10b}$$

Proceeding similarly, the updates for $\{\mu_i^{\mathbf{F}}\}$, $\{\Xi_i^{\mathbf{F}}\}$, and $\{\Xi_i^{\mathbf{U}}\}$ remain the same as in (2.15), while the updates for $\{\mu_i^{\mathbf{U}}\}$ become:

$$\mu_i^{\mathbf{U}} = \hat{\beta}\Xi_i^{\mathbf{U}} \sum_{\tau \in \Omega_i'} \mu_\tau^{\mathbf{V}}(y_{i\tau} - \mu_e^{i\tau}). \tag{3.11}$$

Finally, the updates for $\Xi^{\mathbf{V}}$ remain the same but the updates of $\mu^{\mathbf{V}}$ change. Specifically, the low complexity updates via LDL-decomposition remain mostly the same, except for the modified definition of $\mathbf{v}_\tau$ in (2.20) which now looks like

$$\mathbf{v}_\tau = \hat{\beta} \sum_{i \in \Omega_\tau} (y_{i\tau} - \mu_e^{i\tau}). \tag{3.12}$$

The full robust subspace filtering algorithm is summarized in Algorithm 2. The predictions for $y_{i\tau}$ for $i \notin \Omega_\tau$ and for $\tau \geq t+1$ are obtained as in (2.22) and (2.23), respectively.

3.5 Results

The GPS data that is collected using probe vehicles may be corrupted by noise and may often contain outliers which need to be removed before further processing is performed. To mitigate the performance degradation due to outliers, we employ the robust variational Bayesian subspace filtering (RVBSF) that models the presence of outliers in the data in the sparse outlier matrix $\mathbf{E}$. To test the RVBSF algorithm, on a given day, we randomly sample a certain p_o percentage of the already sampled traffic data $\mathbf{x}_{i,\tau}$ and replace these values with $o_{i,\tau}$ as follows:

$$\mathbf{o}_{i,\tau} = \max(\mathbf{x}_{i,\tau-1}, \mathbf{x}_{i,\tau+1}) + c\,\mu_t. \tag{3.13}$$

Algorithm 2: Robust Variational Bayesian Subspace Filtering

Initialize $\alpha, \gamma, \beta, \mathbf{v}$, $sub = 1$, Ω_τ, Ω'_i, $\Xi^{\mathbf{U}}, \mu^{\mathbf{U}}, \Xi^{\mathbf{V}}, \mu^{\mathbf{V}}, \Xi^{\mathbf{F}}_{diag}, \mu^{\mathbf{F}} \Lambda_1, \mu_1$,
$\hat{\mathbf{Y}} = \mu^{\mathbf{U}}(\mu^{\mathbf{V}})^T$
while $Y_{conv} < 10^{-5}$ **do**
    $\mathbf{Y_{old}} = \hat{\mathbf{Y}}$
    $\Gamma = diag(\gamma)$
    if $sub == 1$ **then**
        Update using (2.21)
        $sub = 2$
        Update using (2.10a), (2.11), (2.15a), (2.15b) $\forall\ 1 \leq i \leq r$
    else if $sub == 2$ **then**
        Update using (2.13), (2.15c), (2.15d), (2.10b) $\forall\, 1 \leq i \leq m$ $sub = 3$
    end
    else
        Update using (3.10a), (3.10b) $\forall\ 1 \leq i \leq m$, $\forall\ 1 \leq \tau \leq t$
        $sub = 1$
    end
    $\hat{\mathbf{Y}} = \mu^{\mathbf{U}}(\mu^{\mathbf{V}})^T$
    Update using (3.8)
    $Y_{conv} = \frac{\|\mathbf{Y} - \mathbf{Y_{old}}\|_F}{\|\mathbf{Y_{old}}\|_F}$
end
return $(\hat{\mathbf{Y}}, \Xi^{\mathbf{U}}, \mu^{\mathbf{U}}, \Xi^{\mathbf{V}}, \mu^{\mathbf{V}}, \Xi^{\mathbf{F}}_{diag}, \mu^{\mathbf{F}})$

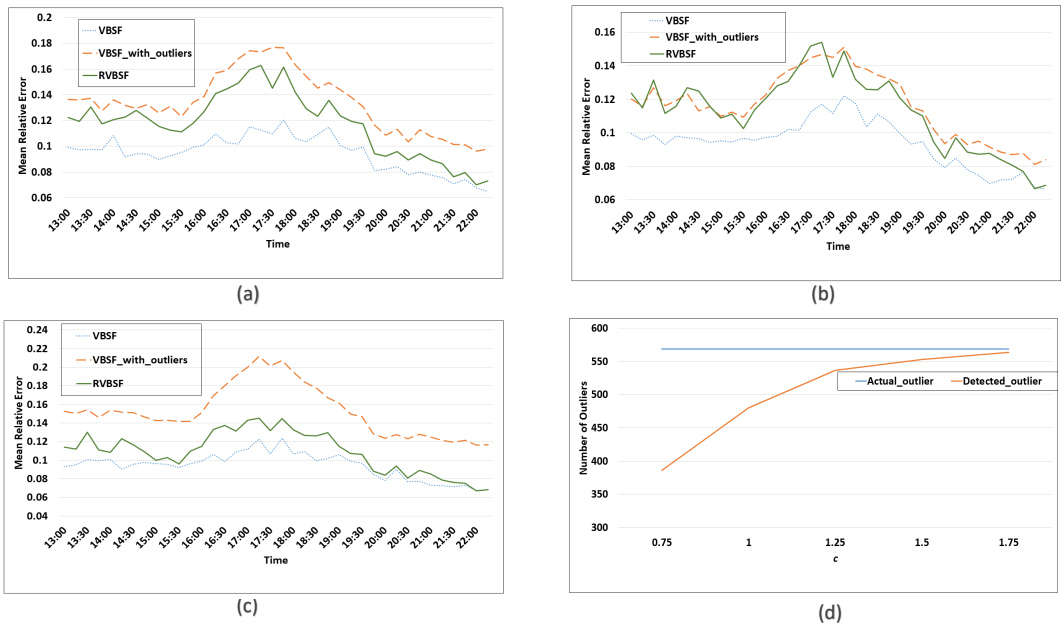


Fig. 3.4 Robust Bayesian subspace filtering for traffic data
(a) Comparison for VBSF and RVBSF with 5% outliers and $c = 0.75$, (b) Comparison for VBSF and RVBSF with 2% outliers and $c = 0.75$ (c) Comparison of VBSF and RVBSF for $c = 1.25$, (d) Number of outliers detected for different outlier values

In other words, the outlier is created by adding a large value $c\,\mu_t$ to the maximum of $\mathbf{x}_{i,\tau-1}$ and $\mathbf{x}_{i,\tau+1}$. Here, μ_t is the mean of observed entries at time t and c is a scaling parameter. The RVBSF algorithm is then applied to solve the real time traffic estimation problem. The detected artificial outliers are those points residing in the matrix $\mathbf{E}$.

The accuracy of outlier detection depends on the outlier value as shown in Fig. 3.4d. The value of c for simulations is chosen from the set $[0.75, 1, 1.25, 1.5, 1.75]$. We compare the robust VBSF (termed as RVBSF) with VBSF for two scenarios. First, when no outliers are added (VBSF), second, when outliers are present in the data but only VBSF was used (VBSF_with_outliers). Table 3.1 summarises the overall performance of the RVBSF algorithm. Understandably, RVBSF improves over VBSF when outliers are present, but is still worse than the MRE of VBSF for the case when no outliers were present. We also compare the performance of RVSF with MAD_VBSF and VBSF_MAD. MAD_VBSF first detect and remove the outliers based on heuristics (MAD filter) and then apply VBSF.

VBSF_MAD first apply VBSF, then detect and remove the outliers based on heuristics (MAD filter). For 25% missing entries, $p_o = 5\%$ and $c = 0.75$, the plots in Fig. 3.4a illustrate the performance of the RVBSF algorithm. Similarly for 75% of missing entries, $p_o = 2\%$ the results are shown in Fig. 3.4b. When $p_o = 5\%$ and $c = 0.75$, we observe that RVBSF detects outliers reasonably well vis-a-vis VBSF_with_outliers. Similar observation holds when outlier values increase as shown in Fig. 3.4c and Fig. 3.4d.

	$c = 0.75$ $p_o = 5\%$	$c = 0.75$ $p_o = 2\%$	$c = 1.5$ $p_o = 2\%$
VBSF	0.09462	0.09457	0.09434
VBSF_outlier	0.13406	0.11643	0.15318
RVBSF	0.11741	0.1127	0.10912
MAD_VBSF	0.1251	0.1151	0.1178
VBSF_MAD	0.1474	0.1289	0.1568

Table 3.1 RVBSF: overall performance

The performance of the proposed RVBSF algorithm is compared with that of OP-RPCA[42] GRASTA[43] and ROSETA[44] in Table 3.2. The RVBSF algorithm performs better than the subspace estimation and tracking algorithms. The difference in performance may be due to a better modeling of the temporal structure available in the data.

	$c = 0.75$ $p_o = 5\%$	$c = 0.75$ $p_o = 2\%$	$c = 1.5$ $p_o = 2\%$
OP-RPCA	0.2594	0.2298	0.2165
ROSETA	0.1859	0.1819	0.1723
GRASTA	0.1493	0.1507	0.1492
RVBSF	0.11741	0.1127	0.10912

Table 3.2 Performance Comparison for Robust Traffic Estimation

Further AQ data may be outlier-corrupted. However, the locations of such outliers are not known apriori. We randomly include a few outliers in the data and then estimate the location as well as their magnitude using the robust version of variational Bayesian subspace filtering.

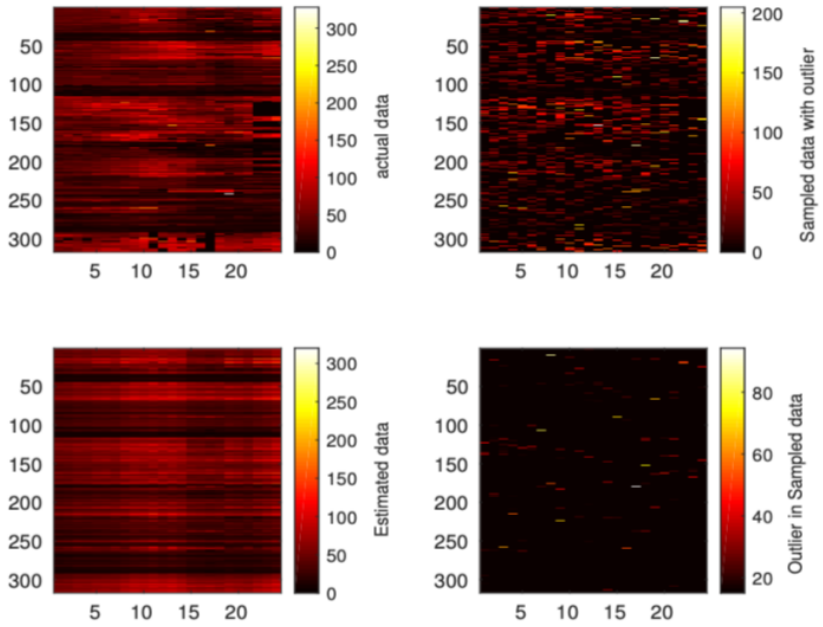


Fig. 3.5 Robust Air Quality Estimation for 50% sampling

A possible limitation of the suggested robust estimation framework is following. While there may be outliers present due to an erroneous speed estimation, there might be cases when the so called outlier value may actually be a real value. The current method may not be able to distinguish between such cases. Hence, a sudden drop in speed along an edge may be treated as an outlier and its possible impact on the traffic of nearby edges be be ignored by the model.

3.6 Summary

In this chapter, Robust Variational Bayesian Subspace Filtering is proposed for outlier removal and data cleansing. We solved the problem of outlier removal as well as estimate the missing observations in the data simultaneously. The estimation algorithm treats the sampled data as the true data and estimates the rest of the entries. Therefore, we observed degradation in estimation performance due to the outliers. We proposed RVBSF where detection and estimation of outliers are done simultaneously. We showed that the

performance of the RVBSF improved in the presence of an outlier as compared to the VBSF. Also, it performs better than the other state of the art methods for robust spatiotemporal estimation.

Chapter 4

Variational Bayesian Subspace Filtering with Subspace Information

4.1 Overview

Copious amounts of sensors are deployed in major cities for sensing the spatio-temporal variation in various urban environment characteristics like air pollution, traffic speed, etc. Formerly, the most common way of sensing data across a city is to deploy many stationary sensors or monitors across the city. Ideally, these static monitors should have a dense spatial coverage to achieve a fine-grained local source apportionment. Deploying static sensors in such large numbers is not only expensive, but also challenging to maintain, and as a result, most cities, especially in the developing world, have a sparse network of sensor monitors [55]. One way to improve the spatio-temporal data coverage is to use drive-by sensing or moving sensors that sample various parts of a region at different time instances [8, 56]. Since one sensor can be used to sample many locations at different timestamps, the actual number of moving sensors needed is just a fraction of the total number of static sensors needed to get the same spatial coverage.

Data acquired from both static and moving sensors contains missing data due to sensor malfunction, irregularity in sensor measurements, etc. Additionally, the drive-by sensing scheme uses relatively fewer sensors resulting in high data "gaps" in both spatial and temporal dimensions [56]. This motivates the problem of extreme matrix completion, where the percentage of data sampled may be as low as 10%. Hence, a natural question to ask is: how to fill the missing entries within a reasonable error range? In addition to the missing entries, sensor measurements can be contaminated with outliers emerging from the sensor malfunctioning, communication errors, or impulse noises. The occurrence of outliers in the measurements can further degrade the performance of data imputation. But, unlike the missing entries, the location and the value of outliers are not known, which makes the problem more challenging. The other question to ask is: how to estimate the missing data while detecting the noisy outliers? In this chapter, we exploit the underlying structure available in the data for extreme matrix completion.

The organisation of this chapter is as follows. We begin in section 4.2 with Variational Bayesian Filtering with Subspace Information (VBFSI), where we introduce the approach for extreme matrix completion. We then provide the literature review in section 4.3. We discuss the proposed VBFSI framework in section 4.4. We provide details of the data and the experiments in section 4.5. Finally, in section 4.6, we conclude the chapter.

4.2 Variational Bayesian Filtering with Subspace Information

In this chapter, we provide a solution to the problem of extreme matrix completion using Variational Bayesian Filtering with Subspace Information. We motivate the problem of spatio-temporal matrix completion as follows. For a given day d, the data is represented in the form of a matrix $\mathbf{X} \in \mathscr{R}^{n \times t}$ where n and t are the numbers of spatial locations and time

slots respectively. Our goal is to estimate the missing entries in the matrix $\mathbf{X}$. We motivate a low-rank robust subspace filtering approach for online matrix imputation, specifically for the extreme scenario of very sparse spatio-temporal sensing. One way of imputation is to exploit the low-rank structure available in the data [31]. However, enforcing only the low-rank structure in the data does not take into account the temporal variation (generally slow variation) of the data in a given location [57]. We incorporate a state-space model to capture the temporal evolution in the data. Moreover, exploiting reliable subspace information from historic data termed as prior subspace information can reduce the sample complexity of matrix completion and improve the imputation performance [58]. We update the subspace on day d using the prior subspace () estimated of the day $d-1$ to capture the periodicity in the data. Incorporating this prior subspace can improve the performance for an extreme case of sparsely sampled data. We use the Variational Bayes approach to update the parameters in an iterative fashion. In our work, the subspace distribution is chosen to allow automatic relevance determination (ARD) and unlike the matrix or tensor completion methods, the algorithm parameters such as rank, noise powers, need not be specified or tuned.

We propose Variational Bayesian Filtering with Subspace Information (VBFSI), a novel matrix completion framework that simultaneously models the low rank nature of the data, temporal evolution through a state space model, and periodicity through subspace tracking. We experimentally show that incorporating the prior subspace over days can improve imputation performance, especially for low data sampling. We also illustrate that our subspace can reduce the sample complexity, thereby motivating VBFSI for extreme matrix completion. Critical parameter, like the rank of the model, is automatically tuned using ARD approach. We also propose a robust version of the VBFSI algorithm for imputing the data in the presence of outliers. We conduct comprehensive experiments on

real-world spatio-temporal datasets that show the efficacy of the proposed method over other state-of-the-art imputation methods.

We compare the performance of the proposed algorithm with various state-of-the-art algorithms on many real-world air quality and traffic datasets. The result shows modeling of the evolution of the subspace leads to improvement in performance even when a small percentage of random measurements are available for the purpose of imputation. A likely impact of our method is that cities with a low sensing budget can perform random drive-by sampling of the urban environment, and the suggested matrix completion framework can provide a reasonably accurate imputation leading to better decision making.

4.3 Related work

Big data matrices can be approximated as low rank matrices [59]. Matrix completion is used to exploit the low rank structure in the data to impute the missing data [31, 60]. Robust PCA is used for matrix completion in the presence of outliers by incorporating a sparse outlier matrix [61, 62]. The traditional matrix completion framework is not applicable for time series data imputation, as it does not take into account the ordering among the temporal embeddings [57]. Autoregressive model can model these temporal embeddings and in turn capture the temporal evolution in the time series data [57, 63, 64]. However, these models fail to capture the prior subspace information that can be exploited to capture periodicity in the time series data. Exploiting the 3-way pattern in the data using tensor completion based frameworks can incorporate the periodic pattern in the data and, in turn improve the imputation performance [31, 65, 66]. However, the temporal evolution and subspace evolution are not modeled in the traditional tensor completion frameworks.

Variational Bayesian approaches are proposed for matrix/tensor completion and robust principal component analysis by modeling the matrix/tensor as low rank [31, 60, 62, 65, 67]. A state space model to capture the temporal evolution is also proposed in [63, 64]. However,

these approaches do not explicitly model the evolution of the subspace to capture the periodicity in the data.

4.4 Proposed Framework

Let $\mathbf{X} \in \mathbb{R}^{n \times t}$ be the data matrix for a day, where n and t denote the number of spatial locations and time stamps, respectively. The low rankness in the data can be imposed using the equation

$$\mathscr{L}_1 = \min_{\mathbf{U},\mathbf{V}} ||\mathbf{P}_\Omega(\mathbf{X} - \mathbf{U}\mathbf{V}^T)||_F \tag{4.1}$$

where $\mathbf{U} \in \mathscr{R}^{n \times r}$ and $\mathbf{V} \in \mathscr{R}^{t \times r}$ and $r =$rank$(\mathbf{X}) << min(n,t)$ implying the low rankness in the data. For sampling percentage of p, set Ω denotes the sampled data containing $p \times n \times t$ samples. $\mathbf{P}_\Omega$ is the indicator matrix where $\mathbf{P}_{ij} = 1 \; \forall (i,j) \in \Omega$. To capture the temporal evolution in the data, we can regularize the columns of $\mathbf{V}$ to follow an autoregressive model.

$$\mathscr{R}(\mathbf{V}) = \sum_{i=1}^{t} ||\mathbf{v}_i - \mathbf{F}\mathbf{v}_{i-1}|| \tag{4.2}$$

To capture the periodicity over days the subspace evolution can be modeled as follows:

$$\mathscr{R}(\mathbf{U}) = \eta \sum_{i=1}^{n} (\mathbf{u}_i - \mathbf{u}_i^{d-1})^T (\Xi_i^{\mathbf{U}_{d-1}})^{-1} (\mathbf{u}_i - \mathbf{u}_i^{d-1}) \tag{4.3}$$

where $\mathscr{R}(\mathbf{U})$ corresponds to the mahalanobis distance between each row vector of $\mathbf{U}^{d-1}$ (subspace estimate of previous day) and $\mathbf{U}$ (current subspace estimate for a given day d). $\Xi^{\mathbf{U}^{d-1}}$ denotes the covariance matrix of $\mathbf{U}^{d-1}$. Here η controls the effect of prior subspace ($\mathbf{U}^{d-1}$, Ξ_i^{d-1}) in the estimation of $\mathbf{U}$.

4.4.1 Spatio Temporal Matrix Completion with Subspace Information (STMC)

In this section, we will obtain a Bayesian framework for spatio-temporal matrix completion. The optimization formulation in (4.1) is equivalent to minimizing the negative log-likelihood function.

$$\mathscr{L}_1 = \min_{\mathbf{U},\mathbf{V}}(-\ln p(\mathbf{X}_\Omega \mid \mathbf{U},\mathbf{V},\beta)) \tag{4.4}$$

where likelihood function on the entries of $\mathbf{X}_\Omega$ can be defined as:

$$p(\mathbf{X}_\Omega \mid \mathbf{U},\mathbf{V},\beta) = \prod_{(i,j)\in\Omega} \mathscr{N}(X_{ij} \mid \mathbf{u}_i.\mathbf{v}_j.^T, \beta^{-1}) \tag{4.5}$$

here β is the noise precision. The prior on the noise is assumed to be non-informative Jeffrey's prior.

$$p(\beta) = \beta^{-1} \tag{4.6}$$

Regularization on the columns of $\mathbf{U}$ defined in (4.3) can be incorporated by initializing a prior on the columns of $\mathbf{U}$.

$$p(\mathbf{U} \mid \gamma) = \prod_{i=1}^{r} \mathscr{N}(\mathbf{u}_i \mid \mathbf{u}_i^{d-1}, \gamma_i^{-1}\mathbf{I}_m) \tag{4.7}$$

Columns of $\mathbf{U}$ are enforced with a sparsity profile using precision γ_i to automate the rank. When γ_i are driven to a large value, this enforces the column mean to be zero and in turn, reduces the rank thereby modeling the low rank in the Bayesian framework. This way of determining the rank on the go is referred to the as the Automatic Rank Determination

[60]. Further, the autoregressive regularization in (4.2) can be modeled as

$$p(\mathbf{V} \mid \mathbf{F}) = \mathcal{N}(\mathbf{v}_1; \mu_1, \Lambda_1) \prod_{\tau=2}^{t} \mathcal{N}(\mathbf{v}_\tau \mid \mathbf{F}\mathbf{v}_{\tau-1}, \mathbf{I}_r) \tag{4.8}$$

$\mathbf{F}$ is assigned multivariate Gaussian priors with column-specific precisions υ.

$$p(\mathbf{F} \mid \upsilon) = \prod_{i=1}^{r} \mathcal{N}(\mathbf{f}_i \mid 0, \upsilon_i^{-1}\mathbf{I}_r) \tag{4.9}$$

Precision variables γ and υ are selected to have Jeffrey's priors

$$p(\gamma_i) = \frac{1}{\gamma_i}, p(\upsilon_i) = \frac{1}{\upsilon_i} \tag{4.10}$$

The overall joint distribution for STMC can be expressed as

$$\begin{aligned} p(\mathbf{X}_\Omega, \mathbf{U}, \mathbf{V}, \mathbf{F}, \beta, \gamma, \upsilon) &= p(\mathbf{X}_\Omega | \mathbf{U}, \mathbf{V}, \beta) p(\mathbf{U} | \gamma) \\ &\times p(\mathbf{V} | \mathbf{F}) p(\mathbf{F} | \upsilon) p(\beta) p(\upsilon) p(\gamma) \end{aligned} \tag{4.11}$$

4.4.2 Robust STMC (RSTMC)

For robust matrix completion, we model the $\mathbf{X} = \mathbf{U}\mathbf{V}^T + \mathbf{E} + \mathbf{N}$, where $\mathbf{E}$ denotes the sparse outlier matrix and $\mathbf{N}$ is the dense error matrix. The conditional distribution of generating the entries of $\mathbf{X}$ can be defined as

$$p(\mathbf{X} \mid \mathbf{U}, \mathbf{V}, \mathbf{E}, \beta) = \prod_{(i,j)\in\Omega} \mathcal{N}(X_{ij} \mid (\mathbf{u}_i.\mathbf{v}_j.^T + E_{ij}), \beta^{-1}) \tag{4.12}$$

Columns of **U** ,**V** and **F** and precision variables γ, β, υ follows the same prior distribution defined in (6-10). Each entry of sparse outlier matrix E_{ij} is assigned a precision α_{ij}.

$$p(\mathbf{E}|\alpha) = \prod_{i|((i,j)\in\Omega)} \prod_{j|((i,j)\in\Omega)} \mathcal{N}(E_{ij} \mid 0, \alpha_{ij}^{-1}) \tag{4.13}$$

where the α_{ij} have the non informative prior

$$p(\alpha_{ij}) = \frac{1}{\alpha_{ij}} \tag{4.14}$$

This works similar to the ARD where instead of column of the matrix, each entry of the matrix is assigned with a precision. Whenever $\alpha_{i,j}$ is driven to a large value , the $E_{i,j} \to 0$ thereby enforcing sparsity. The overall joint distribution for RVBSF is expressed as

$$\begin{aligned} p(\mathbf{X}_\Omega, \mathbf{U}, \mathbf{V}, \mathbf{F}, \mathbf{E}, \beta, \gamma, \upsilon, \alpha) &= p(\mathbf{X}_\Omega|\mathbf{U}, \mathbf{V}, \beta) p(\mathbf{U}|\gamma) \\ &\times p(\mathbf{V}|\mathbf{F}) p(\mathbf{F}|\upsilon) p(\mathbf{E}|\alpha) p(\beta) p(\upsilon) p(\gamma) \end{aligned} \tag{4.15}$$

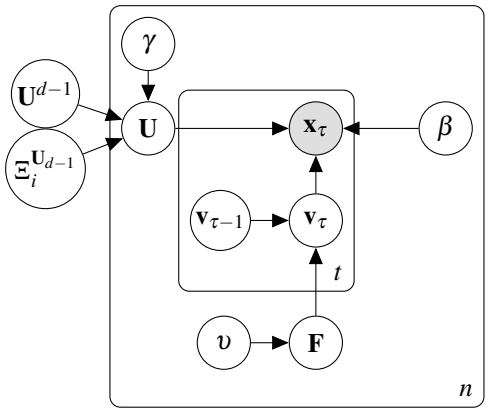


Fig. 4.1 Variational Bayesian Filtering with Subspace Information

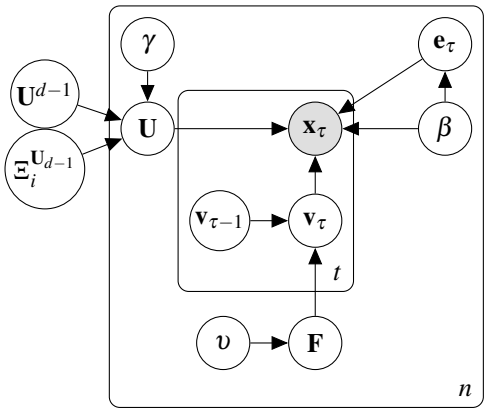


Fig. 4.2 Robust Variational Bayesian Filtering with Subspace Information

4.4.3 Variational Bayesian Inference

Approximate posterior distribution of parameters $\theta := \{\mathbf{U}, \mathbf{V}, \mathbf{F}, \beta, \gamma, \upsilon\}$ and $\theta_R := \{\mathbf{U}, \mathbf{V}, \mathbf{F}, \mathbf{E}, \beta, \gamma, \upsilon, \alpha\}$ for STMC and RSTMC respectively are derived using Variational Inference. We utilize the mean-field approximation, wherein the posterior distribution of STMC is factorized as:

$$p(\theta \mid \mathbf{x}_\Omega) = q_{\mathbf{U}}(\mathbf{U}) q_{\mathbf{V}}(\mathbf{V}) q_{\mathbf{F}}(\mathbf{F}) q_\upsilon(\upsilon) q_\beta(\beta) q_\gamma(\gamma). \tag{4.16}$$

whereas the posterior distribution of parameters θ_R in RSTMC is factorized as:

$$q_{\mathbf{U}}(\mathbf{U}) q_{\mathbf{V}}(\mathbf{V}) q_{\mathbf{F}}(\mathbf{F}) q_{\mathbf{E}}(\mathbf{E}) q_\upsilon(\upsilon) q_\beta(\beta) q_\gamma(\gamma) q_\alpha(\alpha).$$

The posterior distribution of all the parameters is determined by minimizing the Kullback–Leibler divergence of $p(\theta|\mathbf{x}_\Omega)$ from $q(\theta)$, usually via an alternating minimization approach [47].

Since we are exploiting the prior subspace information in our model, we call this framework Variational Bayesian Filtering with Subspace Information (VBFSI) and Robust

Variational Bayesian Filtering with Subspace Information (RVBFSI) respectively. The full graphical model for VBFSI and RVBFSI is depicted in Fig 4.1,4.2.

4.4.4 VBFSI

We use variational inference to estimate the posterior distribution of $q_{\mathbf{U}}$, $q_{\mathbf{V}}$, $q_{\mathbf{F}}$, q_{υ}, q_{β}, and q_{γ} for sampled data $\mathbf{Z} = \mathbf{P}_{\Omega}(\mathbf{X})$. The updates for the posterior distribution of parameters are similar to the updates derived in [60, 64].
The posterior distribution for a row of $\mathbf{U}$ is given by

$$q_{\mathbf{u}_i} = \mathcal{N}(\mathbf{u}_i \mid \mu_i^{\mathbf{U}}, \Xi_i^{\mathbf{U}}) \tag{4.17}$$

The updates of mean and covariance of $\mathbf{U}$ are derived as

$$\begin{aligned}(\Xi_i^{\mathbf{U}})^{-1} = \hat{\gamma}_i \mathbf{I}_r + \hat{\beta} \sum_{\tau|(i,\tau)\in\Omega} (\mu_\tau^{\mathbf{V}} (\mu_\tau^{\mathbf{V}})^T + \Xi_{\tau,\tau}^{\mathbf{V}}) \\ + \eta (\Xi_i^{\mathbf{U}_{d-1}})^{-1}\end{aligned} \tag{4.18}$$

$$\mu_i^{\mathbf{U}} = \Xi_i^{\mathbf{U}} (\hat{\beta} \sum_{\tau|(i,\tau)\in\Omega} \mu_\tau^{\mathbf{V}} Z_{i\tau} + \eta (\Xi_i^{\mathbf{U}_{d-1}})^{-1} \mu_i^{\mathbf{U}_{d-1}}) \tag{4.19}$$

The mean and covariance for the Posterior Distribution of $\mathbf{V}$ are as follows:

$$q_{\mathbf{V}}(\mathbf{V}) = \mathcal{N}(\mathrm{vec}\,(\mathbf{V}) \mid \mu^{\mathbf{V}}, \Xi^{\mathbf{V}}) \tag{4.20}$$

$$\mu^{\mathbf{V}} = \Xi^{\mathbf{V}} \begin{bmatrix} \hat{\beta} \sum_{i|(i,1)\in\Omega} Z_{i,1} \mu_i^{\mathbf{U}} + \Lambda_1^{-1} \mu_1 \\ \hat{\beta} \sum_{i|(i,2)\in\Omega} Z_{i,2} \mu_i^{\mathbf{U}} \\ \vdots \\ \hat{\beta} \sum_{i|(i,t)\in\Omega} Z_{i,t} \mu_i^{\mathbf{U}} \end{bmatrix} \tag{4.21}$$

$$\begin{aligned} \left[\Xi^{\mathbf{V}}\right]^{-1} &= \hat{\beta} \mathrm{Diag}\left(\Xi_{(1)}^{\mathbf{U}}, \ldots, \Xi_{(t)}^{\mathbf{U}}\right) + \\ &+ \begin{bmatrix} \Lambda_1^{-1} & -\hat{\mathbf{F}} & \ldots & 0 \\ -\hat{\mathbf{F}} & \mathbf{I}_r + \Sigma^{\mathbf{F}} & -\hat{\mathbf{F}} & \ldots \\ \vdots & \vdots & & \vdots \\ \ldots & 0 & -\hat{\mathbf{F}} & \mathbf{I}_r \end{bmatrix} \end{aligned} \tag{4.22}$$

The direct inversion of the dense matrix $\Xi^{\mathbf{V}}$ would be costly. The block-tridiagonal structure $(\Xi^{\mathbf{V}})^{-1}$ can be exploited to carry out the updates for $\Xi^{\mathbf{V}}$ in an efficient manner using LDL decomposition [63, 64].

The updates of the posterior distribution of $\mathbf{F}$ are given by

$$q_{\mathbf{f}_i} = \mathcal{N}(\mathbf{f}_i \mid \mu_i^{\mathbf{F}}, \Xi_i^{\mathbf{F}}) \tag{4.23}$$

$$\mu_i^{\mathbf{F}} = [\Xi_i^{\mathbf{F}} (\mu_\tau (\mu_{\tau-1})^T + \Xi_{\tau,\tau-1}^{\mathbf{V}})]_i \tag{4.24a}$$

$$(\Xi_i^{\mathbf{F}})^{-1} = \mathrm{Diag}\,(\hat{\upsilon}) + \sum_{\tau=1}^{t-1} (\mu_\tau (\mu_{\tau-1})^T + \Xi_{\tau,\tau-1}^{\mathbf{V}}) \tag{4.24b}$$

The posterior distribution for hyperparameters $\{\beta, \gamma, \mathbf{v}\}$ are given by

$$q_{\beta}(\beta) = \text{Ga}(\beta; a^{\beta}, b^{\beta}) \tag{4.25a}$$

$$q_{\gamma_i}(\gamma_i) = \text{Ga}(\gamma_i; a_i^{\gamma}, b_i^{\gamma}) \tag{4.25b}$$

$$q_{\upsilon_i}(\upsilon_i) = \text{Ga}(\upsilon_i; a_i^{\upsilon}, b_i^{\upsilon}) \tag{4.25c}$$

where $\text{Ga}(x, a, b)$ denotes the Gamma pdf with parameters a and b. The updates for $\{\beta, \gamma, \mathbf{v}\}$ are given by

$$\hat{\upsilon}_i = \frac{r}{\sum_{k=1}^{r} \left([\mu_k^{\mathbf{F}}]_i^2 + [\Xi_k^{\mathbf{F}}]_{ii} \right)} \tag{4.26a}$$

$$\hat{\gamma}_i = \frac{m+n}{\sum_{k=1}^{m} \left([\mu_k^{\mathbf{U}}]_i^2 + [\Sigma_k^{\mathbf{U}}]_{ii} \right) + \sum_{k=1}^{n} \left([\mu_k^{\mathbf{V}}]_i^2 + [\Sigma_k^{\mathbf{V}}]_{ii} \right)} \tag{4.26b}$$

$$\hat{\beta} = \frac{pnt}{\| \mathbf{Z} - P_{\Omega}(\mathbf{U}\mathbf{V}^T) \|_F^2} \tag{4.26c}$$

We update the mean, covariance of $\mathbf{U}, \mathbf{V}, \mathbf{F}$ and the hyperparameters γ, υ, β iteratively as shown in Algorithm 2

Algorithm 3: VBSFI

Input: $\Xi^{\mathbf{U}_{d-1}}, \mu^{\mathbf{U}_{d-1}}, \mathbf{P}_{\Omega}(\mathbf{X})$

Initialization:$\gamma, \beta, \upsilon, \Xi^{\mathbf{U}}, \mu^{\mathbf{U}}, \Xi^{\mathbf{V}}, \mu^{\mathbf{V}}, \Xi^{\mathbf{F}}, \mu^{\mathbf{F}}, \mathbf{Z}$

1: **while** $X_{conv} < 10^{-5}$ **do**

2: $\mathbf{X_{old}} = \hat{\mathbf{X}}$
Compute $\mathbf{V}, \mathbf{F}, \upsilon, \beta$ using (4.21, 4.22, 4.26a, 4.26c)
Compute $\mathbf{U}, \gamma, \beta$ using (4.18, 4.19, 4.26b, 4.26c)
$\hat{\mathbf{X}} = \mu^{\mathbf{U}}(\mu^{\mathbf{V}})^T$
$X_{conv} = \frac{\left\| \hat{\mathbf{X}} - \mathbf{X_{old}} \right\|_F}{\| \mathbf{X_{old}} \|_F}$

3: **end while**

4: **Output**: $\hat{\mathbf{X}}$

4.4.5 RVBFSI

In this section, we obtain the results for RVBFSI. The posterior distribution of $q_{\mathbf{U}}$, $q_{\mathbf{V}}$, $q_{\mathbf{F}}$, q_{υ}, q_{β}, and q_{γ} takes the same form for $\mathbf{Z} = \mathbf{P}_{\Omega}(\mathbf{X} - \mathbf{E})$ as shown in (17-26c). The posterior distribution for $\mathbf{E}$ takes the following form $\forall (i, j) \in \Omega$.

$$q(E_{ij}) = \mathcal{N}(E_{ij} \mid \mu^{\mathbf{E}}_{ij}, \Xi^{\mathbf{E}}_{ij}) \tag{4.27}$$

$$\mu^{\mathbf{E}}_{i,j} = \hat{\beta}\, \Xi^{\mathbf{E}}_{i,j}(X_{i,j} - \mu^{A}_{i.}(\mu^{B}_{j.})^{T}),\ \Xi^{\mathbf{E}}_{i,j} = \frac{1}{\hat{\beta} + \hat{\alpha_{i,j}}} \tag{4.28}$$

$$\hat{\alpha}^{new}_{i,j} = \frac{1 - \hat{\alpha}^{old}_{i,j}\, \Xi^{\mathbf{E}}_{i,j}}{(\mu^{\mathbf{E}}_{i,j})^{2}} \tag{4.29}$$

$\hat{\alpha}^{new}_{i,j}$ is the fixed-point update for α. This is used in the sparse bayesian learning that leads to much faster convergence and enhanced sparsity [60, 68]. For robust estimation of entries in the presence of outliers, we update the mean, covariance of $\mathbf{U}, \mathbf{V}, \mathbf{F}, \mathbf{E}$ and the hyperparameters $\gamma, \upsilon, \beta, \alpha$ iteratively as shown in Algorithm 3.

Algorithm 4: RVBSFI

Input: $\Xi^{\mathbf{U}_{d-1}}, \mu^{\mathbf{U}_{d-1}}, \mathbf{P}_{\Omega}(\mathbf{X})$

Initialization:$\gamma, \beta, \upsilon, \Xi^{\mathbf{U}}, \mu^{\mathbf{U}}, \Xi^{\mathbf{V}}, \mu^{\mathbf{V}}, \Xi^{\mathbf{F}}, \mu^{\mathbf{F}}, \mu^{\mathbf{E}}, \Xi^{\mathbf{E}}, \alpha, \mathbf{Z} = \mathbf{P}_{\Omega}(\mathbf{X} - \mathbf{E})$

1: **while** $X_{conv} < 10^{-5}$ **do**

2: $\mathbf{X_{old}} = \hat{\mathbf{X}}$
Compute $\mathbf{V}, \mathbf{F}, \upsilon, \beta$ using (4.21, 4.22, 4.26a, 4.26c)
Compute $\mathbf{U}, \gamma, \mathbf{E}, \alpha, \beta$ (4.18, 4.19, 4.26b, 4.26c)
Compute $\mathbf{E}, \alpha, \beta$ using (4.28-4.29, 4.26c)
$\hat{\mathbf{X}} = \mu^{\mathbf{U}}(\mu^{\mathbf{V}})^{T}$
$X_{conv} = \frac{\|\hat{\mathbf{X}} - \mathbf{X_{old}}\|_{F}}{\|\mathbf{X_{old}}\|_{F}}$

3: **end while**

4: **Output**: $\hat{\mathbf{X}}$

4.5 Results

In this section, we will evaluate the performance of VBFSI on various spatiotemporal datasets against the recent state of the art imputation methods. We further compare the performance of RVBFSI in the presence of artificially corrupted outliers.

4.5.1 Experiment Setting

Datasets

We used traffic speed and air quality (PM 2.5) data for performance evaluation.

- Data (DT): Delhi traffic speed data [64]. This data contains traffic speed data of 519 road segments over 60 days with a sampling resolution of 15 min from 7 am to 11 pm in Delhi, India. The data can be organized as a tensor with dimensions $\boldsymbol{R}^{519\times67\times60}$.

- Data (GT): Guangzhou urban traffic speed data [69]. This data contains traffic speed data of 214 road segments over 61 days with a sampling resolution of 10 mins in Guangzhou, China. The data can be organized as a tensor with dimensions $\boldsymbol{R}^{214\times144\times61}$.

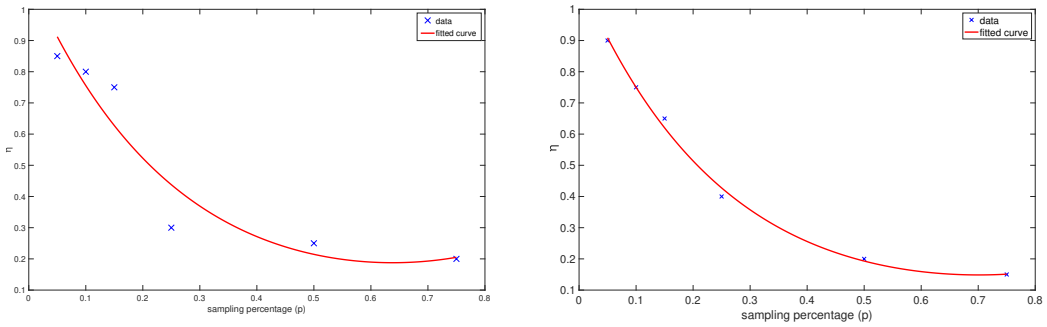
- Data (PT): Pems traffic speed data [70]. This data contains traffic speed data of 228 road segments over 44 days with a sampling resolution of 5 mins in California. We process the data for a sampling resolution of 30 mins. The data can be organized as a tensor with dimensions $\boldsymbol{R}^{228\times48\times44}$.

- Data (CA): China Air Quality data [71]. This data contains the AQI data collected in the cities near Beijing and Guangzhou in China. We pre-process the data and extract the PM2.5 AQI data for 313 locations and 60 days with a sampling resolution of 1 hr. The data can be organized as a tensor with dimensions $\boldsymbol{R}^{313\times24\times60}$.

Parameters Setting

We only tune the parameter η; otherwise, our approach does not require the tuning of any hyperparameters. We grid search the best η for different sampling percentages. Then, we fit the exponential model for η vs p, as shown in Fig. 4.3. We observe that for a higher sampling percentage, imputation performance decreases with an increase in η. After fitting the exponential model for traffic data (DT) η is set as $\eta = 1.09 * exp(-3.87 * p) + 0.00862 * exp(3.76 * p)$ whereas $\eta = 1.282 * exp(-11.18 * p) + 0.0289 * exp(1.74 * p)$ is set for air quality data (CA). We tune η for data (DT) and generalize it to the other two traffic data (PT and GT).

The initial subspace $\mathbf{U}^0$ is calculated using the eight days average for all the datasets. Then we run the algorithm in an online fashion for the next 30 days for all the datasets. All the experiments are run on Matlab with the system configuration of 2.3 GHz and 8 GB RAM.



(a) η vs p performance for Data:DT (b) η vs p performance for Data:CA

Fig. 4.3 Hyperparameter setting for η.

It is observed that the prior subspace have more effect on the traffic data (DT) than air quality data (CA).

Evaluation Metrics

We use the mean relative error (MRE), and root mean square error (RMSE) as evaluation metrics:

$$\text{MRE} = \frac{\left\| \mathbf{X}(\Omega') - \mathbf{X}(\hat{\Omega}') \right\|}{\|\mathbf{X}(\Omega')\|}$$

$$\text{RMSE} = \sqrt{\frac{1}{|\Omega'|} \sum_{(i,j \in \Omega')} (\hat{X_{ij}} - X_{ij})^2}$$

where Ω' represents the set of missing entries.

4.5.2 Baseline Algorithms

We compare our model with recent state of the art matrix and tensor imputation methods.

Matrix completion Frameworks

- **VBSF**: Variational Bayesian Subspace Filtering [64], VBSF proposes an online variational Bayesian formulation to estimate low-rank matrices whose subspace evolves according to a state-space model.
- **VMC**: Variety-based Matrix Completion [72]. VMC exploit low-complexity non-linear structures in the data to estimate the matrix that can be possible high-rank. The high-rank matrix becomes low-rank after mapping each column to a higher dimensional space.

	p %	VBFSI	VBSF	VMC	BCPF	TRLRF	TRMF	BTMF
Data:DT	5%	**0.156 / 4.387**	0.782 / 22.03	0.998 / 28.18	0.164 / 4.6	0.901 / 25.45	0.183 / 5.146	0.157 / 4.394
	15%	**0.135 / 3.796**	0.162 / 4.552	0.97 / 27.39	0.147 / 4.137	0.682 / 19.25	0.151 / 4.24	0.137 / 3.836
	25%	0.127 / 3.576	0.142 / 3.999	0.155 / 4.357	**0.126 / 3.544**	0.415 / 11.72	0.135 / 3.801	0.129 / 3.613
	50%	0.117 / 3.289	0.119 / 3.344	0.131 / 3.687	**0.115 / 3.23**	0.171 / 4.785	0.121 / 3.409	0.119 / 3.342
	75%	0.11 / 3.086	0.11 / 3.099	0.117 / 3.28	**0.109 / 3.076**	0.13 / 3.642	0.117 / 3.262	0.115 / 3.224
Data:PT	5%	**0.144 / 8.608**	1 / 60.08	0.998 / 59.974	0.175 / 10.494	0.94 / 56.47	0.161 / 9.571	0.151 / 9.084
	15%	**0.111 / 6.625**	0.179 / 10.7	0.974 / 58.52	0.147 / 8.836	0.807 / 48.49	0.139 / 8.264	0.118 / 7.06
	25%	**0.101 / 6.026**	0.147 / 8.82	0.152 / 9.13	0.108 / 6.492	0.605 / 36.35	0.118 / 6.995	0.11 / 6.571
	50%	**0.084 / 5.056**	0.097 / 5.79	0.087 / 5.213	0.091 / 5.431	0.168 / 10.07	0.093 / 5.511	0.1 / 6.02
	75%	0.081 / 4.841	0.081 / 4.854	**0.069 / 4.135**	0.081 / 4.848	0.097 / 5.833	0.083 / 4.951	0.097 / 5.81
Data:GT	5%	0.159 / 6.384	1 / 40.31	0.993 / 40.03	0.158 / 6.346	0.863 / 34.76	0.184 / 6.614	**0.131 / 5.244**
	15%	0.121 / 4.854	0.148 / 5.91	0.382 / 15.33	0.138 / 5.541	0.492 / 19.8	0.162 / 5.845	**0.11 / 4.43**
	25%	**0.106 / 4.24**	0.145 / 5.813	0.111 / 4.475	0.114 / 4.583	0.214 / 8.597	0.144 / 5.2	0.103 / 4.126
	50%	**0.088 / 3.547**	0.112 / 4.501	0.09 / 3.616	0.097 / 3.902	0.112 / 4.503	0.128 / 4.624	0.095 / 3.801
	75%	**0.079 / 3.189**	0.1 / 4.027	0.081 / 3.247	0.088 / 3.515	0.091 / 3.652	0.12 / 4.303	0.092 / 3.712
Data:CA	5%	0.439 / 32.44	1 / 76.562	0.998 / 76.464	0.435 / 32.672	0.978 / 74.949	0.434 / 33.915	**0.414 / 31.431**
	15%	0.35 / 25.964	0.396 / 29.762	0.986 / 75.582	**0.341 / 25.471**	0.936 / 72.046	0.369 / 28.735	0.344 / 25.94
	25%	0.304 / 22.578	0.308 / 23.05	0.679 / 50.661	**0.297 / 22.207**	0.886 / 68.426	0.32 / 24.908	0.293 / 22.045
	50%	0.222 / 16.466	0.23 / 17.172	**0.213 / 15.892**	0.237 / 17.646	0.731 / 57.059	0.235 / 18.274	0.248 / 18.483
	75%	0.198 / 14.648	0.2 / 14.675	**0.171 / 12.636**	0.209 / 15.603	0.472 / 36.816	0.197 / 15.249	0.223 / 16.556

Table 4.1 MRE/RMSE scores for data imputation. The best two results are bold and underlined respectively.

- **TRMF**: Temporal regularized matrix factorization [57]. TRMF exploits the autoregressive structure among temporal embeddings $\mathbf{v}_t$. TRMF uses a set (L) containing the lag indices l denoting a dependency between t^{th} and $(t-l)^{th}$ time points. We take the lag as $\{1, 2, T\}$, where T denotes the number of time intervals in a day. We stack $d-8, d-7, \ldots d$ data matrices to predict the samples for d^{th} day, thereby incorporating the dependencies over days and week.

- **BTMF**: Bayesian Temporal Matrix Factorization [73] is a Bayesian extension of TRMF which outperforms TRMF and other imputation methods for traffic data.

Tensor Completion Frameworks

To evaluate the performance of the tensor completion algorithms with VBFSI we use $\boldsymbol{X} \in \boldsymbol{R}^{n \times t \times k}$, a three-way tensor. For an effective comparison between matrix and tensor completion frameworks, k is set as 7 [31]. However, we set the $k = 8$ to capture the weekly pattern, usually observed in traffic data.

- **BCPF**: Bayesian CP Factorization [65]. BCPF is a bayesian tensor-based imputation method that incorporates a sparsity-inducing prior over multiple latent factors. BCPF is effective even for a higher percentage of missing data.

- **TRLRF**: Tensor ring low-rank factors [74] is an efficient and high-performance tensor completion algorithm based on TR(Tensor Ring)decomposition, which employed low-rank constraints on the TR latent space. TRLRF outperforms the state of the art tensor completion algorithm for synthetic and real-world data.

Robust Imputation Frameworks

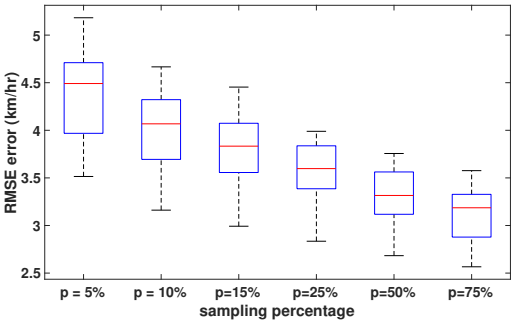
We compare RVBFSI with the following robust imputation methods.

- **RegL$_1$**: Regularized L_1 Augmented Lagrange Multiplier [61] is proposed to approximate a low-rank data matrix in the presence of missing data and outliers.

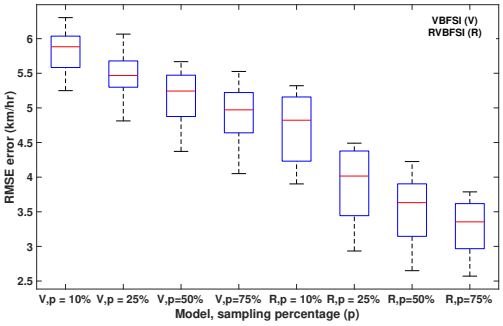
- **BRTF**: Bayesian Robust Tensor Factorization [62] uses a variational Bayesian approach for robust tensor factorization in the presence of missing entries and outliers.

4.5.3 Performance Comparison

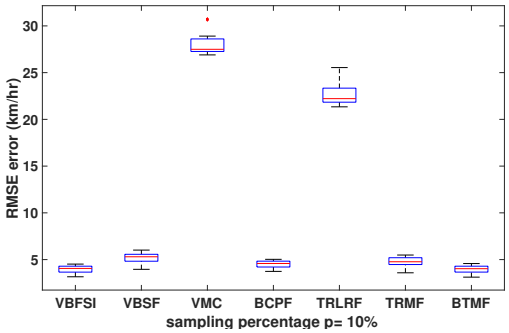
The performance comparison of VBFSI with the current state of the art methods is shown in Table 4.1 and Fig 5.4. The performance of RVBFSI for the imputation task for outlier corrupted data is shown in Table 4.2.



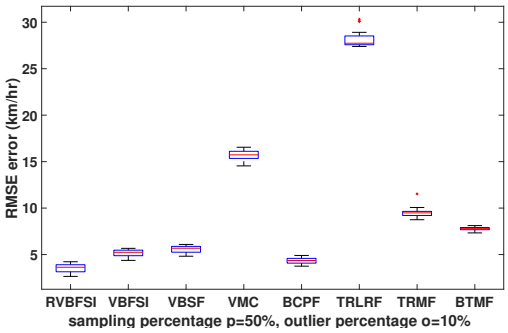
(a) VBFSI for different different sampling p =[0.05, 0.10, 0.25,0.5, 0.75



(b) Comparison of VBFSI and RVBFSI for different p and outlier percentage $o = 10\%$



(c) Imputation performance for $p = 10\%$



(d) Imputation performance for p=50% and o=10%

Fig. 4.4 Sensitivity Analysis for Data:DT

	o=5%				o=10%			
p %	10%	25%	50%	75%	10%	25%	50%	75%
RVFSI	**0.167 / 4.672**	**0.14 / 3.91**	**0.126 / 3.544**	**0.119 / 3.337**	**0.17 / 4.78**	**0.14 / 3.925**	**0.128 / 3.573**	**0.118 / 3.314**
VBFSI	0.188 / 5.277	0.177 / 4.972	0.17 / 4.778	0.158 / 4.427	0.208 / 5.86	0.194 / 5.443	0.184 / 5.173	0.175 / 4.899
VBSF	0.305 / 8.583	0.201 / 5.656	0.177 / 4.98	0.166 / 4.65	0.391 / 11.01	0.237 / 6.65	0.198 / 5.561	0.184 / 5.16
VMC	0.995 / 28.05	0.798 / 22.4	0.368 / 10.35	0.306 / 8.605	0.996 / 28.08	0.937 / 26.41	0.562 / 15.8	0.44 / 12.38
BCPF	0.193 / 5.422	0.164 / 4.597	0.146 / 4.101	0.14 / 3.927	0.205 / 5.763	0.179 / 5.017	0.155 / 4.363	0.148 / 4.146
TRLRF	0.973 / 27.45	0.956 / 26.95	0.934 / 26.34	1.144 / 32.24	0.977 / 27.55	0.969 / 27.33	0.997 / 28.1	1.447 / 40.71
TRMF	0.382 / 10.76	0.419 / 11.78	0.265 / 7.426	0.217 / 6.05	0.565 / 15.91	0.56 / 15.75	0.338 / 9.492	0.291 / 8.139
BTMF	0.226 / 5.828	0.218 / 5.604	0.221 / 5.615	0.218 / 5.591	0.308 / 7.79	0.303 / 7.681	0.304 / 7.794	0.303 / 7.764
RegL_1	0.521 / 14.66	0.489 / 13.78	0.192 / 5.429	0.132 / 3.718	0.699 / 19.69	0.498 / 14.01	0.242 / 6.825	0.155 / 4.387
BRTF	0.243 / 6.829	0.232 / 6.522	0.138 / 3.888	0.131 / 3.706	0.218 / 6.133	0.2 / 5.609	0.152 / 4.285	0.145 / 4.121

Table 4.2 MRE/RMSE scores for imputation of outlier corrupted data (DT), outlier percentage (o) is 5% and 10%.

Comparison with matrix completion methods:VBFSI outperforms VBSF for all the datasets. The performance of VBSF is comparable to VBFSI for higher sampling. In contrast, for lower sampling, the performance of VBSF degrades. VMC experience a similar trend as VBSF, where the performance is comparable for higher sampling and degrades for low sampling. VBFSI outperforms VMC for almost all the cases for traffic data (DT, PT, GT). However, for the air quality data (CA), the performance of VMC is better than VBFSI for a higher sampling percentage. VMC can capture the nonlinearity in the data for a high sampling percentage.

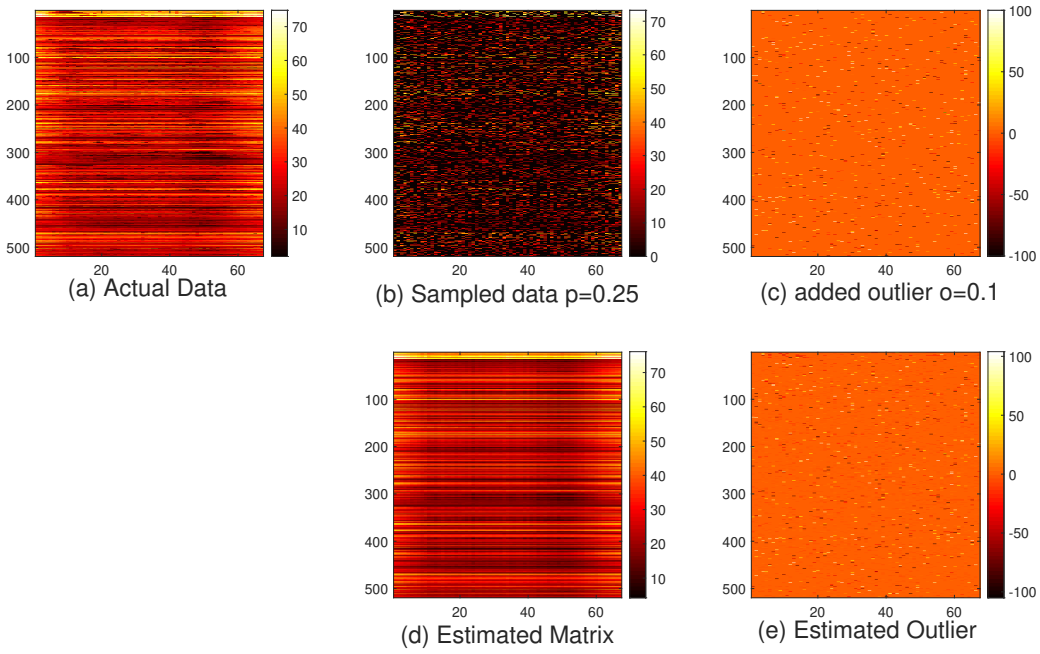


Fig. 4.5 Robust Matrix completion using RVBFSI for p=25% and o=10% (a) Actual Traffic data (DT) matrix $\mathbf{X}$, (b) $\mathbf{X}$ is sampled with 25 % of the entries, (c) 10% of the sampled locations will be corrupted with depicted outlier magnitude and location, (d) Sampled Matrix with outliers are estimated using RVBFSI (e) Sparse outlier matrix estimated using RVBFSI

Comparison with matrix completion frameworks exploiting previous days information: For low sampling percentage, VBFSI performs is comparable to BTMF in most of the cases. However, for higher sampling percentage, VBFSI outperforms BTMF. VBFSI outperforms TRMF in all the scenarios. One of the disadvantages of BTMF and TRMF is that rank is not tuned automatically. Moreover, BTMF uses gibbs sampling to impute the

tensor along with the temporal regularization. Gibbs sampling is relatively slower than the Variational Bayesian approach for parameter estimation [75].

Comparison with tensor completion methods: VBFSI outperforms TRLRF for all the scenarios. VBFSI performance is comparable to BCPF in most of the cases.

Effect of η on the performance of VBFSI

When $\eta = 0$, VBFSI reduces to VBSF. For higher sampling, the performance of VBFSI is comparable to VBSF. However when the sampling is low, then the performance of VBSF degrades. Incorporating even the noisy prior subspace information in the architecture can reduce the sampling complexity of the algorithm by a logarithmic factor [58]. Therefore, for low sampling, VBFSI performs better than VBSF as we have incorporated the prior information in the architecture using η. For lower sampling (p is low), the value of η is high, and it decreases exponentially with the increase in sampling as shown in Fig. 4.3.

Performance analysis in the case of outlier

To compare the performance of VBFSI and RVBFSI in the case of outliers, we artificially add the outliers in the Data: DT. We randomly add 5% and 10% of the outliers in the total sampled data, i.e., the number of outliers is $o \times p$ fraction of the overall data. The entries corrupted with outliers are uniformly distributed between $[-\sigma, \sigma]$, where σ is set as 100 in our experimentation. This artificial injection of outlier is used to evaluate the robustness of the algorithm in the literature [76]. The imputation performance of VBFSI degrades in the presence of outliers (Fig. 4.4(b)). However, RVBFSI can improve the performance of imputation, as shown in Fig. 4.4(b). Moreover, the locations and magnitude of the outliers can be recovered, as shown in Fig. 4.5(e). Performance comparison RVBFSI and VBFSI with other imputation methods are shown in Table 4.2. The performance of VBFSI and BCPF is comparable for all the sampling. In comparison, the performance of VBFSI

is similar to VBSF for high sampling. While RVBFSI outperforms all other imputation methods, including RegL_1 and BRTF significantly.

4.6 Summary

In this chapter, we presented a Bayesian approach towards extreme matrix completion wherein we estimate the underlying temporarily varying subspace using a Variational Bayesian technique. We enforced the low-rank structure to estimate the matrix, where the rank is automatically learned using data by an automatic relevance determination (ARD) approach. The Bayesian approach offers greater flexibility in modeling the data, especially in the case of extreme missing case or fewer samples. The above formulation is jointly coupled with the state space autoregressive model, as well as a penalty function on the slowly varying subspace to model the temporal and periodic evolution in the data. We evaluate the proposed Variational Bayesian with Subspace Information (VBFSI) method to impute matrices in real-world traffic data sets.

Chapter 5

Spatio-Temporal Sampling

5.1 Overview

There are two steps to create a dense spatiotemporal data map for an area. The first is to sample the best representative data samples of an area in space and time, and the second is the extrapolation or imputation of the data based on the sampled set. The second problem is discussed in previous chapters. The inherent temporal and spatial redundancy available in the spatiotemporal data can be used to impute the remaining (incomplete) data by utilizing the matrix completion techniques and obtaining a dense data map in a fraction of the corresponding fixed sensing cost. However, the accuracy of imputation is dependent on the extent to which spatiotemporal sampling captures the inherent structure of the spatiotemporal matrix. There has been limited research on the spatiotemporal sampling problem where the challenge is to pick those sets of paths (using vehicles) that perform representative sampling in space and time.

In this chapter, we address the problem of the choice of efficient drive by sensing routes for the city of Delhi to sample AQI (Air quality index) data. We investigate the usage of public transit buses for drive-by sensing. We propose novel frameworks to pick

those routes that exploit the spatiotemporal structure in the air quality data resulting in an effective imputation and therefore achieving the required dense air quality map.

We begin in section 5.2 with the problem of selective drive-by sensing, where we introduce the problem of sampling with the moving vehicles and further motivate the representative spatiotemporal sampling using moving vehicles. We then provide a background review in section 5.3 of all the terms that are used in this chapter. Rather than running dedicated vehicles to sample the data, we motivate the problem of sampling the data using public transit, i.e. buses. We discussed the bus selection framework for selective drive-by sensing in section 5.4 where we select the best set of buses to sample the spatiotemporal data. We propose two frameworks for representative sampling of spatiotemporal data in section 5.4.2 and 5.4.3.

The best way to test the efficacy of the proposed method is to test the estimation performance on the unsampled data. However, practically it is not possible to access the real-world air quality data without deploying the sensors on the selected set of buses. Therefore, to evaluate the performance of the proposed drive-by sensing framework, we proposed to simulate real-world air quality data by exploiting the smoothness in the spatiotemporal domain in section 5.5.1. Then we evaluate the performance of the sampled spatiotemporal data using the bus selection framework by obtaining the error in the estimation of the dense maps. We have already discussed spatiotemporal estimation frameworks in previous chapters. However, dense air quality maps cannot be computed using the traditional matrix completion framework for the cold start locations. Cold start locations refer to the locations that are not sampled even once in any of the timestamps. We provide a simple extension of the proposed framework in chapter 2 to incorporate the cold start locations estimation in section 5.5.2. Section 5.5 describes all the experimental details to evaluate the performance of the proposed method.

Finally, we summarize the chapter in section 5.6.

5.2 Selective Drive By Sensing

Air pollution has drawn attention in recent years because of its profound effect on people's health [77]. It is crucial to assess the pollution level in a region by monitoring the air quality levels and recommending strategies to combat pollution. There are significant factors that affect the air quality of an area, such as transportation, electricity, fuel uses, industrial parameters such as power plant emissions, etc. Hence, air quality monitoring is required to make strategies for emission control as well as verifying strategies that control emissions.

The most widely used method to monitor air quality is by using static sensing, that is mounting sensors at fixed locations. The temporal resolution of static sensing is high, i.e., the air quality data is available for almost all the sampled timestamps. However, the spatial coverage in a region depends on the number of sensors installed which is generally limited owing to the cost constraints. We can leverage the spatial and temporal correlation in the air quality data to perform cost-effective spatiotemporal monitoring. We can share/multiplex an air quality sensor over multiple locations and perform sequential temporal sensing without losing air quality information. Moving sensor paradigm can be a candidate to obtain such dense AQ (Air Quality) map without needing an expensive setup of hundreds of static monitors [7, 8]. With the technological progress in AQ sensing, it is now possible to put these monitors in buses and other public transit and perform spatiotemporal AQ sensing. However, the moving sensors reading would be spatially and temporally sparse and incomplete because of the unavailability of the moving vehicle across all the spatial locations and timestamps.

Thanks to the inherent temporal and spatial redundancy available in the air quality data, one can potentially impute the remaining (incomplete) data by utilising the matrix completion techniques [31, 78] and obtain a dense air quality map in a fraction of the corresponding fixed sensing cost. However, the quality of imputation crucially relies on

the "quality" of drive-by sensing and the choice of "routes" that vehicles take to sample the air quality.

The problem of dense air quality maps using drive-by sensing is defined as: Given a binary bus occupancy data $\mathscr{Y} \in \{0,1\}^{|\mathscr{L}|\times|\mathscr{T}|\times|\mathscr{B}|}$, where $\mathscr{L}$ denotes the set of locations, $\mathscr{T}$ denotes the set of timestamps and $\mathscr{B}$ denotes the buses. The entry $y_{l,t,b} = 1$ if the bus b samples the location l in the time stamp t. The aim is to pick the best subset of buses $\mathscr{M} \in \mathscr{B}$ that perform representative sampling for the spatiotemporal AQ matrix. Subsequently, by appropriately modeling the spatial and temporal structure present in the AQ data sampled by the set of buses $\mathscr{M}$, one can extrapolate/impute the missing data in the spatiotemporal matrix. In a nutshell, the aim is to pick those buses that exploit the spatiotemporal structure in the air quality data resulting in an effective imputation and thereby creating dense anytime-anywhere AQ map.

Limited work on selecting the optimal set of vehicles for sensing spatiotemporal data models each entry in the spatiotemporal matrix as independent and tries to maximize the number of entries in the matrix [14, 15, 17]. Considering all the locations as independent ignores the correlation across locations, thereby highlighting the sub-optimality of such an approach. Indeed, sampling two consecutive neighbourhood locations is less effective than sampling the diverse spread across locations in an area.

Our work exploits the smoothness in the spatial locations to select buses that sample diverse and representative sets of locations. Further, the air quality data for a given location is smooth in time, i.e., the sensor data varies slowly in time with decreasing correlation as the interval increases. Thereby, sampling the consecutive timestamps of a site is a less effective strategy than sampling at distant time stamps. The proposed Regressive Facility Location (RFL) framework encapsulates the temporal smoothness using an autoregressive time series structure to sample the spatiotemporal data. Also, since the future temporal data is not available to create the dense map at a particular timestamp, we use the causal

temporal smoothness in the proposed RFL framework. This ensures that if a location l is sampled at a given time, other buses can bypass sampling location l and the nearby locations in the subsequent neighboring future time stamps. The Regressive Facility Location (RFL) framework proposes to encapsulate the slowly varying temporal data pattern into the facility location framework and select the buses such that the sampled locations will be representative of the area and will be diverse across all time stamps.

Practically it is not possible to access the real-world air quality data without deploying the sensors on the selected set of buses. Therefore, to evaluate the performance of the proposed RFL drive by sensing framework, we simulate real-world air quality data. Further, we obtain the dense air quality maps using the matrix completion framework from the sampled spatio-temporal data and observe that RFL provide more accurate dense air quality maps.

5.3 Background Review

5.3.1 Vehicle Subset Selection

There has been limited work on selecting the optimal set of vehicles for sensing spatiotemporal data. Authors in [17] propose mobile sensor placement for vehicles to maximize coverage. Authors in [14] propose a drive by sensing framework for taxis and buses. Authors in [15] proposed point of interest oriented (POIs) bus selection algorithm to select buses where the coverage of a bus is defined in terms of the POIs. Authors in [16] proposed the optimal placement of reference monitors to make mobile sensors k-hop calibrable. However, the mobile sensors are fixed and reference monitors are optimally selected. All these framework maps the problem of vehicle subset selection to the maximum cover problem or set cover problem to maximize the spatiotemporal coverage.

5.3.2 Greedy Submodular Maximization

A function is submodular if it is monotonically non-decreasing and exhibits the property of diminishing returns.

Definition 1: A function f is monotonically non decreasing if $\forall\ \mathscr{C} \subseteq \mathscr{D}$

$$f(\mathscr{C}) \leq f(\mathscr{D})$$

Definition 2: A function f: $2^{\mathscr{B}} \rightarrow \mathbf{R}$ is submodular if $\forall\ \mathscr{C} \subseteq \mathscr{D} \subseteq \mathscr{B}$ and every $b \in \mathscr{B} \backslash \mathscr{D}$

$$f(\mathscr{C} \cup b) - f(\mathscr{C}) \geq f(\mathscr{D} \cup b) - f(\mathscr{D})$$

Given a submodular set function f , maximization of f over all subsets of size at most k of the ground set $\mathscr{G}$, i.e. $|\mathscr{G}| = k$,

$$f(\mathscr{V}) = \max_{\mathscr{V}:|\mathscr{V}| \leq k} f(\mathscr{V}) \tag{5.1}$$

is an NP-hard problem [79]. A monotone non-decreasing submodular function solution can be approximated by a greedy algorithm within $(1 - \frac{1}{e})$ of the global maximum [80, 81]. Submodularity is gaining popularity in a large range of combinatorial and machine learning optimization problems. Submodular optimization focus on selecting a subset of items from a whole set to maximize a given submodular function over the set. For these problems of subset selection, the greedy algorithm can always achieve an approximate solution with theoretical guarantees. Submodular subset selection for subset sensor selection [82, 83] is explored where the combinatorial optimization problem of maximizing a submodular function (mutual information, uniform matroids) is proposed between the chosen locations and the locations which are not selected. Thereby, picking the best subset of sensor locations out of the entire set of locations based on maximizing a submodular function

for effective sampling. However, the techniques are explored for static sensor selection. Our aim is to select different locations at distinct time stamps using sensors placed on the vehicles. Thereby we are selecting the best set of vehicles which in turn traverse a path in time. Hence providing representative sampling in space and time.

Facility Location

Facility location is used in the literature for data summarization, clustering, and video summarization based problems to provide the representative summary of the data points [84–86]. Facility location provides not only diverse but representative samples by considering the similarity between the selected and the remaining elements of the dataset. If $S_{i,j}$ denotes the similarity between the points i and j. Then the facility location function for the subset $\mathscr{B}$ is defined as

$$f(\mathscr{B}) = \sum_{i \in \mathscr{V}} \max_{j \in \mathscr{B}} (S_{i,j})$$

where $\mathscr{V}$ is the ground set. The idea is to find the most representative sampled point corresponding to every point in the ground set using the facility location function. Maximizing the facility location function will ensure that sampled points are the most representative samples of the entire set. In terms of clustering, data points selected by the facility location will correspond to the cluster medoids [84]. Also, note that the facility location is submodular; thereby, the greedy algorithm can always achieve an approximate solution with theoretical guarantees. The objective of the selective drive by sensing to sample the spatiotemporal matrix is to pick the most representative locations across all time stamps. The sampled location depends on the vehicle selected. Therefore, we modify the existing facility location framework to our setting and provide the theoretical guarantees by proving that the proposed facility location framework is submodular.

5.3.3 Generating Smooth Graph Signals

To generate a smooth graph signal across nodes, an adjacency matrix can be used. Consider a graph with $\mathscr{L}$ set of nodes/locations, the weighted adjacency matrix can be defined as $\mathbf{G} \in \mathbf{R}^{L \times L}$ can be decomposed as.

$$\mathbf{G} = \mathbf{U}\Sigma\mathbf{U}^T$$

The graph Fourier transform of a signal $x \in \mathbf{R}^L$ is given by $\hat{x}$.

$$\hat{x} = \mathbf{U}^{-1}x$$

The actual signal can be expressed as

$$x = \mathbf{U}\hat{x}$$

The signal is smooth in the time domain if the signal is bandlimited in the frequency domain [87–91],

$$\hat{x}_k = 0 \ \forall k \geq m$$

Therefore

$$x = \mathbf{U}_{(m)}\hat{x} \tag{5.2}$$

where $\mathbf{U}_{(m)} \in \mathbf{R}^{L \times m}$ represents the first m eigenvectors of the matrix $\mathbf{G}$.

The signal $\hat{x}$ can be sampled from random normal distribution as described in the literature [88, 90, 91]. Then using the signal $\hat{x}$ and $\mathbf{U}$, an actual smooth signal can be constructed over the nodes of the graph via Eq. 5.2. The signal generated will be smooth on the locations only. We extend this theory to generate smooth signals across locations and time.

5.3.4 Dense maps using missing data imputation

Air quality data exhibits both spatial and temporal correlation, thereby generating redundancy (not high-rank nature) [92, 93]. Low-rank matrix completion has been proposed to estimate the missing spatiotemporal data [31, 94]. Further state-space model for incorporating the temporal evolution in the matrix completion framework is also proposed in the literature [57, 63, 78]. We have explored spatiotemporal estimation in the previous chapters. These spatiotemporal estimation techniques can be used to provide dense maps by imputing the missing data. However, dense air quality maps cannot be computed using the traditional matrix completion framework for the cold start locations. Cold start locations refer to the set of locations that are never sampled by the selected set of buses. A similarity matrix can be exploited to impute the data for the cold start locations [95, 96]. Section 5.5.2 provided a simple extension of VBSF that is proposed in chapter 2 to impute the cold start locations.

5.4 Bus Selection Framework

Bus selection for drive by sensing is defined as follows: We are given a binary bus occupancy tensor $\mathscr{Y} \in \{0,1\}^{|\mathscr{L}|\times|\mathscr{T}|\times|\mathscr{B}|}$, where $\mathscr{L}$ is the set of locations, $\mathscr{T}$ is the set of timestamps, and $\mathscr{B}$ is the entire set of buses. Buses run on a specific route with a time schedule. The entry $y_{l,t,b} = 1$ denotes the availability of the sensor reading placed at bus b for a particular location l and time slot t. Our work aims to pick k buses from the entire set of buses B for drive by sensing, where $\mathscr{M} \subset \mathscr{B}$ represents the selected set of buses. For a subset of buses $\mathscr{M} \subset \mathscr{B}$, we define a binary spatiotemporal sampling matrix $\Theta \in \{0,1\}^{L\times T}$ as

$$\Theta(\mathscr{M}) = \max(\sum_{k\in\mathscr{M}} \mathscr{Y}_{l,t,k}, 1) \;\; \forall\,(l \in \mathscr{L}),\; \forall\,(t \in \mathscr{T}) \tag{5.3}$$

$\Theta(\mathscr{M})$ is the spatiotemporal sampling matrix where an entry i, j in the matrix denotes that location i is sampled at time stamp j by the set of buses $\mathscr{M}$. The distance between two locations i and j is defined as $\mathbf{D}_{i,j}$. To incorporate the smoothness in the spatial locations, we define the similarity between locations in $\mathbf{S} \in \mathbf{R}^{L \times L}$ as:

$$\mathbf{S}_{i,j} = 1 - \frac{\mathbf{D}_{i,j}}{\max(\mathbf{D})} \tag{5.4}$$

We use the normalized distance between two locations $\frac{\mathbf{D}_{i,j}}{\max(\mathbf{D})}$ between 0 and 1. The smaller the normalized distance between two locations, the higher is the similarity. Another approach is to learn λ in the similarity matrix $\mathbf{S}_{i,j} = \exp^{(-\lambda D_{i,j})}$. An example of similarity matrix is shown in Fig. 5.2 (b) where as the distance between the locations increases similarity decreases.

5.4.1 Baseline methods

Max Coverage (MC)

In the classical max coverage problem, we are given different sets and numbers of subsets k to be selected as input. Each set contains some set of elements and these elements can be common within different sets. The idea is to select k of these sets so that the max number of elements are covered. We model our problem as the max coverage problem where each entry in the sampling matrix Θ is considered as an element[17]. We have the set of buses, and each bus e samples some elements in the matrix $\Theta(e)$. Max coverage framework selects the k number of buses, set $\mathscr{M} \subset \mathscr{B}$ where $|\mathscr{M}| = k$ such that entries in the sampling matrix $\Theta(\mathscr{M})$ are maximized. The maximum coverage problem is NP-hard and submodular; there exists a greedy heuristic that provides a solution within an approximation ratio of $(1 - \frac{1}{e})$[97]. We use percentage coverage as a measure for the max coverage algorithm. The gain for Max Coverage to maximize the occupancy of the selected buses $\mathscr{M}$ is defined

as

$$PC(\mathscr{M}) = \frac{\sum_{i\in\mathscr{L}, j\in\mathscr{T}} \Theta(\mathscr{M})_{i,j}}{L \times T} * 100 \tag{5.5}$$

The greedy approach for the Max Coverage is shown in Algorithm 5. The input of the algorithm are k, $\mathscr{B}$, $\mathscr{Y}$. The aim is to pick the best subset of buses that maximizes Eq 5.5. The best subset of buses are greedily selected in $\mathscr{M}$ as shown in Algorithm 5.

Algorithm 5: Maximum Coverage

Input: m, $\mathscr{B}$, $\mathscr{Y}$
Output: $\mathscr{M} \subset \mathscr{B}$ of size m
Initialization:$\mathscr{M} \leftarrow \phi$
for i =1 to k **do**
  for each $e \in \mathscr{B}$ **do**
    $\Theta(\mathscr{M} \cup e) = max(\sum_{m\in\mathscr{M}\cup e} \mathscr{Y}_{i,j,m}, 1)$
    $f(\mathscr{M} \cup e) = \frac{\sum_{i\in\mathscr{L}, j\in\mathscr{T}} \Theta(\mathscr{M}\cup e)}{L*T} * 100$
  end for
  $e^* = \arg\max_e f(\mathscr{M} \cup e)$
  $\mathscr{M} \leftarrow \mathscr{M} \cup \{e^*\}$
end for

Max Coverage over Location (MCL)

We model our problem as the max coverage over location problem where each location is considered as an element. We have the set of buses, each bus samples some locations over all the timestamps. Max Coverage over location framework selects the k number of buses, set $\mathscr{M} \subset \mathscr{B}$ where $|M| = k$ such that maximum locations are sampled. In max coverage over location, we are maximizing the number of locations sampled by the selected buses, while in max Coverage framework, we are maximizing the entries of the sampling matrix Θ to increase the coverage over location as well as time. We use Percentage Stop coverage as a measure for the Max Coverage over Location algorithm as shown in (5.6). Max coverage framework treats every entry in the matrix Θ as independent, thereby sampling different

time stamps for an already sampled location will be the same as sampling a different location. Max Coverage over Location samples diverse set of locations as compared to the Max coverage problem.

$$PSC(\mathscr{M}) = \frac{\sum_{i \in \mathscr{L}} max\{\sum_{k \in \mathscr{M}, j \in \mathscr{T}} \mathscr{Y}_{i,j,k}, 1\}}{L} * 100 \tag{5.6}$$

Algorithm 6: Maximum Coverage over location

Input: $m,\ B,\ \mathscr{Y}$
Output: $\mathscr{M} \subset \mathscr{B}$ of size m
Initialization:$\mathscr{M} \leftarrow \phi$
for i =1 to k **do**
  for each $e \in \mathscr{B}$ **do**
    $x(\mathscr{M} \cup e) = max(\sum_{m \in \mathscr{M} \cup e, j \in \mathscr{T}} \mathscr{Y}_{i,j,m}, 1)$
    $f(\mathscr{M} \cup e) = \frac{\sum_{i \in \mathscr{L}} x(\mathscr{M} \cup e)}{L} * 100$
  end for
  $e^* = \arg\max_e f(\mathscr{M} \cup e)$
  $\mathscr{M} \leftarrow \mathscr{M} \cup \{e^*\}$
end for

5.4.2 Proposed Facility Location over space (FLS)

Max coverage and max coverage location treat every location as independent while selecting the set of buses, which sample the spatiotemporal matrix. However, a correlation exists across the locations readings; the sensor readings will be smooth over neighboring locations. We thereby use Facility Location over space (FLS) to model the smoothness in the locations using a similarity matrix.

For the selected set of buses $\mathscr{M}$, the sampling matrix is defined in (5.3). For a time stamp t, the sampled locations by the set of buses $\mathscr{M}$ is defined as $\theta_t(\mathscr{M})$,

$$\theta_t(\mathscr{M}) = \{i\} \ \forall \ \Theta_{i,t}(\mathscr{M})|_{=1}$$

$$FLS(\mathscr{M}) = \frac{\sum_{t\in\mathscr{T}}\sum_{l\in\mathscr{L}} \pi_t^l(\mathscr{M})}{L\times T} \tag{5.7}$$

$$\pi_t^l(\mathscr{M}) = \max_{\forall m\in\theta_t(\mathscr{M})}(\mathbf{S}_{l,m}) \tag{5.8}$$

We compute the maximum pairwise similarity between all the locations ($i \in \mathscr{L}$) and the sampled set locations denoted by $\theta_t(\mathscr{M})$ for a timestamp t. The gain defined in (5.7) is maximized when the pairwise similarities between the locations and the nearest chosen location in the selected set of buses are maximized for all the timestamps. This ensures that we pick the buses that sample the locations that are representative of the entire area for all the timestamps.

Facility location finds the assignment similarity of each location to one representative location traversed by the selected set of buses $\mathscr{M}$.

Theorem 1: The function defined in (5.7) is monotone submodular. Therefore a greedy heuristic provides a solution within an approximation ratio of $(1-\frac{1}{e})$.
Proof: Let θ_t be the set of locations that correspond to sampled locations in the sampling matrix Θ for time stamp (column) t. The FLS gain is defined as:

$$FLS(\mathscr{M}) = \frac{\sum_{t\in\mathscr{T}}\sum_{l\in\mathscr{L}} \pi_t^l(\mathscr{M})}{L\times T} \tag{5.9}$$

where

$$\pi_t^l(\mathscr{M}) = \max_{\forall m \in \theta_t(\mathscr{M})} (\mathbf{S}_{l,m}) \tag{5.10}$$

Monotone non decreasing : A function f is submodular if $\forall\ \mathscr{C} \subseteq \mathscr{D}$

$$f(\mathscr{C}) \leq f(\mathscr{D})$$

Proof:

Let $\mathscr{C}$ and $\mathscr{D}$ be the subset of buses following $\forall\ \mathscr{C} \subseteq \mathscr{D} \subseteq \mathscr{B}$. We define $\Theta(\mathscr{C})$ and $\Theta(\mathscr{D})$ as the sampling matrix for the subset of buses $\mathscr{C}$ and $\mathscr{D}$.

Let θ_t be the set of locations that corresponds to 1 in the sampling matrix $\Theta(\mathscr{C})$ for the time stamp t and let $\theta_t^{'}$ be the set of locations that corresponds to 1 in the sampling matrix $\Theta(\mathscr{D})$ for the time stamp t.

$$\theta_t = \{i\}\ \forall\ \Theta_{i,t}(\mathscr{C})|_{=1}$$

$$\theta_t^{'} = \{i\}\ \forall\ \Theta_{i,t}(\mathscr{D})|_{=1}$$

Since $\mathscr{C} \subseteq \mathscr{D}$, therefore $\theta_t \subseteq \theta_t^{'}\ \forall\ t = 1$ to T.

For all $l \in \mathscr{L}, t \in \mathscr{T}$ we have

$$\max_{\forall m \in \theta_t} (\mathbf{S}_{l,m}) \leq \max_{\forall m \in \theta_t^{'}} (\mathbf{S}_{l,m}) \tag{5.11}$$

Hence $f(\mathscr{C}) \leq f(\mathscr{D})$

Submodular : A function f is submodular if $\forall\ \mathscr{C} \subseteq \mathscr{D} \subseteq \mathscr{B}$ and $b \in \mathscr{B} \backslash \mathscr{D}$

$$f(\mathscr{C} \cup b) - f(\mathscr{C}) \geq f(\mathscr{D} \cup b) - f(\mathscr{D})$$

Consider a bus $b \in \mathscr{B} \setminus \mathscr{D}$, We add the bus b in both $\mathscr{C}$ and $\mathscr{D}$ and analyse the gain. Let the bus traverse the set of location G_t for a time stamp t.

$$G_t = \{i\} \ \forall \ \Theta_{i,t}(b)|_{=1}$$

For all $l \in L$ and $t = 1$ to T we have

$$\begin{aligned} \pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) &= \max_{\forall m \in \theta_t \cup G_t} (\mathbf{S}_{l,m}) - \max_{\forall m \in \theta_t} (\mathbf{S}_{l,m}) \\ &= \max(0, \ \max_{\forall m \in G_t} (\mathbf{S}_{l,m}) - \max_{\forall m \in \theta_t} (\mathbf{S}_{l,m})) \end{aligned} \tag{5.12}$$

Since

$$\max_{\forall m \in \theta_t} (\mathbf{S}_{l,m}) \leq \max_{\forall m \in \theta_t'} (\mathbf{S}_{l,m})$$

Therefore,

$$\begin{aligned} \pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) &= \max(0, \ \max_{\forall m \in G_t} (\mathbf{S}_{l,m}) - \max_{\forall m \in \theta_t} (\mathbf{S}_{l,m})) \\ &\geq \max(0, \ \max_{\forall m \in G_t} (\mathbf{S}_{l,m}) - \max_{\forall m \in \theta_t'} (\mathbf{S}_{l,m})) \end{aligned} \tag{5.13}$$

$$\pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) \geq \max(0, \ \max_{\forall m \in G_t} (\mathbf{S}_{l,m}) - \max_{\forall m \in \theta_t'} (\mathbf{S}_{l,m})) \tag{5.14}$$

$$\pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) \geq \max_{\forall m \in \theta_t' \cup G_t} (\mathbf{S}_{l,m}) - \max_{\forall m \in \theta_t'} (\mathbf{S}_{l,m}) \tag{5.15}$$

$$\pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) \geq \pi_t^l(\mathscr{D} \cup b) - \pi_t^l(\mathscr{D}) \tag{5.16}$$

From Eqs (5.12-5.16) for all $l \in \mathscr{L}$ and $t = 1$ to $\mathscr{T}$ we have,

$$f(\mathscr{C} \cup b) - f(\mathscr{C}) \geq f(\mathscr{D} \cup b) - f(\mathscr{D})$$

Algorithm 7: Facility Location over Space

Input: $b,\ B,\ \mathscr{Y},\ \mathbf{S}$
Output: $\mathscr{M} \subset \mathscr{B}$ of size b
Initialization:$\mathscr{M} \leftarrow \phi$
for each $e \in \mathscr{B}$ **do**
  $\Theta(\mathscr{M} \cup e) = \max(\sum_{b \in \mathscr{M} \cup e} \mathscr{Y}_{i,j,b}, 1)$
  for each $t \in \mathscr{T}$ **do**
    $\theta_t(\mathscr{M} \cup e) = \{i\}\ \forall\ \Theta_{i,t}(\mathscr{M} \cup e)|_{=1}$
    $\pi_t^l(\mathscr{M} \cup e) = \max_{\forall m \in \theta_t(\mathscr{M} \cup e)} (\mathbf{S}_{l,m})\ \forall l \in \mathscr{L}$
  end for
  $f(\mathscr{M} \cup e) = \frac{\sum_{t \in \mathscr{T}} \sum_{l \in \mathscr{L}} \pi_t^l(\mathscr{M} \cup e)}{LT}$
  $e^* = \arg\max_e f(\mathscr{M} \cup e)$
  $\mathscr{M} \leftarrow \mathscr{M} \cup \{e^*\}$
end for

5.4.3 Proposed Regressive Facility Location (RFL)

FLS selects the buses that sample representative locations across all timestamps. However, fails to capture the temporal correlation while selecting the set of buses and treat all timestamps as independent. The sensor observations of a location over time are not independent and vary slowly over time. The FLS framework can be modified to incorporate the space and time similarity as shown in Fig. 5.1.

The modified framework incorporates the temporal causal similarity i.e., the future time data is not available to infer the previous time stamps using ρ. The Facility location incorporating the space and time causal similarity is called Facility Location over Space-Time (FLST). The similarity coefficients for location i and timestamp j is denoted in Fig. 5.1. Blue edges in the graph denote the similarity between different locations for a

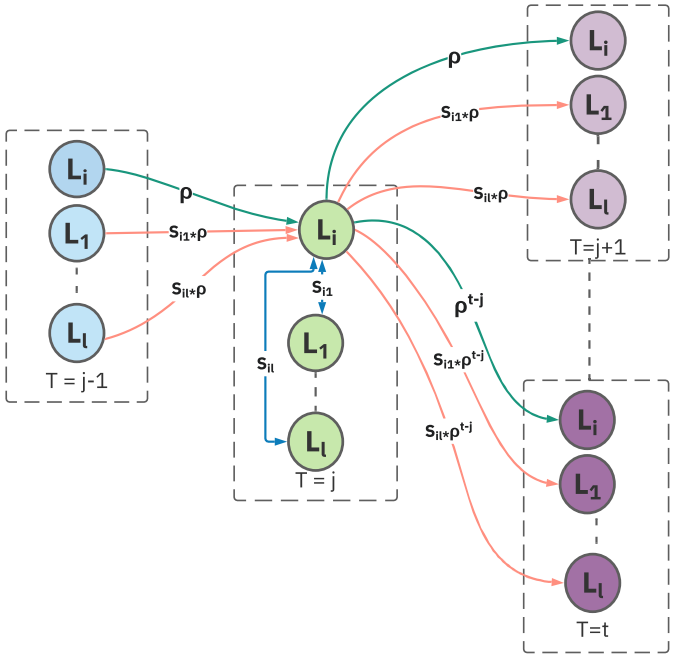
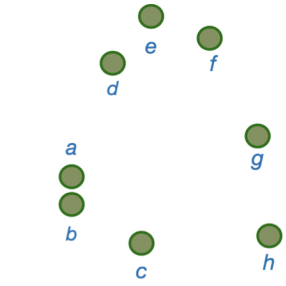


Fig. 5.1 Spatial and temporal similarity



(a) Different locations in the space (G)

	a	b	c	d	e	f	g	h
a	1	0.95	0.7	0.7	0.6	0.5	0.7	0.6
b	0.95	1	0.85	0.65	0.55	0.45	0.66	0.65
c	0.7	0.85	1	0.5	0.4	0.3	0.7	0.7
d	0.7	0.65	0.5	1	0.9	0.8	0.6	0.55
e	0.6	0.55	0.4	0.9	1	0.9	0.55	0.5
f	0.5	0.45	0.3	0.8	0.9	1	0.7	0.55
g	0.7	0.66	0.7	0.6	0.55	0.7	1	0.7
h	0.6	0.65	0.7	0.55	0.5	0.55	0.7	1

(b) Similarity matrix **S** for G

a	
b	1
c	
d	
e	1
f	
g	1
h	

(c) Θ_1

a	0.95
b	1
c	0.85
d	0.9
e	1
f	0.9
g	1
h	0.7

(e) π_1 (FLS)

a	0.95
b	1
c	0.85
d	0.9
e	1
f	0.9
g	1
h	0.7

(g) π_1 (RFL)

a	
b	
c	1
d	
e	
f	1
g	
h	

(d) Θ_2

a	0.7
b	0.85
c	1
d	0.8
e	0.9
f	1
g	0.7
h	0.7

(f) π_2 (FLS)

a	0.9
b	0.95
c	1
d	0.85
e	0.95
f	1
g	0.95
h	0.7

(h) π_2 (RFL)

Fig. 5.2 (a) Represents the locations of an example graph G, (b) The Similarity matrix S for Graph G, (c)-(d) Sampling matrix Θ for time t=1 and 2, (e)-(f) π_t for FLS, (g)-(h) π_t for RFL.

particular timestamp j. Note that $\rho = 0$ is a special case for FLS, and only blue edges are used for similarity in the FLS framework. FLST models the inter-temporal similarity for a particular location using ρ and is denoted by green edges in Fig 5.1. The inter-location and inter-temporal similarity are denoted by orange edges. Directed edges represent the similarity from one node to the other. Note that since we are modeling the temporal causal similarity, reading at location i is relevant only for the locations at future timestamps. The gain for the FLST is defined as

$$FLST(\mathscr{M}) = \frac{\sum_{t\in\mathscr{T}}\sum_{l\in\mathscr{L}} \pi_t^l(\mathscr{M})}{L\times T} \tag{5.17}$$

$$\pi_t^l(\mathscr{M}) = \max_{j,m\in\Theta(\mathscr{M})} (\mathbf{S}_{l,m} * \mathbf{T}_{t,j}) \tag{5.18}$$

where the Temporal causal similarity is defined as

$$\mathbf{T}_{t,j} = \begin{cases} \rho^{t-j}, & t \geq j \\ 0, & t < j \end{cases} \tag{5.19}$$

It is computationally expensive to compute the (5.17) since the comparison for every location and time is done for all the sampled data Θ. Therefore, we propose a fast algorithm that optimizes the gain defined in (5.17), with a reduced computational complexity. Let θ_t be the set of locations that correspond to sampled locations in the sampling matrix Θ for time stamp t. For $\pi_0 = 0$, $\rho \in [0,1]$, the regressive facility location gain is defined by

$$RFL(\mathscr{M}) = \frac{\sum_{t\in\mathscr{T}}\sum_{l\in\mathscr{L}} \pi_t^l(\mathscr{M})}{L\times T} \tag{5.20}$$

$$\pi_t^l(\mathscr{M}) = \max((\mathbf{S}_{l,m})_{\forall m \in \theta_t(\mathscr{M})}, \rho\, \pi_{t-1}^l(\mathscr{M}))\ \forall l \in \mathscr{L} \tag{5.21}$$

Corollary 1: FLST gain defined via (5.22) is equivalent to the RFL gain defined via (5.35).

$$\lambda_t^l(\mathscr{M}) = \max_{m,j \in \Theta(\mathscr{M})} (\mathbf{S}_{l,m} * \mathbf{T}_{t,j}) \tag{5.22}$$

$$\mathbf{T}_{t,j} = \begin{cases} \rho^{t-j}, & t \geq j \\ 0, & t < j \end{cases} \tag{5.23}$$

$$\pi_t^l(\mathscr{M}) = \max((\mathbf{S}_{l,m})_{\forall m \in \theta_t(\mathscr{M})}, \rho\, \pi_{t-1}^l(\mathscr{M}))\ \forall l \in \mathscr{L} \tag{5.24}$$

Proof:

For all $l \in \mathscr{L}$ and $t = 1$ using (5.22) we have,

$$\lambda_1^l(\mathscr{M}) = \max(\max(\mathbf{S}_{l,m})_{\forall m \in \theta_1(\mathscr{M})}, V)\ \forall l \in \mathscr{L} \tag{5.25}$$

$$V = \max(\mathbf{S}_{l,m})_{\forall m \in \theta_j(\mathscr{M})}\ \ (\forall j > 1) = 0$$

Therefore,

$$\lambda_1^l(\mathscr{M}) = \max((\mathbf{S}_{l,m})_{\forall m \in \theta_1(\mathscr{M})})\ \forall l \in \mathscr{L} \tag{5.26}$$

$$\lambda_1^l(\mathscr{M}) = \pi_1^l(\mathscr{M})$$

For all $l \in \mathscr{L}$ and $t = 2$ we have,

$$\lambda_2^l(\mathscr{M}) = \max(\max(\mathbf{S}_{l,m})_{\forall m \in \theta_2(\mathscr{M})}, \max((\mathbf{S}_{l,m})_{\forall m \in \theta_1(\mathscr{M})}) * \rho, 0) \tag{5.27}$$

$$\lambda_2^l(\mathscr{M}) = \max(\max(\mathbf{S}_{l,m})_{\forall m \in \theta_2(\mathscr{M})}, \pi_1^l(\mathscr{M}) * \rho) \tag{5.28}$$

$$\lambda_2^l(\mathscr{M}) = \max((\mathbf{S}_{l,m})_{\forall m \in \theta_2(\mathscr{M})}, \rho\,\pi_1^l(\mathscr{M})) = \pi_2^l(\mathscr{M}) \tag{5.29}$$

For $l \in \mathscr{L}$ and $t = n$ suppose it is true that $\lambda_n^l(\mathscr{M})) = \pi_n^l(\mathscr{M})$ and,

$$\begin{aligned}\lambda_n^l(\mathscr{M}) = \max(&\max(\mathbf{S}_{l,m})_{\forall m \in \theta_n(\mathscr{M})}, \\ &\max((\mathbf{S}_{l,m})_{\forall m \in \theta_{n-1}(\mathscr{M})}) * \rho, \ldots \max((\mathbf{S}_{l,m})_{\forall m \in \theta_1(\mathscr{M})}) * \rho^{n-1})\end{aligned} \tag{5.30}$$

For $l \in \mathscr{L}$ and $t = n+1$ we have,

$$\begin{aligned}\lambda_{n+1}^l(\mathscr{M}) = \max(&\max(\mathbf{S}_{l,m})_{\forall m \in \theta_{n+1}(\mathscr{M})}, \\ &\max((\mathbf{S}_{l,m})_{\forall m \in \theta_n(\mathscr{M})}) * \rho, \ldots \max((\mathbf{S}_{l,m})_{\forall m \in \theta_1(\mathscr{M})}) * \rho^{n})\end{aligned} \tag{5.31}$$

From (5.30),

$$\lambda_{n+1}^l(\mathscr{M}) = \max(\max(\mathbf{S}_{l,m})_{\forall m \in \theta_{n+1}(\mathscr{M})}, \rho * \lambda_n^l(\mathscr{M})) \tag{5.32}$$

Hence,

$$\lambda_{n+1}^l(\mathscr{M}) = \max((\mathbf{S}_{l,m})_{\forall m \in \theta_{n+1}(\mathscr{M})}, \rho\,\pi_n^l(\mathscr{M})) = \pi_{n+1}^l(\mathscr{M}) \tag{5.33}$$

RFL encapsulates the temporal correlation to calculate the gain for π_t^l by using π_{t-1}^l. The idea for incorporating the temporal sampling diversity is if the location is sampled at time t_1, then the subsequent neighboring time sampling would not be the best informative sampling as the data is smooth over time. In RFL, we incorporated the smoothness over time as shown in Fig. 5.2. Given that a location g is sampled at time $t = 1$, then the π_2 gain in the next timestamp is lower in FLS than RFL. Moreover, suppose we have two

adjacent locations a and b, that are neighboring nodes with high similarity. In that case, sampling the location b at timestamp t_1 will also provide information for the location a at subsequent neighboring time stamps as shown in Fig. 5.2. The greedy approach for the Facility location over space is shown in Algorithm 3.

Theorem 2: The Function defined in (5.34) is monotone submodular. Therefore a greedy heuristic provides a solution within an approximation ratio of $(1-\frac{1}{e})$.

Proof: Let θ_t be the set of locations corresponding to sampled locations in the sampling matrix Θ for time stamp t. For $\pi_0 = 0$, $\rho \in [0,1]$, the regressive facility location gain is defined by

$$RFL(\mathscr{M}) = \frac{\sum_{t\in\mathscr{T}}\sum_{l\in\mathscr{L}} \pi_t^l(\mathscr{M})}{L\times T} \tag{5.34}$$

where

$$\pi_t^l(\mathscr{M}) = \max((\mathbf{S}_{l,m})_{\forall m\in\theta_t(\mathscr{M})}, \rho\,\pi_{t-1}^l(\mathscr{M}))\ \forall l \in \mathscr{L} \tag{5.35}$$

Monotone non decreasing A function f is submodular if $\forall\ \mathscr{C} \subseteq \mathscr{D} \subseteq \mathscr{B}$

$$f(\mathscr{C}) \leq f(\mathscr{D})$$

Proof:

Let $\mathscr{C}$ and $\mathscr{D}$ be the subset of buses following $\forall\ \mathscr{C} \subseteq \mathscr{D} \subseteq \mathscr{B}$. We define $\Theta(\mathscr{C})$ and $\Theta(\mathscr{D})$ as the sampling matrix for the subset of buses $\mathscr{C}$ and $\mathscr{D}$.

Let θ_t be the set of locations that corresponds to 1 in the sampling matrix $\Theta(\mathscr{C})$ for the time stamp t and Let $\theta_t^{'}$ be the set of locations that corresponds to 1 in the sampling matrix $\Theta(\mathscr{D})$ for the time stamp t.

$$\theta_t = \{i\}\ \forall\ \Theta_{i,t}(\mathscr{C})|_{=1}$$

$$\theta_t^{'} = \{i\}\ \forall\ \Theta_{i,t}(\mathscr{D})|_{=1}$$

For all $l \in \mathscr{L}$ and $t = 1$ we have

$$\max_{\forall m \in \theta_1} (\mathbf{S}_{l,m}) \leq \max_{\forall m \in \theta_1'} (\mathbf{S}_{l,m}) \tag{5.36}$$

For $t = 2$ we have from using (5.11),

$$\max((\mathbf{S}_{l,m})_{\forall m \in \theta_2}, \rho \max_{\forall m \in \theta_1} (\mathbf{S}_{l,m})) \leq \max((\mathbf{S}_{l,m})_{\forall m \in \theta_2'}, \rho \max_{\forall m \in \theta_1'} (\mathbf{S}_{l,m})) \tag{5.37}$$

for time stamp t we have, from using (5.11)

$$\begin{aligned} &\max((\mathbf{S}_{l,m})_{\forall m \in \theta_t}, \rho \max_{\forall m \in \theta_{t-1}} (\mathbf{S}_{l,m}), \ldots, \rho^{t-1} \max_{\forall m \in \theta_1} (\mathbf{S}_{l,m})) \\ &\leq \max((\mathbf{S}_{l,m})_{\forall m \in \theta_t'}, \rho \max_{\forall m \in \theta_{t-1}'} (\mathbf{S}_{l,m}), \ldots, \rho^{t-1} \max_{\forall m \in \theta_1'} (\mathbf{S}_{l,m})) \end{aligned} \tag{5.38}$$

Hence $f(\mathscr{C}) \leq f(\mathscr{D})$

Submodular A function f is submodular if $\forall\ \mathscr{C} \subseteq \mathscr{D} \subseteq \mathscr{B}$ and $b \in \mathscr{B} \backslash \mathscr{D}$

$$f(\mathscr{C} \cup b) - f(\mathscr{C}) \geq f(\mathscr{D} \cup b) - f(\mathscr{D})$$

Consider a bus $b \in \mathscr{B} \backslash \mathscr{D}$, We add the bus b in both $\mathscr{C}$ and $\mathscr{D}$ and analyse the gain. Let the bus traverse the set of location G_t for a time stamp t.

$$G_t = \{i\}\ \forall\ \Theta_{i,t}(b)|_{=1}$$

For all $l \in \mathscr{L}$ and $t = 1$ to $\mathscr{T}$ we have

$$\begin{aligned}
&\pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) = \\
&\max((\mathbf{S}_{l,m})_{\forall m \in \theta_t \cup G_t}, \rho \max_{\forall m \in \theta_{t-1} \cup G_{t-1}}(\mathbf{S}_{l,m}), \ldots, \rho^{t-1} \max_{\forall m \in \theta_1 \cup G_1}(\mathbf{S}_{l,m})) - \\
&\max((\mathbf{S}_{l,m})_{\forall m \in \theta_t}, \rho \max_{\forall m \in \theta_{t-1}}(\mathbf{S}_{l,m}), \ldots, \rho^{t-1} \max_{\forall m \in \theta_1}(\mathbf{S}_{l,m}))
\end{aligned} \tag{5.39}$$

Let

$$K = \max((\mathbf{S}_{l,m})_{\forall m \in \theta_t}, \rho \max_{\forall m \in \theta_{t-1}}(\mathbf{S}_{l,m}), \ldots, \rho^{t-1} \max_{\forall m \in \theta_1}(\mathbf{S}_{l,m}))$$

$$\begin{aligned}
&\pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) = \\
&\max((\mathbf{S}_{l,m})_{\forall m \in \theta_t \cup G_t}, \rho \max_{\forall m \in \theta_{t-1} \cup G_{t-1}}(\mathbf{S}_{l,m}), \ldots, \rho^{t-1} \max_{\forall m \in \theta_1 \cup G_1}(\mathbf{S}_{l,m})) - K
\end{aligned} \tag{5.40}$$

$$\begin{aligned}
&\pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) = \\
&\max(0, (\mathbf{S}_{l,m})_{\forall m \in G_t} - K, \rho \max_{\forall m \in G_{t-1}}(\mathbf{S}_{l,m}) - K, \ldots, \rho^{t-1} \max_{\forall m \in G_1}(\mathbf{S}_{l,m}) - K)
\end{aligned} \tag{5.41}$$

let

$$K' = \max((\mathbf{S}_{l,m})_{\forall m \in \theta_t'}, \rho \max_{\forall m \in \theta_{t-1}'}(\mathbf{S}_{l,m}), \ldots, \rho^{t-1} \max_{\forall m \in \theta_1'}(\mathbf{S}_{l,m}))$$

From (5.38) since $K \leq K'$ Therefore,

$$\begin{aligned}
&\pi_t^l(\mathscr{C} \cup b) - \pi_t^l(\mathscr{C}) \geq \\
&\max(0, (\mathbf{S}_{l,m})_{\forall m \in G_t} - K', \rho \max_{\forall m \in G_{t-1}}(\mathbf{S}_{l,m}) - K', \ldots, \rho^{t-1} \max_{\forall m \in G_1}(\mathbf{S}_{l,m}) - K')
\end{aligned} \tag{5.42}$$

$$\pi_t^l(\mathscr{C}\cup b)-\pi_t^l(\mathscr{C})\geq$$
$$\max((\mathbf{S}_{l,m})_{\forall m\in\theta_t\cup G_t},\rho\max_{\forall m\in\theta_{t-1}\cup G_{t-1}}(\mathbf{S}_{l,m}),\ldots,\rho^{t-1}\max_{\forall m\in\theta_1\cup G_1}(\mathbf{S}_{l,m}))-K' \tag{5.43}$$

$$\begin{aligned}&\pi_t^l(\mathscr{C}\cup b)-\pi_t^l(\mathscr{C})\geq\\&\max((\mathbf{S}_{l,m})_{\forall m\in\theta_t'\cup G_t},\rho\max_{\forall m\in\theta_{t-1}'\cup G_{t-1}}(\mathbf{S}_{l,m}),\ldots,\rho^{t-1}\max_{\forall m\in\theta_1'\cup G_1}(\mathbf{S}_{l,m}))\\&-\max((\mathbf{S}_{l,m})_{\forall m\in\theta_t'},\rho\max_{\forall m\in\theta_{t-1}'}(\mathbf{S}_{l,m}),\ldots,\rho^{t-1}\max_{\forall m\in\theta_1'}(\mathbf{S}_{l,m}))\end{aligned} \tag{5.44}$$

$$\pi_t^l(\mathscr{C}\cup b)-\pi_t^l(\mathscr{C})\geq\pi_t^l(\mathscr{D}\cup b)-\pi_t^l(\mathscr{D}) \tag{5.45}$$

From Eqs (5.39-5.45) for all $l\in\mathscr{L}$ and $t=1$ to $\mathscr{T}$ we have,

$$f(\mathscr{C}\cup b)-f(\mathscr{C})\geq f(\mathscr{D}\cup b)-f(\mathscr{D})$$

Computational Complexity: For each $l\in\mathscr{L}$ and $t\in\mathscr{T}$ computing $\pi_t^l(\mathscr{M})$ via (5.8) requires Λ comparisons where $\Lambda<L$ hence the computational complexity for each iteration of the greedy algorithm for all the buses in FLS is $O(\Lambda LTB)$. For each $l\in\mathscr{L}$ and $t\in\mathscr{T}$ computing $\pi_t^l(\mathscr{M})$ via (5.22) requires ΛT comparisons, hence a cost of $O(\Lambda LT^2B)$ for FLST. We reduce the computation complexity of RFL by computing $\pi_t^l(\mathscr{M})$ via (5.35) to $O(\Lambda LTB)$.

Algorithm 8: Regressive Facility Location

Input: $m,\ \mathscr{B},\ \mathscr{Y},\ \rho,\ \mathbf{S}$
Output: $\mathscr{M} \subset \mathscr{B}$ of size m
Initialization:$\mathscr{M} \leftarrow \phi$; $\pi_0 = 0$
for i =1 to k **do**
  for each $e \in \mathscr{B}$ **do**
    $\Theta(\mathscr{M} \cup e) = \max(\sum_{m \in \mathscr{M} \cup e} \mathscr{Y}_{i,j,m}, 1)$
    for each $t \in \mathscr{T}$ **do**
      $\theta_t(\mathscr{M} \cup e) = \{i\}\ \forall\ \Theta_{i,t}(\mathscr{M} \cup e)|_{=1}$
      Compute $\pi_t^l(\mathscr{M} \cup e)$ using (5.35) $(\forall l \in \mathscr{L})$
    end for
    $f(\mathscr{M} \cup e) = \frac{1}{L \times T} \sum_{t=1}^{T} \sum_{l=1}^{L} \pi_t^l(\mathscr{M} \cup e)$
  end for
  $e^* = \arg\max_e f(\mathscr{M} \cup e)$
  $\mathscr{M} \leftarrow \mathscr{M} \cup \{e^*\}$
end for

5.5 Experimentation

5.5.1 Simulating Real World AQ data

Similarity matrix

To simulate the real-world AQ data on the set of location $\mathscr{L}$ and timestamps $\mathscr{T}$, we learn the similarity matrix based on the static real world of Delhi [98]. It contains the AQ data for 33 locations for 3 months. We define the entry of the similarity matrix $\mathbf{G}_{i,j} = \exp^{(-\lambda d_{i,j})}$ where $d_{i,j}$ is the distance between the two locations i and j. We learn the parameter λ using linear regression and it is observed to be 0.07676 for the AQ data [98]. We then compute the similarity matrix $\mathbf{G}$ for the set of location $\mathscr{L}$ based on the actual distance between locations $d_{i,j}$. We learn the temporal similarity matrix from the data [98] and use it as a temporal similarity matrix $\mathbf{H}$. The learned similarity matrix is shown in the Fig. 5.3

As we can see that the AQ data shows a high correlation in the neighboring time stamps. The motivation of the Regressive Facility Location is to model this correlation while selecting the buses to sample the locations in different time stamps.

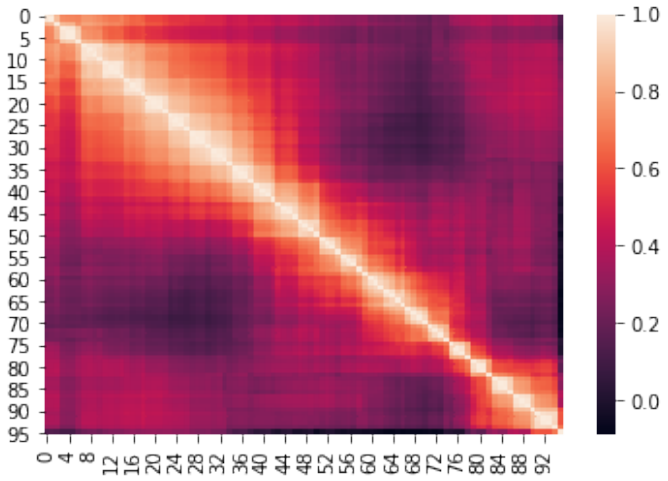


Fig. 5.3 Similarity matrix **H** across time

To generate a spatiotemporal matrix **Y** that varies smoothly over space and time, we use two variants described in the following sections.

Simulated Data 1

- We first factorize the actual matrix $\mathbf{Z} \in \mathbf{R}^{L \times T}$ as

$$\mathbf{Z} = \mathbf{A}\mathbf{B}^T$$

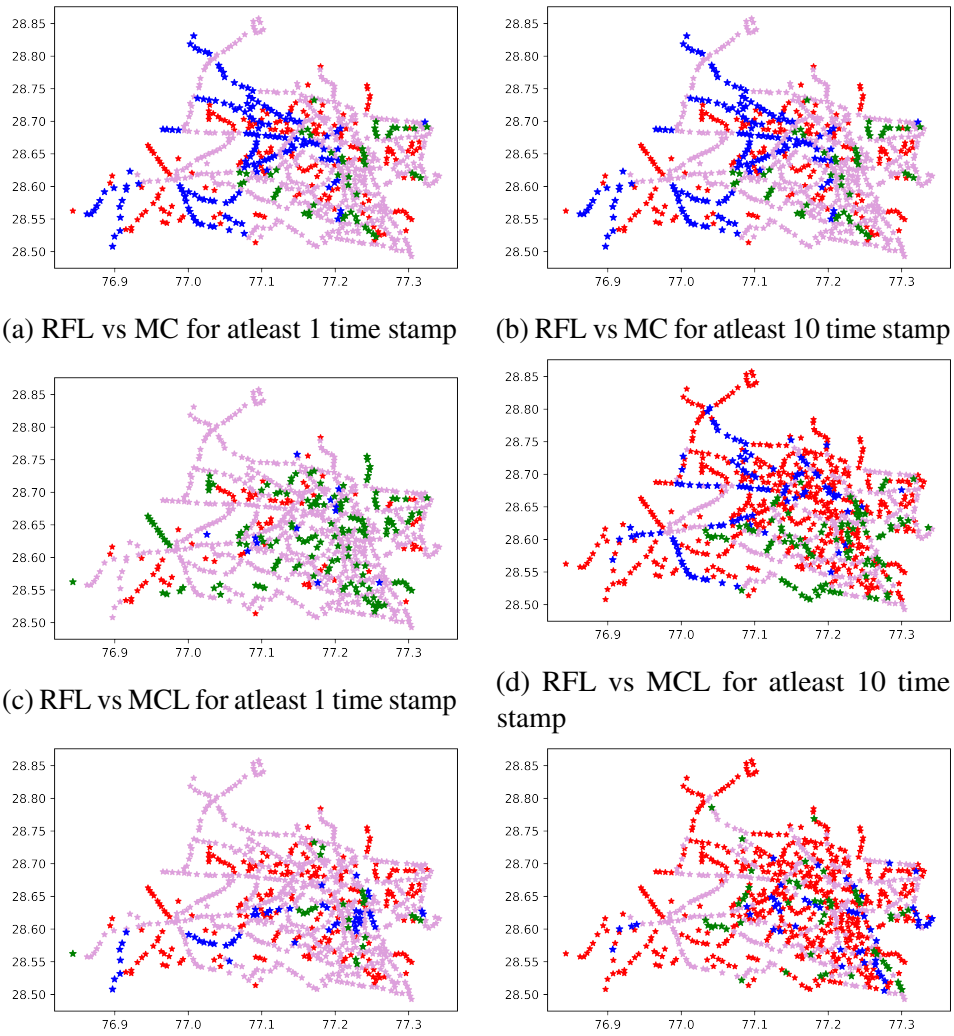
  where $\mathbf{A} \in \mathbf{R}^{L \times r}$ and $\mathbf{B} \in \mathbf{R}^{T \times r}$.

- We construct the matrix **A** for the locations using the **G** and a bandlimited signal. The similarity matrix **G** can be decomposed as

$$\mathbf{G} = \mathbf{U}\Sigma\mathbf{U}^T$$

  We generate the matrix **A** with rank r as defined in (5.2) as.

$$\mathbf{a}_i = \mathbf{U}_{(m)}\hat{\mathbf{a}}_i \ \ (\forall i = 1 \text{ to } r)$$



(a) RFL vs MC for atleast 1 time stamp

(b) RFL vs MC for atleast 10 time stamp

(c) RFL vs MCL for atleast 1 time stamp

(d) RFL vs MCL for atleast 10 time stamp

(e) RFL vs FLS for atleast 1 time stamp

(f) RFL vs FLS for atleast 10 time stamp

Fig. 5.4 Coverage plots for all the locations of Delhi for selected bus k=30. Red points denote the point not sampled by both RFL and comparison Framework, Pink points denote the points sampled by both RFL and comparison Framework, Blue points denote the points sampled by RFL but not the comparison framework, Green points denote the points sampled by comparison framework but not the RFL

where a_i represents the i^{th} column of matrix $\mathbf{A}$, a_i is sampled from a random normal distribution with standard deviation (0.5) .

- Similarly we construct the matrix $\mathbf{B}$ for time stamps using $\mathbf{H}$ and a bandlimited signal.

$$\mathbf{H} = \mathbf{V}\Lambda\mathbf{V}^T$$

$$\mathbf{b}_i = \mathbf{V}_{(n)}\hat{\mathbf{b}}_i \ \ (\forall i = 1 \text{ to } r)$$

- The overall data $\mathbf{Y}$ is generated using $\mathbf{A}$, $\mathbf{B}$ and noise signal n distributed as zero mean and std of 0.001.

$$\mathbf{Y} = \mathbf{A}\mathbf{B}^T + \mathbf{N}$$

We run the experiments for randomly generated 30 spatiotemporal matrix $\mathbf{Y}$, where m is randomly chosen from 5 to 15, n is chosen from 5 to 15, r is chosen randomly from 20 to 30.

Simulated Data 2

To generate a smooth signal over space and time we used the framework described in paper [99].

$$\mathbf{a}_t = \mathbf{U}_{(m)}\hat{\mathbf{a}}_t \tag{5.46}$$

The observed signal can be generated by the following Eqs:

$$\mathbf{y}_t = \mathbf{z}_t + \mathbf{n}_t \tag{5.47}$$

$$\mathbf{z}_t = \mathbf{R}\mathbf{z}_{t-1} + \mathbf{a}_t \tag{5.48}$$

The state transition matrix $\mathbf{R}$ is defined as a general diagonal matrix $\mathbf{R} = diag(c_1, c_2, \ldots c_T)$, where each c represents the autocorrelation coefficient that describes the time correlation of the data with the delayed (one time lag) data.We randomly generate 30 spatiotemporal matrix with c as 1, m is randomly chosen from 5 to 15 and r is chosen randomly from 20 to 30.

5.5.2 Creating Dense Maps from Sampled Data

A dense AQ map can be created by first selecting the $\mathscr{M}$ set of buses and then sampling the data $\hat{\mathbf{Y}}$ from the simulated ground truth AQ matrix $\mathbf{Y}$ based on the sampling matrix $\Theta(\mathscr{M})$. The final step is to impute the missing data in the matrix $\hat{\mathbf{Y}}$. To evaluate the performance, we compute the MRE score for the missing data imputation. There exists a problem of cold start while imputation, i.e., some of the locations are not sampled at all by the selected set of buses. Therefore, we use an extended version of the matrix imputation method that handles cold start cases. The matrix completion framework (VBMC) proposed in paper [94] imposes a low rank structure on the data to impute the missing data as:

$$\mathscr{L}_1 = \min_{\mathbf{A},\mathbf{B}} ||\mathbf{P}_\Omega(\mathbf{Y} - \mathbf{A}\mathbf{B}^T)||_F \tag{5.49}$$

where $\mathbf{A} \in \mathscr{R}^{L\times r}$ and $\mathbf{B} \in \mathscr{R}^{T\times r}$ and r =rank$(\mathbf{Y})$ $<<$ $min(L,T)$ Further, VBSF [78] adds a regularization on the matrix $\mathbf{B}$ to incorporate the temporal evolution in addition to the low rankness ((5.52)) as

$$\mathscr{R}(\mathbf{B}) = \sum_{i=1}^{T} ||\mathbf{b}_i - \mathbf{F}\mathbf{b}_{i-1}|| \tag{5.50}$$

However, these framework does not incorporate for the cold start locations. Incorporating the similarity matrix $\mathbf{G}$ along with the low-rank matrix completion framework can tackle the cold start problem [95, 96].

$$\mathscr{L}_2 = \min_{\mathbf{A},\mathscr{C}} ||(\mathbf{G} - \mathbf{A}\mathscr{C}^T)||_F \tag{5.51}$$

We use Variational Bayesian Matrix Completion (Cold start) and Variational Bayesian Subspace Filtering (Cold start) as the extended matrix completion frameworks that handle the cold start location data imputation to evaluate the performance of the dense AQ maps.

We impute the missing data using VBMC(CS) where we optimize the Eqs. (5.52, 5.54). We also impute the missing data using VBSF(CS) where we optimize the Eqs. (5.52, 5.53, 5.54). The matrix completion framework (VBMC) proposed in paper [94] imposes a low rank structure on the data to impute the missing data as:

$$\mathscr{L}_1 = \min_{\mathbf{A},\mathbf{B}} ||\mathbf{P}_\Omega(\mathbf{Y} - \mathbf{A}\mathbf{B}^T)||_F \tag{5.52}$$

where $\mathbf{A} \in \mathscr{R}^{L\times r}$ and $\mathbf{B} \in \mathscr{R}^{T\times r}$ and $r =$rank$(\mathbf{Y}) << min(L,T)$ Further, VBSF in chapter 2, add a regularization on the matrix $\mathbf{B}$ to incorporate the temporal evolution in addition to the low rankness ((5.52)) as

$$\mathscr{R}(\mathbf{B}) = \sum_{i=1}^{T} ||\mathbf{b}_i - \mathbf{F}\mathbf{b}_{i-1}|| \tag{5.53}$$

However, these framework does not incorporate for the cold start locations. Incorporating the similarity matrix $\mathbf{G}$ along with the low rank matrix completion framework can tackle the cold start problem [95, 96]. Completion of the unobserved entries in the matrix $\mathbf{Y}$ and transduction of knowledge from these entries to cold-start locations via similarity matrice $\mathbf{G}$ is carried out simultaneously using

$$\mathscr{L}_2 = \min_{\mathbf{A},\mathbf{C}} ||(\mathbf{G} - \mathbf{A}\mathbf{C}^T)||_F \tag{5.54}$$

Refer to the update equations in the Algorithm 1 in chapter 2. We showed the changed updated equations below: The update for i^{th} column of $\mathbf{A}$ for the cold start matrix completion is as follows:

$$\Xi_i^{\mathbf{A}} = \left(\hat{\gamma}_i \mathbf{I}_r + \hat{\beta} \sum_{\tau \in \Omega_i'} (\mu_\tau^{\mathbf{B}} (\mu_\tau^{\mathbf{B}})^T + \Xi_{\tau,\tau}^{\mathbf{B}}) + \hat{\beta}_1 (\mu^{\mathbf{C}} (\mu^{\mathbf{C}})^T + \Xi^{\mathbf{C}}) \right)^{-1} \tag{5.55}$$

$$\mu_i^{\mathbf{A}} = \Xi_i^{\mathbf{A}}(\hat{\beta} \sum_{\tau \in \Omega_i'} \mu_\tau^{\mathbf{B}} y_{i\tau} + \hat{\beta}_1 \mu^{\mathbf{C}} g_i) \tag{5.56}$$

The update for β_1 is as follows:

$$\hat{\beta}_1 = \frac{pL^2}{\| \mathbf{G} - \mathbf{A}\mathbf{C}^T \|_F^2} \tag{5.57}$$

The updates Eqs. for $\mathbf{C}$ is as follows:

$$\Gamma = diag(\gamma), \qquad \Xi^{\mathbf{C}} = (< \beta_1 >< \mathbf{A}^{\mathbf{T}}\mathbf{A} > +\Gamma)^{-1} \tag{5.58}$$

$$< \mathbf{C} > =< \beta_1 > \mathbf{G} < \mathbf{A} > \Xi^{\mathbf{C}}, \tag{5.59}$$

5.5.3 Dataset

We use the Delhi Bus GTFS data [9] to obtain the bus occupancy tensor defined by $\mathscr{Y} \in \{0,1\}^{L \times T \times B}$

- **Locations** (L): There are total 3210 bus stops. We sample L stops such that the minimum distance between the stops should be d. The total number of locations L is 824 when $d = 500$ m.

- **Time stamps** (T): We use 10 min sampling from 6 am to 10 pm, resulting in T=96 as the buses run during this time.

- **Buses** (B): The number of buses B is 1476.

The entries in $y_{l,t,b} = 1$ if bus b is in the 500 meter radius of location l for the time stamp t.

5.5.4 Evaluation Metrics

The evaluation metrics used for the performance comparison are defined as

- Percentage Stop coverage(PSC) is defined in Eq. (5.6).
- Percentage Coverage (PC) is defined in Eq. (5.5)
- FLS gain defined in (5.7)
- RFL gain defined in (5.34) for $\rho = 0.98$
- Mean Relative Error (MRE): $\frac{\|\mathbf{Y}-\hat{\mathbf{Y}}\|}{\|\mathbf{Y}\|}$, where $\hat{\mathbf{Y}}$ is the estimate of $\mathbf{Y}$.

Mean Relative Error is used to evaluate the performance of imputed dense AQ maps.

	$k=20$				$k=35$				$k=50$			
	PSC	PC	FLS	RFL	PSC	PC	FLS	RFL	PSC	PC	FLS	RFL
Random bus	56.796	4.612	90.202	91.936	59.102	6.593	90.379	91.563	64.078	10.584	92.515	93.579
Max Coverage	55.218	7.357	91.867	93.163	68.932	11.983	93.801	94.693	79.49	16.117	95.089	95.764
Max Cov Loc	80.704	5.141	90.719	92.593	92.233	7.756	92.734	94.143	96.359	10.293	93.935	94.941
FLS	60.316	6.279	93.164	94.159	73.908	10.685	94.924	95.673	75.728	14.482	95.804	96.363
RFL($\rho = 0.95$)	58.981	6.349	93.122	94.201	71.845	10.628	94.885	95.654	75.85	14.629	95.785	96.392
RFL($\rho = 0.98$)	60.68	6.504	93.16	94.324	72.816	10.761	94.875	95.724	79.369	14.759	95.754	96.414
RFL($\rho = 0.99$)	64.442	6.394	92.842	94.223	73.908	10.68	94.812	95.691	79.126	14.525	95.665	96.385
RFL($\rho = 1$)	71.845	4.949	91.019	92.795	86.529	8.628	93.375	94.614	93.204	12.141	94.472	95.443

Table 5.1 Performance Comparison for selecting k buses

5.5.5 Performance Comparison

The performance comparison of the Regressive Facility location with other baseline methods is shown in Table 5.1 and Fig 5.4. Performance on the imputing the complete dense map is shown in Table 5.2, 5.3.

Effect of ρ in RFL:

As the ρ increases, PSC increases. For $\rho = 1$, the performance is comparable to the MCL, since if a location is sampled for a timestamp, the gain corresponding to sampling other

	Simulated Data 1						
k	Random	MC	MCL	FLS	RFL95	RFL98	RFL1
20	70.595	39.333	55.582	31.71	31.395	28.507	55.577
35	47.062	22.29	27.938	10.69	11.065	10.664	21.209
50	31.6	14.193	16.62	7.465	7.367	6.983	13.997
75	23.546	11.275	11.296	5.312	5.07	5.065	10.832
100	18.844	10.375	8.65	4.289	4.271	4.112	7.517
	Simulated Data 2						
k	Random	MC	MCL	FLS	RFL95	RFL98	RFL1
20	70.239	40.13	40.225	27.68	28.03	25.85	43.70
35	40.26	19.906	16.293	8.395	9.433	8.563	13.75
50	26.674	11.699	10.822	5.228	5.62	4.251	8.126
75	19.46	8.448	7.197	2.736	2.627	2.532	7.263
100	14.732	8.035	5.785	1.973	1.817	1.79	4.487

Table 5.2 MRE for Dense Map using VBMC(CS)

	Simulated Data 1						
k	Random	MC	MCL	FLS	RFL95	RFL98	RFL1
20	90.788	37.538	73.922	36.585	36.749	33.602	76.477
35	44.446	21.008	26.995	10.934	11.609	10.124	19.386
50	27.766	13.303	15.125	7.018	7.492	6.574	11.678
75	20.34	10.068	10.15	4.528	4.586	4.52	8.842
100	16.33	9.52	7.187	3.809	3.769	3.684	6.385
	Simulated Data 2						
k	Random	MC	MCL	FLS	RFL95	RFL98	RFL1
20	77.619	35.275	53.419	30.424	31.479	28.864	57.799
35	39.429	20.8	19.972	9.817	11.193	9.297	14.858
50	28.65	13.526	11.627	5.801	6.398	4.904	8.027
75	21.873	9.704	7.38	2.581	2.591	2.581	5.178
100	16.922	9.201	5.243	1.76	1.638	1.524	3.547

Table 5.3 MRE for Dense Map using VBSF(CS)

timestamps will be zero; therefore MCL will select the buses that sample different locations to increase the Percentage stop coverage (PSC). However, the performance of RFL ($\rho = 1$) worsens for PC, FLS and RFL. For $\rho = 0.95$, the performance is comparable to the FLS as the effect of temporal correlation decreases with time in RFL for lower values of ρ. Since the FLS does not incorporate the temporal correlation in the framework and can sample the same set of representative locations across timestamps, we observe a reduced PSC for RFL ($\rho = 0.95$) and FLS. We observe similar performance for RFL($\rho = 0.98$)

and RFL($\rho = 0.99$). However, we observe an improved FLS score for RFL($\rho = 0.98$) throughout as compared to RFL($\rho = 0.99$). Therefore we evaluate the performance of RFL for $\rho = 0.98$ rather than $\rho = 0.99$ for dense map creation.

Coverage Plot:

To demonstrate the representations in space and time, we plot the coverage plot as shown in Fig 5.4. Every point on the plot is one of the bus stop locations. Fig 5.4 (a-c) denotes the locations that are sampled for atleast one timestamp, thereby illustrating the overall stop coverage. Fig 5.4 (d-f) represents the locations that are sampled for atleast 10 timestamps, thereby demonstrating the temporal coverage for the sampled stops. Pink points denote the locations sampled by both the RFL and the compared framework. Blue points denote the point sampled by the RFL but not the compared framework, green points denote the points sampled by the comparison framework but not the RFL and red points denote the points not sampled by RFL and the compared framework. From Fig. 5.4(a,d), it is observed that the MC does not sample a diverse set of locations. MCL provides improved spatial coverage, however the temporal coverage is worse as compared to RFL as shown in Fig. 5.4(b,e). It can also be observed that RFL samples a diverse set of locations than FLS as shown in Fig 5.4(c,f).

Dense AQ map MRE:

The MRE scores for creating dense AQ map are shown in Table 5.2, 5.3. For simplicity, we denote the RFL($\rho = 0.95$) as RFL95. It is observed that RFL($\rho = 0.98$) outperforms all the baseline algorithms for dense map creation VBMC(CS) and VBSF(CS). Since the generated data have temporal correlation thereby VBSF(CS) performance is better than VBMC(CS) for most of the cases, especially for higher bus sampling. However, for the

lower sampling of buses (20), it is observed that VBMC(CS) performs better as there are significantly lower samples to learn the temporal pattern along with the low rankness.

5.6 Summary

This chapter describes our proposed framework to sample spatiotemporal data using moving vehicles to create dense spatiotemporal maps. To obtain a dense air quality map, the cities may require only a small fraction of moving sensors as compared to all static sensors setups. The spatiotemporal correlation in the air quality data can be leveraged to select the moving sensor for sampling and to facilitate an effective spatiotemporal extrapolation on the sampled data.

This chapter proposes a Facility Location over space and Regressive Facility Location for drive-by sensing to select the set of buses that samples the spatiotemporal AQ data. We show that the proposed selective drive-by sensing algorithm gains are submodular; therefore, a greedy heuristic provides a solution within an approximation ratio of $(1-\frac{1}{e})$. It is shown that the chosen set of buses using the proposed framework Regressive Facility Location provides representative coverage across space and time. We further obtain the dense AQ maps using the matrix completion framework from the sampled spatiotemporal data and observe that Regressive Facility Location provides more accurate dense AQ maps.

Chapter 6

Spatio-Temporal Prediction

6.1 Overview

In this chapter, we motivate predicting future spatiotemporal data by exploiting the historic spatiotemporal patterns for the ETA prediction usecase. Accurate expected time of arrival (ETA) information is crucial in maintaining the quality of service of public transit. The quality of any public transport system is crucially dependent on its reliability. One of the primary metrics that measure the reliability of a public transit is the predictability of its vehicles in reaching specific pre-determined locations (bus stops in bus-based transit). A predictable and a reliable public transportation also attracts more users [100] thereby increasing the economic viability of the transit as well as reducing congestion in the road, a huge urban challenge, especially in the developing world. From the point of view of the transit operators, predictability is key to maintaining its efficiency. For instance, predicting bus-bunching in real time also helps in avoiding it. From the passengers perspective, predictability is generally measured by the accurate determination of the expected time of arrival (ETA) of a vehicle on a route. With the advent of smart phones and GPS enabled public transport, it is now easier to show the ETAs of various routes on the phone applications, especially of those routes which are not very frequent. Not just for the bus

based public transport routes, the ETA determination is necessary and vital even for the private on-demand transportation companies like Shuttl (India) and Chariot (US and India), as well as cab companies such as Lyft, Uber, and Ola.

The ETA estimation problem is as follows. Given the current location of the vehicle and its future trajectory, what is the expected time of its arrival in a given set of locations? In this chapter, we focus the ETA estimation problem only for bus based public transport routes, especially those that make frequent stops. One of the ways to estimate the ETA to a location is to predict the future traffic speed using the historic speed data on the given route. This approach, however, has several limitations. The first limitation is the lack of availability of the future traffic speed data across the city network. Large commercial entities like Google and Microsoft provide APIs that provide a prediction for future speed data in a city, but they may be both expensive and inaccurate. More importantly, these APIs are heavily biased towards cars. In reality, the speed profile for buses is often different than that of cars. The other significant limitation in using these speed profiles is the fact that a bus would often make many stops, which needs to be accounted in the calculations for ETA. In summary, the ETA calculation would be erroneous due to the unavailability of stopping time and inaccurate road speed information. In this chapter, we propose that one can build a simple yet effective system for predicting ETA for a given bus route by using only the past GPS trajectories of a given route using Mask-CNN.

The organisation of this chapter is as follows. We begin in section 6.2 with Mask-CNN, where we introduce the approach of a generative autoregressive models to predict the ETA. We then provide the literature review in section 6.3. We discussed the proposed framework in section 6.4. We provide details of the data and the experiments in section 6.5. Finally in section 6.6, we conclude the chapter.

6.2 Mask-CNN

In this chapter, we employ a generative autoregressive model to predict the ETA in a given trip conditional on the travel times between stops so far in the trip. The generative model is trained using the historic data for the previous trips on the same route. To this end, we form an ETA matrix, whose rows are the trips on a given day and whose columns contain the time taken to travel between consecutive bus stops. As an example, an ETA matrix corresponding to one day for a route operating T trips a day with K stops will be of the size $T \times K$. Our approach is inspired by pixel-CNN [101] which is used to learn the joint distribution of the underlying ETA. Here, the ETA matrix has a temporal (causal) nature and each row of the matrix can only be filled sequentially. We modify the pixel-CNN approach to suit the special structure in the ETA estimation task. We explored two different mask instead of using the traditional mask proposed in [101]. Masking also helps in controlling the level of dependencies from past used for the prediction of future values. We make the masking operation automatic thereby making the system plug and play. The generative model used in our work has been demonstrated [101] to explicitly model complex probability distributions that fits the training data and handles both noisy as well as the missing data case well. Through this approach, we integrate both stopping time as well as travel time in the training data itself. To the best of our knowledge, conditional generative models have not been explored for ETA prediction. The results indicate that we can make a simple and reliable ETA prediction mechanism by using only bus GPS data without relying on any other external speed data.

In this chapter, we propose a novel ETA prediction algorithm based on generative autoregressive model that integrates both traffic speed corresponding to the bus as well as the stopping time in one framework. The algorithm utilizes only the historic GPS data from the same route and can be independently implemented irrespective of availability of traffic data in the city. The key algorithm parameters can be tuned automatically thereby

increasing the ease of implementation in the real-world scenarios. Finally, we implement our algorithm on the real-world transit data available in Delhi, India and it outperforms other traffic prediction based ETA algorithms.

6.3 Related Work

Most ETA estimation works [102–106] in the literature are based on future speed prediction relying on the historic speed data from cars/cabs across the network. As noted before, the structure of the city network available to the car based traffic speed prediction models may not be valid for the bus routes which also need to allow for stopping times at the bus stops. For instance, authors in [106] employ an autoregressive moving average (ARIMA) model that relies on the fact the future link (road segment) speeds can be accurately predicted by a linear combination of past speeds from few other links. The variability of link speed profiles of buses from our data set indicate a much higher dimension. Moreover, many of the above methods rely on structured “big" data which simply may not be available for bus route networks. Further, some previous approaches based on individual road segment based travel time estimation assumes that the travel time on consecutive road segments as independent [102–105, 107–109], may not always be true as illustrated in recent works based on machine learning [110, 111].

ML techniques based on K nearest neighbor approach was used to predict the travel time in [112]. Authors in [113] suggest a Kalman filtering method. In [114], support vector regression was used to predict the travel time. Gradient boosting method was proposed in [115]. However, the above methods mainly incorporate temporal nature of the data, while the spatial dependencies were not explicitly modeled.

Recently, deep learning based methods have been proposed and have shown state of the art performance. Recurrent neural networks (RNN) in [116] and long short term memory (LSTM) in [117] have been proposed to predict the travel time. Spatio-Temporal Hidden

Markov Models (STHMM) are used to model correlations among different traffic time series in [110]. Recently, deep end to end travel time estimation (DeepTTE) [111] uses raw GPS data and also captures the local spatial dependencies like weather conditions, driving habits, start time, day of the week and an RNN is used to learn the temporal dependencies on the feature map generated by the geo convolutional layer. This model predicts the travel time for a complete path, but when generating the individual estimation of road segments, it does not incorporate the spatial correlation in the road segments. All the deep learning model need an extensive data set to train the non-linearity in the models. They also need considerable effort in turning parameters.

Closer to the problem taken in this work, authors in [118] and [119] use historical bus trajectories for predicting speed across future road segments. Authors in [120–122] uses the historical trajectories to predict the ETA based on the similarity in pattern using k-NN classifier, kalman filter, clustering and Auto-Regressive Integrated Moving Average (ARIMA) techniques. Authors in [123] model bus travel time as the sum of the median of historical bus travel times, random variations in travel time over time, and a model evolution error. Historical bus travel time is obtained using the k-NN algorithm, and the random variations in travel time are captured using particle filtering. All these model capture the temporal aspect of the ETA prediction but doesn't model the spatial correlation of the current trip. We compare our method with LSTM following a similar setting where based on the current trip and the past trip, future trip eta is predicted. This way of modelling doesn't require historical pattern search.

6.4 Proposed Method

6.4.1 Problem Formulation

We first discuss the problem formulation for the ETA estimation problem. To this end, we first introduce key notations. A bus route is defined as an ordered list of bus stops $(k = 1, ...K)$, where $k = 1$ is the source and $k = K$ is the destination stop respectively. Each route is undertaken by several trips in a day where each trip ideally should run according to a predetermined timetable. But, these trips may not adhere to the timetable due to various reasons. Let there be a total of T such trips in a route. We denote the travel time between stops $k-1$ to k by $t_{\tau,k}$ during the τ^{th} trip where, $(\tau = 1, \ldots, T)$. The ETA estimation problem tackled in this work is the following. Assuming the τ^{th} trip is in progress, and the bus is near the k^{th} stop, what is the ETA for the remaining stops on the route. Mathematically, we would like to predict $\hat{t}_{\tau,(k+\Delta)}$ where $\Delta = \{1, ...K-k\}$. We assume the availability of the historical travel time data for the bus route. Note that historical data could be both noisy as well as incomplete. As we shall see later, we use this historical data to train our prediction model. Further, we also assume travel times of the current trip $t_{\tau,(1:k-1)}$, and the previous trips *of the same day* $t_{(1:\tau-1),(1:k-1)}$ is also available. Mathematically, given the prediction model, we use $t_{\tau,(1:k-1)}$ and the previous trips i.e. $t_{(1:\tau-1),(1:k-1)}$ to predict $\hat{t}_{\tau,(k+\Delta)}$, for $\Delta = 1$ to $K-k$.

One way of estimating the ETA $t_{\tau,(1:k)}$ for the τ^{th} trip is to employ the maximum a posteriori (MAP) estimation method.

$$\hat{t}_{\tau,(k+\Delta)} = \underset{\hat{t}_{\tau,(k+\Delta)}}{\arg\max}\; p\left(\hat{t}_{\tau,(k+\Delta)} \,|\, t_{\tau,(1:k-1)}, t_{(1:\tau-1),(1:k-1)}\right). \tag{6.1}$$

where, $p(.)$ denote the conditional probability density function (pdf) of the travel times from stop k to $k+\Delta$, $\Delta = 1$ to $K-k$, at the τ^{th} trip conditional on previous travel times

of the current trip and the previous trips on the same day. The above method is also optimal assuming the above mentioned conditional pdfs are known and the MAP estimator can be evaluated. However, estimation of such vast number of pdfs for each route is intractable and the MAP computation becomes impractical. To alleviate the above issues, we employ what is known as the generative modeling approach to learn the conditional distributions required in (6.1) in a single and unified framework thereby making ETA estimation sufficiently accurate while enabling low complexity computations that require less data. These learned distributions is then sampled to estimate the ETA values for the future stops in a trip.

6.4.2 Generative Modeling for traffic prediction

Generative modeling is a powerful way of estimating the distribution of the data in an unsupervised way. Last few years have witnessed tremendous research efforts towards generative modeling of the data [101, 124, 125]. Generative models generally employ deep learning frameworks to learn an approximation of the true distribution of the data. In this work, we utilise an autoregressive generative model to explicitly learn the distribution of the ETA data.

Our model is inspired by pixel-CNN [101], where the authors learn a distribution of images. An image is nothing but a matrix of pixel values. The distribution of natural images is thus the joint distribution of all the pixels of a matrix. Formally, let $\mathbf{X} = (x_1, x_2, x_3, ..., x_{n^2})$ be a matrix shown in Fig. 6.1. Using the chain-rule, the joint distribution of the matrix X is

$$p(\mathbf{X}) = \prod_{i=1}^{n^2} p\left(x_i \,|\, x_1, ..., x_{i-1}\right) \tag{6.2}$$

In words, if $\mathbf{X}$ is an image, the very first pixel x_1 is independent, albeit with a distribution, the second pixel x_2 dependent on first, third depends on the first and second and so on. In

x_1	x_2			x_n
x_{n+1}				x_{2n}
		x_i		
				x_n^2

Fig. 6.1 Matrix X

summary, the matrix is ordered as a sequence of points where the probability distribution of one point depends on the observed values of the previous points. The generation proceeds row by row and pixel by pixel. Similarly, we can determine the probability of pixel x_i conditioned on $x_{i-1}...x_1$.

Likewise, the travel times of a bus route can be seen as an image of size $T \times K$ with rows as trips and the columns denote the travel times between consecutive stops. In a way, one day of a bus route travel time matrix can be seen as a single image. Consequently, we can learn the distribution of these ETA matrices using generative models like in [101] by suitably modifying to suit the specifics of the ETA estimation problem. To learn the generative model, we use mask-convolutional neural networks (CNN) based autoregressive model. CNN based models are well known and widely studied for capturing local correlation in images for classification tasks [126].

The architecture of mask-CNN is fundamentally simple with two blocks in it — the training and the real-time ETA estimation blocks. The mask-CNN is utilised to learn the generative model in the training block using the historical route travel time data. Once the model is trained, we use it to compute ETA for a given route. The observed ETA values after the completion of the trip is fed to the training block which simultaneously adapts the

model that is used for the future trips. We first briefly describe CNN, followed by detailed discussion of the usage of these models for the ETA estimation problem. Before discussing the mask-CNN framework in detail, we provide a simple example to discuss inference based on generative models.

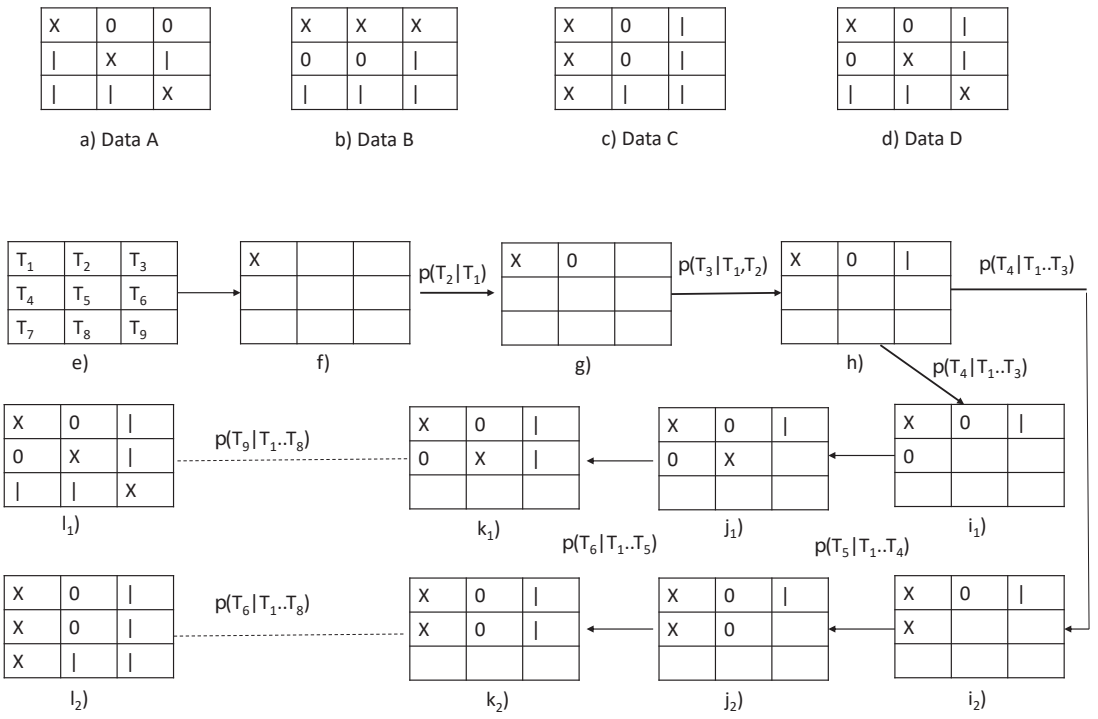


Fig. 6.2 Example dataset with 4 elements and inferencing the dataset

6.4.3 Generative Modeling: an example

Consider a data set with 4 elements as shown in Figs. (6.2a - 6.2d). Each element of this data set is a 3×3 matrix whose entries are ordered from T_1 to T_9, where each of the element of the matrix comes from the set $S = \{0, \times, |\}$. Assume that we can estimate the joint probability distribution $p(T_1, T_2, \ldots, T_9)$ for the data in Fig. 6.2(a-d). The inference after training is shown in Figs. 6.2(e- l). Suppose we observe (Fig. 6.2f) $T_1 = \times$. We can

now predict the values from T_2, T_3 to T_9 onwards conditional on $T_1 = \times$ using the trained generative model. Note that there are two paths to take once we observe $T_1 = \times$, with different conditional probabilities. It can be easily inferred from the data (Figs. (6.2a - 6.2d)), that probability that T_2 will be 0 is higher as compared from $\times$. Further on observing T_2 (which may not be same as what was predicted before), we update the probabilities for T_3 to T_9 and the sequential prediction process continues as we observe more variable. The next question therefore is, how to model the joint and the conditional distributions?

6.4.4 Convolutional Neural Networks (CNN) for ETA

Convolutional neural networks (CNN) is a class of deep learning models that has provided state of the art performance in various image/video classification tasks. CNN captures the local spatial coherence by “convolving" a local 2-D area with filters and thereby absorbing the spatial dependencies in an image. Intuitively, filters perform the task of feature extraction from the matrix (or an image). Many filters can be passed through these matrices, each picking a different set of features. For example, a horizontal line or a vertical line filter. Convolutional networks employ these filters, slices of the matrix feature space, and map them one by one. In other words, they create a map of each place where these features occur. A general CNN architecture (Figs. 6.3) has many layers. Following convolution operation through filters, the resulting matrix is passed through layers containing nonlinear transforms like *tanh* or a rectified linear unit (*ReLU*) that is generally applied to each element of the matrix.

Similar to an image, one of the major motivation for using a CNN for the ETA estimation problem is the natural spatial and temporal correlation available in the ETA data of bus trips for a given route. However, using the "regular" convolution filter may imply that we end up using points for the convolution operation that may not have been generated yet. This implies that we may break the causality of the system. For instance, to

predict the ETA between stops $j-1$ to j, we cannot use the data for stops j to $j+1$, as that trip has not happened yet. This challenge can be overcome by using an appropriate mask along with convolution operation that maintains the causality of the system. Also, masking gives the flexibility to restrict the dependencies. One of the contributions of our work is to automate the selection of an appropriate filter that decides the optimal dependencies for the ETA estimation task. The dependencies that are captured using the CNN are nothing but the conditional probability distribution function that we seek to obtain. We now discuss the proposed mask-CNN model and the ETA estimation problem in more detail.

6.4.5 Mask-CNN architecture

The mask-CNN architecture is a fully convolutional network of seven layers that preserves the spatial resolution of its input throughout the layers and outputs a conditional distribution at each location. We first define the input provided for the training of the architecture. The time taken to travel between any two consecutive stops in the evening may not be dependant on the time taken in the morning. Therefore, we can divide everyday trip data into smaller overlapping chunks of window size H.

Let K denote the total number of stops in a route. Similar to an image, we define the collected bus ETA data as a 2-D matrix of dimension $H \times K$, one for each day, whose rows are the trips on a route in a day and the columns contain the travel time between two consecutive stops. In the case of an image, a pixel generally takes value in the range of 0 to 255. The traffic ETA matrix can be seen as an image with ETA values ranging from 0 to C (seconds). The value of C is decided based on the maximum possible value of ETA and quantization levels l.

The architecture of mask-CNN is shown in Fig. 6.3 where the input to the model is $H \times K$ ETA matrix and the corresponding output is a $H \times K \times C$ tensor. Here H is the window size for the number of routes and K is the number of bus stops. Applying a

softmax layer on the above tensor generates an output tensor $H \times K \times C$ corresponding to the probabilities of each pixel taking C values. Finally, the value with maximum probability is chosen. The first layer is a mask A CNN layer with filter dimension $F \times F$ with a total of N filters, padding as p and stride as 1. See Fig. 6.4. Next $k-1$ layers after the first layer is Mask B layer with filter dimension $L \times L$ with n number of filters, padding as p_1 and stride as 1. ReLU activation function is used after every convolution layer. The last convolution layer FC is a fully connected layer with filter size 1. The number of filters in the fully connected layer FC is equal to C. The end layer in the mask-CNN layer is the softmax layer which assigns the probability to all the discrete variables C and output is the discrete variable with the highest probability.

The overall architecture of the masked CNN for traffic state prediction is as follows

1. First layer is the Mask A CNN layer with filter dimension $F \times F$.
2. There are $k-1$ Mask B CNN layers with filter dimension $L \times L$.
3. *ReLU* activation is followed by every convolution layer.
4. At the output stage there is a fully connected convolution layer followed by a C-way softmax layer.

Masking

There are two masks used in the masked CNN, mask A and mask B. Mask A is the first layer in the mask CNN and shows the effect of already predicted ETA points on the point that we are about to predict sequentially. Mask B is used in rest of the layers. For mask B the connection with the about to be predicted pixel point is also included. Mask A and B for 5×5 filter are shown in Fig. 6.4 (a, b) where 0 denotes that the future dependencies are removed from the prediction. In mask A and mask B, the entries $M_{i,j}$ can be 0 or 1 based

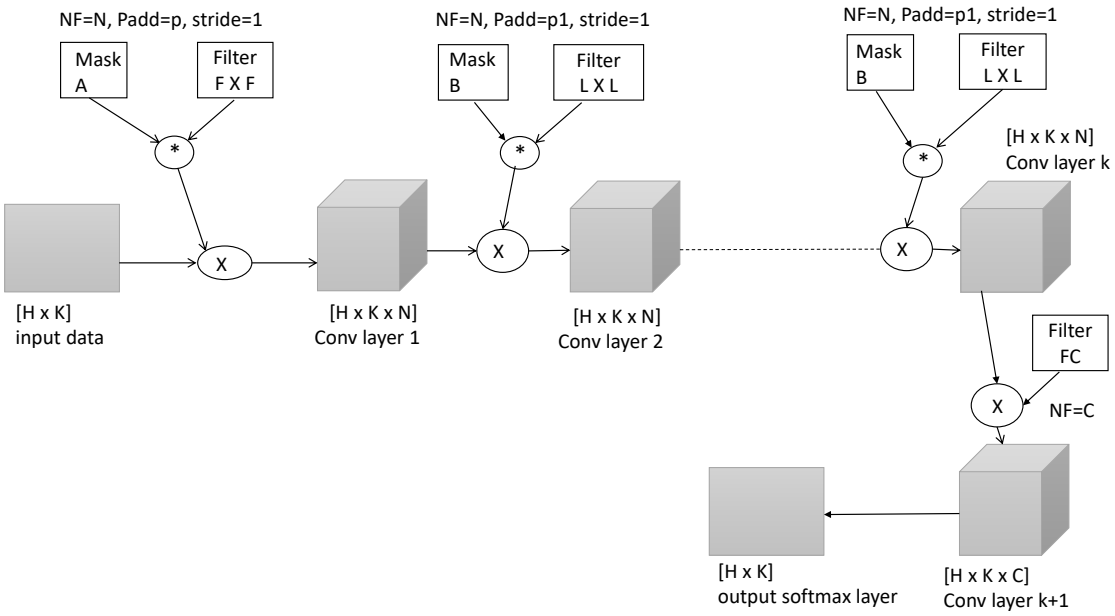


Fig. 6.3 Architecture of mask-CNN

on how we want to model the past dependencies for the next prediction. We employ three Masks (Mask 1, Mask 2 and Mask 3) for ETA prediction with different $M_{i,j}$.

6.4.6 ETA prediction using the trained model

Once the model is trained using the historic travel time data, we are now ready to provide ETA estimation for every trip in the route. Fig. 6.5 explains the inference process with a simple example of a route with 4 stops for the t^{th} trip. At the beginning of the trip, the ETAs for various stops is generated using sampling from the joint distribution that is trained using historic data. However, as soon the bus crosses the first stop, the subsequent sequential generation of ETAs would take into account the actual travel time $t_{t,1}$ to the first step. Similarly, the ETAs are updated when the trip crosses second and the third stops. We predict the ETA as:

$$\hat{t}_{tr,(k+1)} \leftarrow \textbf{Model}\left(t_{tr,(1:k)}, t_{(1:tr-1),(1:K)}\right) \tag{6.3}$$

M_{11}	M_{12}	M_{13}	M_{14}	M_{15}
M_{21}	M_{22}	M_{23}	M_{24}	M_{25}
M_{31}	M_{32}	0	0	0
0	0	0	0	0
0	0	0	0	0

a) Mask A

M_{11}	M_{12}	M_{13}	M_{14}	M_{15}
M_{21}	M_{22}	M_{23}	M_{24}	M_{25}
M_{31}	M_{32}	1	0	0
0	0	0	0	0
0	0	0	0	0

b) Mask B

1	1	1	1	1
1	1	1	1	1
1	1	0/1	0	0
0	0	0	0	0
0	0	0	0	0

c) Mask 1

0	0	0	0	0
0	0	1	0	0
0	1	0/1	0	0
0	0	0	0	0
0	0	0	0	0

d) Mask 2

0	0	1	0	0
0	0	1	0	0
1	1	0/1	0	0
0	0	0	0	0
0	0	0	0	0

e) Mask 3

Fig. 6.4 Different masks used in mask-CNN

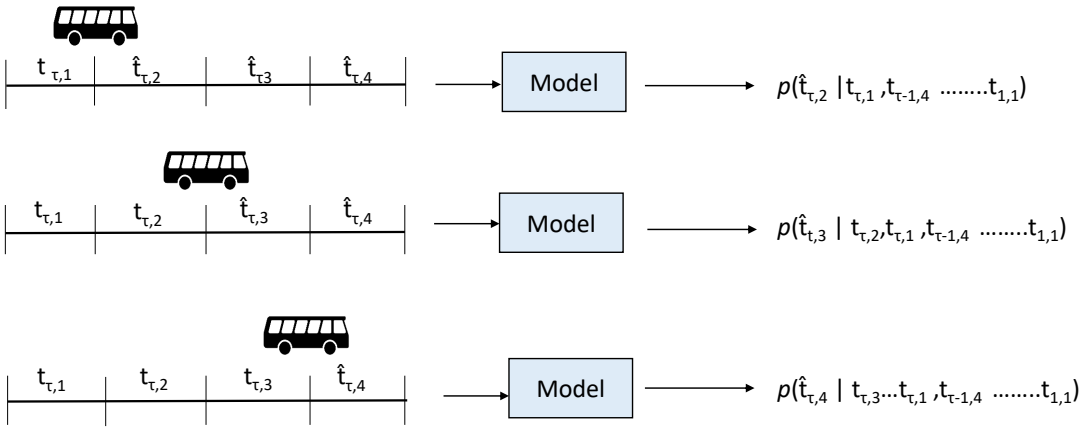


Fig. 6.5 Inferencing the Travel Time

6.5 Results

We now discuss the performance of the proposed mask-CNN algorithm for the ETA estimation task for a bus route network. We compare our technique with the state-of-the-art

approaches like time series prediction, deep learning, as well as the matrix completion approaches below:

1. ARIMA (Autoregressive Integrated Moving Average) [106].

2. LSTM (Long Short Term Memory) [117] is one of the recent methods to compute the ETA in public transit as well as cabs. We used the architecture shown in fig 6.6.

3. VBSF (Variational Bayesian Subspace Filtering) [127]: Online matrix completion frameworks are not only employed to fill the missing entries of a matrix but also for prediction of the future columns. VBSF is one of the matrix completion algorithm that is used for traffic estimation and prediction and is shown to outperform other similar techniques.

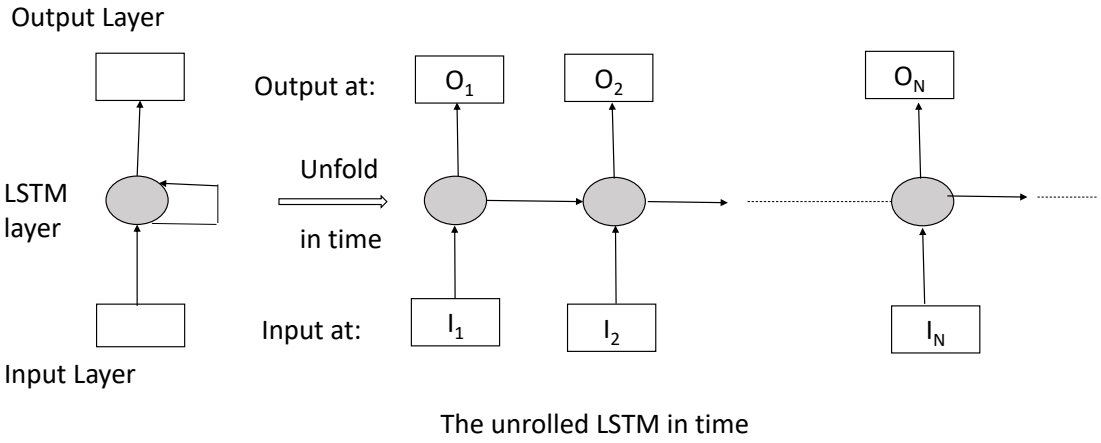


Fig. 6.6 The LSTM architecture is unrolled along time to describe a complete trip

6.5.1 Dataset

Our dataset consists of travel time information of three bus routes in New Delhi as drawn in Fig. 6.7. The lengths of these routes are approximately 30 km, 22 km and 20 km. Each route operates around 40 trips per day with nearly fixed starting timetable. The bus route

is made of a sequence of stops, and we collect the arrival time and the departure time for these stops. We divide everyday day into smaller overlapping chunks of $h = 10$ trips. Of the three months of collected data, we use two months of data for training and the third month data is used for evaluation of all the algorithms.

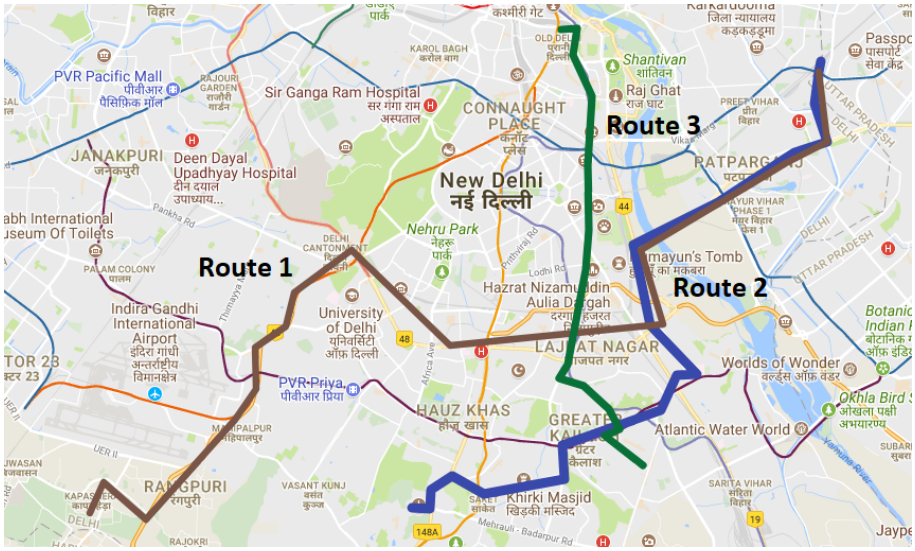


Fig. 6.7 Routes used for data collection

6.5.2 Training Parameters

We employ a variety of masks based on the dependencies we want to capture in the dataset. We use three different kinds of the mask in our evaluation (mask A1 and B1 for mask 1, mask A2 and B2 for mask 2, mask A3 and B3 for mask 3).The masks 1, 2 and 3 for filter dimension 5 is shown in Fig. 6.4 (c, d, e) respectively, where the middle element in Fig. 6.4 (c, d, e) is 0 for mask A and 1 for mask B.

To train our model, we use a batch size of 32 with a learning rate of 0.01 along with RMSprop optimizer [128]. We test three filter dimension values of 7, 5 and 3 for both F and L in mask A and B. Also, the number of blocks $k-1$ for mask B is set to 6. The number of classes for softmax layer C is taken as 128, 256 and 512. Stride in the CNN is taken as 1, while zero padding (p, $p1$) for filter size 7 is taken as 3, for filter size 5 is

taken as 2, and for filter size 3 is taken as 1. We first remove the outliers from the training data and fix the maximum ETA for a stop as 1024 for our data. The number of classes $C = 1024/l$ is decided based on the travel time data where l is the quantization level . In our case, we fix the value of C using the grid search as shown in Table 6.1.

6.5.3 ETA Estimation

We use the standard mean absolute percentage error (MAPE), root mean squared error (RMSE) and mean absolute error (MAE) as our performance metrics defined as follows:

$$\text{MAE} = \frac{1}{TK}\sum_{k=1}^{K}\sum_{\tau=1}^{T}|\hat{t}_{\tau,k} - t_{\tau,k}|$$

$$\text{MAPE} = \frac{1}{TK}\sum_{k=1}^{K}\sum_{\tau=1}^{T}\frac{|\hat{t}_{\tau,k} - t_{\tau,k}|}{|t_{\tau,k}|} \times 100\%$$

$$\text{RMSE} = \frac{1}{T}\sum_{\tau=1}^{T}\sqrt{\frac{\sum_{k=1}^{K}(\hat{t}_{\tau,k} - t_{\tau,k})^2}{K}}$$

The comparative performance of different filters, masks, and number of classes are shown in Table 6.1. Based on these results, we tune the filter dimension as 5, mask as 2 and softmax classes as 512.. Mask 2 and filter dimension of 5 performs better than the other mask and filter because there are lesser dependencies of the road segments far away from the predicted road segment. Note all these parameters can easily be auto-tuned to make the system manual-tuning free.

To evaluate the performance of the mask-CNN approach, we compare it with Mean, ARIMA, LSTM and VBSF, since these methods outpeform other methods in their class. Comparing our approach with other deep learning approaches is not possible due to a relatively smaller data set that we deal with. Further as mentioned before, we do not possess other parameters like driver id, weather information, etc. that methods like [111, 129] use.

	MAPE
Filter 3 ,mask 1 , classes-256	0.2927
Filter 3 ,mask 2 , classes-256	0.2815
Filter 3 ,mask 2 , classes-512	0.2619
Filter 5 ,mask 1, classes-512	0.27114
Filter 5 ,mask 2 , classes-512	0.23991
Filter 5 ,mask 3 , classes-512	0.24208
Filter 7 ,mask 2 , classes-512	0.27078

Table 6.1 Performance comparison for different filter size and Quantization classes

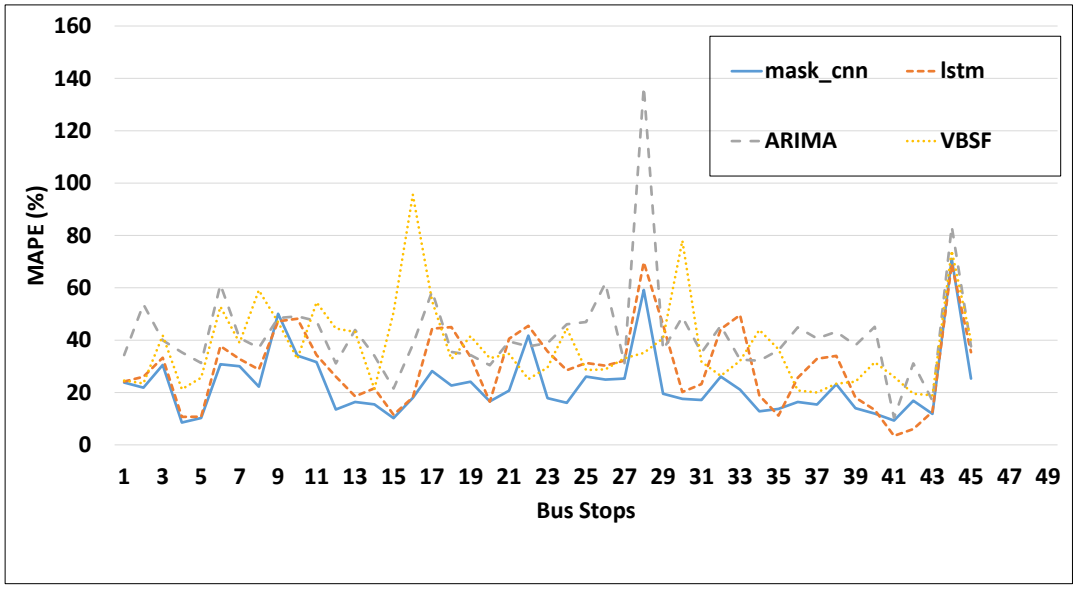


Fig. 6.8 Comparison of Masked CNN for a bus route

Finally, none of the methods use the online prediction mechanism as the trip progresses. Table 6.2 and Fig. 6.8 provide the required comparison. Fig. 6.8 shows the variation in performance concerning all the stops in the route for the first route. The other two routes behave similarly. LSTM is popular method used for the time series data. We show that mask-CNN performs better than LSTM for the bus ETA problem. For eg. in case of route 1, the average ETA for the stops are 150 secs. There are 7 stops with standard deviation higher than 100 secs and 16 stops with standard deviation less than 50 secs. The prediction for the stops with high variation contribute to high error. On an average the error in the

	Route 1			Route 2		
	MAPE (%)	MAE (sec)	RMSE (sec)	MAPE (%)	MAE (sec)	RMSE (sec)
Mean	42.78	51.42	66.21	72.48	88.45	98.48
ARIMA	48.64	53.77	69.42	68.42	73.84	84.18
VBSF	37.02	50.62	65.27	74.96	99.35	121.26
LSTM	29.587	38.59	56.65	52.82	65.76	79.97
mask-CNN	23.991	30.40	45.84	46.24	62.15	76.60

	Route 3		
	MAPE (%)	MAE (sec)	RMSE (sec)
Mean	45.67	48.55	59.69
ARIMA	47.44	49.37	60.75
VBSF	45.2	48.31	59.05
LSTM	41.09	43.25	52.15
mask-CNN	36.05	39.42	49.89

Table 6.2 Performance Comparison

ETA prediction for LSTM is 38 secs while for mask-CNN is 30 sec. We demonstrate that our algorithm performs better than the ARIMA, LSTM and VBSF in most of the cases.

The advantage of the mask-CNN model is that it is not designed empirically for different times. Our model does not require any information regarding the time, day, driver profiles and other complex features required by other models [111, 129] to model the ETA prediction. The only information to train the model is the travel time data between stops. Mask-CNN captures the dependency in the data temporally as well as spatial by representing the ETA as a image and modeling the same with different masks.

6.6 Summary

In this chapter, we investigated a deep learning based generative model to estimate the ETA of a bus trip in real time. We train a model for each individual route using historical data of trips collected over two months. We observed that we could learn a reasonably

accurate joint distribution of the ETA variables across day and bus stops. Our model is easy to implement for transit agencies, adaptive and utilises the real-time information of the trip as well. It has a great potential to be used in places where other dense traffic data set is not available.

Chapter 7

Conclusion and Future Directions

In conclusion, this thesis provides a sustainable solution for effective sensing using public transit and can deliver reliable real-time and future data to the user. The solutions proposed in this thesis can be utilized to provide a spatiotemporal data map of an area, where a user can avail the data for any location and time period. We proposed a spatiotemporal sampling framework to collect data using buses and develop models to estimate and predict the spatiotemporal data for enhanced user experience. In this thesis, we explored three problems towards providing an effective strategy for sensing and providing spatiotemporal data, i.e., spatiotemporal estimation, spatiotemporal sampling and spatiotemporal prediction.

In chapter 2, Variational Bayesian Subspace Filtering a bayesian model is proposed that fit sequential multivariate measurements arising from a low-dimensional time varying subspace for spatiotemporal estimation. In chapter 3, a Robust Variational Bayesian Subspace Filtering is proposed to predict the missing data in the presence of outliers. We also predict the location and value of the outliers in the data. In chapter 4, we proposed a Variational bayesian approach for extreme matrix completion for a high percentage of missing data. We exploit the subspace information of previous days to reduce the sample complexity and improve the estimation performance for extreme matrix completion.

In chapter 5, we proposed regressive facility location based drive by sensing, an efficient vehicle selection framework for effective spatiotemporal sampling. Regressive facility location based drive-by sensing, is an efficient vehicle selection framework that incorporates the smoothness in neighbouring locations and autoregressive time correlation while selecting the optimal set of vehicles for effective spatiotemporal sampling. In our work, we investigate the usage of public transit buses for drive-by-sensing. We pick those routes using the proposed regressive facility location that exploits the spatiotemporal structure in the air quality data resulting in an effective imputation and therefore achieving the required dense air quality map. We illustrate that the proposed method samples the representative spatiotemporal data in turn reducing the extrapolation error. Our approach, therefore, has the potential to provide cost-effective dense air quality maps.

In Chapter 6, we proposed a deep learning based generative model that learns the probability distribution of spatiotemporal ETA data across trips and conditional on the current trip information predicts the ETA information on the go. We present a simple but robust model for spatiotemporal ETA prediction for a bus route that only relies on the historical data of the particular route. We propose a system that generates ETA information for a trip and updates it as the trip progresses based on the real-time information. Our plug and play model not only captures the non-linearity of the task well but that any transit agency can use without needing any other external data source.

7.1 Future Research Directions

In this section, we presented approaches and the possible research directions that can be considered for further work. Below are some of these potential opportunities in detail.

- Calibrated Drive-By sensing: In chapter 5, we discussed the problem of efficient drive-by sensing, where we proposed to pick the best set of buses for spatio-temporal

sampling. Installing low-cost sensors on buses for drive-by sensing is a cost-effective solution for air quality monitoring. However, these sensors need to be calibrated from time to time. Removing the sensors and installing them again is a cumbersome task. Also, how often these sensors should be calibrated is not known. Calibrated Drive by Sensing can be a potential solution where the low-cost sensor can be calibrated on the go. Delhi has around 30 High-Cost Monitors installed at different locations. Low-cost sensors can be calibrated with these high-quality Air monitoring stations. However, limited routes buses can be 1 hop calibrated with these High-Cost Monitors. A bus b is said to be k hop calibrated when there exist a link between the $k-1$ hop calibrated bus and the bus b. Selecting the best set of buses that can be k hop calibrated is an important problem to tackle in terms of drive-by sensing.

- Robust Drive-by sensing: Drive-by sensing proposed in chapter 5, consider the bus occupancy data for a day and assume that the buses will follow the same pattern for the coming days. To incorporate the variations for different days, training or building an algorithm on a few days and testing a robust version's performance for the coming days are required.

- High-Rank Matrix Completion: We proposed Variational Bayesian Subspace Filtering, a low-rank subspace filtering approach for missing data estimation and outlier removal in chapter 2,3. If the data defining the matrix belongs to a structure having fewer degrees of freedom than the entire dataset, that structure provides redundancy that can be leveraged to complete the matrix. This typical low-rank subspace assumption is not always satisfied. The original matrix can possibly be high rank, but it becomes low rank after mapping each column to a higher dimensional space of monomial features [130, 131]. We can further explore this direction to improve the performance of our Spatiotemporal Estimation.

- Scalable ETA prediction: In chapter 6, we discuss the ETA prediction algorithm for a single route in Delhi, India. However, there are around 543 routes in Delhi. Training a Deep Auto regressive model and deploying it individually for 543 routes is not a scalable option. Therefore, our future work will focus on predicting the future ETA for all the routes at the same time using a Graph convolution-based framework. To train the ETA framework to a large network, we will employ graph sampling during training. Also, the ETA prediction algorithm uses Mean square error as the cost function. However, in a practical scenario, user would prefer waiting a bit longer than missing the bus because of underestimation of ETA. We will focus on these practical issues in our subsequent ETA prediction work.

References

[1] Yu Zheng, Furui Liu, and Hsun-Ping Hsieh. U-air: When urban air quality inference meets big data. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1436–1444. ACM, 2013.

[2] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(3):38, 2014.

[3] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2267–2276. ACM, 2015.

[4] IISD. The road to sustainable transport, 2021.

[5] Global Sustainable Transport Conference. Mobilizing sustainable transport for development, 2016.

[6] Waldo R Tobler. A computer movie simulating urban growth in the detroit region. *Economic geography*, 46(sup1):234–240, 1970.

[7] Amin Anjomshoaa, Fábio Duarte, Daniël Rennings, Thomas J Matarazzo, Priyanka deSouza, and Carlo Ratti. City scanner: Building and scheduling a mobile sensing platform for smart city services. *IEEE Internet of Things Journal*, 5(6):4567–4579, 2018.

[8] David Hasenfratz, Olga Saukh, Christoph Walser, Christoph Hueglin, Martin Fierz, Tabita Arn, Jan Beutel, and Lothar Thiele. Deriving high-resolution urban air pollution maps using mobile sensor nodes. *Pervasive and Mobile Computing*, 16:268–285, 2015.

[9] OTD-Delhi. Open transit data - delhi, 2020.

[10] Simone Mora, Amin Anjomshoaa, Tom Benson, Fábio Duarte, and Carlo Ratti. Towards large-scale drive-by sensing with multi-purpose city scanner nodes. In *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, pages 743–748. IEEE, 2019.

[11] Amin Anjomshoaa, Fábio Duarte, Daniël Rennings, Thomas J Matarazzo, Priyanka deSouza, and Carlo Ratti. City scanner: Building and scheduling a mobile sensing platform for smart city services. *IEEE Internet of Things Journal*, 5(6):4567–4579, 2018.

[12] Charul, U. Bhatt, P. Biyani, and K. Rajawat. Online variational bayesian subspace filtering. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5057–5061, 2019.

[13] Xinyu Chen, Yixian Chen, and Zhaocheng He. Urban traffic speed dataset of guangzhou, china, March 2018.

[14] Dhruv Agarwal, Srinivasan Iyengar, Manohar Swaminathan, Eash Sharma, Ashish Raj, and Aadithya Hatwar. Modulo: Drive-by sensing at city-scale on the cheap. In *Proceedings of the 3rd ACM SIGCAS Conference on Computing and Sustainable Societies*, pages 187–197, 2020.

[15] Yi Gao, Wei Dong, Kai Guo, Xue Liu, Yuan Chen, Xiaojin Liu, Jiajun Bu, and Chun Chen. Mosaic: A low-cost mobile sensing system for urban air quality monitoring. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE, 2016.

[16] Kaibo Fu, Wei Ren, and Wei Dong. Multihop calibration for mobile sensing: K-hop calibratability and reference sensor deployment. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pages 1–9. IEEE, 2017.

[17] Junade Ali and Vladimir Dyo. Coverage and mobile sensor placement for vehicles on predetermined routes: A greedy heuristic approach. 2017.

[18] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Found. Comp. Math.*, 9(6):717–772, 2009.

[19] Laura Balzano, Robert Nowak, and Benjamin Recht. Online identification and tracking of subspaces from highly incomplete information. In *Proc. of IEEE Allerton*, pages 704–711, Sept. 2010.

[20] S Derin Babacan, Martin Luessi, Rafael Molina, and Aggelos K Katsaggelos. Sparse Bayesian methods for low-rank matrix estimation. *IEEE Trans. Signal Process.*, 60(8):3964–3977, 2012.

[21] Paris V Giampouras, Athanasios A Rontogiannis, Konstantinos E Themelis, and Konstantinos D Koutroumbas. Online sparse and low-rank subspace learning from incomplete data: A Bayesian view. *Signal Processing*, 137:199–212, 2017.

[22] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):11:1–11:37, 2011.

[23] Xinghao Ding, Lihan He, and Lawrence Carin. Bayesian robust principal component analysis. *IEEE Trans. Image Process.*, 20(12):3419–3430, 2011.

[24] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Trans. Pattern Anal. Mach. Intell*, 35(1):208–220, 2013.

[25] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Proc. of NIPS*, pages 847–855. Barcelona, Spain., Dec. 2016.

[26] Markos Papageorgiou, Christina Diakaki, Vaya Dinopoulou, Apostolos Kotsialos, and Yibing Wang. Review of road traffic control strategies. *Proc. of the IEEE*, 91(12):2043–2067, 2003.

[27] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proc. of the National Academy of Sciences*, 114(3):462–467, 2017.

[28] Li Qu, Li Li, Yi Zhang, and Jianming Hu. PPCA-based missing data imputation for traffic flow volume: A systematical approach. *IEEE Trans. Intell. Transp. Syst.*, 10(3):512–522, 2009.

[29] Li Qu, Yi Zhang, Jianming Hu, Liyan Jia, and Li Li. A BPCA based missing value imputing method for traffic flow volume data. In *Proc. of the IEEE Symp. Intelligent Vehicles*, pages 985–990, 2008.

[30] Huachun Tan, Yuankai Wu, Bin Shen, Peter J Jin, and Bin Ran. Short-term traffic prediction based on dynamic tensor completion. *IEEE Trans. Intell. Transp. Syst.*, 17(8):2123–2133, 2016.

[31] Muhammad Tayyab Asif, Nikola Mitrovic, Justin Dauwels, and Patrick Jaillet. Matrix and tensor based methods for missing data estimation in large traffic networks. *IEEE Trans. Intell. Transp. Syst.*, 17(7):1816–1825, 2016.

[32] Veerabhadran Ramanathan and Gregory Carmichael. Global and regional climate changes due to black carbon. *Nature geoscience*, 1(4):221, 2008.

[33] Robert D Brook, Sanjay Rajagopalan, C Arden Pope III, Jeffrey R Brook, Aruni Bhatnagar, Ana V Diez-Roux, Fernando Holguin, Yuling Hong, Russell V Luepker, Murray A Mittleman, et al. Particulate matter air pollution and cardiovascular disease: an update to the scientific statement from the american heart association. *Circulation*, 121(21):2331–2378, 2010.

[34] Aaron J Cohen, Michael Brauer, Richard Burnett, H Ross Anderson, Joseph Frostad, Kara Estep, Kalpana Balakrishnan, Bert Brunekreef, Lalit Dandona, Rakhi Dandona, et al. Estimates and 25-year trends of the global burden of disease attributable to ambient air pollution: an analysis of data from the global burden of diseases study 2015. *The Lancet*, 389(10082):1907–1918, 2017.

[35] Zhi Zhang, Guo Liang, Yan-jie Dai, Xiang-uang Dong, and Ping-xin Wang. A short-term user load forecasting with missing data. In *2018 International Conference on Mechanical, Electronic and Information Technology*, pages 395–400, Apr. 2018.

[36] Jason T Parker, Philip Schniter, and Volkan Cevher. Bilinear generalized approximate message passing Part I: Derivation. *IEEE Trans. Signal Process.*, 62(22):5839–5853, 2014.

[37] Jason T Parker, Philip Schniter, and Volkan Cevher. Bilinear generalized approximate message passing Part II: Applications. *IEEE Trans. Signal Process.*, 62(22):5854–5867, 2014.

[38] Bo Xin, Yizhou Wang, Wen Gao, and David Wipf. Exploring algorithmic limits of matrix rank minimization under affine constraints. *IEEE Trans. Signal Process.*, 64(19):4960–4974, 2016.

[39] Linxiao Yang, Jun Fang, Huiping Duan, Hongbin Li, and Bing Zeng. Fast low-rank Bayesian matrix completion with hierarchical gaussian prior models. *IEEE Trans. Signal Process.*, 66(11):2804–2817, 2018.

[40] Jaakko Luttinen. Fast variational Bayesian linear state-space model. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 305–320. Springer, 2013.

[41] Zhanyu Ma, Andrew E Teschendorff, Arne Leijon, Yuanyuan Qiao, Honggang Zhang, and Jun Guo. Variational Bayesian matrix factorization for bounded support data. *IEEE Trans. Pattern Anal. Mach. Intell*, 37(4):876–889, 2015.

[42] Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust PCA via outlier pursuit. In *Proc. of NIPS*, pages 2496–2504, Vancouver, Canada, Dec. 2010.

[43] Jun He, Laura Balzano, and John Lui. Online robust subspace tracking from partial information. *arXiv preprint arXiv:1109.3827*, 2011.

[44] Hassan Mansour and Xin Jiang. A robust online subspace estimation and tracking algorithm. In *Proc. of the IEEE ICASSP*, pages 4065–4069, Apr. 2015.

[45] S Derin Babacan, Martin Luessi, Rafael Molina, and Aggelos K Katsaggelos. Sparse Bayesian methods for low-rank matrix estimation. *IEEE Trans. Signal Process.*, 60(8):3964–3977, 2012.

[46] M. T. Asif, N. Mitrovic, L. Garg, J. Dauwels, and P. Jaillet. Low-dimensional models for missing data imputation in road networks. In *Proc. of the IEEE ICASSP*, pages 3527–3531, May. 2013.

[47] Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[48] Matthew J. Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, University of London London, 2003.

[49] Dimitris G Tzikas, Aristidis C Likas, and Nikolaos P Galatsanos. The variational approximation for Bayesian inference. *IEEE Signal Process. Mag.*, 25(6):131–146, 2008.

[50] Masa-Aki Sato. Online model selection based on the variational Bayes. *Neural Computation*, 13(7):1649–1681, 2001.

[51] Paul Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

[52] Xinyu Chen, Zhaocheng He, Yixian Chen, Yuhuan Lu, and Jiawei Wang. Missing traffic data imputation and pattern discovery with a bayesian augmented tensor factorization model. *Transportation Research Part C: Emerging Technologies*, 104:66–77, 2019.

[53] Thierry Bouwmans and El Hadi Zahzah. Robust pca via principal component pursuit: A review for a comparative evaluation in video surveillance. *Computer Vision and Image Understanding*, 122:22–34, 2014.

[54] Chinh Dang and Hayder Radha. Rpca-kfe: Key frame extraction for video using robust principal component analysis. *IEEE Transactions on Image Processing*, 24(11):3742–3753, 2015.

[55] Sotiris Vardoulakis, Norbert Gonzalez-Flesca, Bernard EA Fisher, and Koulis Pericleous. Spatial variability of air pollution in the vicinity of a permanent monitoring station in central paris. *Atmospheric Environment*, 39(15):2725–2736, 2005.

[56] Amin Anjomshoaa, Fábio Duarte, Daniël Rennings, Thomas J Matarazzo, Priyanka deSouza, and Carlo Ratti. City scanner: Building and scheduling a mobile sensing platform for smart city services. *IEEE Internet of Things Journal*, 5(6):4567–4579, 2018.

[57] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in neural information processing systems*, pages 847–855, 2016.

[58] Armin Eftekhari, Dehui Yang, and Michael B Wakin. Weighted matrix completion and recovery with prior subspace information. *IEEE Transactions on Information Theory*, 64(6):4044–4071, 2018.

[59] Madeleine Udell and Alex Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019.

[60] S Derin Babacan, Martin Luessi, Rafael Molina, and Aggelos K Katsaggelos. Sparse bayesian methods for low-rank matrix estimation. *IEEE Transactions on Signal Processing*, 60(8):3964–3977, 2012.

[61] Yinqiang Zheng, Guangcan Liu, Shigeki Sugimoto, Shuicheng Yan, and Masatoshi Okutomi. Practical low-rank matrix approximation under robust l 1-norm. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1410–1417. IEEE, 2012.

[62] Qibin Zhao, Guoxu Zhou, Liqing Zhang, Andrzej Cichocki, and Shun-Ichi Amari. Bayesian robust tensor factorization for incomplete multiway data. *IEEE transactions on neural networks and learning systems*, 27(4):736–748, 2015.

[63] Jaakko Luttinen. Fast variational Bayesian linear state-space model. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 305–320. Springer, 2013.

[64] Charul, U. Bhatt, P. Biyani, and K. Rajawat. Online variational bayesian subspace filtering. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5057–5061, 2019.

[65] Qibin Zhao, Liqing Zhang, and Andrzej Cichocki. Bayesian cp factorization of incomplete tensors with automatic rank determination. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1751–1763, 2015.

[66] Xinyu Chen, Zhaocheng He, and Lijun Sun. A bayesian tensor decomposition approach for spatiotemporal traffic data imputation. *Transportation research part C: emerging technologies*, 98:73–84, 2019.

[67] Linxiao Yang, Jun Fang, Huiping Duan, Hongbin Li, and Bing Zeng. Fast low-rank bayesian matrix completion with hierarchical gaussian prior models. *IEEE Transactions on Signal Processing*, 2018.

[68] Michael E Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244, 2001.

[69] Xinyu Chen, Yixian Chen, and Zhaocheng He. Urban traffic speed dataset of guangzhou, china [data set].zenodo., 2018.

[70] Yixian Chen and Xinyu Chen. A large scale pems traffic speed dataset (version v1) [data set].zenodo., 2020.

[71] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th SIGKDD conference on Knowledge Discovery and Data Mining*, August 2015.

[72] Greg Ongie, Rebecca Willett, Robert D Nowak, and Laura Balzano. Algebraic variety models for high-rank matrix completion. In *International Conference on Machine Learning*, pages 2691–2700, 2017.

[73] Lijun Sun and Xinyu Chen. Bayesian temporal factorization for multidimensional time series prediction. *arXiv preprint arXiv:1910.06366*, 2019.

[74] Longhao Yuan, Chao Li, Danilo Mandic, Jianting Cao, and Qibin Zhao. Tensor ring decomposition with rank minimization on latent space: An efficient approach for tensor completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9151–9158, 2019.

[75] Tim Salimans, Diederik Kingma, and Max Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*, pages 1218–1226, 2015.

[76] Xiaojie Guo and Zhouchen Lin. Low-rank matrix recovery via robust outlier estimation. *IEEE Transactions on Image Processing*, 27(11):5316–5327, 2018.

[77] Pirlea Florina and Huang Wendy Ven-dee. The global distribution of air pollution. https://datatopics.worldbank.org/world-development-indicators/stories/the-global-distribution-of-air-pollution.html, 2019.

[78] Charul Paliwal, Uttkarsha Bhatt, Pravesh Biyani, and Ketan Rajawat. Traffic estimation and prediction via online variational bayesian subspace filtering. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[79] Satoru Fujishige. *Submodular functions and optimization*. Elsevier, 2005.

[80] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 182–196. Springer, 2007.

[81] Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 570–579. IEEE, 2011.

[82] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(2), 2008.

[83] Manohar Shamaiah, Siddhartha Banerjee, and Haris Vikalo. Greedy sensor selection: Leveraging submodularity. In *49th IEEE conference on decision and control (CDC)*, pages 2572–2577. IEEE, 2010.

[84] Sebastian Tschiatschek, Rishabh K Iyer, Haochen Wei, and Jeff A Bilmes. Learning mixtures of submodular functions for image collection summarization. In *Advances in neural information processing systems*, pages 1413–1421, 2014.

[85] Sebastian Tschiatschek, Rishabh K Iyer, Haochen Wei, and Jeff A Bilmes. Learning mixtures of submodular functions for image collection summarization. In *Advances in neural information processing systems*, pages 1413–1421, 2014.

[86] Ehsan Elhamifar. Sequential facility location: Approximate submodularity and greedy algorithm. In *International Conference on Machine Learning*, pages 1784–1793. PMLR, 2019.

[87] Aamir Anis, Akshay Gadde, and Antonio Ortega. Efficient sampling set selection for bandlimited graph signals using graph spectral proxies. *IEEE Transactions on Signal Processing*, 64(14):3775–3789, 2016.

[88] Siheng Chen, Rohan Varma, Aliaksei Sandryhaila, and Jelena Kovavcevi'c. Discrete signal processing on graphs: Sampling theory<? pub _newline=""? *IEEE transactions on signal processing*, 63(24):6510–6523, 2015.

[89] Siheng Chen, Rohan Varma, Aarti Singh, and Jelena Kovavcevic. Signal recovery on graphs: Fundamental limits of sampling strategies. *IEEE Transactions on Signal and Information Processing over Networks*, 2(4):539–554, 2016.

[90] Abolfazl Hashemi, Rasoul Shafipour, Haris Vikalo, and Gonzalo Mateos. Accelerated sampling of bandlimited graph signals. *arXiv preprint arXiv:1807.07222*, 2018.

[91] Ajinkya Jayawant and Antonio Ortega. A distance-based formulation for sampling signals on graphs. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6318–6322. IEEE, 2018.

[92] Paul D Sampson, Adam A Szpiro, Lianne Sheppard, Johan Lindström, and Joel D Kaufman. Pragmatic estimation of a spatio-temporal air quality model with irregular monitoring data. *Atmospheric Environment*, 45(36):6593–6606, 2011.

[93] Wanli Min and Laura Wynter. Real-time road traffic prediction with spatio-temporal correlations. *Transportation Research Part C: Emerging Technologies*, 19(4):606–616, 2011.

[94] S Derin Babacan, Martin Luessi, Rafael Molina, and Aggelos K Katsaggelos. Sparse bayesian methods for low-rank matrix estimation. *IEEE Transactions on Signal Processing*, 60(8):3964–3977, 2012.

[95] Tinghui Zhou, Hanhuai Shan, Arindam Banerjee, and Guillermo Sapiro. Kernelized probabilistic matrix factorization: Exploiting graphs and side information. In *Proceedings of the 2012 SIAM international Conference on Data mining*, pages 403–414. SIAM, 2012.

[96] Iman Barjasteh, Rana Forsati, Dennis Ross, Abdol-Hossein Esfahanian, and Hayder Radha. Cold-start recommendation with provable guarantees: A decoupled approach. *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1462–1474, 2016.

[97] Dorit S Hochbaum. Approximating covering and packing problems: set cover, vertex cover, independent set, and related problems. In *Approximation algorithms for NP-hard problems*, pages 94–143. 1996.

[98] CPCB. Delhi real time air quality data, 2015.

[99] Yueliang Liu, Wenbin Guo, Kangyong You, Lei Zhao, Tao Peng, and Wenbo Wang. Graph learning for spatiotemporal signals with long-and short-term characterization. *IEEE Transactions on Signal and Information Processing over Networks*, 6:699–713, 2020.

[100] Candace Brakewood and Kari Watkins. A literature review of the passenger benefits of real-time transit information. *Transport Reviews*, pages 1–30, 2018.

[101] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

[102] Raffi Sevlian and Ram Rajagopal. Travel time estimation using floating car data. *arXiv preprint arXiv:1012.4249*, 2010.

[103] Corrado De Fabritiis, Roberto Ragona, and Gaetano Valenti. Traffic estimation and prediction based on real time floating car data. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 197–203. IEEE, 2008.

[104] Muhammad Tayyab Asif, Justin Dauwels, Chong Yang Goh, Ali Oran, Esmail Fathi, Muye Xu, Menoth Mohan Dhanya, Nikola Mitrovic, and Patrick Jaillet. Spatiotemporal patterns in large-scale traffic speed prediction. *IEEE Transactions on Intelligent Transportation Systems*, 15(2):794–804, 2014.

[105] Mahmood Rahmani, Erik Jenelius, and Haris N Koutsopoulos. Route travel time estimation using low-frequency floating car data. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 2292–2297. IEEE, 2013.

[106] Daniel Billings and Jiann-Shiou Yang. Application of the arima models to urban roadway travel time prediction-a case study. In *Systems, Man and Cybernetics, 2006. SMC'06. IEEE International Conference on*, volume 3, pages 2529–2534. IEEE, 2006.

[107] Aude Hofleitner, Ryan Herring, Pieter Abbeel, and Alexandre Bayen. Learning the dynamics of arterial traffic from probe data using a dynamic bayesian network. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1679–1693, 2012.

[108] Bei Pan, Ugur Demiryurek, and Cyrus Shahabi. Utilizing real-world transportation data for accurate traffic prediction. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 595–604. IEEE, 2012.

[109] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, Fei-Yue Wang, et al. Traffic flow prediction with big data: A deep learning approach. *IEEE Trans. Intelligent Transportation Systems*, 16(2):865–873, 2015.

[110] Bin Yang, Chenjuan Guo, and Christian S Jensen. Travel cost inference from sparse, spatio temporally correlated time series using markov models. *Proceedings of the VLDB Endowment*, 6(9):769–780, 2013.

[111] Jieping Ye Zheng Wang, Kun Fu. Learning to estimate the travel time, 2018.

[112] Jiwon Myung, Dong-Kyu Kim, Seung-Young Kho, and Chang-Ho Park. Travel time prediction using k nearest neighbor method with combined data from vehicle detector system and automatic toll collection system. *Transportation Research Record*, 2256(1):51–59, 2011.

[113] Steven I-Jy Chien and Chandra Mouly Kuchipudi. Dynamic travel time prediction with real-time and historic data. *Journal of transportation engineering*, 129(6):608–616, 2003.

[114] Chun-Hsin Wu, Chia-Chen Wei, Da-Chun Su, Ming-Hua Chang, and Jan-Ming Ho. Travel time prediction with support vector regression. In *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE*, volume 2, pages 1438–1442. IEEE, 2003.

[115] Yanru Zhang and Ali Haghani. A gradient boosting method to improve travel time prediction. *Transportation Research Part C: Emerging Technologies*, 58:308–324, 2015.

[116] Jingyuan Wang, Qian Gu, Junjie Wu, Guannan Liu, and Zhang Xiong. Traffic speed prediction and congestion source exploration: A deep learning method. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 499–508. IEEE, 2016.

[117] Yanjie Duan, Yisheng Lv, and Fei-Yue Wang. Travel time prediction with lstm neural network. In *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, pages 1053–1058. IEEE, 2016.

[118] Wang-Chien Lee, Weiping Si, Ling-Jyh Chen, and Meng Chang Chen. Http: a new framework for bus travel time prediction based on historical trajectories. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, pages 279–288. ACM, 2012.

[119] Hongjian Wang, Yu-Hsuan Kuo, Daniel Kifer, and Zhenhui Li. A simple baseline for travel time estimation using large-scale trip data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 61. ACM, 2016.

[120] Lelitha Vanajakshi. Performance comparison of bus travel time prediction models across indian cities. *Transportation Research Record*, 1:12, 2018.

[121] B Anil Kumar, R Jairam, Shriniwas S Arkatkar, and Lelitha Vanajakshi. Real time bus travel time prediction using k-nn classifier. *Transportation Letters*, 11(7):362–372, 2019.

[122] Hima Elsa Shaji, Arun K Tangirala, and Lelitha Vanajakshi. Evaluation of clustering algorithms for the prediction of trends in bus travel time. *Transportation Research Record*, 2672(45):242–252, 2018.

[123] B Dhivyabharathi, B Anil Kumar, and Lelitha Vanajakshi. Real time bus arrival time prediction system under indian traffic condition. In *2016 IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, pages 18–22. IEEE, 2016.

[124] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[125] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[126] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[127] Charul, Uttkarsha Bhatt, Pravesh Biyani, and Ketan Rajawat. Online variational bayesian subspace filtering. In *Proc. of the IEEE ICASSP*, May. 2019.

[128] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[129] Dong Wang, Junbo Zhang, Wei Cao, Jian Li, and Yu Zheng. When will you arrive? estimating travel time based on deep neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[130] Greg Ongie, Rebecca Willett, Robert D Nowak, and Laura Balzano. Algebraic variety models for high-rank matrix completion. *arXiv preprint arXiv:1703.09631*, 2017.

[131] Jicong Fan and Madeleine Udell. Online high rank matrix completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8690–8698, 2019.