Online Calibration for Simulation-Based Dynamic Traffic Assignment: towards Large-Scale and Real-Time Performance

by

Haizheng Zhang

B.E., Tsinghua University (2013)
S.M., Massachusetts Institute of Technology (2016)

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Transportation

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2018

Signature of author .....
Department of Civil and Environmental Engineering
August 17, 2018

Certified by .....
Moshe E. Ben-Akiva
Edmund K. Turner Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by .....
Heidi Nepf
Donald and Martha Harleman Professor of Civil and Environmental Engineering
Chair, Graduate Program Committee

Online Calibration for Simulation-Based Dynamic Traffic Assignment: towards Large-Scale and Real-Time Performance

by

Haizheng Zhang

Submitted to the Department of Civil and Environmental Engineering
on August 17, 2018, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Transportation

Abstract

The severity of traffic congestion is increasing each year in the US, resulting in higher travel times, and increased energy consumption and emissions. They have led to an increasing emphasis on the development of tools for traffic management, which intends to alleviate congestion by more efficiently utilizing the existing infrastructure. Effective traffic management necessitates the generation of accurate short-term predictions of traffic states and in this context, simulation-based Dynamic Traffic Assignment (DTA) systems have gained prominence over the years. However, a key challenge that remains to be addressed with real-time DTA systems is their scalability and accuracy for applications to large-scale urban networks.

A key component of real-time DTA systems that impacts scalability and accuracy is online calibration which attempts to adjust simulation parameters in real-time to match as closely as possible simulated measurements with real-time surveillance data. This thesis contributes to the existing literature on online calibration of DTA systems in three respects: (1) modeling explicitly the stochasticity in simulators and thereby improving accuracy; (2) augmenting the State Space Model (SSM) to capture the delayed measurements on large-scale and congested networks; (3) presenting a gradient estimation procedure called partitioned simultaneous perturbation (PSP) that utilizes an assumed sparse gradient structure to facilitate real-time performance. The results demonstrate that, first, the proposed approach to address stochasticity improves the accuracy of supply calibration on a synthetic network. Second, the augmented SSM improves both estimation and prediction accuracy on a congested synthetic network and the large-scale Singapore expressway network. Finally, compared with the traditional finite difference method, the PSP reduces the number of computations by 90% and achieves the same calibration accuracy on the Singapore expressway network. The proposed methodologies have important applications in the deployment of real-time DTA systems for large scale urban networks.

Thesis Supervisor: Moshe E. Ben-Akiva
Title: Edmund K. Turner Professor of Civil and Environmental Engineering

Acknowledgments

First of all, I would like to express my deepest gratitude to Prof. Moshe Ben-Akiva. Your extraordinary intelligence and continuous support has made my PhD life at MIT unique and fantastic. It is a great honor and pleasure to work with you.

I express my sincere gratitude to Dr. Ravi Seshadri, who backs me up all the time along this colorful journey. I would never make this far without your continuous support. Great thanks go to committee members Prof. Constantinos Antoniou, Prof. Francisco Pereira, Prof. Saurabh Amin and Prof. Carolina Osorio. You are extremely helpful not only with my thesis, but also with every step in the research.

I would like to thank Katherine Rosa and Eunice Kim for their help in detailed but necessary administrative affairs. Special thanks to Prof. Markus Buehler and Kiley Clapper for the honor to be supported by Civil and Environmental Engineering.

Thanks to my best friends (and lovely roommates) Hongyi Zhang and Tianli Zhou. It was a great pleasure to meet you, know you and share with you my experience at MIT. Great thanks to Hongyi for his excellent knowledge and intelligence, from which I benefit substantially. Thanks also go to my friends at ITS Lab and MIT: Zhen Yang, Wen Jian, Xuenan Ni, Xiaoyan Shen, Xia Miao, Linsen Chong, Chao Zhang, Zhan Zhao, Li Jin, Manxi Wu, Xiang Song, Peiyu Jing, Yundi Zhang, Yihang Sui, Mazen Danaf, Eytan Gross, Ajinkya Ghorpade, Samarth Gupta, Isabel Viegas, Arun Prakash, Carlos Azevedo, Kyungsoo Jeong, and Bilge Atasoy.

Last but not least, I would like to thank Weichen Liu, for the endless love and support whenever I need. My sincerest gratitude goes to my parents, for you will always be there for me no matter what happens. You carefully express your love, usually without many words, yet no more words in the world can be greater than your love. Forever shall I remain indebted to you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Congestion is an important issue in transportation systems. Traffic congestion is a scenario that commuters experience daily, and its severity is rapidly increasing each year in the United States. During peak hours in 2016, trips took 35% more time on average than non-peak hours, while the percentage was only 20% in 2010 (Schrank et al., 2015). According to FHWA (2016), the average duration of congestion in traffic systems is 4.7 hours daily in 2016, compared with 4.3 hours in 2009 (FHWA, 2009). Apart from the time delay, congestion exacerbates air pollution, energy consumption and emissions. The extra fuel expenditure due to congestion was 19 gallons annually for an average vehicle in 2014, which is a 4-gallon increase from 2010 (Schrank et al., 2015). Congestion incurred an estimated $160 billion annual cost for extra time and fuel in 2014, and the cost is expected to be $192 billion in 2020.

Traffic management is a sustainable and effective alternative to alleviate congestion, given that the traditional practice to build more roads is untenable nowadays because of physical and economic constraints. A Transportation Management Center (TMC) usually serves as the nerve center to manage freeway and arterial traffic and mitigate congestion. TMCs obtain real-time traffic surveillance data; provide information to travelers; and generate control strategies or guidance on freeway tolls, ramp-metering signals and intersections. To make these strategies effective and reliable, TMCs should be able to *predict* short-term traffic conditions, which form the basis of proactive control strategies. Thus, a *model* that captures the traffic system

is necessary for traffic management. Such a model should be able to *estimate* traffic conditions from real-time surveillance data and *predict* future conditions based on the estimation. Dynamic Traffic Assignment (DTA) systems have long been considered effective tools in this regard. A DTA system is one that captures the evolution of traffic conditions with a synthesis of demand and supply models. DTA systems assign time-dependent traffic demand to road networks and determine the traffic conditions through modeling the interactions between demand and supply.

While DTA systems can predict short-term traffic conditions, the prediction accuracy relies on the quality of *online calibration*, which aims at estimating and predicting DTA model parameters using the real-time surveillance data. Online calibration is a key component of real-time DTA systems that is crucial in replicating real traffic conditions and thus providing accurate predictions. However, the accuracy usually comes with a price of *complexity*. The complexity of online calibration constrains the successful deployment of DTA systems in three aspects: robustness of accurate prediction, large-scale applicability and real-time performance. In this thesis, we address these three aspects in online calibration with the aim of improving prediction accuracy and addressing computational complexity for DTA.

1.1 Dynamic Traffic Assignment

Traffic assignment is a modeling process which aims to determine the traffic states in the network. It involves assigning demand to road networks, modeling travelers' behavior and estimating network conditions and travel times. Traffic assignment involves two key model components: demand and supply. The demand model dictates the assignment of origin-destination (OD) flows to different routes. Based on the assigned flows, the supply model determines how traffic flow advances in the network. There are two types of traffic assignment models: static and dynamic.

Static traffic assignment assumes that the demand and supply models stay the same for the modeling period and thus, describes the steady state traffic conditions in the network (Chiu et al., 2011). During the peak period, the traffic volumes are

determined by a fixed origin-destination (OD) matrix, and the supply model gives travel times for each link based on a volume-delay function. The static model does not explicitly represent detailed traffic dynamics such as queuing and vehicle movements. Specifically, there are no constraints on the link flow volumes, as the inflow always equals the outflow. Thus, it is impossible to accurately model traffic flow conditions in congestion.

In DTA, the interactions between demand and supply are time-dependent. The demand module generates time-dependent trips. Then, based on the demand assigned to each segment in the network, the supply module dynamically models traffic conditions. For example, there is a fundamental diagram that determines the traffic speed on each segment given different traffic volumes on it. When the outflow reaches the capacity for a segment, a queue will form at the end of the segment. Following this, on adjacent segments, congestion propagation and queue dissipation are explicitly modeled with high fidelity. Thus, DTA can capture various traffic conditions (either steady state or transition to/from congestion) and determine their progression across time and space. In traffic management operations, DTA models are particularly favorable since they overcome the drawbacks in static traffic assignment. Since the late 1970s, DTA has evolved substantially into an important tool for estimating and predicting dynamic traffic flows on road networks.

Figure 1-1 presents the general framework of DTA systems (Ben-Akiva et al., 2010a). The demand and supply modules receive inputs and surveillance data from the management system, which is usually deployed at TMCs. Inputs include the network representation, historical time-dependent OD matrices, supply parameters, traveler behavior parameters, incident or event information, weather conditions, and traffic control strategies. Surveillance data include real-time field measurements, such as traffic flow counts, average speeds, segment densities, and link travel times. The demand-supply interaction is captured through simulation in the DTA system (or mathematical formulation for analytical DTA). Finally, the DTA system generates traffic conditions that match the surveillance data so as to provide an accurate prediction of future conditions.

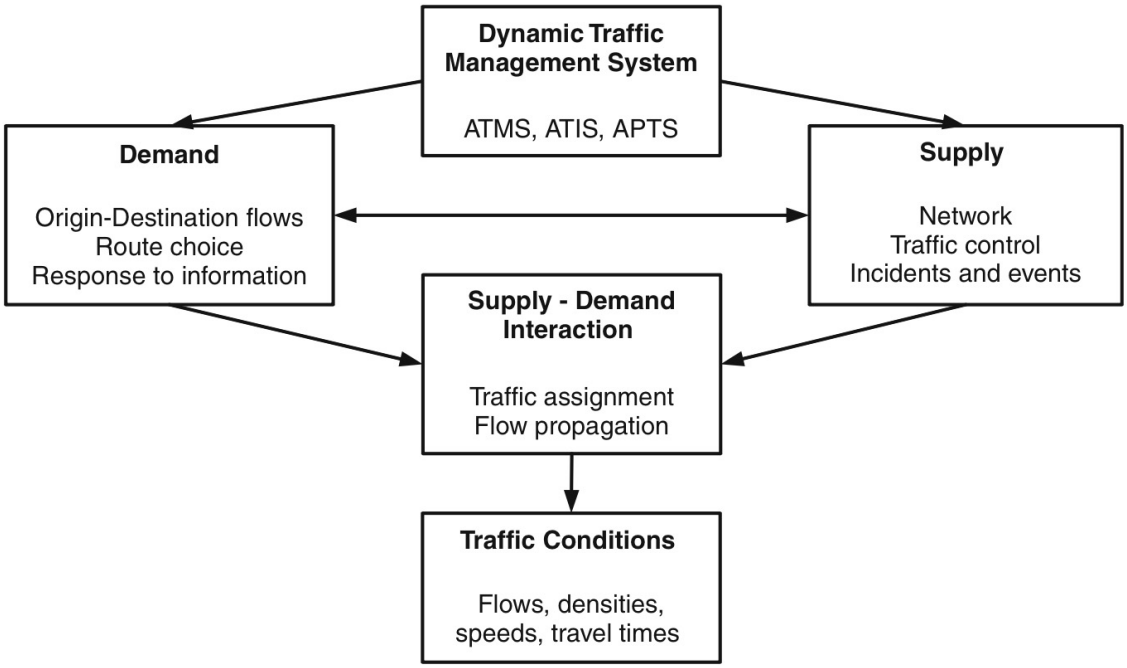


Figure 1-1: General DTA framework (Ben-Akiva et al., 2010a)

Various DTA models have been proposed in the literature which essentially belong to two categories: analytical and simulation-based (Peeta & Ziliaskopoulos, 2001). Most analytical DTA models express the route assignment problem as a mathematical optimization with the target of user equilibrium, system optimum or their variants. Simulation-based DTA captures traffic flow dynamics using a simulator. The main difference between analytical and simulation-based DTA is the approach adopted to model route-choice decisions, queue accumulation or dissipation, vehicle movements and flow conservation. These processes are discrete and stochastic. Thus, capturing them adds extra complexity to an analytical model, and it may lose close-form properties. On the other hand, simulations can easily handle the physical constraints and discrete decisions via rules and random sampling. Thus, owing to higher fidelity in replicating traffic conditions, simulation-based DTA has gained wide acceptability in real-world deployments (Mahmassani, 2001; Ben-Akiva et al., 2010a, 2010b). In view of the aforementioned advantages of simulation-based DTA, it forms the basis of this thesis.

1.2 Challenges of DTA Deployment

1.2.1 Online Calibration for DTA

A key component of real-time DTA systems is *online calibration* which refers to the determination of model parameters in real-time so that the DTA system replicates as closely as possible current traffic conditions implied by the surveillance data. These parameters are crucial to the demand and supply modules (of the DTA system) and necessary for accurate estimation and prediction of traffic conditions, which as noted before are the basis of traffic management systems in TMCs.

Apart from the requirement of prediction accuracy, several other problems restrict the broad deployment of DTA systems. Three key challenges identified by Peeta & Ziliaskopoulos (2001) are:

(1) Robustness: incorporating randomness,

(2) Large-scale network applicability, and

(3) Real-time performance.

1.2.2 Robustness: Incorporating Randomness

While traffic prediction plays a significant role in traffic management, the robustness of the predictions is no less important. Robustness is crucial and requires the appropriate characterization of the various sources of uncertainty which arise due to: (1) inherent stochasticity in the supply and demand simulators of the DTA system, whose randomness results from departure times and route choice decisions on demand side and vehicle movement models on supply side; (2) measurement error or noise in the surveillance data; and (3) modeling errors in the calibration process. The randomness from these sources accumulate and pose a critical challenge in the online calibration process since it involves fitting a stochastic simulator to noisy measurements. Thus, in order to generate accurate and robust traffic state estimations and predictions, it

is imperative that the online calibration process handles and mitigates if possible, the stochasticity arising from the aforementioned sources.

1.2.3 Large-Scale Network Applicability

TMCs are interested in DTA's ability to model large-scale networks since global traffic control is more effective than local control. Thus, one future direction of DTA is the deployment to large-scale networks. It requires the DTA system be able to handle time-dependent parameters that may be numerous, in the order of tens of thousands. *Computational tractability* will be a key issue when the parameter space increases. Regarding online calibration, the large dimension of parameter space and the increased complexity of the DTA system pose additional challenges to estimate and predict model parameters. Also, the faster growth of parameters than the surveillance data exacerbates this issue. For example, in the case of OD estimation, when the area of a network increases, the origin and destination nodes and surveillance sensors grow linearly, but the number of OD pairs increase quadratically. In conclusion, the application of DTA systems to large-scale networks is challenging, and it affects the *computational tractability* of online calibration.

1.2.4 Real-Time Performance

As the scale of the network expands, the computational time is likely to increase as well. Nevertheless, traffic estimation and prediction have to occur on time to respond to changes in traffic conditions. The real-time requirement also comes from the TMCs to monitor and manage traffic. Most DTA systems model traffic flows by splitting time into intervals, typically within a rolling-horizon framework (Figure 1-2). For these DTA systems, real-time performance requires that the simulation and parameter estimation should finish within the current interval. For instance, for a DTA system with 5-minute estimation intervals, it receives surveillance data at 8:00 for the time interval 7:55-8:00. Next, the DTA system needs to update model parameters using the surveillance data, make traffic predictions, and generate control strategies for the

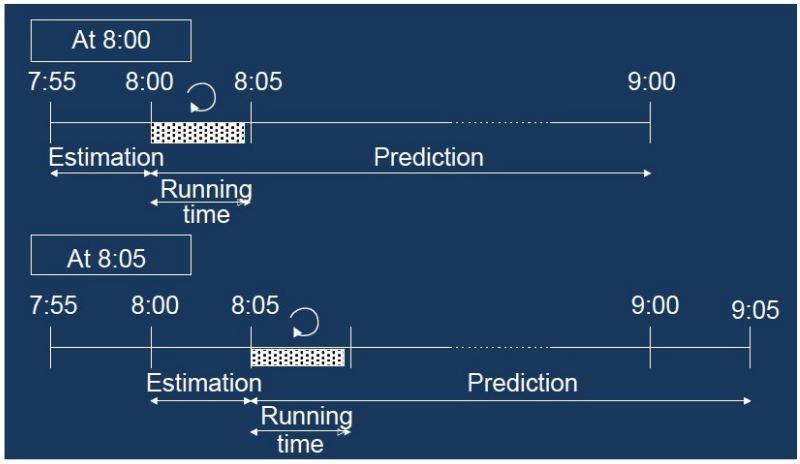


Figure 1-2: The rolling horizon framework for traffic estimation and prediction in DTA systems

prediction interval. All these tasks need to finish within the estimation interval length (i.e., 5 minutes). Then at 8:05, the DTA receives new surveillance data, with which all the tasks will be performed again.

On the other hand, from the perspective of TMCs, estimation intervals smaller than 5 minute may be preferable for an instant and effective response in non-recurrent and urgent scenarios such as accidents and emergencies. Hence, the future direction for real-time deployment is to decrease the interval length which implies a smaller time budget for the state estimation, state prediction and control generation. Moreover, reducing the interval length makes online calibration more difficult in the following two ways. The first is less information captured in each interval about traffic conditions. With the same sparsity of sensors in the road network, a shorter interval results in less flow for each sensor. The second is increased delay in modeling and expanded parameter space. Some parameters will only be observable in later intervals. Thus, the delayed relations have to be modeled across time steps, and the number of parameters increases. These two issues also exist for large-scale networks, where sensors are sparse; and congested networks, where traffic flows hit less sensors than free flows.

In a nutshell, with the purpose of accurate traffic prediction, DTA systems have three significant challenges in their future deployment to real-world applications.

These challenges will persist, and many endeavors will be made to solve them in the next few years.

1.3 Thesis Contributions

This thesis aims at addressing three significant challenges and push the frontiers of online calibration for simulation-based DTA. Specifically, we focus on the online calibration problem and contribute to the existing literature in the following respects.

- The research identifies, quantifies and investigates the *stochasticity* in DTA simulators. The thesis also proposes methods to address the stochasticity issue, which yields better estimation accuracy for online supply calibration.

- The research extends the online calibration framework to handle measurement issues on *large-scale networks*. The state augmentation technique is able to model the measurement delay and improve the validity of the underlying state space model. This approach is also able to deal congestion scenarios with small simulation intervals.

- The research presents a sparse gradient estimation procedure to facilitate *real-time performance* for online calibration. It significantly reduces the computational complexity, while maintaining estimation and prediction accuracy, as reported in the case study with a large-scale network.

1.4 Thesis Outline

The thesis structure is listed below. Chapter 2 summarizes and comments on the recent developments in online calibration with particular emphasis on Kalman filtering algorithms. Chapter 3 quantifies and analyzes the stochasticity within the simulation-based DTA system. The chapter also presents two remedies to mitigate the impacts of stochasticity and provide robust and accurate traffic estimations. Chapter 4 sheds light on the modeling of delayed traffic systems, where the impact of parameters is

only captured in the measurements several intervals later due to long travel times in large-scale networks and congestion. A simple traffic assignment example demonstrates the effect of delay. Following this observation, we present the state augmentation technique that addresses this issue. We demonstrate the applicability of the technique on the traffic assignment example and a synthetic case study with a small congested network. In Chapter 5, we discuss the gradient estimation process for the identification of DTA systems. Then, we present a sparse gradient estimation technique called partitioned simultaneous perturbation to accelerate online calibration. Chapter 6 presents a case study for online calibration on a large-scale network: the Singapore expressways. Some practical considerations are also discussed. Finally, this thesis ends with conclusions and future research directions.

Chapter 2

Recent Developments in Online Calibration for DTA

Calibration for DTA involves the estimation of model parameters to fit surveillance data such that the traffic conditions in the DTA system represent the real world. As discussed in Section 1.2, the real-time deployment of DTA requires the calibration process to be online, where surveillance data arrive in batches, and calibration is performed in real-time with each batch. The goal of online calibration is to estimate model parameters so that the DTA model can represent the real-time traffic scenario, in the sense of minimizing the discrepancy between model outputs and surveillance data. There are two requirements for the solution approach to be truly "online": (1) model parameters for an interval are updated only based on data up to that time, i.e., the algorithm does not "foresee" data; and (2) the calibration for one interval will complete in less time than the interval length, i.e., the calibration is faster than real-time data generation.

In this chapter, we start with the literature review of online calibration for DTA. Next, we present the online calibration framework based on the state space model and focus on the critical system identification for simulation-based DTA. Following this, we introduce the Kalman filtering based solution approach, and finally, the chapter is summarized.

2.1 Online Calibration in Literature

There has been extensive research on calibration for DTA systems. Existing approaches to model DTA systems broadly fall into two categories: analytical and simulation-based. The critical difference lies in whether there exists a direct closed-form relation between model parameters and measurements. If no analytical relation is available, the key to calibration is modeling the DTA system as mathematical functions on which optimization algorithms can rely. Given the analytical functions, the calibration problem is essentially a regression task that aims at estimating parameters to fit measurements. Here we need to clarify that although the topic of this thesis is simulation-based DTA, reviewing the literature for general DTAs is still helpful because they may share the same intuition.

Extensive studies have focused on the online calibration problem, but not many algorithms have been proven to be efficient and scalable. The state space model (SSM) is a prominent candidate that achieves both. Recent research has applied the state space models to online calibration problems, with the Kalman filtering framework as the solution approach.

The State Space Model

The state space model or hidden Markov model is a time-series model that describes the transition process and observation process of the state variables. Model parameters in DTA for each interval comprise the *state*, which evolves according to a transition relation. At each time step, *observations* bear a relationship with the *state*, which determines the measurement relation at that time. Since the states may not be directly observable, they are also called "hidden states". The goal of the state space model is to infer and predict the *states* from the *observations*.

In order to model the transition equation, there have been numerous approaches proposed in the literature. Ashok & Ben-Akiva (1993) define a stationary time series model for the transition equation. It requires offline calibrated OD flows to serve as historical parameters. Based on the same idea, Ashok & Ben-Akiva (2000) formulated

an autoregressive (AR) process to the fourth degree on the deviations. The authors applied Kalman filtering techniques to estimate and predict OD demand in real-time with satisfactory results. Wang & Papageorgiou (2005) formulated both demand and supply parameters in a stochastic macroscopic model; a random walk transition model is applied to estimate traffic conditions on freeway stretches. Zhou & Mahmassani (2007) assumed a stationary random process with constant mean and variance and demonstrated its performance as a transition equation for OD estimation on a test network in Irvine. On the other hand, the stationary time series model may fail when the pattern of parameters is different from historical values. In such cases, we can apply an uninformative random walk model to the absolute values of model parameters, which assumes no historicals as priors. Cremer & Keller (1987); Chang & Wu (1994) assumed a random walk to make predictions on dynamic split proportions for route choice. The authors concluded that this assumption worked well in terms of prediction accuracy and stability. However, the authors used a scenario where demand changes slowly. Thus, the result may not reflect the trends of time-dependent OD flows in reality. It is still advisable to apply deviations from historical parameters whenever available to incorporate maximum historical information.

As for the measurement equation, its specification depends on the DTA system. In OD estimation, most research applies the assignment matrix to describe the measurement model (Ashok, 1996; Zhou & Mahmassani, 2007). For supply and route choice parameters, analytical and simulation-based DTAs utilize distinct approaches. Since closed-form relations exist for analytical DTAs, the measurement equations can be derived explicitly. Two representative traffic flow models in analytical DTA include the CTM model (Daganzo, 1995) and LWR model (Richards, 1956; Lighthill & Whitham, 1955), which are expressed in the form of partial-differential equations. Thus, either closed-form or numerical solutions are available to describe the measurement equation under current traffic conditions. In the case of simulation-based DTAs, it is difficult to formulate a closed-form relation due to the complex nonlinear and stochastic nature of the simulator. An approach to solve this problem is via system identification: approximating the simulation-based DTA with mathematical models.

In order words, the DTA system is now treated as a "black box" and an analytical model is estimated between input parameters and simulated measurements. The most widely used model is a simple linear relationship, in which case the identification task reduces to gradient estimation. Since no prior analytical form is assumed, the gradient estimation approach is generic and can handle all types of input parameters and measurements (Antoniou et al., 2004, 2006). Nevertheless, its drawback is in computational complexity: the number of function evaluations required can be as large as parameter dimensions.

Based on these specifications for the state space model, the solution approach for a linear state space model is Kalman filtering. Antoniou (2004) applied an extended Kalman filter (EKF), unscented Kalman filter (UKF) and limiting EKF (LimEKF) in a case study involving two freeway stretches in the UK and California. The author calibrated demand and supply parameters for a simulation-based DTA with flow volumes and speed data. Regarding computational performance, gradient estimation comprises a major part of the computation. The LimEKF was reported to have superior computational performance with complexity $O(1)$, due to using an offline estimated Kalman gain matrix and hence, had online performance. EKF and UKF have a similar computational complexity of $O(n)$, where n is the number of calibration parameters. The results showed that the EKF outperforms UKF and LimEKF in terms of estimation and prediction accuracy. Thus, the author concluded that EKF is still the most straightforward approach, despite the time complexity and the linear approximation. However, with a freeway stretch, the case study has only 80 parameters for each 15-minute time interval. Since the goal of online calibration is real-time performance, the approaches are yet to be proven on large-scale networks with larger parameter dimensions and short time intervals.

Another recent development in offline calibration that may have applications in the online case is the use of meta-models (Osorio & Bierlaire, 2013; C. Zhang et al., 2017). The idea is to model the objective function (usually the divergence between real measurements and simulation) with an analytical approximation. The analytical model is macroscopic and problem-specific. A general-purpose parametric function

(e.g., polynomials) is also included in the analytical model to allow for the imperfect problem-specific modeling. Then, scaling parameters for the problem-specific and general-purpose functions are estimated from traffic simulation for a given period. The meta-models work as a hybrid of the analytical form and mathematical approximation. Thus, it may yield benefits of both approaches and is a promising direction for future research in online calibration.

2.2 Online Calibration Framework

In this section, we first present the state space model in more detail. Next, we discuss some recent developments and additional assumptions to improve the model. Finally, we comment on system identification, or the gradient estimation procedure for the state space model.

2.2.1 The State Space Model

As introduced in Section 2.1, the state space model (SSM) is a Markov model depicting the evolution of *state* variables and their relation to *observations*. A graphical illustration of SSM is given in Figure 2-1, where the white nodes represent *hidden states*, and shaded nodes denote *observations*.

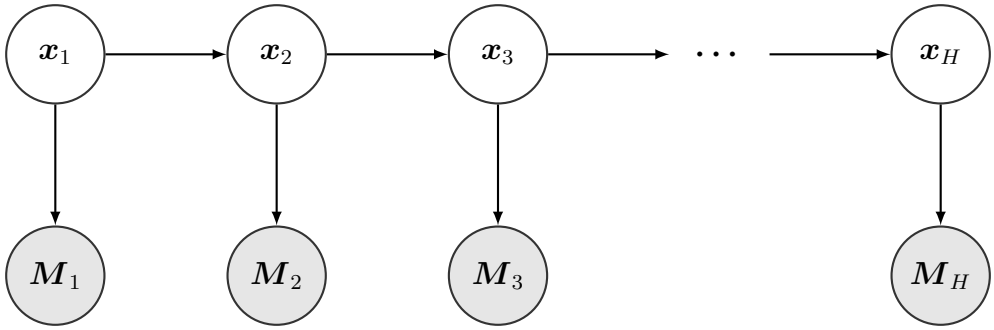


Figure 2-1: State space model (hidden Markov model)

Following the discussion of transition and measurement equations in Section 2.1, transition equations in SSMs are assumed as first order autoregressive (AR(1)) models

whereas measurement equations in the SSM are generic and capture all types of parameters (e.g., demand and supply). In the online calibration context for simulation-based DTA, the measurement model is the simulator that converts input parameters to simulated measurements. The mathematical formulation for the model is given by:

$$\boldsymbol{x}_h = \boldsymbol{f}_h(\boldsymbol{x}_{h-1}) + \boldsymbol{w}_h \tag{2.1}$$

$$\boldsymbol{M}_h = \boldsymbol{g}_h(\boldsymbol{x}_h) + \boldsymbol{v}_h \tag{2.2}$$

where, Equation (2.1) is the transition equation, and Equation (2.2) is the measurement equation. Additional notation is defined below.

- $\boldsymbol{f}_h(\cdot)$ and $\boldsymbol{g}_h(\cdot)$ are general functions that determine the transition and measurement relations
- h: discretized interval index, $h \in \mathcal{H} = \{1, 2, ..., H\}$, where $\mathcal{H}$ is the set of simulation intervals, where time is discretized into indices
- $\boldsymbol{x}_h$: states of time interval h
- $\boldsymbol{M}_h$: measurements/observations in the time interval h
- $\boldsymbol{w}_h, \boldsymbol{v}_h$ are random errors that are zero mean and independent of each other

We have two comments on the SSM. First, the SSM does not assume a functional form of equations, for the sake of generality. Thus, the transition $\boldsymbol{f}_h(\cdot)$ and measurement equations $\boldsymbol{g}_h(\cdot)$ are abstract functions that may have different forms for each interval h. Second, the transition and measurement equations only have dependencies on one state. The unique dependency is the Markovian assumption or memoryless assumption, which simplifies the model and makes state estimation easier.

The Idea of Deviations

The idea of deviations is a way to define the states and make use of time-dependent historical parameters as default values. Ashok & Ben-Akiva (1993) proposed this idea, and since then, it has been widely tested and applied (Ashok & Ben-Akiva,

2000; Bierlaire & Crittin, 2004; Antoniou, Ben-Akiva, & Koutsopoulos, 2007). The idea is simple: we obtain the deviations as new states by subtracting historicals from parameters. Thus, the transition equation only needs to capture the evolution of deviations. In comparison, when directly defining the parameters as states, it is difficult for an autoregressive (AR) process to account for the evolving trend for all cases throughout the modeling period. As an example, the OD flows in the "building" phase and the "fading" phase of peak hours are difficult to model with a same time-invariant AR process, simply because of distinct transition patterns. On the other hand, deviations as states may be easier to model, as the trend is already incorportated in the historicals. Thus, the deviations provide a simple way to incorporate temporal and spatial patterns in the DTA parameters. In the following notation, we define the state vector $\Delta \boldsymbol{x}_h$ and measurement $\Delta \boldsymbol{M}_h$ in deviations.

$$\Delta \boldsymbol{x}_h = \boldsymbol{x}_h - \boldsymbol{x}_h^H \tag{2.3}$$

$$\Delta \boldsymbol{M}_h = \boldsymbol{M}_h - \boldsymbol{g}_h(\boldsymbol{x}_h^H) \tag{2.4}$$

where, $\boldsymbol{g}_h(\boldsymbol{x}_h^H)$ represents the historical measurement values. Based on this definition of deviations, the transition equation Equation (2.1) and measurement equation Equation (2.2) now become:

$$\Delta \boldsymbol{x}_h = \boldsymbol{f}_h(\Delta \boldsymbol{x}_{h-1}) + \boldsymbol{w}_h \tag{2.5}$$

$$\Delta \boldsymbol{M}_h = \boldsymbol{g}_h(\boldsymbol{x}_h) - \boldsymbol{g}_h(\boldsymbol{x}_h^H) + \boldsymbol{v}_h \tag{2.6}$$

$$= \boldsymbol{g}'_h(\Delta \boldsymbol{x}_h) + \boldsymbol{v}_h \tag{2.7}$$

where, the $\boldsymbol{g}'(\cdot)$ is another general function to model the observation of the new state vector $\Delta \boldsymbol{x}_h$.

After subtracting the historical values, the deviations $\Delta \boldsymbol{x}_h$ and $\Delta \boldsymbol{M}_h$ can more reasonably be approximated with random variables of 0 mean, as they represent the day-to-day fluctuations around the historicals. Thus, the $\boldsymbol{w}_h, \boldsymbol{v}_h$ terms are more likely

to be 0 mean (Ashok & Ben-Akiva, 1993) and we can make the following assumptions:

$$\mathbb{E}[\boldsymbol{w}_h] = \mathbf{0}, \quad \forall h \in \mathcal{H} \tag{2.8}$$

$$\mathbb{E}[\boldsymbol{v}_h] = \mathbf{0}, \quad \forall h \in \mathcal{H} \tag{2.9}$$

$$\mathbb{E}[\boldsymbol{w}_h \boldsymbol{v}_h^T] = \mathbf{0}, \quad \forall h \in \mathcal{H} \tag{2.10}$$

$$\mathbb{E}[\boldsymbol{w}_h \boldsymbol{w}_h^T] = \boldsymbol{Q}_h, \quad \forall h \in \mathcal{H} \tag{2.11}$$

$$\mathbb{E}[\boldsymbol{v}_h \boldsymbol{v}_h^T] = \boldsymbol{R}_h, \quad \forall h \in \mathcal{H} \tag{2.12}$$

Further, as in Ashok & Ben-Akiva (1993), we assume that the error terms across different time steps are uncorrelated:

$$\mathbb{E}[\boldsymbol{w}_h \boldsymbol{w}_k^T] = \mathbf{0}, \quad \forall h, k \in \mathcal{H}, h \neq k \tag{2.13}$$

$$\mathbb{E}[\boldsymbol{v}_h \boldsymbol{v}_k^T] = \mathbf{0}, \quad \forall h, k \in \mathcal{H}, h \neq k \tag{2.14}$$

State Augmentation and Approximation

The SSM in Equations (2.1) and (2.2) satisfies the Markovian assumption, where $\boldsymbol{f}_h(\cdot), \boldsymbol{g}_h(\cdot)$ only have one state as arguments, implying that direct dependencies on more than one previous state is not possible. However, in DTA models and transportation systems, this assumption rarely holds. As an example, long trips on the network will still be captured by the surveillance system a few (e.g., q) intervals after they begin. This dependency naturally violates the Markovian assumption, because $\boldsymbol{g}_h(\boldsymbol{x}_h)$ should also have the previous states $\boldsymbol{x}_{h-1:h-q}$ as arguments. In this sense, the SSM is somewhat "myopic" in that it attempts to explain all the surveillance data $\boldsymbol{M}_h$ with the parameters $\boldsymbol{x}_h$ in the same interval. Thus, longer trips are likely to be ignored, and the state estimation is biased.

To extend the model, Ashok (1996) presented a technique called *state augmentation*. The technique creates *augmented states* in the form of a sliding window that include parameters in q intervals. The measurements are kept the same for each

interval. Thus, the model accounts for the missing relationships implicit in longer trips. The augmented state space model (augmented SSM) comprises the following equations:

- Transition equation

$$\boldsymbol{x}_h = \boldsymbol{f}_{h-1}(\boldsymbol{x}_{h-1}, ..., \boldsymbol{x}_{h-p}) + \boldsymbol{w}_h \tag{2.15}$$

- Measurement equation

$$\boldsymbol{M}_h = \boldsymbol{g}_h(\boldsymbol{x}_h, ..., \boldsymbol{x}_{h-q+1}) + \boldsymbol{v}_h \tag{2.16}$$

where, p is the number of previous states that are believed to have relations with $\boldsymbol{x}_h$; q is the number of states related to current measurement $\boldsymbol{M}_h$; $\boldsymbol{w}_h$ and $\boldsymbol{v}_h$ are error terms, which represent the transition and measurement errors. Note that p and q are not the same, and the resulting state should use whichever is greater as the number of intervals to include. Note that the parameter space for each time step is now at least q times greater since for each $\boldsymbol{M}_h$ we need to update $\boldsymbol{x}_h, ..., \boldsymbol{x}_{h-q+1}$.

The *state augmentation with approximation* essentially ignores the augmentation in the measurement equation, resulting in Equation (2.2), as proposed in Ashok & Ben-Akiva (2000). It assumes that $\boldsymbol{x}_h$ will be correctly estimated when $\boldsymbol{M}_h$ is first used. This assumption is a strong argument that still neglects the transportation system delay. Ashok & Ben-Akiva (2000) demonstrated that augmented SSM is not more beneficial than the approximation on a 32-kilometer expressway compared with SSM. It was also concluded that augmented SSM is not particularly useful when "most of the information" in the surveillance data is utilized the first time they are seen.

We comment on the augmented SSM for DTAs. First, the augmented SSM is also an SSM. It solves the modeling disadvantages of the original SSM by capturing the time delay in the transportation system. The augmented SSM theoretically should yield a more accurate state estimation and thus, is more likely to provide

reliable predictions. Second, it increases the parameter space in each time step. The computational complexity will increase q times if we compare Equation (2.16) with Equation (2.2). Finally, state augmentation with approximation demonstrated similar performance but at a significantly lower computational cost on an expressway stretch. However, large-scale networks presumably have numerous long trips with high travel times, especially in congestion where they are delayed.

Linearization of State Space Models

We have presented the SSM and augmented SSM with the state augmentation technique but the specific functional form of $\boldsymbol{f}(\cdot)$ and $\boldsymbol{g}(\cdot)$ remains to be identified. In this thesis, they are approximated with linear functions, which follows the logic of linearization in the extended Kalman filter (EKF). While more complex nonlinear models exist, the EKF is a well-studied and effective solution approach that relies on a linear SSM for the nonlinear case. More complex models include approaches like the unscented Kalman filter (UKF) and particle filter (PF). Studies have reported that UKF did not result in a significant difference from EKF in traffic predictions (St-Pierre & Gingras, 2004). PF and UKF have been reported to be more time-consuming than EKF (Hegyi et al., 2006, 2007). In view of these considerations, we assume a linear relationship for the SSM.

$\boldsymbol{f}_h$ and $\boldsymbol{g}_h$ for the linear augmented SSM are given in the following equations. Note that the original SSM is a special case for $p = 1, q = 1$:

$$\Delta \boldsymbol{x}_h = \sum_{k=h-1}^{h-p} \boldsymbol{F}_h^k \Delta \boldsymbol{x}_k + \boldsymbol{w}_h \tag{2.17}$$

$$\Delta \boldsymbol{M}_h = \sum_{k=h}^{h-q+1} \boldsymbol{H}_h^k \Delta \boldsymbol{x}_k + \boldsymbol{v}_h \tag{2.18}$$

where, $\boldsymbol{F}_h^k$ is a square matrix, representing the effect of $\Delta \boldsymbol{x}_k$ on $\Delta \boldsymbol{x}_h$; If the autoregressive transition model holds throughout the period, p matrices $(\boldsymbol{F}_1, ..., \boldsymbol{F}_p)$ completely determine the transition equation for all time intervals. In practice, we often assume this due to model parsimony. $\boldsymbol{H}_h^k$ is a gradient approximation of the

simulator that describes the impact of $\Delta \boldsymbol{x}_k$ on $\Delta \boldsymbol{M}_h$.

We make some comments on the computational tractability. The dimensionality is $n_x \times n_x$ for $\boldsymbol{F}_h^k$. In practice, a diagonal matrix may be assumed for $\boldsymbol{F}_h^k$, because of the difficulty in estimating the complete matrix in practice. The dimension of $\boldsymbol{H}_h^k$ is $n_M \times n_x$, and a typical gradient estimation procedure based on finite difference will need $O(n_x)$ runs for simulation-based DTA. We will discuss the details of gradient estimation in Section 2.2.2.

We make some critical comments on the linearization procedure. First, an autoregressive (AR) model represents the transition equation. Higher order AR models are beneficial for the accuracy of transition equations, but requires offline estimation. Second, the gradient estimation procedure is necessary for $\boldsymbol{H}_h^k$, and the computational complexity can be a significant issue as the number of parameters n_x and augmentation degree q increases.

We conclude this section with three remarks. First, the state transition model is presented with a generic function $\boldsymbol{f}_h(\cdot)$ and approximated with linear models. Second, the idea of deviations is an elegant framework that utilizes the temporal trend in historicals. Thus, it should be applied when historical values are available. Finally, the augmented SSM is beneficial in capturing long trips, but the computational complexity restricts its application. Hence an approximation is usually employed in practice.

2.2.2 System Identification

Although simulation-based DTA can efficiently model complex traffic interactions, a critical issue is the lack of an analytical formulation. We cannot derive it in a closed form, because of the complexity in simulating demand, supply and their interactions. Unfortunately, online calibration algorithms typically require a mathematical model for the DTA system, most notably in the application of the measurement equation. The model-building procedure is called *system identification*, in which $\boldsymbol{g}_h(\cdot)$ is treated as a "black box" between $\boldsymbol{x}_h$ and $\boldsymbol{M}_h$, and we attempt to find a mathematical model that describes it based on function evaluations of $\boldsymbol{g}_h(\cdot)$. Following the discussion

in the linear SSM model, we need to estimate all $\boldsymbol{H}_h^k$s for the completeness of the measurement equation.

Gradient Estimation

Under the linear assumption in the SSM model, the system identification task is to determine the gradient/Jacobian matrix $\boldsymbol{H}_h^k$ in Equation (2.18). The $\boldsymbol{H}_h^k$ matrix needs to be determined in each time step h because it changes with network state. Since no closed form is available, we have to rely on simulations of $\boldsymbol{g}_h(\cdot)$. In this thesis, the *gradient estimation* task for simulation-based DTA is defined as approximating the gradient $\boldsymbol{H}_h^k$ (a.k.a. Jacobian, H matrix) with function evaluations of the simulator (i.e., $\boldsymbol{g}_h(\cdot)$).

Although the calibration of DTA models has been studied for more than 20 years, the gradient estimation task for simulation-based DTA has received relatively less attention. In the literature, the most widely used approach is the *finite difference* method which has a computational complexity of order $O(n)$, where n is the number of parameters. A more computationally efficient approach is *simultaneous perturbation* which, unfortunately, yields inaccurate gradient estimates and hence, adversely impacts calibration accuracy. There have been efforts using a fixed Kalman gain as presented in Antoniou (2004) that gives immediate results, but we still need to update the system gradient for each interval to consider changes in traffic conditions.

In the following paragraphs, we present the specific methods for gradient estimation.

Finite Difference

The finite difference (FD) is a widely applied numerical method to calculate the gradient. Assuming $\boldsymbol{g}_h(\cdot)$ is the measurement vector of dimension m and $\boldsymbol{x}_h$ is the parameter vector of dimension n, the gradient is a matrix of dimension $(m \times n)$. Then, the gradient matrix shown in Equation (2.36) can be calculated by FD in Equations (2.21) and (2.22). Here we add another subscript i for $\boldsymbol{g}_{h,i}$ and j for $\boldsymbol{x}_{h,j}$ to denote the ith and jth element of each vector, where $i = 1, 2, ..., m$ and $j = 1, 2, ..., n$.

$$\boldsymbol{H}_h^k = \begin{bmatrix} \frac{\partial}{\partial x_{k,1}} g_{h,1} & \cdots & \frac{\partial}{\partial x_{k,n}} g_{h,1} \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x_{k,1}} g_{h,m} & \cdots & \frac{\partial}{\partial x_{k,n}} g_{h,m} \end{bmatrix} \tag{2.19}$$

$$= \begin{bmatrix} \frac{\partial}{\partial x_{k,1}} \boldsymbol{g}_h & \cdots & \frac{\partial}{\partial x_{k,n}} \boldsymbol{g}_h \end{bmatrix} \tag{2.20}$$

$$\text{where, } \frac{\partial}{\partial x_{k,j}} \boldsymbol{g}_h = \frac{\boldsymbol{g}_h(\boldsymbol{x}_k + \boldsymbol{\delta}_j) - \boldsymbol{g}_h(\boldsymbol{x}_k - \boldsymbol{\delta}_j)}{2\delta_j} \tag{2.21}$$

$$\boldsymbol{\delta}_j = [0, 0, ..., \delta_j, ..., 0]^\top \tag{2.22}$$

Note $\frac{\partial}{\partial x_{k,j}} g_{h,i}$ denotes the partial derivative element of $g_{h,i}$ to $x_{k,j}$. The method shown in Equation (2.21) is the *central finite difference*. $\boldsymbol{\delta}_j$ is the *perturbation vector*, and indicates that the vector $\boldsymbol{x}_h$ is perturbed at the jth element with size δ_j; Equation (2.21) approximates the ith column of the $\boldsymbol{H}_h^k$ matrix. In simulation-based DTA, the simulation run substitutes $\boldsymbol{g}_h$. Observe that with two evaluations of $\boldsymbol{g}(\cdot)$, we obtain the gradient for $\boldsymbol{x}_{k,j}$. Thus, the central FD needs $2n$ calculations for $\boldsymbol{H}_h^k$, with each $\boldsymbol{g}(\cdot)$ as one basic operation. Notice that the unit of complexity is a single run of simulation. Depending on the network size and number of simulated vehicles, the time needed for one run can be very different.

Simultaneous Perturbation

Another method to calculate the $\boldsymbol{H}_h^k$ is called simultaneous perturbation (SP). It originates from the idea of SPSA (Spall, 1992). Instead of perturbing the vector $\boldsymbol{x}_h$ in each dimension, SP perturbs all dimensions at the same time. Following the same representation as FD, SP for each column of the gradient is given by:

$$\frac{\partial}{\partial x_{k,j}} \boldsymbol{g}_h = \frac{\boldsymbol{g}_h(\boldsymbol{x}_k + \boldsymbol{\delta}) - \boldsymbol{g}_h(\boldsymbol{x}_k - \boldsymbol{\delta})}{2\delta_j} \tag{2.23}$$

$$\boldsymbol{\delta} = [\delta_1, \delta_2, ..., \delta_j, ..., \delta_n]^\top \tag{2.24}$$

The perturbation vector $\boldsymbol{\delta}$ has a different size δ_j for each dimension. δ_js also have different signs, because each dimension is randomly perturbed in either the positive or negative direction. Note that all the columns in $\boldsymbol{H}_h^k$ have the same numerator in Equation (2.23), so we only need twice the evaluation of $\boldsymbol{g}_h(\cdot)$. Thus, to obtain an approximate of $\boldsymbol{H}_h^k$, we only need one calculation. However, since all columns have the same numerator vector, they are linearly dependent. Thus, the rank of $\boldsymbol{H}_h^k$ is 1, which will be uninformative for each iteration when the parameter space is large.

2.3 Solution Approaches

The SSM has been comprehensively studied in the literature, and algorithms in the Kalman filter family can estimate the state vector efficiently. When the transition and measurement equations are linear, Kalman filters are proven to be the optimal linear state estimator with the objective of minimizing the mean squared error (MSE) for each time step (Ashok, 1996). Under the condition of Gaussian errors, it minimizes the MSE among all (linear or nonlinear) estimators. Under the case that measurement model is a linear approximation of nonlinear DTA models, Kalman filter is called the extended Kalman filter (EKF). While it is difficult to guarantee the optimality for the EKF, the method is computationally tractable (with polynomial complexity) and useful in many practical applications (Antoniou, 2004; St-Pierre & Gingras, 2004; Hegyi et al., 2006).

There have been several Kalman filter variants applied to solve the state-space formulation in the context of online calibration. Here the extended Kalman filter algorithm is reviewed first and its connection to the state-space model is made explicit. Then its variants are summarized and commented upon. Last but not least, the drawbacks of current practices of EKF are addressed and this leads to the next section.

2.3.1 Extended Kalman Filter

Without loss of generality for augmented SSM and the definition of deviations, the basic equations are:

- Transition equation

$$\boldsymbol{X}_h = \boldsymbol{f}_{h-1}\left(\boldsymbol{X}_{h-1}\right) + \boldsymbol{W}_h \tag{2.25}$$

which is linearized by:

$$\boldsymbol{X}_h = \boldsymbol{\Phi}_{h-1}\boldsymbol{X}_{h-1} + \boldsymbol{W}_h \tag{2.26}$$

- Measurement equation

$$\boldsymbol{M}_h = \boldsymbol{g}_h\left(\boldsymbol{X}_h\right) + \boldsymbol{V}_h \tag{2.27}$$

which is linearized by:

$$\boldsymbol{M}_h = \boldsymbol{\Theta}_h\boldsymbol{X}_h + \boldsymbol{V}_h \tag{2.28}$$

where, Equations (2.26) and (2.28) are the state-space formulations that can be extended to augmented states with the deviation definition, using the following notation:

$$\boldsymbol{X}_h = \left[\boldsymbol{x}_h^\top, \boldsymbol{x}_{h-1}^\top, ..., \boldsymbol{x}_{h-r+1}^\top\right]^\top \tag{2.29}$$

$$\boldsymbol{\Theta}_h = \left[\boldsymbol{H}_h^h, \boldsymbol{H}_h^{h-1}, ..., \boldsymbol{H}_h^{h-r+1}\right] \tag{2.30}$$

$$\boldsymbol{\Phi}_h = \begin{bmatrix} \boldsymbol{F}_h^{h-1} & \boldsymbol{F}_h^{h-2} & \cdots & \boldsymbol{F}_h^{h-r} \\ \boldsymbol{I}_{(r-1)n\times(r-1)n} & & \boldsymbol{0}_{(r-1)n\times n} & \end{bmatrix} \tag{2.31}$$

where, the *degree of augmentation* is $r = \max\{p, q\}$ and n is the number of DTA parameters for each interval.

$\boldsymbol{W}_h$ and $\boldsymbol{V}_h$ are uncorrelated continuous variables. We usually assume them as zero mean and multivariate Gaussian with covariance matrix $\boldsymbol{Q}_h$ and $\boldsymbol{R}_h$, respectively. Compared with the assumptions in the state space model, the EKF further assumes the Gaussian distribution, which is necessary for closed-form state estimators. Although it may be difficult to prove, the Gaussian assumption is widely used in

inference tasks for continuous variables, because it is the only assumption that yields a closed form solution.

The solution algorithm of EKF is displayed below.

Algorithm 1 Extended Kalman Filter

Initialize

$$\hat{\boldsymbol{X}}_{0|0} = \boldsymbol{X}_0 \tag{2.32}$$

$$\boldsymbol{P}_{0|0} = \boldsymbol{P}_0 \tag{2.33}$$

for $h = 1$ to H **do**

Time Update

Predicted state estimate

$$\hat{\boldsymbol{X}}_{h|h-1} = \boldsymbol{\Phi}_{h-1}\hat{\boldsymbol{X}}_{h-1|h-1} \tag{2.34}$$

Predicted covariance estimate

$$\boldsymbol{P}_{h|h-1} = \boldsymbol{\Phi}_{h-1}\boldsymbol{P}_{h-1|h-1}\boldsymbol{\Phi}_{h-1}^{\top} + \boldsymbol{Q}_h \tag{2.35}$$

Measurement Update

INPUT: real-time measurement $\boldsymbol{M}_h$

Measurement equation linearization

$$\boldsymbol{\Theta}_h = \left.\frac{\partial \boldsymbol{g}_h}{\partial \boldsymbol{X}}\right|_{\hat{\boldsymbol{X}}_{h|h-1}} \tag{2.36}$$

Near-optimal Kalman gain

$$\boldsymbol{K}_h = \boldsymbol{P}_{h|h-1}\boldsymbol{\Theta}_h^{\top}\left(\boldsymbol{\Theta}_h\boldsymbol{P}_{h|h-1}\boldsymbol{\Theta}_h^{\top} + \boldsymbol{R}_h\right)^{-1} \tag{2.37}$$

Updated state estimate

$$\hat{\boldsymbol{X}}_{h|h} = \hat{\boldsymbol{X}}_{h|h-1} + \boldsymbol{K}_h\left(\boldsymbol{M}_h - \boldsymbol{g}_h(\hat{\boldsymbol{X}}_{h|h-1})\right) \tag{2.38}$$

Updated covariance estimate

$$\boldsymbol{P}_{h|h} = \boldsymbol{P}_{h|h-1} - \boldsymbol{K}_h\boldsymbol{\Theta}_h\boldsymbol{P}_{h|h-1} \tag{2.39}$$

OUTPUT: posterior estimates $\hat{\boldsymbol{x}}_{h|h}$ and $\boldsymbol{P}_{h|h}$

end for

Note that the notation $\hat{\boldsymbol{X}}_{h|h-1}$ and $\hat{\boldsymbol{X}}_{h|h}$ indicates the estimate of random vector $\boldsymbol{X}_h$ before and after seeing the surveillance data at h, given we are currently at

$h-1$. Similarly, $\boldsymbol{P}_{h|h-1}$ and $\boldsymbol{P}_{h|h}$ are the corresponding estimate of $\boldsymbol{P}_h$. For the EKF algorithm, the input parameters are:

- $\boldsymbol{X}_0$: initial starting point (guess) of the state vector at time $h = 0$
- $\boldsymbol{P}_0$: initial covariance matrix (guess) of $\boldsymbol{X}_0$
- $\boldsymbol{Q}_h$: time-variant covariance matrix of $\boldsymbol{w}_h$, $h \in \mathcal{H}$
- $\boldsymbol{R}_h$: time-variant covariance matrix of $\boldsymbol{v}_h$, $h \in \mathcal{H}$

We briefly summarize the steps of the EKF algorithm. Assume that the initial state estimate $\hat{\boldsymbol{X}}_{0|0}$ and the covariance matrix estimate $\hat{\boldsymbol{P}}_{0|0}$ are available according to Equations (2.32) and (2.33). The *time update* provides the prediction (i.e., prior estimate) of the state $\hat{\boldsymbol{X}}_{h|h-1}$ and its covariance matrix $\boldsymbol{P}_{h|h-1}$ for the next time step (Equations (2.34) and (2.35)). Subscript $h|h-1$ indicates that we observe measurements at time $h-1$ and we predict for time h. When new measurements $\boldsymbol{M}_h$ are available, the *measurement update* utilize them to update the predictions in Equations (2.38) and (2.39), which yields posterior state estimate $\hat{\boldsymbol{X}}_{h|h}$ and its covariance estimate $\boldsymbol{P}_{h|h}$. Observe that the EKF algorithm only involves matrix operations except Equation (2.36). Thus, excluding the complexity of gradient estimation, EKF is a polynomial time algorithm that handles real-time measurements.

For the gradient estimation procedure in Equation (2.36), either the FD or SP can be applied. We name the resulting EKF FD-EKF and SP-EKF, respectively. Compared with the FD-EKF, SP-EKF improves the computational time, but the approximated gradient matrix will be inaccurate, as discussed in previous sections. Due to this characteristic and given that our aim is to obtain accurate parameter estimates, this thesis bases upon FD-EKF, to obtain the most accurate gradient estimation.

2.3.2 Constrained Kalman Filter

A recent extension of the Kalman filtering framework to model constraints on state variables is the constrained extended Kalman filter (CEKF) introduced in H. Zhang

(2016) and H. Zhang et al. (2017). In Kalman filters, the variables are assumed to be unconstrained Gaussian. Thus, it is possible that EKF yields unreasonable parameters. For example, OD flows should never be negative, and speed-density parameters should not be negative in the fundamental diagram. The authors explicitly modeled the constraints on state vectors through a post-filtering quadratic optimization in the Kalman filtering framework.

Based on the Gaussian assumption of random errors, the state variables also follow Gaussian distributions. For the unconstrained state variables, the Kalman filter estimate $\hat{\boldsymbol{X}}_{h|h}$ is the mean of the distribution that the state vector $\boldsymbol{x}_h$ follows. The posterior covariance matrix $\boldsymbol{P}_{h|h}$ indicates the covariance of $\boldsymbol{X}_h$. Thus, $\boldsymbol{x}_h \sim \mathcal{N}(\hat{\boldsymbol{X}}_{h|h}, \boldsymbol{P}_{h|h})$, with the probability density function of:

$$f_X(\boldsymbol{X}) = \frac{1}{\sqrt{(2\pi)^n |\boldsymbol{P}_{h|h}|}} \exp\left\{-\frac{1}{2}(\boldsymbol{X} - \hat{\boldsymbol{X}}_{h|h})^\top {\boldsymbol{P}_{h|h}}^{-1}(\boldsymbol{X} - \hat{\boldsymbol{X}}_{h|h})\right\} \tag{2.40}$$

where, n is the dimension of vector $\boldsymbol{X}$.

Now we consider the constraints. $\boldsymbol{X}_{h|h}$ follows the Gaussian distribution and is subject to constraints. Thus, the optimal solution is not $\hat{\boldsymbol{X}}_{h|h}$ if any violation of the constraints occurs. Under the objective of the maximum *a posteriori* (MAP) probability density, the optimal solution is given by:

$$\max_{\boldsymbol{X}} f_X(\boldsymbol{X}) \Leftrightarrow \min_{\boldsymbol{X}} (\boldsymbol{X} - \hat{\boldsymbol{X}}_{h|h})^\top \boldsymbol{P}_{h|h}^{-1} (\boldsymbol{X} - \hat{\boldsymbol{X}}_{h|h}) \tag{2.41}$$

$$\text{s.t. } \boldsymbol{D}\boldsymbol{X} \leq \boldsymbol{d} \tag{2.42}$$

where, $\boldsymbol{D}$ is a known $s \times n$ constant matrix, s is the number of constraints, n is the dimension of the state vector, and $s \leq n$.

The authors also compared this approach with the "truncation" approach that sets the invalid elements to the bounding condition. The truncation neglects the correlation between elements in the state vector, while the proposed approach utilizes

the correlation during the optimization.

As demonstrated on the Singapore Expressway network in H. Zhang et al. (2017), the CEKF significantly outperforms the EKF, where the EKF tends to overestimate the demand, due to the truncation process that neglects state covariance.

As a conclusion, in our online DTA calibration, we should apply the constrained Kalman filter technique whenever possible.

2.4 Summary

In this chapter, we reviewed the recent developments in online calibration for DTA. We close with several significant comments. First, extensive research has demonstrated the state space model/Kalman filtering framework as a powerful tool. Recent developments in state definition, state augmentation and modeling constraints have enriched its applicability to various situations. Second, although simulation-based DTA has superior fidelity, the majority of research on DTA calibration is based on analytical DTA, while the simulator unpredictability is less discussed. Third, although the Kalman filter has been successfully applied in many different DTA calibration contexts, the computational performance of gradient estimation is a bottleneck for real-time deployment. Lastly, related to the previous point, the scalability of online calibration on large-scale networks remains to be realized. In the following chapters, we attempt to advance the state-of-the-art in simulation-based DTA towards large-scale and real-time performance.

Chapter 3

Supply Calibration Considering Simulation Stochasticity

The supply module is a key component of simulation-based Dynamic Traffic Assignment (DTA) systems. Supply parameters in mesoscopic traffic simulators typically include traffic dynamics or fundamental diagram parameters and segment capacities whereas in microscopic simulators they include car-following and lane-changing model parameters. These parameters of the supply simulator primarily describe vehicle movement and queue formation/dissipation, and in conjunction with the demand simulation module, generate traffic measurements such as flow counts, average speeds and link travel times. Consequently, the supply parameters are crucial in accurately modeling traffic conditions, particularly congestion. Although in general, supply parameters are static given that they depend on road segment characteristics that seldom change, urban transportation networks are in fact frequently subject to non-recurrent supply changes due to incidents and weather conditions which necessitate online updates of the parameters. Incidents or road constructions will lead to lane closures, which significantly affects segment throughput. Weather conditions may decrease visibility and traction, in which cases people drive more carefully, leading to reduced speed measurements and increased headways.

The necessity of the supply calibration to be *online* lies in the fact that these parameters usually change in an unpredictable manner. Incidents are typically un-

predictable, in the sense of the time of occurrence, duration, severity, number of lanes affected. Weather can be predicted, but the accurate duration and impact on traffic systems are hard to quantify beforehand. While scheduled road constructions are mostly predictable, the actual execution could still differ from the schedule for various reasons. Hence, the most reliable and straightforward source to monitor supply is still the real-time surveillance system. The online supply calibration should infer the underlying parameter changes from real-time surveillance data, and readily evaluate if the supply changes fit the data. For real-time deployment, this online process needs to be executed for each interval to reflect the supply changes instantly.

Crucial as the supply calibration is, it is also challenging. The challenges of OD estimation such as nonlinearity, stochasticity and time-delay also apply to supply calibration. However, there is a subtle difference. A linear function will reasonably approximate the relation between OD and sensor flows, as the fraction of OD flows contributing to sensors changes slowly (Ashok & Ben-Akiva, 2000). On the contrary, the relationship between supply parameters and measurements is nonlinear. While linear relations are mostly employed to approximate the nonlinearity, the approximation exacerbates the uncertainty and stochasticity. In this regard, the calibration procedure should handle the stochasticity carefully to accurately quantify the supply changes.

In this chapter, we focus on the role of simulation stochasticity in online supply calibration. Stochasticity leads to uncertainty in the simulated measurements, which is an important consideration when fitting the simulation to real-world observations. The chapter is organized as follows. We first discuss related literature on supply calibration not covered in Chapter 2, and motivate the analysis of simulation stochasticity. Then, we attempt to quantify the stochasticity, followed by an error analysis in the Kalman filtering framework. Lastly, we present two methods to reduce the effect of simulation stochasticity and demonstrate their performance with a synthetic case study.

3.1 Literature Review and Problem Definition

We first recall the general calibration problem definition: given traffic surveillance observations from the real world, adjust the DTA model parameters such that the discrepancy between the real-world observations and the simulated measurements is minimized. In this section, we summarize existing work on online supply calibration, and identify gaps in the literature pertaining to simulation-based DTA systems. Finally, we draw attention to the issue of simulation stochasticity.

3.1.1 Literature of Online Supply Calibration

In the context of online supply calibration, moderate research has been conducted for DTA systems. As noted previously, the online calibration problem involves two key tasks: (1) system identification that specifies the mathematical model of the DTA system; and (2) application of a suitable algorithm to calibrate parameters that utilizes the mathematical formulation. Task (1) is clearly a prerequisite for (2) and given that (2) has already been extensively discussed in Chapter 2, this review focuses on (1) in the specific context of online supply calibration.

The supply module in various DTA systems utilizes either analytical formulations or simulation. The key difference lies in whether there is a closed-form relation between parameters and measurements. In the following sub-sections, we review the existing research, with a focus on *system identification* for both analytical and simulation-based DTAs.

Analytical DTA

For analytical DTAs, typically, close-form relations exist between the model parameters and measurements. A well-known example is the Cell Transmission Model (CTM) which employs a macroscopic supply model that captures traffic dynamics with nonlinear differential equations (Daganzo, 1995). Since the model has a closed form, the analytical relations or numerical solutions can be obtained quickly. Thus, with the explicit relations, we can formulate the model using the state space frame-

work. Wang & Papageorgiou (2005) employed a random walk model as the transition equation and deduced the partial derivatives for the measurement equation. The authors applied the model on a freeway stretch using the extended Kalman filter. The estimation results were satisfactory for the segment and boundary variables, and the time-dependent measurements were well-fitted. The case study demonstrated the EKF's capability of tracking traffic states under various traffic conditions.

Simulation-Based DTA

On the other hand, it is generally harder to obtain analytical relations for simulation-based models. No closed-form relationship is available due to random sampling and the complex demand-supply interactions in the simulation models. When modeling the analytical relationship, previous research has employed either *a priori knowledge* or *an approximation procedure*.

First, examples of utilizing *prior knowledge* include transfer function models, which are bivariate linear models between traffic flow speeds and densities for each segment (Tavana & Mahmassani, 2000). The coefficient parameters are estimated offline to match the real scenario. Huynh et al. (2002) extended the work of Tavana & Mahmassani (2000) and applied an adaptive process to the transfer function where the parameters are updated online. The authors also proposed a nonlinear least squares optimization formulation for the update and concluded that an adaptive process for the transfer function is beneficial, either with or without the nonlinear optimization. However, in the simulation, the authors modified the supply module by replacing the Greenshields model with the simpler transfer function. This replacement is based on the assumption that the transfer function model is a good approximation of the Greenshields model which may not always hold, thus affecting predictive power of the model relative to the original simulation-based model.

The second approach, the *approximation procedure* for system identification involves building models from data using statistical methods. The procedure usually involves fitting a parametric model and when a linear model is assumed, the procedure is *gradient estimation*. The unknown parameters are the gradient matrix

or Jacobian that describes the system based on a first order approximation of the relationship between measurements and parameters. The finite difference method for gradient estimation which was described in the previous chapter is a straightforward yet widely-used numerical method to compute the gradient. Each parameter is perturbed independently and the resulting change in target variables quantifies the impact on measurements. This yields one column in the Jacobian matrix. Antoniou (2004) applied the Extended Kalman Filter for the supply calibration problem using a finite difference method for gradient computation. The EKF was able to accurately predict speeds on a corridor network under sunny and rainy weather conditions.

In summary, online supply calibration has received relatively less attention in the literature. The most widely used method is still the Extended Kalman Filter using the finite difference method for gradient estimation (system identification) which has successfully been applied to small networks. In this thesis, we adopt a similar EKF based approach utilizing the finite difference method for system identification. Moreover, an issue which has not been addressed in the literature is that of simulator stochasticity and its impact on both gradient estimation and the Kalman Filter model. We discuss this in more detail in the following section.

3.1.2 Motivation for Quantifying Simulation Stochasticity

Stochasticity has not been systematically addressed in the context of online calibration of simulation-based DTA systems. A common approach to address stochasticity is to average the results from multiple runs or "replications" of the simulation. While averaging more simulations effectively reduces noise, the computational burden may be unacceptable for online applications as the network scale grows. Furthermore, the impacts of stochasticity may be more severe in large complex networks, thus requiring a higher number of replications which may not be computationally feasible given the time constraints in online applications. Thus, quantifying the stochasticity of simulations offline is crucial because it represents the confidence of each simulated result and hence, plays an important role in term of minimizing the discrepancy between simulation and real-world observations.

A simple method to quantify stochasticity is through variance-covariance matrices. In the Kalman filtering framework for online DTA calibration, the covariance matrices for the transition equation ($\boldsymbol{Q}$) and measurement equation ($\boldsymbol{R}$) control the confidence of each model. Finding suitable values for them is called *filter tuning*. It is generally known that the Kalman filter is highly sensitive to them. However as of now, there is no simple guideline to identify the "correct" error covariance matrices. While guidelines exist for preventing divergence of Kalman filters (Schneider & Georgakis, 2013), in many applications, Kalman filters still need manual tuning by trial and error. This is primarily because there is no mature adaptive filtering method that simply works for every field of application (Ananthasayanam et al., 2016) leading to numerous ad hoc settings for the filter and difficulty in guaranteeing performance. Within the field of DTA calibration, filter tuning has also received less attention and $\boldsymbol{R}$ matrices are usually assumed to be time-invariant for simplicity. By quantifying the simulation stochasticity, we aim to provide a more systematic characterization of the covariance matrix $\boldsymbol{R}$ for Kalman filters.

Furthermore, the gradient estimation procedure is also greatly impacted by simulation stochasticity, but usually ignored. This is because the finite difference approach does not specifically consider the error in function evaluations, thus leading to another unaccounted source of error that should be incorporated in $\boldsymbol{R}$ in the measurement equation.

Based on the aforementioned motivating factors, it is necessary to analyze the error caused by simulation stochasticity. Considering this, we may better understand the error covariance matrices, and potentially give some guidance on Kalman filter tuning for online calibration of DTA systems.

3.1.3 Supply Calibration Problem Definition Considering Simulation Stochasticity

We now restate the supply calibration problem under simulation stochasticity. Given traffic surveillance observations from the real world, adjust the DTA supply param-

eters such that the discrepancy between the observations and the *expectation* of the simulation is minimized considering simulation stochasticity. Specifically, we consider simulation stochasticity in two steps of the EKF algorithm: (1) gradient estimation when performing finite difference; (2) simulated measurement error when calculating the Kalman innovation (prediction residual).

We close this section with the following comments. First, the stochasticity of simulation determines our confidence in the simulated measurements, which is crucial when minimizing the discrepancy against real observations. Second, this stochasticity has not been extensively studied in the DTA calibration literature, and the impact may be underestimated. Lastly, quantifying the stochasticity will also improve estimates of the error covariance in the Kalman filtering framework. This may lead to better calibration performance. In the remainder of this chapter, we first quantify simulation stochasticity, then conduct an error analysis to shed some light on error covariances and finally propose some remedies to reduce its impact on calibration for simulation-based DTA.

3.2 Quantifying Simulation Stochasticity

The source of simulation stochasticity is the extensive use of random number generators to mimic stochastic processes. First and foremost, some facts about random generators are helpful to understand. The generator can produce a sequence of pseudo random numbers, which appear to be samples drawn from a certain distribution. However, the sequence is in fact deterministic, because computers can perform deterministic operations efficiently. The generator is implemented such that an initial random seed controls which predetermined sequence to produce. To make a pseudo random generator a good approximation of a true random one, the seeds are usually selected from a function of true random events like current date and time, or the amount of time between keyboard strokes.

In simulation-based DTA, there is usually an initial seed for the random generator to start with. The generated random numbers are used for various operations:

sampling Poisson process for vehicle departure times, sampling route choice decisions with random utility models for pre-trip, en-route choices, etc. Thus, the random numbers affect the spatial-temporal patterns of traffic, which in turn affect the simulated measurements.

3.2.1 Experimental Procedure

To quantify the variation of simulated measurements caused by the use of random numbers, we conduct an experiment with the following steps:

1. Draw random numbers to comprise a seed pool;
2. For each seed in the pool, run a simulation with the selected seed while keep the same demand and supply parameters;
3. Compare the difference of simulated measurements and calculate variance-covariance matrix across seeds.

DTA System & Road Network

We select DynaMIT, a state-of-the-art mesoscopic traffic simulation as our DTA system. In DynaMIT, each segment has 7 supply parameters to describe the modified Greenshields model: free flow speed V_f, jam density K_{jam}, alpha α, beta β, segment capacity c, minimum speed V_{min}, minimum density K_{min}. Thus, the speed-density relationship is dictated by the modified Greenshields model:

$$V = \begin{cases} V_f & , K \leq K_{min} \\ \max\left\{V_{min}, V_f\left[1 - \left(\frac{K-K_{min}}{K_{jam}}\right)^{\beta}\right]^{\alpha}\right\} & , K > K_{min} \end{cases} \tag{3.1}$$

The segment capacity c controls the number of vehicles that can leave the segment in a unit time interval.

For a proof of concept, the simulations are based on a synthetic network with 2 OD pairs and 8 segments. The network topology is given in Figure 3-1. Each segment

has a sensor that can capture the mean speed and aggregate flow for each simulation or estimation interval. Basic information about segment lengths, free flow speeds and free flow travel time is presented in Table 3.1. On the demand side, Table 3.2 presents the mainstream and off-ramp OD flow statistics for the whole simulation period 14:00-19:00.

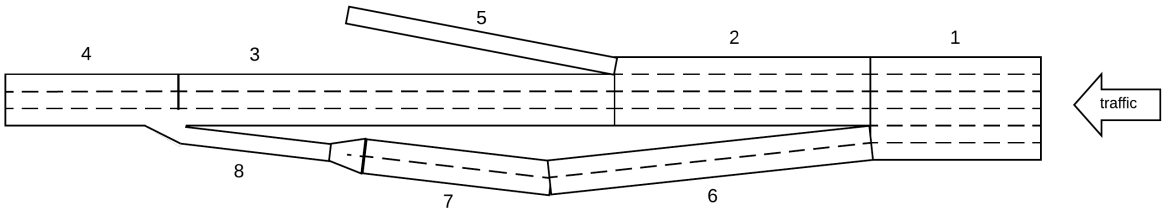


Figure 3-1: Toy road network, traffic going to left

Table 3.1: Specifications of each segment on the toy network

Segment ID	1	2	3	4	5	6	7	8
Length (meter)	297.5	553.8	493.1	351.2	408.6	666.7	377.3	183.0
Free flow speed (mph)	71	66	75	75	60	70	70	60
Minimum travel time (second)	9.37	18.8	14.7	10.5	15.2	21.3	12.1	6.82

Table 3.2: Demand statistics for simulation period 14:00-19:00

OD pair	OD flows at percentile (veh/hour)					Mean OD flow (veh/hour)
	10%	25%	50%	75%	90%	
Mainstream	3670	3882	4086	4446	4940	4220
Off-ramp	0	168	336	480	708	350

3.2.2 Stochasticity Measures

RMSNs on Simulations with Different Seeds

For the proof of concept, we simulate 5 hours of traffic with 6 seeds treating the simulation result with the first seed as the benchmark, and calculate the time-dependent Root Mean Square Normalized Error (RMSN) for the result from each seed against the benchmark. RMSN is defined in Equation (3.2), where y_t is the true value:

$$\text{RMSN} = \sqrt{\frac{\sum_{t=1}^{T}(\hat{y}_t - y_t)^2}{n}} \Bigg/ \frac{\sum_{t=1}^{T} y_t}{n} \tag{3.2}$$

First, we investigate the impact of the simulation interval length (or horizon length). The length determines how frequently the sensors report measurements. In our experiment, two simulation intervals are selected: 5 minutes and 15 minutes. We would expect the 15-minute interval to have less stochasticity since the measurements are averaged over a longer interval. Table 3.3 presents the RMSNs compared to result from Seed 1.

Table 3.3: RMSNs compared to seed 1 for simulated measurements

Interval	Measurement	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Average
5 minutes	Flow volume	0	5.45%	5.28%	5.33%	4.81%	4.74%	5.12%
	Sensor speed	0	21.9%	18.6%	20.8%	19.4%	20.9%	20.3%
	Link travel time	0	21.6%	23.7%	23.1%	21.5%	22.1%	22.4%
15 minutes	Flow volume	0	3.04%	3.42%	3.20%	4.07%	3.33%	3.41%
	Sensor speed	0	9.69%	12.7%	11.1%	10.1%	10.6%	10.8%
	Link travel time	0	13.4%	14.5%	15.0%	15.6%	14.9%	14.7%

As we expected, the stochasticity for 5-minute aggregates is greater than 15-minute aggregates. From the table, it is also evident that the variation of link travel time and sensor speed is greater than traffic flow volume. This implies that noise of different measurements should be handled differently, because there is no single percentage magnitude that describes their variations.

The figures in Figure 3-2 compare the measurements from Seed 1 and Seed 2 with a scatter plot. Each point is the measurement from the same sensor in the same interval. Thus, more points close to diagonal implies less stochasticity. The figures support our conclusions above that (1) 5-minute measurements (left) have more variability than 15-minute ones (right), (2) link travel times and speeds have more variation than flow volumes. Additionally, it can be observed that the variation is larger for moderate speeds (30-50 mph) and large travel times (over 50 seconds).

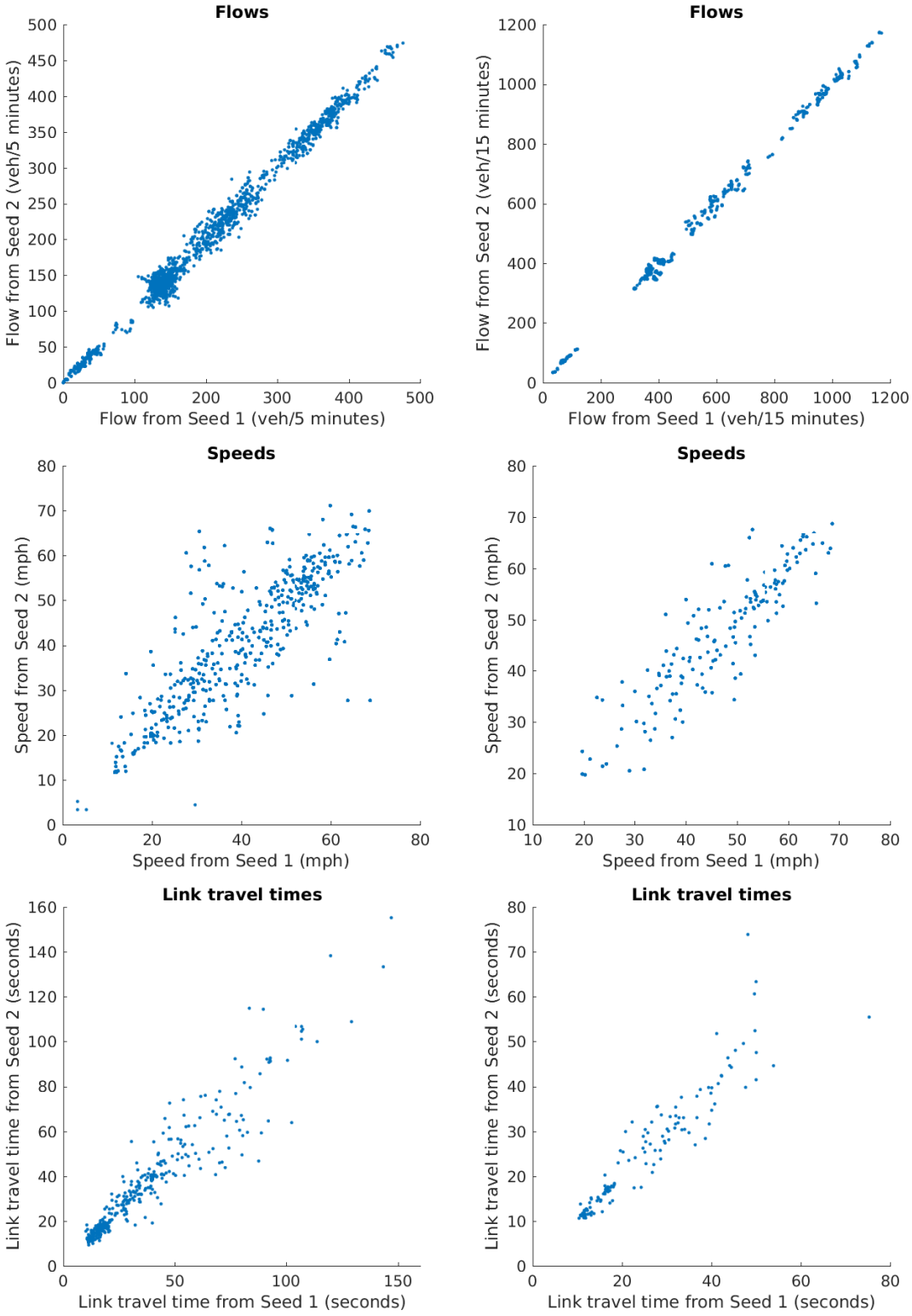


Figure 3-2: Scatter plot for measurements from Seed 1 and Seed 2 in all intervals. Left: 5 minute intervals, right: 15 minute intervals

Standard Deviations of Measurements in a Sample Interval

We now examine the measurements in more detail. We focus on the interval after 1 hour of the warm-up period: 15:00-15:05 or 15:00-15:15, depending on the interval length. We increase the number of simulations to 30 with different seeds to reduce the noise in variance estimator. All other parameters and inputs in the 30 simulations are exactly the same to ensure measurement stochasticity only comes from the random seed. For presentation of the results, apart from mean and standard deviation (SD), we use the Coefficient of Variation (CV) as a metric (standard deviation divided by mean). The results for 5 and 15 minute simulation intervals are summarized in Tables 3.4 and 3.5.

Table 3.4: Mean, standard deviation (SD) and coefficient of variance (CV) for traffic measurements for 15:00-15:05 from 30 runs with different seeds

	Segment ID	1	2	3	4	5	6	7	8
Flow	Mean	378.1	220.1	203.6	331.8	22.17	156.7	146.8	142.0
(veh/5 min)	SD	2.468	8.291	8.024	8.827	3.260	8.354	7.636	5.574
	CV	**0.65%**	**3.8%**	**3.9%**	**2.7%**	**15%**	**5.3%**	**5.2%**	**3.9%**
	Segment ID	1	2	3	4	5	6	7	8
Speed	Mean	53.35	42.31	43.13	29.54	52.95	51.91	35.33	30.78
(mph)	SD	3.755	2.895	5.013	8.279	2.108	10.788	6.806	4.764
	CV	**7.0%**	**6.8%**	**12%**	**28%**	**4.0%**	**21%**	**19%**	**15.5%**
	Segment ID	1	2	3	4	5	6+7	8	
Link TT	Mean	12.28	31.42	26.56	29.27	17.90	55.52	14.91	
(seconds)	SD	0.841	3.310	3.312	7.253	0.878	6.654	2.014	
	CV	**6.9%**	**11%**	**12%**	**25%**	**4.9%**	**12%**	**14%**	

* Segment 6 and 7 are on the same link, thus the link travel time cannot be separated

Table 3.5: Mean, standard deviation (SD) and coefficient of variance (CV) for traffic measurements for 15:00-15:15 from 30 runs with different seeds

	Segment ID	1	2	3	4	5	6	7	8
Flow	Mean	976.2	590.3	542.0	933.0	49.47	387.1	390.3	390.9
(veh/15 min)	SD	1.540	17.77	17.69	9.750	2.432	16.69	15.53	15.37
	CV	**0.16%**	**3.0%**	**3.3%**	**1.0%**	**4.9%**	**4.3%**	**4.0%**	**3.9%**
	Segment ID	1	2	3	4	5	6	7	8
Speed	Mean	58.11	48.29	46.23	33.44	55.62	58.95	44.51	32.88
(mph)	SD	1.533	2.153	4.303	7.628	0.9409	7.656	5.241	3.523
	CV	**2.6%**	**4.5%**	**9.3%**	**23%**	**1.7%**	**13%**	**12%**	**11%**
	Segment ID	1	2	3	4	5	6+7	8	
Link TT	Mean	11.20	26.20	25.36	23.59	16.85	44.21	12.79	
(seconds)	SD	0.3146	1.125	2.596	5.859	0.3345	3.270	1.391	
	CV	**2.8%**	**4.3%**	**10%**	**25%**	**2.0%**	**7.4%**	**11%**	

* Segment 6 and 7 are on the same link, thus the link travel time cannot be separated

We make two general observations from Tables 3.4 and 3.5. First, when the interval is increased from 5 to 15 minutes, CV for most measurements decrease. Second and more importantly, the measurements generally have different variabilities. SD or CV do not have the same magnitude across different segments even for the same measurement type. This implies that there may not be a simple rule when setting the diagonals of $\boldsymbol{R}$ matrix to capture simulation stochasticity. This further necessitates quantifying the simulation stochasticity for different segments.

Next, we examine the differences in variability across segments in more detail. From both tables, the mean speeds on Segment 4 are lower than most segments. This may be a result of severe reduction in lanes: Segment 1 has 6 lanes but Segment 4 only has 3. Thus, congestion is likely to happen on Segment 4. It is also noticeable that the speed on Segment 4 has significantly greater variance than others. A hypothesis that may explain this evidence is that congested segments have more variance in measurements due the simulation stochasticity.

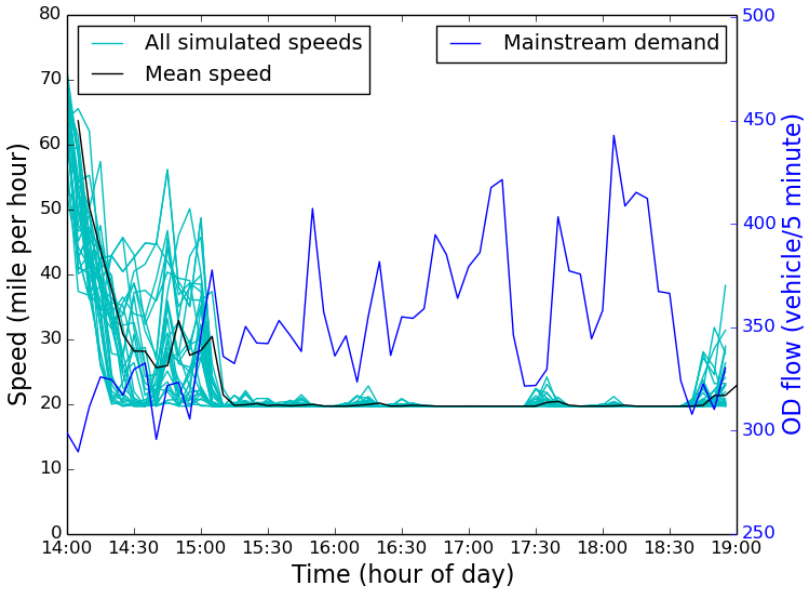


Figure 3-3: Speed measurements on Segment 4 and the mainstream OD flow assigned in each 5-minute interval

This hypothesis can be supported by Figure 3-3, which presents the simulated speeds and mainstream demand throughout the simulation period. Congestion starts

to form at the beginning and stabilizes at 15:15. During this transition, the speed variance is severe. However, after the transition to the congested regime, the variance is surprisingly small (e.g., 15:15-16:00). Additionally, when the OD flow decreases and the congestion is alleviated, there is a tendency that variance will increase, as seen for 14:30-15:00, 17:30 and 18:45. Thus it is likely that the transition between congestion and free flow is prone to simulation stochasticity.

Covariance of Measurements in the Sample Interval

The previous section presented the variance of measurements, now we focus on the covariance. In this discussion, we attempt to examine the measurement errors' relation to each other by quantifying the off-diagonals of the covariance matrix. This analysis should reveal the spatial relations across measurements caused by simulation stochasticity.

As in Tables 3.4 and 3.5, we can calculate the covariance matrix from the 30 runs with different seeds for interval 15:00-15:05 and 15:00-15:15. The covariance for measurement $Cov(\boldsymbol{M}_h)$ in a chosen interval h is given by:

$$Cov(\boldsymbol{M}_h) = \mathbb{E}\left[\left(\boldsymbol{M}_h - \overline{\boldsymbol{M}}_h\right)\left(\boldsymbol{M}_h - \overline{\boldsymbol{M}}_h\right)^\top\right] \tag{3.3}$$

$$\simeq \frac{1}{n-1}\sum_{i=1}^{n}\left(\boldsymbol{M}_h^{(i)} - \overline{\boldsymbol{M}}_h\right)\left(\boldsymbol{M}_h^{(i)} - \overline{\boldsymbol{M}}_h\right)^\top \tag{3.4}$$

where, n is number of random seeds to simulate with. $\boldsymbol{M}_h^{(i)}$ is the measurement vector for ith simulation instance. $\overline{\boldsymbol{M}}_h = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{M}_h^{(i)}$ is the mean measurement vector over n different instances of simulation with distinct seeds. Equation (3.4) is the sample covariance, which is an estimator for the true covariance.

These covariances of measurements for interval 15:00-15:05 and 15:00-15:15 are exhibited with heat maps in Figure 3-4. Blue blocks represent a positive covariance while red blocks indicate a negative covariance.

From Equation (3.4), we can see the covariance measures the deviations from

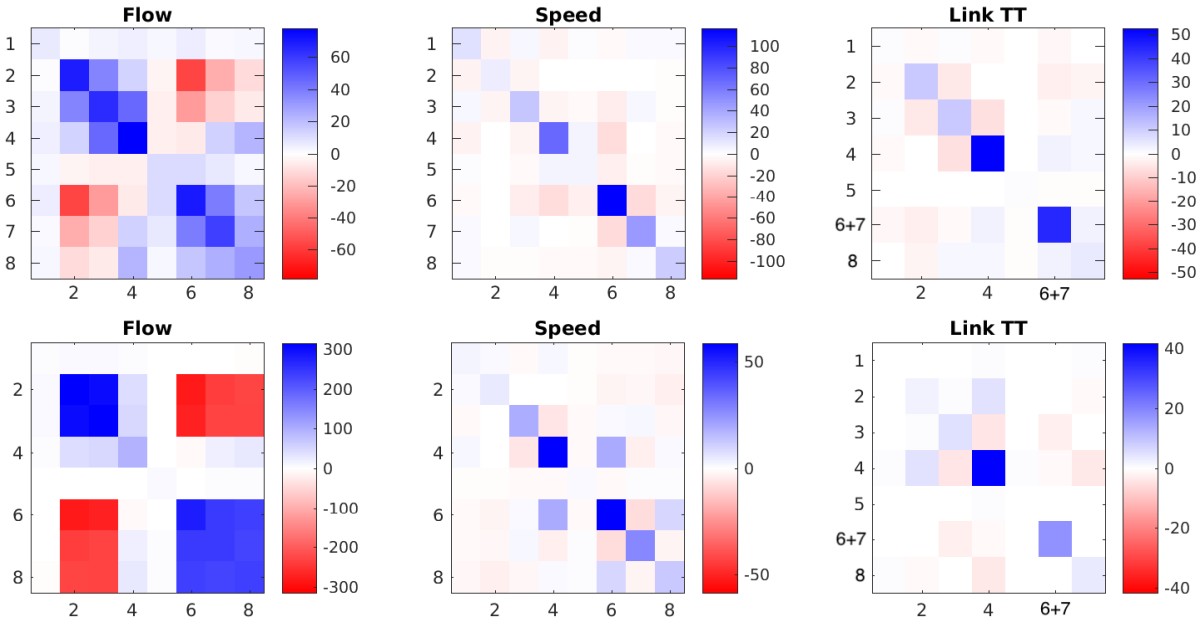


Figure 3-4: Covariance matrix of measurements for 15:00-15:05 (top) and 15:00-15:15 (bottom), after 1 hour warm-up simulation (links have the same id as segments except the one containing segment 6 and 7, denoted by "6+7")

an "averaged" simulated scenario, which is assumed to be the **expectation** of all simulated scenarios across different seeds. We comment on the negative covariances in the flow heat map. Flows of Segment 2&3 and flows of Segment 6,7&8 are two alternative routes for the same OD pair in Figure 3-1. When total demand is fixed, less vehicles on one route will result in more on the other. Another observation regarding speed and link travel time is that Segment 4 and 6 have more stochasticity than other segments. This may due to the difference in times at which congestion forms in different simulations.

An overall examination of Figure 3-4 suggests that there is a certain variance-covariance structure for each measurement type. Here we contrast this observation with the systematic measurement covariance–namely the $\boldsymbol{R}$ matrix–in the Kalman filtering framework. In most applications, $\boldsymbol{R}$ is heuristically set based on researchers' belief. Since it is difficult to propose a covariance structure from heuristics, $\boldsymbol{R}$ is usually assumed as diagonal with presumed magnitudes. However, the heat maps suggest that magnitude of diagonals can be significantly different. Besides, the co-variance may play an important role in capturing spatial correlations across different

simulated instances.

3.2.3 Summary

We close this section with three major observations. First, aggregating over a longer time period helps reduce simulation stochasticity, especially for speed and link travel time measurements. Second, for the given network and the demand inputs, there is higher simulation stochasticity in speeds and link travel times during the transition between free flow and congestion, and vice versa. Last but not least, the variance magnitude is different across sensors, probably due to differing traffic states at different locations. The covariance structure is also different across measurement types. This implies a calculated covariance matrix from multiple simulations is probably more accurate than ad hoc error covariance settings in terms of accurately capturing measurement stochasticity.

3.3 Error Analysis for Kalman Filtering Equations

3.3.1 State Space Model

Having investigated the nature of simulation stochasticity, we now discuss its connection to error covariances in the Kalman filtering framework. In this section, we first review the State Space Model and then focus on how to capture uncertainty in the simulation in the model.

The general State Space Model for online calibration of DTA systems is represented with abstract functions in Equations (2.15) and (2.16). We review them here for convenience.

$$\boldsymbol{x}_h = \boldsymbol{f}_{h-1}(\boldsymbol{x}_{h-1:h-p}) + \boldsymbol{w}_h \tag{3.5}$$

$$\boldsymbol{M}_h = \boldsymbol{g}_h(\boldsymbol{x}_{h:h-q+1}) + \boldsymbol{v}_h \tag{3.6}$$

where the function $\boldsymbol{f}_{h-1}(\cdot)$ captures the transition relations from previous p inter-

vals to interval h, and $\boldsymbol{g}_h(\cdot)$ represents the DTA system that transforms the inputs/parameters into simulated measurements. $\boldsymbol{M}_h$ denotes the real-world observations from traffic surveillance systems. Thus, $\boldsymbol{v}_h$ is the error term for the gap between $\boldsymbol{M}_h$ and simulation $\boldsymbol{g}_h(\cdot)$, which is usually assumed to include field measurement errors.

When we consider simulation stochasticity, we should minimize the discrepancy between real-world observations $\boldsymbol{M}_k$ and the *expectation* of simulated measurements. Thus, the $\boldsymbol{g}_h(\cdot)$ in Equation (3.6) no longer represents one instance of the simulation, but denotes the *expectation* of all possible simulations with different random seeds. We introduce the following notation and assumptions:

1. Let $\boldsymbol{S}_h(\cdot, \omega)$ denote the simulated measurements with ω as random seed for interval h, where $\omega \in \Omega$ and Ω is the set of all random seeds;
2. Denote $\boldsymbol{\Sigma}_h = Cov(\boldsymbol{S}_h(\cdot, \omega)) = \mathbb{E}_\omega[\boldsymbol{S}_h(\cdot, \omega)\boldsymbol{S}_h(\cdot, \omega)^\top]$
3. $\boldsymbol{g}_h(\cdot) = \mathbb{E}_\omega[\boldsymbol{S}_h(\cdot, \omega)]$

In a similar manner, we further denote the stochasticity with an error term $\boldsymbol{\epsilon}_h(\omega)$ such that $\boldsymbol{\epsilon}_h(\omega) = \boldsymbol{g}_h(\cdot) - \boldsymbol{S}_h(\cdot, \omega)$, the new measurement equation would be given by:

$$\boldsymbol{M}_h = \boldsymbol{S}_h(\boldsymbol{x}_{h:h-q+1}, \omega) + \boldsymbol{\epsilon}_h(\omega) + \boldsymbol{v}_h \tag{3.7}$$

For the simplicity of the following discussion, we drop ω and let $\boldsymbol{S}_h(\cdot)$ denote the measurements of one arbitrary simulation instance. This simplification is valid when we use a randomly drawn random seed, which is exactly the case for each simulation interval h. Thus, the seeds ω in different intervals can be deemed independent of each other, albeit in a pseudo-random sense. In light of this, $\mathbb{E}[f(\boldsymbol{\epsilon}_h)] = \mathbb{E}_\omega[f(\boldsymbol{\epsilon}_h(\omega))]$, as the left hand side is the expectation over all cases possible, which is a superset of drawing ω from Ω.

We make the following additional assumptions about the error terms:

1. $\mathbb{E}[\boldsymbol{\epsilon}_h] = \mathbb{E}_\omega[\boldsymbol{\epsilon}_h(\omega)] = \mathbf{0}$

2. $Cov(\boldsymbol{\epsilon}_h) = \mathbb{E}[\boldsymbol{\epsilon}_h {\boldsymbol{\epsilon}_h}^\top] = \mathbb{E}_\omega[\boldsymbol{\epsilon}_h(\omega) \boldsymbol{\epsilon}_h(\omega)^\top] = \boldsymbol{\Sigma}_h$

3. $\mathbb{E}[\boldsymbol{\epsilon}_h {\boldsymbol{v}_h}^\top] = \mathbf{0}$

where, the last equality is a strong assumption. But it is valid if we assume $\boldsymbol{v}_h$ is the measurement error from the surveillance system. Thus, the error for simulation stochasticity $\boldsymbol{\epsilon}$ is independent of $\boldsymbol{v}_h$. In addition, we have already assumed the following for the State Space Model.

1. $\mathbb{E}[\boldsymbol{v}_h] = \mathbf{0}$

2. $\mathbb{E}[\boldsymbol{v}_h {\boldsymbol{v}_h}^\top] = \boldsymbol{R}_h$

Based on these characteristics, we can group two error terms and a simple derivation yields:

1. $\mathbb{E}[\boldsymbol{\epsilon}_h + \boldsymbol{v}_h] = \mathbf{0}$

2. $\mathbb{E}[(\boldsymbol{\epsilon}_h + \boldsymbol{v}_h)(\boldsymbol{\epsilon}_h + \boldsymbol{v}_h)^\top] = \boldsymbol{\Sigma}_h + \boldsymbol{R}_h$

which applies to the following measurement equation:

$$\boldsymbol{M}_h = \boldsymbol{S}_h(\boldsymbol{x}_{h:h-q+1}) + \boldsymbol{\epsilon}_h + \boldsymbol{v}_h \tag{3.8}$$

Thus, we have derived the measurement equation considering simulation stochasticity. Note when we use Kalman filtering techniques, $\boldsymbol{\epsilon}_h + \boldsymbol{v}_h$ should be assumed as Gaussian variables, and $\boldsymbol{\Sigma}_h + \boldsymbol{R}_h$ will be used as the measurement covariance matrix.

3.3.2 Gradient Estimation

Simulator stochasticity also needs to be explicitly accounted for in the gradient estimation procedure. Recall that this involves a linear approximation of the measurement equation which requires computation of the system Jacobian (also referred to as the H matrix). Specifically, the linear analytical model is given by the following equation

(note for the simplicity of notation, without loss of generality, the augmented state vector $\boldsymbol{x}_{h:h-q+1}$ is reduced to $\boldsymbol{x}_h$),

$$\partial \boldsymbol{M}_h = \boldsymbol{H}_h(\boldsymbol{x}_h - \boldsymbol{x}_h^H) + \boldsymbol{\eta}_h + \boldsymbol{\epsilon}_h + \boldsymbol{v}_h \tag{3.9}$$

where, previously defined notation applies. The additional $\boldsymbol{\eta}_h$ comes from the linear approximation. While this term is not the focus of our discussion, it is beneficial to identify it as a source of uncertainty for the completeness of the analysis.

Based on the above derivations, we conclude this section with the following comments. First, the error term of the linear SSM model comprises different sources of uncertainty, including field-measurement noise $\boldsymbol{v}_h$, simulation stochasticity $\boldsymbol{\epsilon}_h$ and linearization error $\boldsymbol{\eta}_h$.Second, considering all the cases, the error covariance in the measurement model is at least $\boldsymbol{\Sigma}_h + \boldsymbol{R}_h$. Lastly, the gradient estimation (obtaining $\boldsymbol{H}_h$) is based on simulation and hence, will also suffer from simulation stochasticity. This is addressed in more detail in the following section, where we will discuss how simulation stochasticity affects $\boldsymbol{H}_h$ and how to reduce the impact.

3.4 Stochasticity in Gradient Estimation

In this section, we quantify the impact of stochasticity on the gradient matrix which is critical given that it represents the linear relationship between parameters and measurements. If the gradient matrix suffers from large noise, it is likely to significantly affect the performance of the Kalman filter. In the literature, the estimated gradient is usually directly used, and the issue of noise in the linearization procedure is seldom discussed. We attempt to analyze the impact of noise arising from simulation stochasticity on the gradient.

We first show the evidence of stochasticity in the H matrix. Then we conduct an analysis to quantify the simulation stochasticity, followed by an experiment that verifies the analysis. Finally we attempt to minimize the impact via some guidance

based on our analysis.

3.4.1 Evidence of Stochasticity in the Gradient Matrix

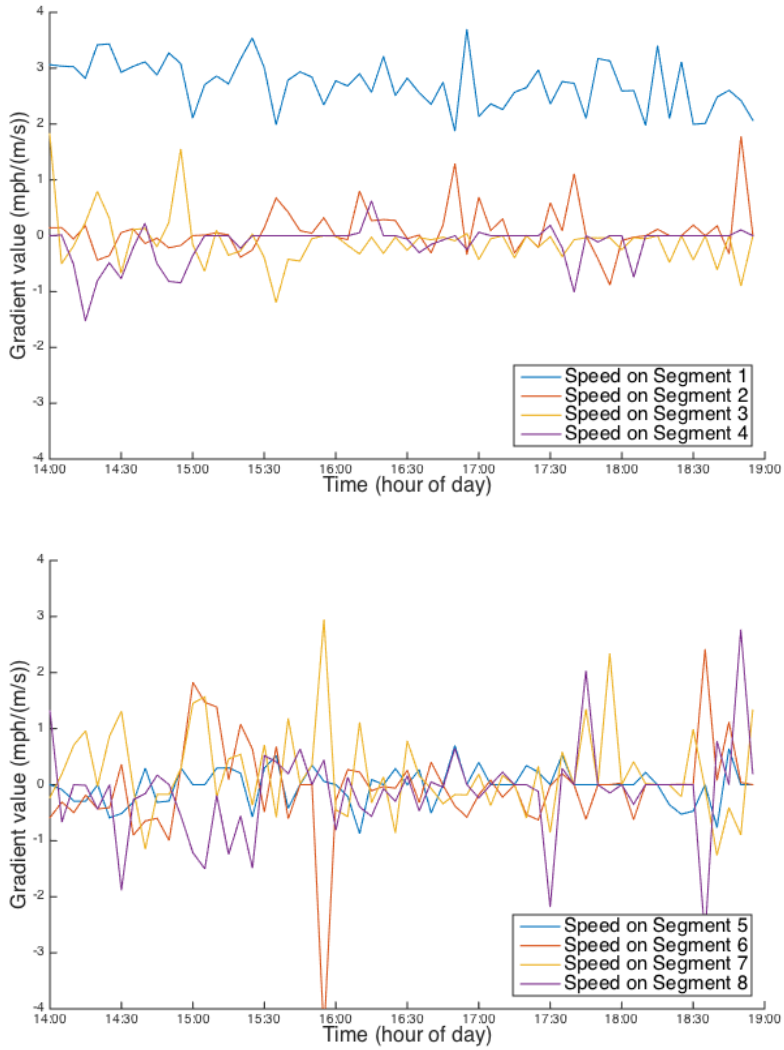


Figure 3-5: The impact of free flow speed V_f in Segment 1 on all segments in different simulation intervals

As a result of simulation stochasticity for each run, the gradient matrix calculated from finite difference by two runs of simulation is also stochastic. The key question now lies in how much impact it has and how this impact can be mitigated. Figure 3-5 presents one column of the gradient matrix for all time intervals, calculated by the finite difference. The parameter being calibrated is free flow speed V_f for Segment 1

on the toy network in Figure 3-1, and measurements are speeds on all segments. The unit of the vertical axis is mile per hour (mph) over meter per second (m/s), because the parameter V_f follows the International System of Units (SI) while measurements follow United States customary units. Thus, the gradient value of 2.237 means a 1 m/s change in V_f will increase the speed measurement by 1 mph.

Now we comment on Figure 3-5. First, we expect V_f to have a positive impact on the speed of the same segment, and it is verified by Figure 3-5. Second, there is no stable positive/negative relation between V_f of Segment 1 and other segments. The severe fluctuations and abrupt sign change indicate noisy gradients, thus high uncertainty in the linear relationship. It is likely that the noise is related to simulation stochasticity. Thus an error analysis on the gradient matrix is necessary and helpful.

3.4.2 Error Analysis

Based on the analysis of the previous section, we now examine the impact of stochasticity on the H matrix (gradient). We continue to use the finite difference method to obtain the H matrix.

We first recall the Equations (2.21) to (2.22).

$$\boldsymbol{H}_h = \left[\begin{array}{ccc} \frac{\partial \boldsymbol{g}_{h,1}}{\partial \boldsymbol{x}_{h,1}} & \cdots & \frac{\partial \boldsymbol{g}_{h,1}}{\partial \boldsymbol{x}_{h,n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \boldsymbol{g}_{h,m}}{\partial \boldsymbol{x}_{h,1}} & \cdots & \frac{\partial \boldsymbol{g}_{h,m}}{\partial \boldsymbol{x}_{h,n}} \end{array}\right]\Bigg|_{\boldsymbol{x}_h = \hat{\boldsymbol{x}}_{h|h-1}} \tag{3.10}$$

$$\text{where, } \boldsymbol{H}_{h(:,j)} = \frac{\boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} + \boldsymbol{\delta}_j) - \boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} - \boldsymbol{\delta}_j)}{2\delta_j} \tag{3.11}$$

$$\boldsymbol{\delta}_j = [0, 0, ..., \delta_j, ..., 0]^\top \tag{3.12}$$

Now we focus on Equation (3.11) and substitute the *expected* simulation $\boldsymbol{g}_h(\cdot)$ with $\boldsymbol{S}_h(\cdot) + \boldsymbol{\epsilon}_h$. This yields:

$$\boldsymbol{H}_{h(:,j)} = \left(\boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} + \boldsymbol{\delta}_j) - \boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} - \boldsymbol{\delta}_j)\right)/2\delta_j \tag{3.13}$$

$$= \left(\boldsymbol{S}_h(\hat{\boldsymbol{x}}_{h|h-1} + \boldsymbol{\delta}_j) - \boldsymbol{S}_h(\hat{\boldsymbol{x}}_{h|h-1} - \boldsymbol{\delta}_j)\right)/2\delta_j + \left(\boldsymbol{\epsilon}_h - \boldsymbol{\epsilon}'_h\right)/2\delta_j \tag{3.14}$$

where, we made the assumption that $\boldsymbol{\epsilon}_h$ and $\boldsymbol{\epsilon}'_h$ are two independent random variables following the same distribution with zero mean and covariance $\boldsymbol{\Sigma}_h$. $\left(\boldsymbol{\epsilon}_h - \boldsymbol{\epsilon}'_h\right)/2\delta_j$ is the error term for column j of the H matrix. Hence,

1. $\mathbb{E}[\left(\boldsymbol{\epsilon}_h - \boldsymbol{\epsilon}'_h\right)/2\delta_j] = \mathbf{0}$
2. $Cov\left(\left(\boldsymbol{\epsilon}_h - \boldsymbol{\epsilon}'_h\right)/2\delta_j\right) = \mathbb{E}[\left(\boldsymbol{\epsilon}_h - \boldsymbol{\epsilon}'_h\right)\left(\boldsymbol{\epsilon}_h - \boldsymbol{\epsilon}'_h\right)^\top]/4\delta_j^2 = \boldsymbol{\Sigma}_h/2\delta_j^2$

Now we attempt to shed some light on the effect of the error term. Here we define $\hat{\boldsymbol{H}}_{h(:,j)} = \left(\boldsymbol{S}_h(\hat{\boldsymbol{x}}_{h|h-1} + \boldsymbol{\delta}_j) - \boldsymbol{S}_h(\hat{\boldsymbol{x}}_{h|h-1} - \boldsymbol{\delta}_j)\right)/2\delta_j$, as it is actually an approximation of $\boldsymbol{H}_{h(:,j)}$ by finite difference. Thus, $\hat{\boldsymbol{H}}_{h(:,j)} = \boldsymbol{H}_{h(:,j)} + \left(\boldsymbol{\epsilon}'_h - \boldsymbol{\epsilon}_h\right)/2\delta_j$, where $\boldsymbol{H}_{h(:,j)}$ is the signal and $\left(\boldsymbol{\epsilon}'_h - \boldsymbol{\epsilon}_h\right)/2\delta_j$ is the noise. We first examine $\left(\boldsymbol{\epsilon}'_h - \boldsymbol{\epsilon}_h\right)/2\delta_j$. It has covariance $\boldsymbol{\Sigma}_h/2\delta_j^2$, and the magnitude only depends on the perturbation size δ_j. Thus, a small δ_j will magnify the covariance matrix, resulting in more noise. Next, we focus on $\boldsymbol{H}_{h(:,j)}$. We claim that the local perturbation size δ_j does not affect $\boldsymbol{H}_{h(:,j)}$ significantly. It is based on the fundamental assumption of the extended Kalman filter: the slope of the function $\boldsymbol{g}_h(\cdot)$ does not change significantly around $\boldsymbol{x}_{h|h-1}$ so that a linear function can approximate $\boldsymbol{g}_h(\cdot)$ locally. In a global sense, the signal-to-noise ratio $\left(\boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} + \boldsymbol{\delta}_j) - \boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} - \boldsymbol{\delta}_j)\right) / \left(\boldsymbol{\epsilon}'_h - \boldsymbol{\epsilon}_h\right)$ also explains the necessity for a large δ_j.

Based on the above analysis, to minimize the effect of simulation stochasticity, we need to choose greater δ_j. On the other hand, δ_j should not be too large, for $\boldsymbol{H}_{h(:,j)}$ to be a local approximation. As an example, some supply parameters have a small magnitude, in which case the perturbation size is even smaller, often $o(1)$. Thus, the noise $\left(\boldsymbol{\epsilon}_h - \boldsymbol{\epsilon}'_h\right)/2\delta_j$ may dominate $\hat{\boldsymbol{H}}_{h(:,j)}$. Another example is OD flow parameters. However, the perturbation δ_j on OD is usually $O(10)$. For this reason, the H matrix for OD flows is usually less affected by simulation stochasticity.

3.4.3 Experimental Verification

In order to verify our analysis, we conduct an experiment to obtain the standard deviation of each element in the H matrix via 30 runs of simulation with different seeds. The following procedure is conducted:

1. Random select 30 seeds to form a pool;
2. Specify the parameters to investigate, in our case, free flow speed V_f;
3. Determine the perturbation size δ, in our case, 2% and 10% of the initial values are used for each V_f;
4. For each seed, run the first interval of simulation with calibration and record the H matrix calculated;
5. Calculate the element-wise standard deviation of these 30 H matrices and compare the results for different perturbation sizes.

The reason for running only the first interval is to ensure that all runs (for different seeds) start with the same network state (empty network). If on the other hand, subsequent intervals were used, the initial traffic states would be different, because the value of previous estimates $\boldsymbol{x}_{h-1|h-1}$ are determined by a noisy H matrix.

Figures 3-6 and 3-7 presents the mean and standard deviation heat maps from these 30 simulation results for two percentage perturbation sizes. It is clear that with larger perturbation, the standard deviations for a majority of the elements in the H matrix are reduced significantly. Another important observation is that the mean for off-diagonal elements is close to zero, but the standard deviation has a greater magnitude. In such cases, the off-diagonals are very noisy, providing an explanation of the trends observed in Figure 3-5.

We have quantified simulation stochasticity with an error term in each column of the H matrix. Since the standard deviation of the error is inversely proportional to the perturbation size, in order to decrease the impact of simulation stochasticity on the H matrix, the perturbation size should be increased. However, increasing the

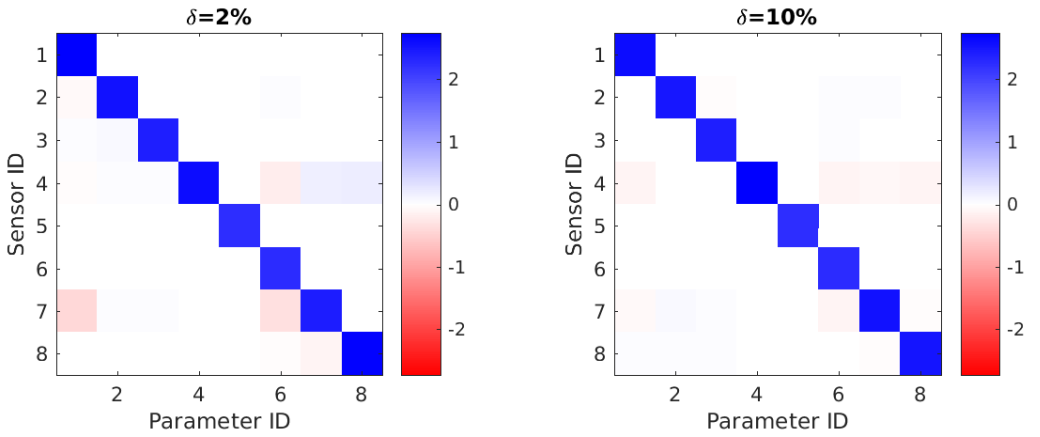


Figure 3-6: Mean of each element in the H matrix for two percentage perturbations δ on segment free flow speed V_f

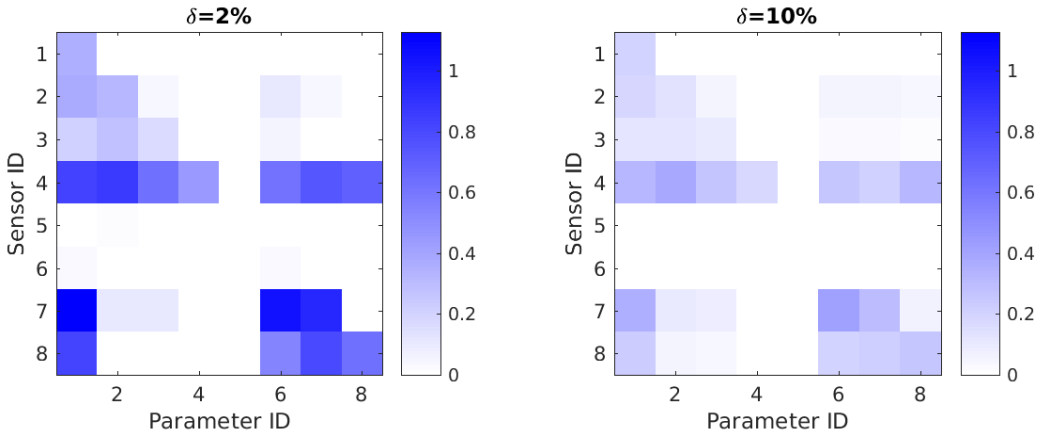


Figure 3-7: Standard deviation of each element in the H matrix for two percentage perturbations δ on segment free flow speed V_f

perturbation size adversely impacts the accuracy of the local function approximation. Moreover, this method may not work for sparse H matrices, where noise still exists in the elements that should be zero. These elements are dominated by noise because the true "signal" is 0. In the remainder of this chapter, we attempt to mitigate this with an enforced H matrix structure.

3.5 Solution Approaches Considering Simulation Stochasticity

3.5.1 Incorporating Simulation Covariance

Based on the observations in Section 3.3.1, the new measurement error covariance should be set to $\boldsymbol{\Sigma}_h + \boldsymbol{R}_h$ so as to consider simulation stochasticity. In addition, the form of the error covariance implies that it does not simply rely on a heuristic setting of $\boldsymbol{R}_h$, which makes it more robust.

As stated in the literature review, the $\boldsymbol{\Sigma}_h$ matrix is usually assumed to be time-invariant for simplicity. This is incorrect because the traffic states are different for each time interval, and simulation stochasticity depends on the current traffic states. However, accurate $\boldsymbol{\Sigma}_h$ matrices for each interval h are rarely available in practice. In this approach, we can rely on the outputs from offline simulations to provide a universal $\boldsymbol{\Sigma}$ for online calibration for the whole simulation period. Although this compromises calibration performance, it is significantly less intensive in terms of data requirements.

This can be done by calculating the covariance matrix over all different seeds for each time interval h, and then averaging over all simulation intervals $h \in \{1, 2, ..., N\}$. Specifically, the selected covariance comes from the mean of covariance matrices $\boldsymbol{\Sigma}_h$ for each interval, given by:

$$\boldsymbol{\Sigma} = \frac{1}{N} \sum_{h=1}^{N} \boldsymbol{\Sigma}_h \tag{3.15}$$

$$= \frac{1}{N} \sum_{h=1}^{N} Cov(\boldsymbol{M}_h) \tag{3.16}$$

where, $Cov(\boldsymbol{M}_h)$ is given by Equation (3.4).

The implementation is straightforward and simply involves replacing $\boldsymbol{R}_h$ with $\boldsymbol{\Sigma} + \boldsymbol{R}_h$. The only drawback is that we need to compute the error covariance $\boldsymbol{\Sigma}$

offline. When computing $\mathbf{\Sigma}$, it is preferable to run simulations with demand and supply parameters that match the "true" parameters as closely as possible to ensure accuracy of the covariance matrix. This could be done when offline calibrated demand and default values for supply are available. If they are not available offline, an online estimation process for $\mathbf{\Sigma}_h$ is helpful, which could come from offline runs with calibrated demand and supply for previous intervals. This will be a direction of future work.

3.5.2 Enforcing H Matrix Structure with a H Mask

From the mean H matrix in Figure 3-7, it is obvious that only the diagonals should be non-zero. However, the standard deviation of the off-diagonals indicates that each sample of H matrix is likely to have non-zero off-diagonals. In light of this, we can assume an H matrix structure–which we call H mask–to force these elements to be zero. The procedure is to first identify a H mask, then apply the mask to update the H matrix, which is to be used in the measurement equation.

Identifying Non-Zero H Elements with t-Test

In order to determine whether each element is significantly different from zero, we define the null hypothesis as the element having a zero mean. We can use the Student's t-test for each element of the H matrix (although debatable, discussed later). Then the test statistics are given by:

$$t_{(i,j)} = \left(\overline{\boldsymbol{H}}_{h,(i,j)} - 0\right) \Bigg/ \left(\frac{\boldsymbol{\sigma}_{h,(i,j)}}{\sqrt{N-1}}\right) \text{ for } i = 1, ..., m, j = 1, ..., n \tag{3.17}$$

where, N is the number of simulation runs with different seeds. $\overline{\boldsymbol{H}}_h$ is the average H matrix for those N runs, while $\boldsymbol{\sigma}_h$ is the sample standard deviation matrix calculated element-wise from N H matrices. Subscript (i, j) are the indices in an $m \times n$ matrix. For the same example in the previous section, the t-stat matrix based on Equation (3.17) yield the following heat maps in Figure 3-8, again for two different perturbation sizes. Note the diagonal values over 50 are capped, in order for the

off-diagonal values to be distinguishable from 0.

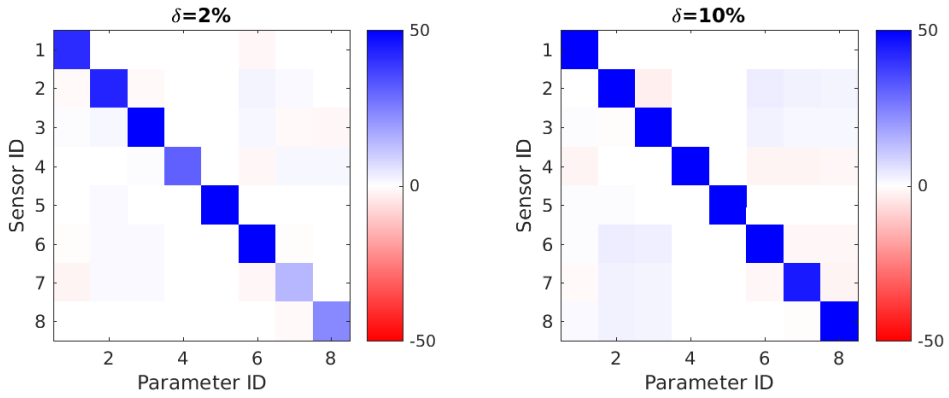


Figure 3-8: t-stats of each element in H matrix from 30 runs of simulation for two perturbation size δ

From the figure we can conclude that diagonals are significantly different from zero, mostly around 50. For off-diagonals, the largest value is 3.44, which has a 0.0018 p-value according to the t distribution with degree of freedom 29. However, it is worth mentioning that using a t-test for every element is problematic, which is called a multiple comparison problem. Assume that we have n true null hypotheses, and we do multiple t-tests with $\alpha = 0.01$. Then the probability of rejecting at least one true null hypothesis is $1 - (1 - \alpha)^n$. For our case in Figure 3-9, $n = 64$ gives 0.474 as the probability of rejecting a true null hypothesis. To mitigate the issue, the Holm-Bonferroni method should be conducted.

Identifying Non-Zero H Elements with Holm-Bonferroni Method

The corresponding p-values for each element are exhibited in Figure 3-9. In Figure 3-10, we plot the critical values according to the Holm-Bonferroni test with the ordered p-values in the log scale on the y axis. It is obvious that the ninth smallest p-value is greater than the critical value. Thus we only reject the null hypotheses for first 8 indices, which correspond to the diagonals.

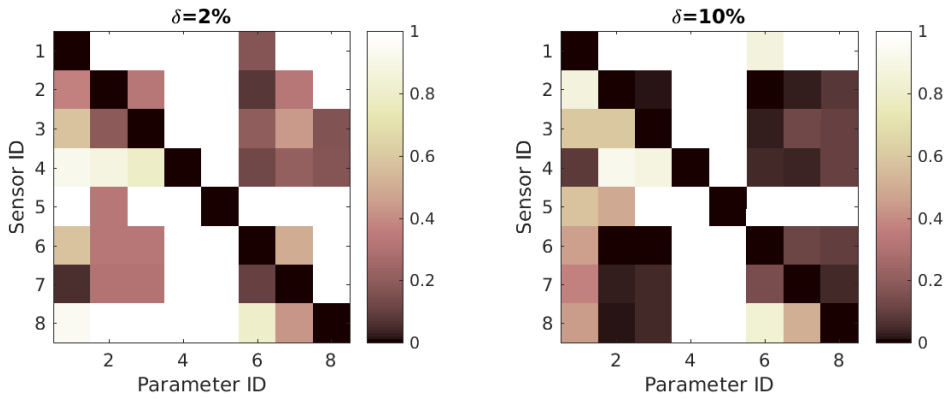


Figure 3-9: p-values of each element in H matrix from 30 runs of simulation for two perturbation size δ

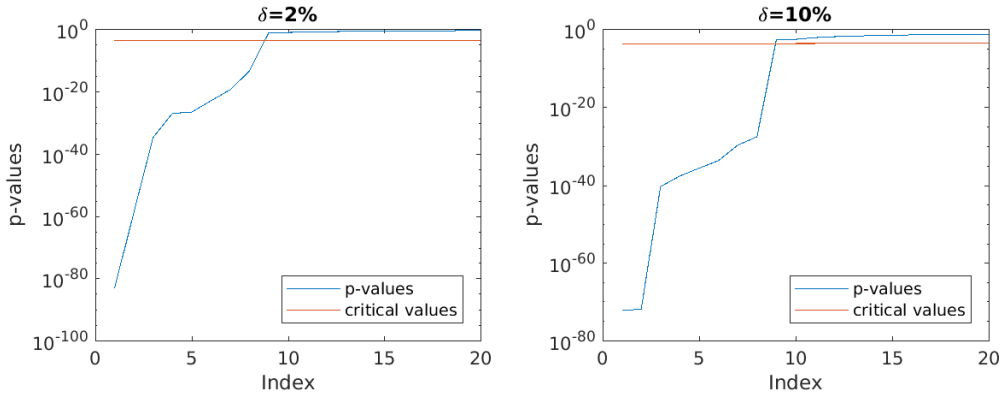


Figure 3-10: The Holm-Bonferroni test to detect H matrix for two perturbation size δ

Applying H Matrix Mask

According to the Bonferroni test, we ensure that the diagonals are non-zero. Hence, we define the H mask $\boldsymbol{H}_{mask}$ as an identity matrix, and use the following update rule:

$$\tilde{\boldsymbol{H}}_h = \boldsymbol{H}_h \circ \boldsymbol{H}_{mask} \tag{3.18}$$

where, $\circ$ is the element-wise matrix product, also known as the Hadamard product or Schur product.

Thus we use $\tilde{\boldsymbol{H}}_h$ in our Kalman filter update procedure. This will significantly increase the sparsity and reduce the noise in some H matrix elements. In our case the off-diagonals are forced to be zeros.

In theory, H matrices in different intervals may have different H masks. However in practice, a universal H mask maybe used because H masks for different intervals may be difficult to obtain. The reason lies in the fact that it is difficult to keep identical all prevailing simulation conditions and quantify stochasticity in a single interval.

3.6 A Synthetic Case Study

In this case study, the aforementioned two approaches are applied on the synthetic network in Figure 3-1: (1) incorporating measurement covariance due to simulation stochasticity; (2) applying H mask to reduce noise. In our experiments, each approach is conducted for one selected set of supply parameters.

3.6.1 Using Simulation Error Covariance Matrix

Data Generation

In this experiment, the parameters are segment capacities for the 8 segments. We assume an autoregressive model with degree 5 (AR(5)) for each segment capacity and generate the time-dependent capacities as the true parameters for DynaMIT. Note that the generation is based on a set of historical values, which are time-invariant default parameters that capture the mean of the simulation period. The AR(5) model is on the deviations from the historicals. Then with a run of DynaMIT with the true parameters, we obtain the sensor measurements, which are the surveillance data for calibration. DynaMIT works as the real world in the data generation procedure.

Calibration Procedure

In our calibration procedure, DynaMIT is the simulation-based DTA to be calibrated. The AR(5) model is assumed known to the calibration algorithm. The following configurations are tested:

(1) Use the default values without online calibration

(2) Use an identity matrix $\boldsymbol{R} = \boldsymbol{I}$ for constrained extended Kalman filter (standard deviation is 1 mph)

(3) Use an diagonal matrix $\boldsymbol{R} = 100\boldsymbol{I}$ for constrained extended Kalman filter (standard deviation is 10 mph)

(4) Use $\boldsymbol{R} = \boldsymbol{\Sigma} + \boldsymbol{I}$ for constrained extended Kalman filter, where $\boldsymbol{\Sigma}$ is the error covariance matrix from stochasticity analysis

(5) Use $\boldsymbol{R} = \boldsymbol{\Sigma} + \boldsymbol{I}$ for constrained extended Kalman filter, change the initial seed for simulation-based DTA to test robustness

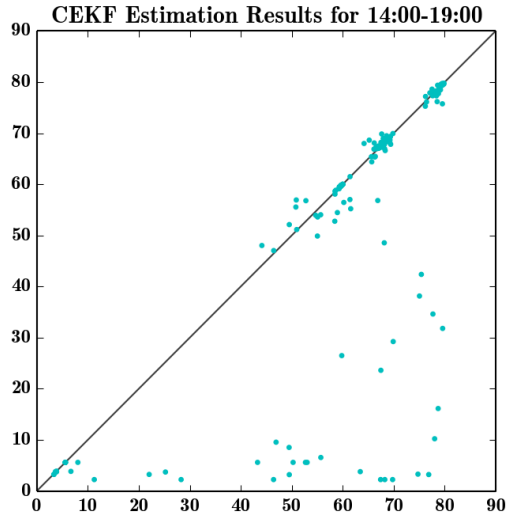
Results

The following results indicate an overall fit of speed in RMSNs (Equation (3.2)). Two RMSNs with different initial seeds are presented.

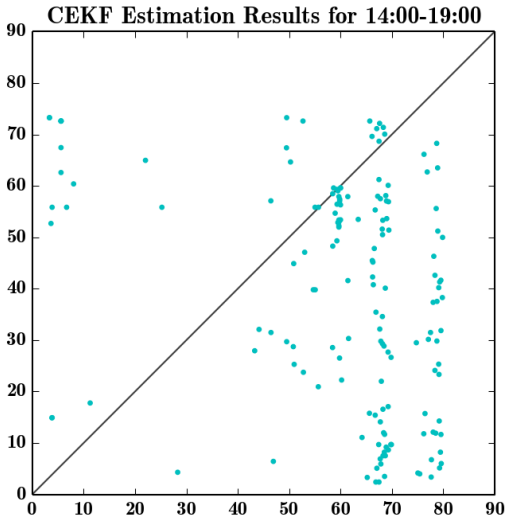
Index	Setting	Speed RMSN
(1)	no calibration, seed 1	35.6%
(2)	$\boldsymbol{R} = \boldsymbol{I}$, Seed 1	67.3%
(3)	$\boldsymbol{R} = 100\boldsymbol{I}$, Seed 1	57.3%
(4)	$\boldsymbol{R} = \boldsymbol{\Sigma} + \boldsymbol{I}$, Seed 1	17.9%
(5)	$\boldsymbol{R} = \boldsymbol{\Sigma} + \boldsymbol{I}$, Seed 2	18.9%

Table 3.6: Extended Kalman filtering result using $\boldsymbol{\Sigma}_h + \boldsymbol{R}_h$ as measurement error covariance for Seed 1 and 2

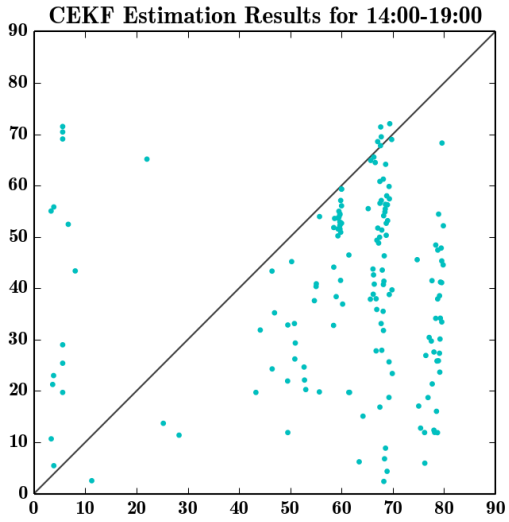
The results verified our analysis. The speed RMSN improves when we use $\boldsymbol{\Sigma}_h + \boldsymbol{R}_h$ as covariance. The improvement is observed for both two seeds, meaning that the calibration algorithm is able calibrate the DTA system for different initial seeds. The performance is verified in scatter plots in Figure 3-11 as well. In experiment (1) some of the speeds are under estimated. While experiment (2) and (3) reported poorly fitted speeds. Experiments (4) and (5) indicated that most of the high speeds are fitted well. However, the speeds with low values were not fitted as well as those with high values, which is probably because of the high simulation stochasticity during transition between free flow and congestion. With high measurement stochasticity, the calibration task is more difficult.



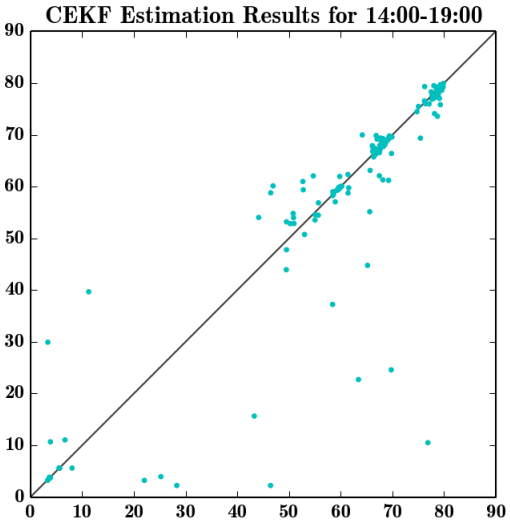
(1) No calibration, Seed 1



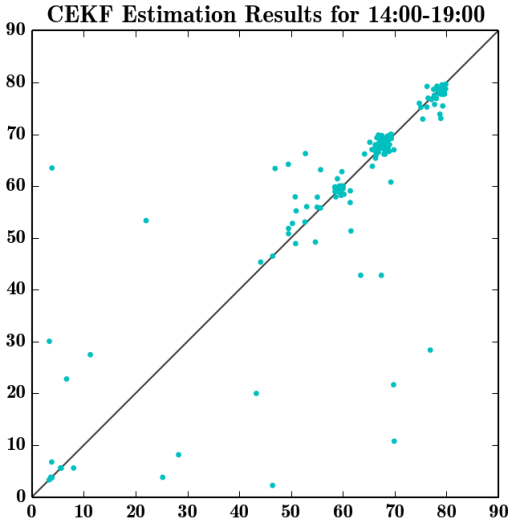
(2) CEKF with $\boldsymbol{R} = \boldsymbol{I}$, Seed 1



(3) CEKF with $\boldsymbol{R} = 100\boldsymbol{I}$, Seed 1



(4) CEKF with $\boldsymbol{R} = \boldsymbol{\Sigma} + \boldsymbol{I}$, Seed 1



(2) CEKF with $\boldsymbol{R} = \boldsymbol{\Sigma} + \boldsymbol{I}$, Seed 2

Figure 3-11: Scatter plot for observed speeds vs estimated speeds

3.6.2 Enforcing Gradient Structure

Now we present the experiments with the idea to enforce the gradient structure with an H mask.

Data Generation

The parameters in this experiment are free flow speeds V_f for 8 segments. We follow the same procedure as discussed in Section 3.6.1. We assume another AR process, and obtain speed measurements as surveillance data for the subsequent calibration.

Calibration Procedure

In the calibration experiments, the idea of the H mask is applied. A benchmark for comparison is the Limiting EKF, where an average gradient matrix is calculated offline and used for all the intervals. The average gradient matrix is computed from gradient matrices obtained from previous runs of online calibration, where the Constrained Extended Kalman Filter with Finite Difference (FD-CEKF) is applied. We apply both models individually and compare it with the case without calibration. The following experiments are conducted:

(1) Use the default values without online calibration

(2) Use the computed gradient matrix from finite difference for online calibration

(3) Use the offline averaged gradient matrix for online calibration

(4) Use $\boldsymbol{H}_{mask}$ to enforce the gradient structure for computed gradient matrix

(5) Apply $\boldsymbol{H}_{mask}$ on the offline averaged gradient matrix for online calibration

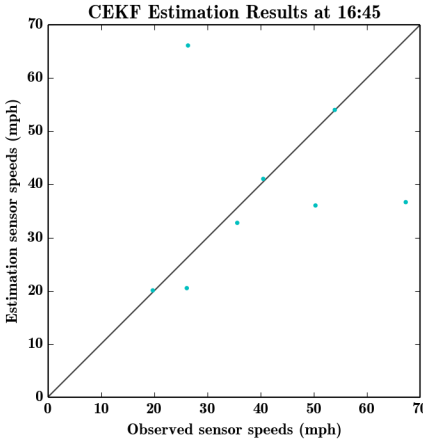
Results

The speed RMSNs are presented in Table 3.7, corresponding to the 5 experiments discussed above. The online calibration that directly uses computed gradient matrix in (2) yields no improvement over the historical free flow parameters (1). Using

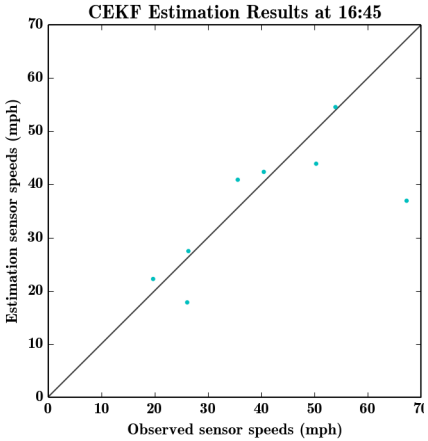
Index	Masked H	Average H	Speed RMSN
(1)	no calibration		23.6%
(2)			23.7%
(3)		✓	21.6%
(4)	✓		**17.7%**
(5)	✓	✓	19.6%

Table 3.7: Extended Kalman filtering result using H mask filtered gradient

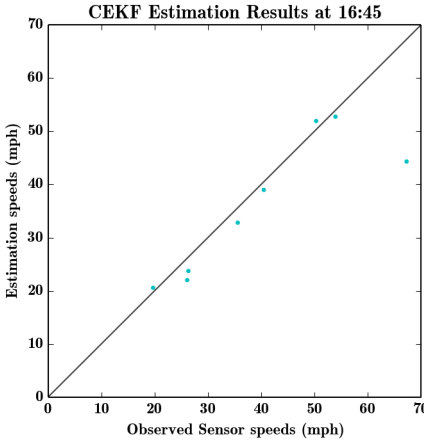
an averaged gradient in (3) decreased the RMSN by 9%, compared with (1). An examination of the average gradient matrix indicates that the noise in the gradient is reduced but not eliminated for the zero elements. This implies that reducing the noise for the gradient will improve the calibration performance. Experiment (3) with $\boldsymbol{H}_{mask}$ yields the lowest RMSN, improving (1) by 25%. This is because applying the $\boldsymbol{H}_{mask}$ eliminates the noise for zero elements, thus making the gradient sparse and the parameter-dependency accurate. Comparing (3) and (4), it is likely that the small magnitude of noise in zero elements still restricts the accuracy of the gradient. Hence, applying $\boldsymbol{H}_{mask}$ in (5) improves over (3). Surprisingly for (5), applying both $\boldsymbol{H}_{mask}$ and averaged gradient yields worse result than applying $\boldsymbol{H}_{mask}$ alone. This may be a result of the lack of an online update of the gradient, because in congestion scenarios, the gradient elements can be significantly different from free flow scenarios.



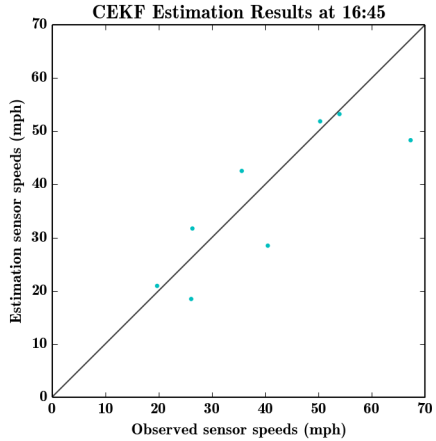
(2) FD-CEKF, 16:45



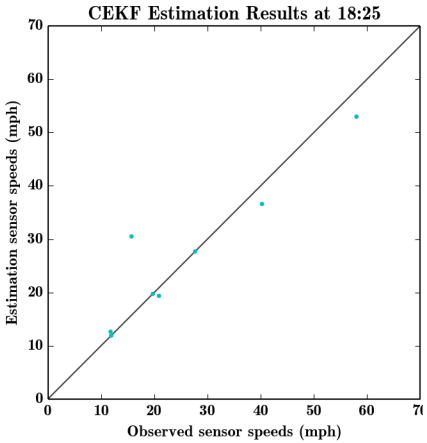
(3) CEKF with average gradient, 16:45



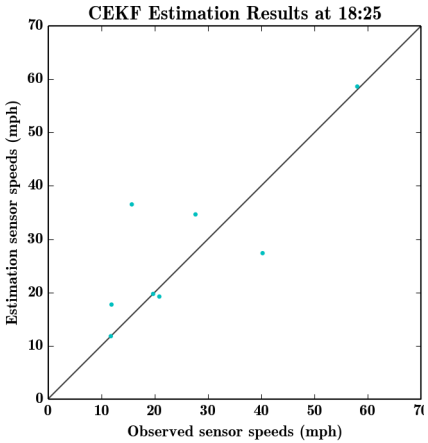
(4) FD-CEKF with $\boldsymbol{H}_{mask}$, 16:45



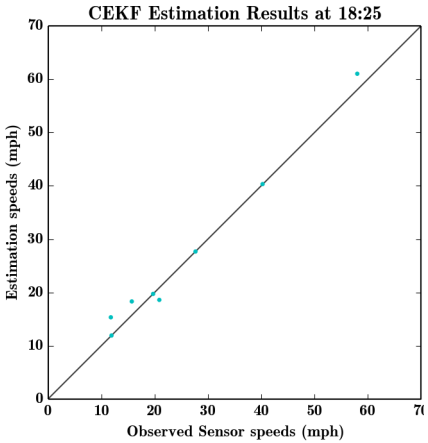
(5) CEKF with average gradient and $\boldsymbol{H}_{mask}$, 16:45



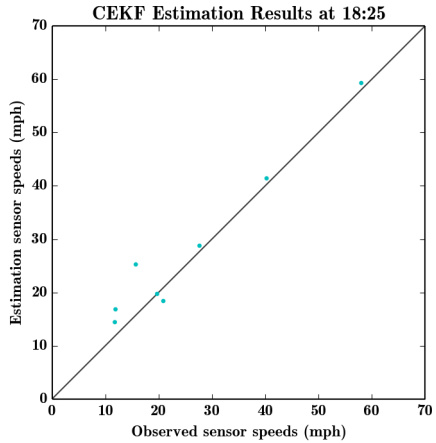
(2) FD-CEKF, 18:25



(3) CEKF with average gradient, 18:25



(4) FD-CEKF with $\boldsymbol{H}_{mask}$, 18:25



(5) CEKF with average gradient and $\boldsymbol{H}_{mask}$, 18:25

Figure 3-12: Scatter plot for observed speeds vs estimated speeds at time 16:45 and 18:25

3.7 Conclusion

In this chapter, we presented the issue of simulation stochasticity with the measurements of flow, speed and link travel time. Then we conducted an error analysis on simulation stochasticity and its impact on gradient estimation. Based on the error analysis, two solution approaches–quantifying simulated measurement covariance and applying gradient structure–were proposed. Two synthetic case studies proved their applicability and demonstrated improvements over existing approaches. There are several major conclusions: (1) speed and link travel time measurements are more prone to simulation stochasticity; (2) the stochasticity is likely to be present during the transition between free flows and congestion; (3) the stochasticity can be mitigated by quantifying the error covariance in the Kalman filtering framework; (4) the stochasticity introduces noise in the gradient estimation procedure, and enforcing a sparse matrix can improve the gradient accuracy, hence yielding better calibration results.

An additional challenge for the future work is how to separate the impact of supply from demand so that we do not overfit parameters to the measurements. For example, the speed reduction on a certain link implies increased average density. It may be caused by demand increase or lane closure. When we just have speed data for this link, we may not be able to identify the true cause. But if we know upstream and downstream links are reporting normal speed measurements, then it is likely to be the lane closure that caused the unusual speeds. Hence when both demand and supply parameters are calibrated together, it is necessary to consider the cause of effects. For this matter, it may also be beneficial to quantify the covariance between impact of supply and demand parameters. It is worth noting that demand-supply calibration is one important direction for future work.

Chapter 4

Towards Large-Scale Networks: Dynamic Bayesian Networks and State Augmentation

In Chapter 2, we presented recent developments of the state space model for DTA calibration. In this chapter, we first discuss the Markovian assumption in the state space model and its drawbacks with a delayed system. Then, we integrate the state space model into a general framework named Dynamic Bayesian Networks (DBNs). Dynamic Bayesian Networks are *directed graphical models* that capture the generation process of time-series data (Murphy, 2002). In particular, we examine a family of DBNs that overcome the Markovian assumption in the state space model. This model family can be solved by the Kalman filter with a technique called state augmentation. Finally, we present a synthetic case study on a toy network to illustrate the improvement over the standard extended Kalman filter and conclude the chapter.

4.1 The Markovian Assumption

In this section, we discuss the role of the Markovian assumption in the state space model using a Dynamic Bayesian Network (DBN) representation and illustrate the drawbacks of this assumption in a time-delayed system such as traffic networks.

We have presented the Extended Kalman filter and its application to online calibration for DTA. The underlying State Space Model (sometimes called hidden Markov model or Kalman filter model) is the focus of discussion in this chapter. For convenience, we present the state space model again in Figure 4-1. The shaded nodes are observed measurements and unshaded ones are latent state variables which cannot be directly measured. Within the state space model, random variables are assumed to be either discrete or continuous and Gaussian. In the DTA calibration literature, almost all the state space formulations assume a continuous parameter space and Gaussian errors. Thus, in this thesis we restrict our focus to continuous Gaussian random variables for the state space model with a DBN representation.

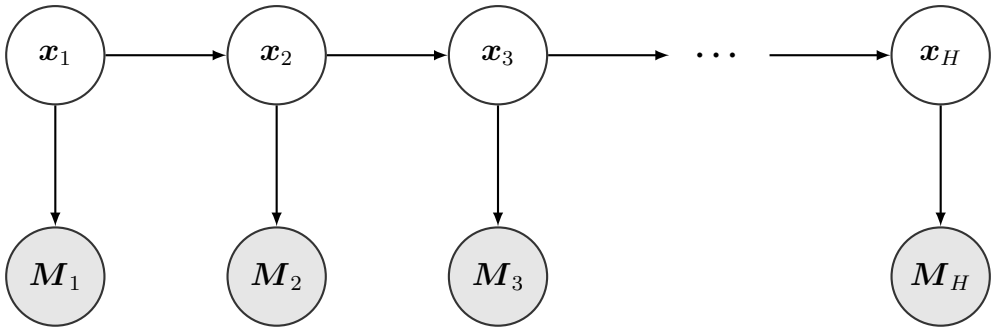


Figure 4-1: State space model with measurements

From a Dynamic Bayesian Network (DBN) perspective, Figure 4-1 exhibits a probabilistic directed graphical model structure that defines the factors of the joint probability: the directed edges depict conditional probability with connected nodes being random variables. Specifically,

$$f(\boldsymbol{x}_{1:H}, \boldsymbol{M}_{1:H}) = f(\boldsymbol{x}_1) f(\boldsymbol{M}_1|\boldsymbol{x}_1) \prod_{i=1}^{H-1} f(\boldsymbol{x}_{i+1}|\boldsymbol{x}_i) f(\boldsymbol{M}_{i+1}|\boldsymbol{x}_{i+1}) \tag{4.1}$$

Thus, a directed graph describes the generation of time-series data with conditional probabilities. This representation easily depicts conditional dependencies. We can intuitively find dependencies by checking the connectivity between nodes. For example in Figure 4-1, $\boldsymbol{x}_2$ uniquely determines $\boldsymbol{x}_3$. In other words, conditioned on

$\boldsymbol{x}_2$, $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$ are independent. Similarly, $\boldsymbol{x}_1$ does not affect $\boldsymbol{M}_2$ given $\boldsymbol{x}_2$. This is the Markovian/memoryless assumption in state space models. On the other hand, it is worth mentioning that $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$ are not independent, because they are both correlated with $\boldsymbol{x}_2$.

Now we shed some light on the Kalman filtering algorithm that solves the state space models. At step h, the prior of $\boldsymbol{x}_{h|h-1}$ is given by the transition equation from $\boldsymbol{x}_{h-1}$, and the posterior estimator $\boldsymbol{x}_{h|h}$ is updated from observing $\boldsymbol{M}_h$. Then $\boldsymbol{x}_{h|h}$ is used as the prior of $\boldsymbol{x}_{h+1|h}$, and the process continues. In the Kalman filter solution approach, previous states are not updated. This is helpful in the online setting, because we only need to update the estimate of $\boldsymbol{x}_h$ for each step. In other words, we reduce complexity of the parameter space from $\boldsymbol{x}_{1:h}$ to $\boldsymbol{x}_h$ for each time slice h.

Although the Markovian assumption simplifies the inference task, it may be problematic when we have a delayed system. Consider the case that the i-th element of latent variable $\boldsymbol{x}_h$–denoted by $\boldsymbol{x}_h(i)$–only has an impact on measurement $\boldsymbol{M}_{h+1}$ in time slice $h+1$. Then, an update of the latent variable $\boldsymbol{x}_h(i)$ with standard Kalman filtering techniques is impossible when we only know $\boldsymbol{M}_h$. We illustrate the example more intuitively with its corresponding DBN representation in Figure 4-2. Such a representation conforms with the measurement equation in the state space model exhibited in Equation (4.2) with $q = 2$. Noticeably the Markovian assumption no longer holds, and the previous structure ignores the true diagonal relations.

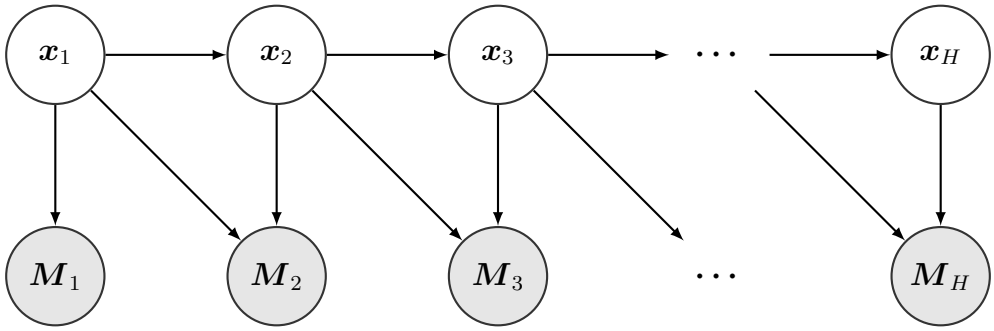


Figure 4-2: A DBN with measurement equation contradicting the Markovian assumption

$$\boldsymbol{M}_h = \boldsymbol{A}_h^h \boldsymbol{x}_h + \boldsymbol{A}_h^{h-1} \boldsymbol{x}_{h-1} + \cdots + \boldsymbol{A}_h^{h-q+1} \boldsymbol{x}_{h-q+1} + \boldsymbol{v}_h \tag{4.2}$$

Similarly, the Markovian assumption may also fail when a state have dependencies on states at 2 or more time slices earlier. For instance, $\boldsymbol{x}_3$ depends on $\boldsymbol{x}_1$, as suggested in Equation (2.17), which is also rewritten here as Equation (4.3). Figure 4-3 shows the corresponding structure in the DBN representation for $p = 2$.

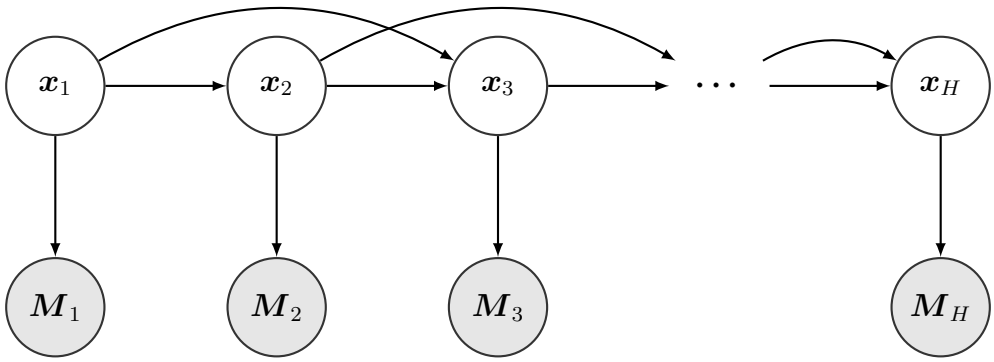


Figure 4-3: A DBN with transition equation contradicting Markovian assumption

$$\boldsymbol{x}_h = \boldsymbol{F}_h^{h-1} \boldsymbol{x}_{h-1} + \boldsymbol{F}_h^{h-2} \boldsymbol{x}_{h-2} + \cdots + \boldsymbol{F}_h^{h-p} \boldsymbol{x}_{h-p} + \boldsymbol{w}_h \tag{4.3}$$

Now we work through an OD estimation example to illustrate the impact of violating the Markovian assumption.

4.2 OD Estimation Example Violating the Markovian Assumption

In this section, we present an example for OD estimation. The true model is shown in Figure 4-2, which violates the Markovian assumption. We now analyze the impact of neglecting the relations in the true model by estimating OD flows with the model in Figure 4-1.

4.2.1 Toy road network example and basic assumptions

Figure 4-4 exhibits a toy road network and two OD pairs. $s1$ and $s2$ are two flow-count sensors that report aggregated flow within each 5-minute time interval. The objective is to infer OD flows in each time interval after measuring sensor flow counts.

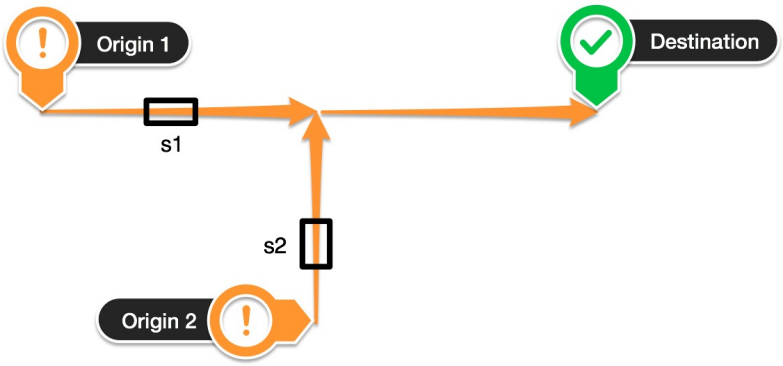


Figure 4-4: A road network example and sensor placement that ensures no delay in capturing the states

In this example, we make three assumptions:

(1) Each link takes 1 time interval to traverse;

(2) All vehicles will travel the same distance within each interval, meaning a sensor either captures all or nothing from an OD pair in each interval;

(3) There is no measurement error in sensor flow counts.

Table 4.1 exhibits an example of the OD and sensor flows in two intervals. Note $s1$ only captures O_1D in the same interval and $s2$ captures O_2D. The OD flow inference is instant: we can read off measurements as OD flows. We make an important observation that the system has no time-delay and the state space model in Figure 4-1 is accurate.

Now we introduce delay in the time at which measurements capture the state vector: we change the sensor placement scheme to the one shown in Figure 4-5. The measurements are listed in Table 4.2. The key change is that now $s3$ captures O_1D and O_2D in the previous interval. This introduces correlation between states and

Table 4.1: Example OD and sensor flows for toy network

	t=1	t=2
O_1D	30	24
O_2D	20	18
s1	30	24
s2	20	18

measurements across time intervals, making the Markovian assumption invalid. We can still read off $s2$ to estimate O_2D, but we have no information about O_1D at $t = 1$ unless we also know $s3$ at $t = 2$.

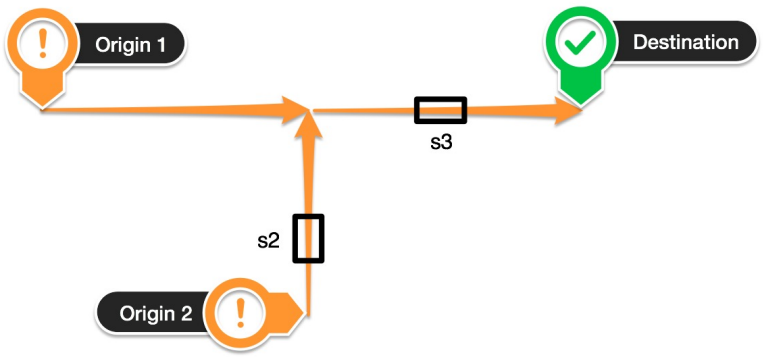


Figure 4-5: A sensor placement scheme with lag between OD and flow counts

Table 4.2: Example OD and sensor flows for toy network

	t=1	t=2
O_1D	?	?
O_2D	20	18
s2	20	18
s3	0	50

4.2.2 Iterations of the Kalman filter with the toy example

In the remainder of this section, several iterations of the Kalman filter are presented. The settings are listed in Equation (4.4). Based on the Kalman filter update rule, the first iteration gives us Equation (4.5) at $t = 1$.

$$
\begin{aligned}
\boldsymbol{x}_h =& \boldsymbol{F}\boldsymbol{x}_{h-1} + \boldsymbol{w} \\
\boldsymbol{M}_h =& \boldsymbol{A}\boldsymbol{x}_h + \boldsymbol{v} \\
\boldsymbol{x} = \begin{pmatrix} O_1D \\ O_2D \end{pmatrix}; \boldsymbol{M} =& \begin{pmatrix} s2 \\ s3 \end{pmatrix} \\
\boldsymbol{x}_{0|0} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}; \boldsymbol{F} = \begin{bmatrix} 0.8 & 0 \\ 0 & 0.9 \end{bmatrix}; \boldsymbol{A} =& \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \\
\boldsymbol{P}_{0|0} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}; \boldsymbol{Q} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}; \boldsymbol{R} =& \begin{bmatrix} \epsilon & 0 \\ 0 & \epsilon \end{bmatrix}, \epsilon \ll 1
\end{aligned}
\tag{4.4}
$$

$$
\begin{aligned}
\boldsymbol{x}_{1|0} = \boldsymbol{F}\boldsymbol{x}_{0|0} =& \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
\boldsymbol{P}_{1|0} = \boldsymbol{F}\boldsymbol{P}_{0|0}\boldsymbol{F}^\top + \boldsymbol{Q} =& \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \\
\boldsymbol{S}_1 = \boldsymbol{A}\boldsymbol{P}_{1|0}\boldsymbol{A}^\top + \boldsymbol{R} =& \begin{bmatrix} 10 & 0 \\ 0 & \epsilon \end{bmatrix} \\
\boldsymbol{K}_1 = \boldsymbol{P}_{1|0}\boldsymbol{A}^\top\boldsymbol{S}_1^{-1} =& \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \\
\boldsymbol{x}_{1|1} = \boldsymbol{x}_{1|0} + \boldsymbol{K}_1(\boldsymbol{M}_1 - \boldsymbol{A}\boldsymbol{x}_{1|0}) =& \begin{pmatrix} 0 \\ 20 \end{pmatrix} \\
\boldsymbol{P}_{1|1} = \boldsymbol{P}_{1|0} - \boldsymbol{K}_1\boldsymbol{A}\boldsymbol{P}_{1|0} =& \begin{bmatrix} 10 & 0 \\ 0 & \epsilon \end{bmatrix}
\end{aligned}
\tag{4.5}
$$

From the first iteration, it is obvious that O_1D cannot be estimated from the measurement update. Hence, based on the above calculated results, we present the second iteration. The updates for $t = 2$ are in Equations (4.6).

$$
\begin{aligned}
\boldsymbol{x}_{2|1} &= \boldsymbol{F}\boldsymbol{x}_{1|1} = \begin{pmatrix} 0 \\ 18 \end{pmatrix} \\
\boldsymbol{P}_{2|1} &= \boldsymbol{F}\boldsymbol{P}_{1|1}\boldsymbol{F}^\top + \boldsymbol{Q} = \begin{bmatrix} 16.4 & 0 \\ 0 & 10 \end{bmatrix} \\
\boldsymbol{S}_2 &= \boldsymbol{A}\boldsymbol{P}_{2|1}\boldsymbol{A}^\top + \boldsymbol{R} = \begin{bmatrix} 10 & 0 \\ 0 & \epsilon \end{bmatrix} \\
\boldsymbol{K}_2 &= \boldsymbol{P}_{2|1}\boldsymbol{A}^\top \boldsymbol{S}_2^{-1} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \\
\boldsymbol{x}_{2|2} &= \boldsymbol{x}_{2|1} + \boldsymbol{K}_2(\boldsymbol{M}_2 - \boldsymbol{A}\boldsymbol{x}_{2|1}) = \begin{pmatrix} 0 \\ 18 \end{pmatrix} \\
\boldsymbol{P}_{2|2} &= \boldsymbol{P}_{2|1} - \boldsymbol{K}_2\boldsymbol{A}\boldsymbol{P}_{2|1} = \begin{bmatrix} 16.4 & 0 \\ 0 & \epsilon \end{bmatrix}
\end{aligned}
\tag{4.6}
$$

As expected, O_1D was not updated at $t = 2$. In addition, we have two major observations: 1) the estimate of O_1D only relies on the transition model. Even in our case where the transition model is perfect, a biased initial point $\boldsymbol{x}_0 = \boldsymbol{0}$ and no measurement update result in a biased estimate; 2) the posterior variance of O_1D in $\boldsymbol{P}_{h|h}$ does not decrease with t, because the error from transition model accumulates when there is no update from measurements.

We conclude this section with the claim that failing to model measurement correlation across intervals could lead to no update for hidden states. We also demonstrated the growth of the estimated variance. In the following sections, the state augmentation technique will be presented and applied to the same example.

4.3 Solution Approach

4.3.1 The State Augmentation Technique

The transition and measurement equation in a delayed traffic system (Equations (4.2) and (4.3)) presented in Okutani & Stephanedes (1984); Ashok & Ben-Akiva (1993) will lead to the model shown in Figure 4-2. It contradicts the Markovian assumption

and thus, cannot be solved directly by Kalman filtering techniques. Fortunately, this is only true if we define parameters $\boldsymbol{x}_h$ as the state vector. According to (Ashok & Ben-Akiva, 1993), we could augment the state to include parameters at different time slices. In DBN terms, we create a super latent node for adjacent time slices, for example in Figure 4-6. The *degree of augmentation* is defined as the number of intervals to include when constructing the super node. The degree of augmentation is the maximum degree in the transition and measurement equation. Thus, the additional edges contradicting the Markovian assumption are absorbed into the edges between the super nodes. This would enforce that there is no edge between super latent nodes and measurements at different time slices. For instance, the relation between $\boldsymbol{x}_2$ and $\boldsymbol{M}_3$ is represented in edge $\{\boldsymbol{x}_2, \boldsymbol{x}_3\} \rightarrow \boldsymbol{M}_3$; the transition between $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$ is captured in edge $\{\boldsymbol{x}_1, \boldsymbol{x}_2\} \rightarrow \{\boldsymbol{x}_2, \boldsymbol{x}_3\}$. The resulting model captures both structures in Figures 4-2 and 4-3 are shown in Figure 4-6.

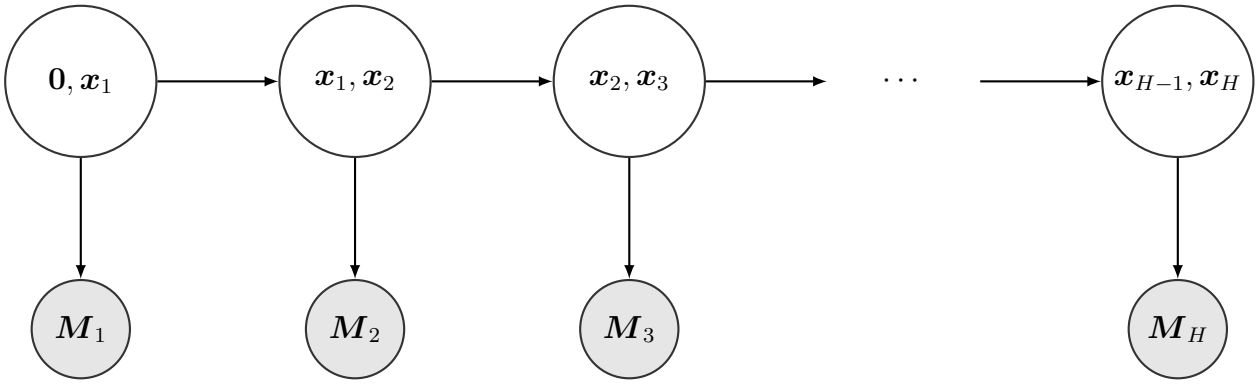


Figure 4-6: A state space model with augmented states, mitigating the issue posed in Figure 4-2 and Figure 4-3

Corresponding to the augmentation of states, the measurement and transition equation become Equation (4.7), according to (Ashok, 1996).

$$\begin{aligned} \boldsymbol{\mathcal{X}}_h &= \boldsymbol{\Phi}_{h-1}\boldsymbol{\mathcal{X}}_{h-1} + \boldsymbol{W}_h \\ \boldsymbol{M}_h &= \boldsymbol{\mathcal{A}}_h\boldsymbol{\mathcal{X}}_h + \boldsymbol{v}_h \end{aligned} \tag{4.7}$$

where, *the degree of augmentation* is $r = \max\{p, q\}$. Assuming n is the length of $\boldsymbol{x}_h$ and m is the length of $\boldsymbol{M}_h$, we define the following quantities:

Augmented state:

$$\boldsymbol{\mathcal{X}}_h = \begin{pmatrix} \boldsymbol{x}_h \\ \boldsymbol{x}_{h-1} \\ \vdots \\ \boldsymbol{x}_{h-r+1} \end{pmatrix} \tag{4.8}$$

Transition matrix:

$$\boldsymbol{\Phi}_h = \begin{bmatrix} \boldsymbol{\mathcal{F}_h} & \\ \boldsymbol{I}_{(r-1)n\times(r-1)n} & \boldsymbol{0}_{(r-1)n\times n} \end{bmatrix} \tag{4.9}$$

where,

$$\boldsymbol{\mathcal{F}}_h = \begin{bmatrix} \boldsymbol{F}_h^{h-1} & \boldsymbol{F}_h^{h-2} & \cdots & \boldsymbol{F}_h^{h-r} \end{bmatrix}_{n\times rn}$$

if $p < r$, $\boldsymbol{F}_h^{h-j} = \boldsymbol{0}_{n\times n}, \forall j = p+1, ..., r$

$$\begin{aligned} \boldsymbol{\mathcal{A}}_h &= \left[\boldsymbol{A}_h^h, \boldsymbol{A}_h^{h-1}, \cdots, \boldsymbol{A}_h^{h-r+1}\right]_{m\times rn} \\ \boldsymbol{W}_h &= \begin{pmatrix} \boldsymbol{w}_h \\ \boldsymbol{0}_{(r-1)\times n} \end{pmatrix} \end{aligned} \tag{4.10}$$

We have some critical comments: (1) with augmentation of the states, the dimension of covariance matrix $\boldsymbol{P}$ is now $r \times r$ times greater, making the matrix multiplication more cumbersome. Hence a more computationally complex Kalman filter iteration may be disadvantageous to real-time deployment. (2) At each step, parameters in previous intervals are adjusted along with current ones. The relation between these parameters and measurements are revealed in the augmented Jacobian $\boldsymbol{\mathcal{A}}_h$ in the measurement equation. However, obtaining them needs more effort. When using finite difference, the number of required runs for $\boldsymbol{\mathcal{A}}_h$ is $O(nr)$.

4.3.2 State Augmentation on the OD Estimation Example

With the state augmentation technique, we revisit the OD estimation problem. The updated settings are listed in Equation (4.11).

$$
\begin{aligned}
\boldsymbol{\mathcal{X}}_h =& \boldsymbol{\mathcal{F}}\boldsymbol{\mathcal{X}}_{h-1} + \boldsymbol{W} \\
\boldsymbol{M}_h =& \boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{X}}_h + \boldsymbol{v} \\
\boldsymbol{\mathcal{X}}_h = \begin{pmatrix} \boldsymbol{x}_h \\ \boldsymbol{x}_{h-1} \end{pmatrix}; & \boldsymbol{M} = \begin{pmatrix} s2 \\ s3 \end{pmatrix} \\
\boldsymbol{\mathcal{X}}_{0|0} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}; & \boldsymbol{\mathcal{F}} = \begin{bmatrix} 0.8 & 0 & 0 & 0 \\ 0 & 0.9 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}; \boldsymbol{\mathcal{A}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \\
\boldsymbol{P}_{0|0} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}; & \boldsymbol{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}; \boldsymbol{R} = \begin{bmatrix} \epsilon & 0 \\ 0 & \epsilon \end{bmatrix}, \epsilon \ll 1
\end{aligned}
\tag{4.11}
$$

The outcome of the first iteration is as follows:

$$
\begin{aligned}
\boldsymbol{\mathcal{X}}_{1|0} = \boldsymbol{\mathcal{F}}\boldsymbol{\mathcal{X}}_{0|0} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad & \boldsymbol{P}_{1|0} = \boldsymbol{\mathcal{F}}\boldsymbol{P}_{0|0}\boldsymbol{\mathcal{F}}^\top + \boldsymbol{Q} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
\boldsymbol{S}_1 = \boldsymbol{A}\boldsymbol{P}_{1|0}\boldsymbol{A}^\top + \boldsymbol{R} = \begin{bmatrix} 10 & 0 \\ 0 & \epsilon \end{bmatrix}, & \boldsymbol{K}_1 = \boldsymbol{P}_{1|0}\boldsymbol{A}^\top\boldsymbol{S}_1^{-1} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \\
\boldsymbol{\mathcal{X}}_{1|1} = \boldsymbol{\mathcal{X}}_{1|0} + \boldsymbol{K}_1(\boldsymbol{M}_1 - \boldsymbol{A}\boldsymbol{\mathcal{X}}_{1|0}) = \begin{pmatrix} 0 \\ 20 \\ 0 \\ 0 \end{pmatrix}, \quad & \boldsymbol{P}_{1|1} = \boldsymbol{P}_{1|0} - \boldsymbol{K}_1\boldsymbol{A}\boldsymbol{P}_{1|0} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & \epsilon & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}
\end{aligned}
\tag{4.12}
$$

And a second iteration gives:

$$\boldsymbol{\mathcal{X}}_{2|1} = \boldsymbol{\mathcal{F}}\boldsymbol{\mathcal{X}}_{1|1} = \begin{pmatrix} 0 \\ 18 \\ 0 \\ 20 \end{pmatrix}, \quad \boldsymbol{P}_{2|1} = \boldsymbol{\mathcal{F}}\boldsymbol{P}_{1|1}\boldsymbol{\mathcal{F}}^\top + \boldsymbol{Q} = \begin{bmatrix} 16.4 & 0 & 8 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 8 & 0 & \epsilon \end{bmatrix}$$

$$\boldsymbol{S}_2 = \boldsymbol{A}\boldsymbol{P}_{2|1}\boldsymbol{A}^\top + \boldsymbol{R} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, \quad \boldsymbol{K}_2 = \boldsymbol{P}_{2|1}\boldsymbol{A}^\top\boldsymbol{S}_2^{-1} = \begin{bmatrix} 0 & 0.8 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{4.13}$$

$$\boldsymbol{\mathcal{X}}_{2|2} = \boldsymbol{\mathcal{X}}_{2|1} + \boldsymbol{K}_2(\boldsymbol{M}_2 - \boldsymbol{A}\boldsymbol{\mathcal{X}}_{2|1}) = \begin{pmatrix} 24 \\ 18 \\ 30 \\ 20 \end{pmatrix}, \quad \boldsymbol{P}_{2|2} = \boldsymbol{P}_{2|1} - \boldsymbol{K}_2\boldsymbol{A}\boldsymbol{P}_{2|1} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

With the result $\boldsymbol{\mathcal{X}}_{2|2}$, it can be concluded that the true OD values for $t = 1$ and $t = 2$ are recovered, although the $O_2D = 24$ at $t = 2$ is not directly from $\boldsymbol{M}_3$ but due to the perfect transition equation and perfect estimates for $t = 1$.

Thus, we demonstrate the power of state augmentation with a simple network. When the sensor placement is configured such that not all OD flows are captured by sensors in the same interval, state augmentation may benefit the OD estimation problem by identifying sensor-OD flow relations across intervals. For this toy network, solving the standard state space model cannot correctly estimate some OD values, while state augmentation gives exact estimates for those values.

4.4 Synthetic Case Study

4.4.1 Synthetic Network and Data Generation

We demonstrate the performance of state augmentation with an online OD estimation example on the toy network shown in Figure 3-1. However, the supply parameters in Table 3.1 need to be modified to make the augmented model valid. Under free flow conditions in Table 3.1, the main stream OD travel time is 54 and 60 seconds for

two routes. Thus, 80% of the traffic flow will be captured in a 5-minute simulation interval. We propose the following network specifications in Table 4.3 by reducing the free flow speeds. After the reduction, the main stream OD travel time is reduced to 75 and 84 seconds, which is around 40% increase. With this change, the mainstream OD travel time will exceed 300 seconds after 30 minutes simulation under the moderate demand in Table 3.2. In such case, the congestion will make the augmented model valid. The network topology is also presented here again for convenience in Figure 4-7.

Table 4.3: Specifications of each segment on the toy network with reduced free flow speeds

Segment ID	1	2	3	4	5	6	7	8
Length (meter)	297.5	553.8	493.1	351.2	408.6	666.7	377.3	183.0
Free flow speed (mph)	50	50	50	50	20	50	50	50
Minimum travel time (second)	13.31	24.8	22.1	15.7	45.7	29.8	16.9	8.19

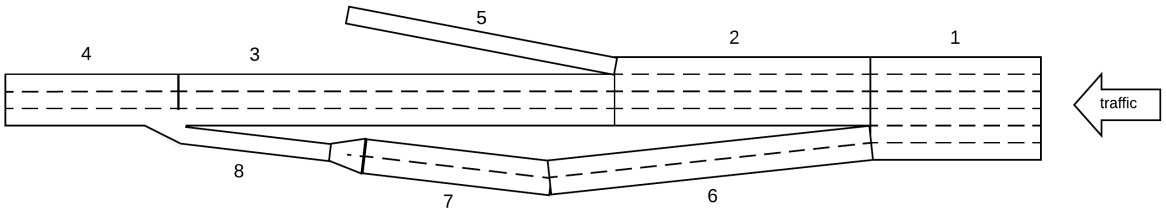


Figure 4-7: Toy road network, traffic going to left

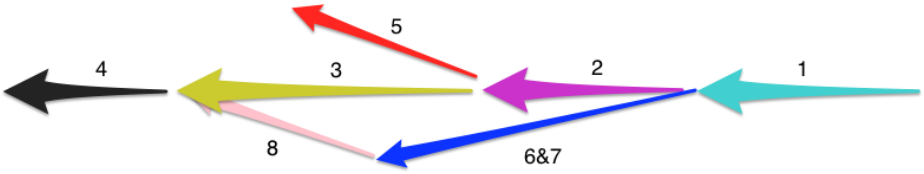


Figure 4-8: Topology of segments in the same color as Figure 4-9

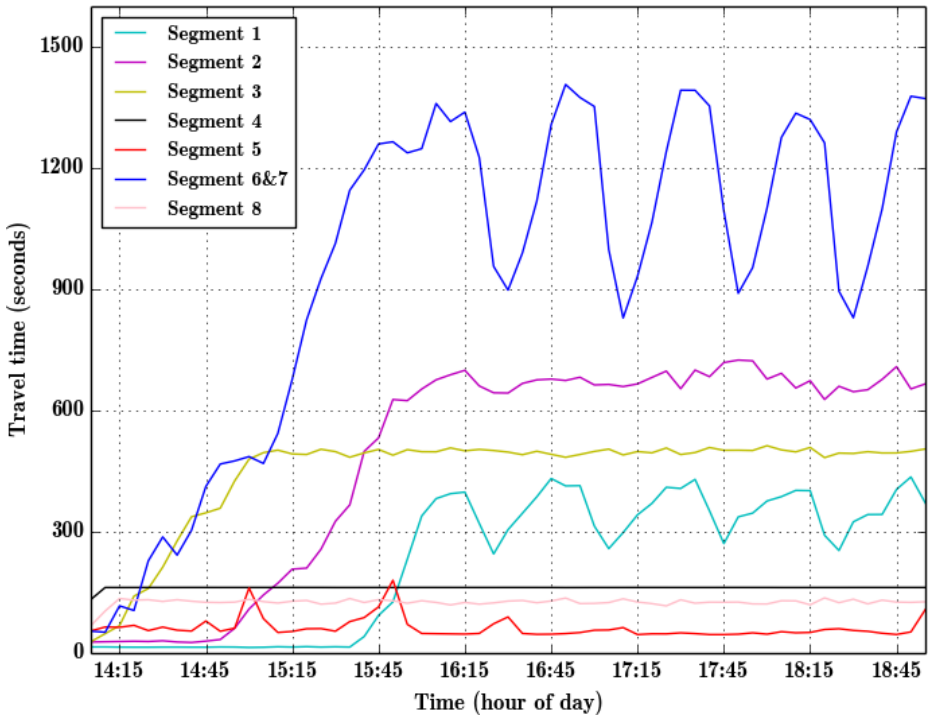


Figure 4-9: Link travel times on the toy network with the modified supply parameters

Figure 4-9 presents the link travel times obtained by assigning demand in Table 3.2 with the reduced free flow speeds. It is evident that the congestion from Segment 4 propagates backward to Segment 8 and 3, and then affects Segments 6, 7 and 2 and finally 1. The oscillation in Segment 6, 7 and 2 may be because of stop and go traffic conditions and may be affected by the simulator stochasticity. But it is certain that the travel time of Segment 1 will exceed 5 minutes when congestion is present. In such cases, the traffic flow only passes sensors on Segment 2 in the next interval. It takes over 900 seconds for congested flows to reach Segment 5. Thus, we have constructed the test scenario that violates the Markovian assumption.

We briefly summarize the data generation process. The parameters are time-dependent OD flows. The surveillance data are flow counts in every 5 minutes for the simulation period 14:00-19:00. Similar to the example in Section 3.6.1, we treat DynaMIT as the real world that generates the surveillance data.

4.4.2 Calibration Procedure

Based on the synthetic data, we perform online calibration with the following settings of the Kalman filter. Since the true time-dependent demand is known, we can obtain a true AR process. According to the Akaike information criterion (AIC), the best model was found be AR(5) which is hereafter used for the transition equation. For the gradient estimation, finite difference is used. As for the solution algorithm, the Constrained Extended Kalman Filter is applied. To compare different degrees of state augmentation, the following three models are considered:

(1) CEKF(1): CEKF with original state space model, AR(5) transition model

(2) CEKF(2): CEKF with 2nd-order augmented state space model, AR(5) transition model

(3) CEKF(5): CEKF with 5th-order augmented state space model, AR(5) transition model

We have a comment on the implementation of the AR model. As discussed in Chapter 2, when applying the Kalman filtering technique with AR models, the convention is to use the approximation of state augmentation. It is essentially keeping the transition model but not augmenting the state for measurement update. Thus, it is possible to augment the state to a degree that is lower than the transition AR degree.

4.4.3 Results

We present the RMSN results in the following table:

Table 4.4: Flow RMSN for state estimation and predictions for 15:00-19:00

Experiment	Estimation	Prediction RMSN		
	RMSN	Step 1	Step 2	Step 3
CEKF(1)	13.5%	21.0%	26.2%	34.7%
CEKF(2)	9.8%	18.8%	24.2%	31.9%
CEKF(5)	10.8%	15.4%	19.3%	26.6%

Section 4.4.3 illustrates the performance of the three models with the same AR(5) transition equation. For state estimation, CEKF(2) and CEKF(5) have smaller errors than CEKF(1), while CEKF(2) has the best estimation accuracy. However, for prediction performance, CEKF(5) outperforms CEKF(2), which in turn outperforms CEKF(1). This is probably because the CEKF(5) model estimates OD flows more accurately in the congestion scenario, after 16:00 (see Figure 4-9).

Figure 4-10 presents the scatter plots for simulation period 15:00-19:00 where points closer to the diagonal indicate a better fit. For the state estimation result in the first row, points in CEKF(2) (middle) and CEKF(5) (right) are closer to the diagonal line than CEKF(1). For prediction results in the second to fourth row, CEKF(1) has more points below the diagonal, which means it tends to underestimate the flow. This is reasonable because CEKF(1) is "myopic" and can only see the OD flows' influence on the same interval. In congestion scenarios, the estimated gradient is close to zero, because perturbing the input OD flows does not change the saturated flow rate. Thus, CEKF(1) does not increase the flow because it is irrelevant. However, CEKF(5) can capture the long-term affect of changing OD flows. Specifically, after perturbing OD flows, although the first-order gradient is zero, the higher order gradients still capture the impact of the perturbation. Thus, CEKF(5) will utilize the higher order gradients for OD calibration. This results in the more balanced fit in the prediction scatter plots.

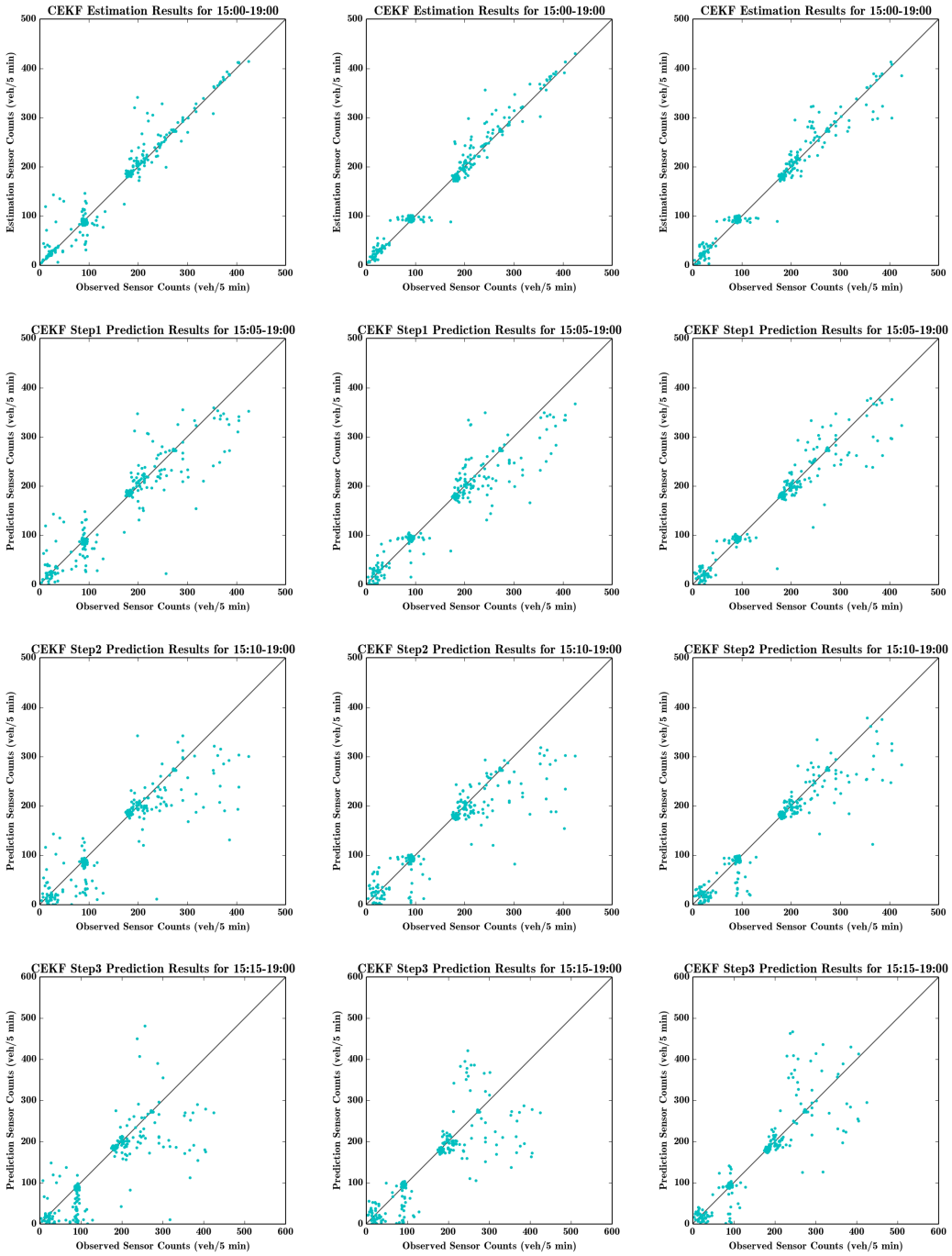


Figure 4-10: Scatter plot for estimated/predicted vs observed flow counts: left: CEKF(1), middle: CEKF(2), right: CEKF(5)

4.5 Conclusion

In this chapter, we discussed the drawbacks of the original state space model and showed with a simple example that in some situations, certain states are not identifiable. We then presented the state augmentation technique within the Dynamic Bayesian Network framework. With the augmented state space model, the Kalman filter is capable of updating parameters in previous intervals, thus having a "long-term" view. Finally, we presented a synthetic case study in a scenario with congestion and demonstrated the power of the augmented state space model. A case study with real world data will be discussed in Chapter 6.

Chapter 5

Towards Real-Time Performance: Accelerating Gradient Estimation

In Chapter 4, we presented the state augmentation technique to account for the delayed observation of hidden states. Employing state augmentation will increase the dimension of the state vector and consequently, the gradient matrix (H matrix, or system Jacobian). This generally means that the gradient estimation process will be more time-consuming which is a challenge for real-time applications. For example, each 5-minute interval in the case study in Chapter 6 needs around 30 minutes of computational time, even without state augmentation on a 20 core server. A large proportion of the computational time consumed by online calibration is spent on gradient estimation (finite difference is applied to obtain the H matrix). Thus, direct application of the FD-EKF and state augmentation to real-time DTA systems is impossible. In this chapter, we attempt to accelerate the gradient estimation process by finding more computationally efficient approaches.

The structure of this chapter is as follows. First, based on the sparse structure of the gradient matrices, a Partitioned Simultaneous Perturbation (PSP) approach is proposed to approximate Finite Difference (FD), based on Simultaneous Perturbation(SP). Then, a time-series model for the H matrices in different intervals is proposed, with the intention of reducing total number of runs for gradient estimation. Following that, a non-parametric approach based on K-Nearest-Neighbors (KNN) is

discussed. Finally the results are presented and commented upon.

5.1 Partitioned Simultaneous Perturbation

In case of the simultaneous perturbation approach introduced in Section 2.2.2, the inaccurate gradient estimation comes from the fact that perturbations on different parameters will have impacts on the same measurement. This will result in systematic overestimation or underestimation. For instance, if perturbing two parameters cancels out their effects, the obtained gradient will be zero for both of them. Hence it is generally better to perturb two parameters simultaneously only if they both do not affect the same set of sensors.

A simple idea is to divide different parameters into partitions such that in each partition, any two parameters are not captured by any common sensors. Its feasibility, especially on large networks, can be intuitively understood with the following example. We assume a large road network with a large number of OD pairs and segments. The OD pairs within the west would not be captured by the surveillance data on the east and vice versa. The same conclusion holds for segment supply parameters, as they will only affect local traffic conditions. Hence, we can group one parameter in the west and another in the east into the same partition, followed by simultaneous perturbation. This partitioning guarantees that no sensors are affected by perturbations from other parameters in the same partition, and thus, the influence on each sensor is neither exaggerated nor canceled.

5.2 Related Work of Gradient Estimation

There has been a lot of research in gradient estimation. As summarized in Fu (2015, 2002), there are several methods to obtain a gradient matrix. The author divided the approaches for stochastic gradient estimation into two categories: indirect and direct. An indirect estimator obtains an approximation of the true gradient value, and it relies on only function evaluations of the system. The direct estimation approach

attempts to obtain the true gradient with the help of derivations of the stochasticity for each case-specific problem. In our online calibration framework, traffic simulators are treated as a black-box and the mapping of inputs and measurements cannot be derived in closed-form. Thus, the direct approach is not applicable to the gradient estimation in DTA calibration. Hence, the focus is on the indirect estimators. As mentioned in Fu (2015) and also reviewed in Section 2.2.2, the two major approaches for indirect gradient estimation are finite difference and simultaneous perturbation. As mentioned in the beginning of this section, finite difference has great computational complexity. While the simultaneous perturbation approach is more efficient, its drawback is that the resulting gradient matrix has only rank 1. Thus, it is extremely noisy and significantly less accurate when compared to finite difference. In summary, for gradient estimation, simultaneous perturbation and finite difference are at two extremes of the trade-off between computational complexity and approximation accuracy. Thus, it is necessary to enrich the family of methods for indirect gradient estimation with other approaches that are both accurate and computationally tractable.

While the estimation of sparse Jacobian was discussed thoroughly in Coleman & Moré (1983), the use of the partitioned simultaneous perturbation (PSP) idea in DTA calibration was first proposed by Huang (2010). According to the author's case study, PSP-EKF was reported to be 10 times faster than FD-EKF although, as expected, the calibration result is less accurate. A heuristic approach is described to conduct the partitioning based on previous estimated gradients. However, the author did not consider its generalization for all parameters (both OD and supply parameters), primarily due to the difficulty in identifying a correct structure for the true gradient matrix. Moreover, the work also lacked an analysis of the PSP and differences from the FD. In this section, we address these undiscussed issues via a thorough development of the PSP approach for gradient estimation.

5.2.1 Problem Definition

First and foremost, it is beneficial to summarize the composition of the gradient matrix (H matrix). Each H matrix has m rows corresponding to m measurements and n columns for n parameters. These dimensions are the same across time intervals. The finite difference (FD) approach perturbs each parameter twice to obtain one column of H.

The partitioned simultaneous perturbation is an approach for the gradient estimation problem. It aims to approximate finite difference with the least computations possible, assuming knowledge of the gradient structure. PSP comprises 3 procedures/subproblems: (1) gradient structure identification; (2) parameter partitioning; (3) simultaneous perturbation for gradient estimation. We define each problem as follows.

(1) **Gradient structure identification**: obtain the incidence matrix $\boldsymbol{H}_{inc}$ to identify the sparse structure of the gradient matrix.

(2) **Parameter partitioning**: divide n parameters into minimum p partitions such that no two parameters in the same partition relate to any same measurements. The parameters in these p partitions should be mutually exclusive and collectively exhaustive.

(3) **Simultaneous perturbation for gradient estimation**: for each partition, perturb all belonging parameters in two opposite directions and calculate the difference in measurements to form each column of a compressed matrix $\boldsymbol{C}$ of dimension $m \times p$.

Now we analyze each problem and discuss solution approaches in the following section.

5.3 Solution Approaches

5.3.1 Gradient Structure Identification

The term *structure* in this context refers to the locations of zeros and non-zeros in the gradient matrix. The gradient structure is necessary for the partitioning method to determine which parameters can be grouped together. An incidence matrix $\boldsymbol{H}_{inc}$ is a representation for the gradient structure. $\boldsymbol{H}_{inc,(i,j)}$ is 1 if measurement i and parameter j is related and 0 if not. An incidence matrix is obtained from:

$$\boldsymbol{H}_{inc,(i,j)} = \begin{cases} 1 & \text{if } \boldsymbol{H}_{(i,j)} \neq 0 \\ 0 & \text{if } \boldsymbol{H}_{(i,j)} = 0 \end{cases} \tag{5.1}$$

One may ask about the difference between the gradient incidence matrix $\boldsymbol{H}_{inc}$ and the H mask $\boldsymbol{H}_{mask}$ in Section 3.5.2 and eq. (3.18). Here, we claim a subtle but clear distinction between the $\boldsymbol{H}_{inc}$ and $\boldsymbol{H}_{mask}$. $\boldsymbol{H}_{mask}$ indicates the structure of the *expectation of gradient*, in which case, elements in $\boldsymbol{H}_{mask}$ will be 0 only if they capture pure noise from simulation stochasticity. On the other hand, $\boldsymbol{H}_{inc}$ represents the structure of all possible gradients, even if they only contain noise. In other words, $\boldsymbol{H}_{inc}$ represents the least sparse case during the calibration period so as to separate the impact of parameters on the same sensor. Thus, $\boldsymbol{H}_{inc}$ should be denser than $\boldsymbol{H}_{mask}$. Using the sparser $\boldsymbol{H}_{mask}$ for partitioning neglects the noise in gradients, which yields less partitions but more chance that parameters will impact the same measurement. So it is generally preferable to use $\boldsymbol{H}_{inc}$.

There are two general comments about the gradient structure. First, the partitioning relies on the sparse nature of the gradient. More sparsity means less shared measurements among parameters, thus preferably resulting in less partitions. Second, the gradient structure may change across intervals. Hence, if we assume a gradient structure beforehand, it must cover all possible structures across all intervals. In other words, the overall $\boldsymbol{H}_{inc}$ should be the result of *OR* operation of $\boldsymbol{H}_{inc,h}$ for all

possible interval h. In such a case, the partitioning needs to be done only once.

5.3.2 Parameter Partitioning

Given the gradient incidence matrix, we are ready to perform the partitioning. While the task is simply grouping non-conflicting parameters, it may not be so simple as it seems. In this discussion, we first reduce the issue to a graph coloring problem, followed by a heuristic algorithm for the graph coloring. Then we extend the problem to non-sparse cases where the non-zero gradients are affected by noise.

Graph Coloring Problem

First we reduce the partitioning problem to a graph coloring problem. We recall that each column of $\boldsymbol{H}_{inc}$ is the impact of each parameter on all the measurements. We want to group two parameters that do not affect the same sensor. In other words, any two 1s in the same row disqualifies grouping of the two corresponding parameters. The term *conflict* is used to describe the fact that two parameters have gradient values with any same sensor. We call these rows *conflicting* for a given pair of parameters. In a graphical representation, we denote the parameters as nodes, and each pair of nodes that has *conflicting* rows are connected by edges. In this regard, the partitioning problem is equivalent to finding minimum colors for all the nodes such that no two connected nodes have the same color.

For example, Figure 5-1 presents a gradient incidence matrix, and the corresponding graph representation. The first row in $\boldsymbol{H}_{inc}$ shows that Node 1, 2 and 6 are connected, thus must be assigned with different colors. In this particular example, 3 partitions mean the number of finite difference calculations are reduced from 6 to 3.

There are two major comments about the graph coloring problem, according to Coleman & Moré (1983). First, finding minimum number of colors is NP-hard. Second, there are numerous algorithms that try to find the optimal coloring with heuristics. However, there are cases where any algorithm will perform poorly.

$$\boldsymbol{H}_{inc} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

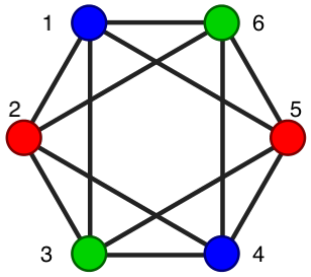


Figure 5-1: A gradient incidence matrix and its corresponding optimal graph with 3 colors

Sequential Graph Coloring Algorithm

Fascinating as the graph coloring problem is, our focus in this thesis is not inventing an algorithm that performs well. Hence, here we present a sequential graph coloring algorithm from Coleman & Moré (1983) that does not guarantee optimality, but has been widely used and analyzed.

Algorithm 2 Sequential Graph Coloring

```
v_1, v_2, ..., v_n are n nodes in the graph;
p = 0;
for k = 1 to n do
    for i = 1 to p + 1 do
        if connected nodes of v_k not assigned Color i then
            Break;
        end if
    end for
    Assign Color i to v_k;
    if i > p then
        p ← p + 1;
    end if
end for
```

The resulting p from the algorithm will be the number of colors. It is a greedy algorithm, and literature has reported the performance depends on the ordering of nodes (k loop in Algorithm 2). According to Coleman & Moré (1983), there exists an ordering of nodes such that the sequential graph coloring method can obtain the optimum.

In our implementation, we perform the partitioning job offline. Specifically, we

run the sequential graph coloring algorithm with multiple random ordering of the nodes. In this way we record the minimum number of colors and the corresponding color assignment used.

Condensing Sparse Gradient

Now we present formulations that describe the gradient condensing process. Assume the graph color assignments are in an $n \times p$ matrix $\boldsymbol{D}$ such that $\boldsymbol{D}$ is also an incidence matrix: the jth row indicates the color assignment of parameter j, and the kth column $\boldsymbol{D}_k$ indicates all the parameters with color k. Since one parameter cannot be assigned to multiple colors, each row would have exactly one element with value 1. Recall the gradient $\boldsymbol{H}$ is $m \times n$. The condensed gradient is given by:

$$\tilde{\boldsymbol{H}} = \boldsymbol{H}\boldsymbol{D} \tag{5.2}$$

Since we ensured the partitioning process, the condensation is lossless.

Inflating Condensed Gradient

Similarly, we present the formulation that the Sparse gradient $\boldsymbol{H}$ could be recovered without loss from condensed gradient $\tilde{\boldsymbol{H}}$ with the help of gradient incidence matrix $\boldsymbol{H}_{inc}$:

$$\boldsymbol{H} = \left(\tilde{\boldsymbol{H}}\boldsymbol{D}^{\top}\right) \circ \boldsymbol{H}_{inc} \tag{5.3}$$

where, $\circ$ is the element-wise product.

5.3.3 Simultaneous Perturbation for Gradient Estimation

Thus our condensed gradient $\tilde{\boldsymbol{H}}$ can be obtained from simultaneous perturbations. The perturbations yield:

$$\tilde{\boldsymbol{H}}_k = \frac{\boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} + \boldsymbol{\delta}_k) - \boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} - \boldsymbol{\delta}_k)}{2\delta_k} \tag{5.4}$$

$$\boldsymbol{\delta}_k = \delta_k \boldsymbol{D}_k \qquad \forall k = 1, 2, ..., p \tag{5.5}$$

$$\boldsymbol{H} = \left(\tilde{\boldsymbol{H}} \boldsymbol{D}^\top\right) \circ \boldsymbol{H}_{inc} \tag{5.6}$$

where, δ_k is the perturbation size for all parameters in the same partition. $\boldsymbol{D}_k$ is the kth column of color assignment $\boldsymbol{D}$ such that only parameter in partition k have non-zero values δ_k.

Without loss of generality, a perturbation size for each parameter in the vector $\boldsymbol{\delta}$ can be achieved by:

$$\tilde{\boldsymbol{H}}_k = \boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} + \boldsymbol{\delta}_k) - \boldsymbol{g}_h(\hat{\boldsymbol{x}}_{h|h-1} - \boldsymbol{\delta}_k) \tag{5.7}$$

$$\boldsymbol{\delta}_k = \boldsymbol{\delta} \circ \boldsymbol{D}_k \qquad \forall k = 1, 2, ..., p \tag{5.8}$$

$$\boldsymbol{H} = \left(\tilde{\boldsymbol{H}} \boldsymbol{D}^\top\right) \circ \boldsymbol{H}_{inc} \circ \left[\frac{1}{2\boldsymbol{\delta}}, \frac{1}{2\boldsymbol{\delta}}, ..., \frac{1}{2\boldsymbol{\delta}}\right]^\top \tag{5.9}$$

where, $\left[\frac{1}{2\boldsymbol{\delta}}, \frac{1}{2\boldsymbol{\delta}}, ..., \frac{1}{2\boldsymbol{\delta}}\right]^\top$ is a $m \times n$ matrix.

So far, we have successfully introduced the steps to perform the partitioned simultaneous perturbation and discussed the existence and reliability of the sparse gradient incidence matrix $\boldsymbol{H}_{inc}$. However, the reduction of parameter number n to p may not be significant. In the next section, we present a real scenario to demonstrate the reduction rate of the procedure.

5.4 Performance on a Large-Scale Network

5.4.1 Test Network

In this section, we conduct an experiment to demonstrate the performance of PSP using DynaMIT on the Singapore Expressway network, displayed in Figure 5-2. The

following figure illustrates the road network topology. It has 4121 OD pairs and 3906 segments.

Figure 5-2: Singapore expressway network

5.4.2 Obtaining Gradient Incidence Matrix

We follow the same procedure as mentioned in Section 5.3 with the focus on gradient structure identification. To obtain a universal gradient incidence matrix throughout the whole simulation period, we run the existing scenario with FD-EKF first and record all the H matrices. Note this procedure may be much longer than real-time, but it is acceptable since this needs only once. As mentioned before, the incidence matrix is the *OR* operation for all the H matrices obtained, for maximum coverage.

5.4.3 Calibration Accuracy and Computational Performance

To compare the accuracy and computation time, we run the calibration with PSP-EKF for the same scenario as we conducted FD-EKF. We run the simulation for 7-10AM, with 5 minute for each interval on the Singapore expressway network. We use real flow count data from the Land Transport Authority (LTA) for this demonstration.

To conduct a fair comparison of the performance of PSP-EKF and FD-EKF, we need to control all other sources of difference, including default values of supply

parameters, initial random seed, simulation period, etc. However, fixing the initial random seed does not necessarily mean the random number sequence will be the same. It is possible that a small disturbance in the calibration result (either demand or flow) will change the random sequence afterwards. Despite these uncertainties in simulation, the gradient matrices obtained by PSP should be close to those by FD and consequently, the overall calibration errors should be similar.

Accuracy

In response to the expectations, we view the accuracy of PSP approach in two aspects: (1) the PSP approximation to FD; (2) the overall calibration results.

The simulation results show that for the first 10 intervals, the gradient estimation from PSP is identical to that from FD. However, for later intervals, as the number of unequal elements increases, so does the magnitude of difference.

In terms of calibration accuracy, Table 5.1 indicates that the performance is similar for both methods.

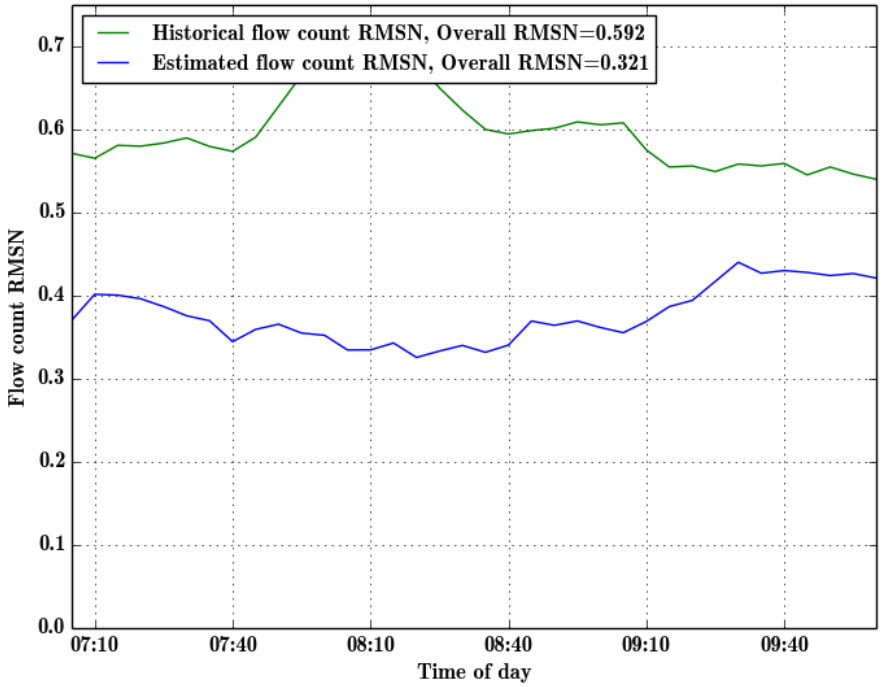
Table 5.1: Calibration accuracy comparison for FD-CEKF and PSP-CEKF

Method	Estimation RMSN	Prediction RMSN		
		1 step	2 step	3 step
No calibration	59.7%	59.7%	59.7%	59.7%
FD-CEKF	32.1%	34.0%	36.3%	38.3%
PSP-CEKF	32.9%	34.7%	37.0%	39.0%

Figure 5-3 shows the PSP and FD calibration results interval by interval. Their performance is very similar, despite the uncertainty of random seeds in simulation and imperfect gradient estimation. This concludes that the PSP method to estimate the gradient matrix approximates the FD well in the application of the Kalman filter.

Computation Complexity

For the traditional central finite difference, we need 4121 pairs of simulation to estimate the gradient in each interval. With the PSP approach, we managed to reduce



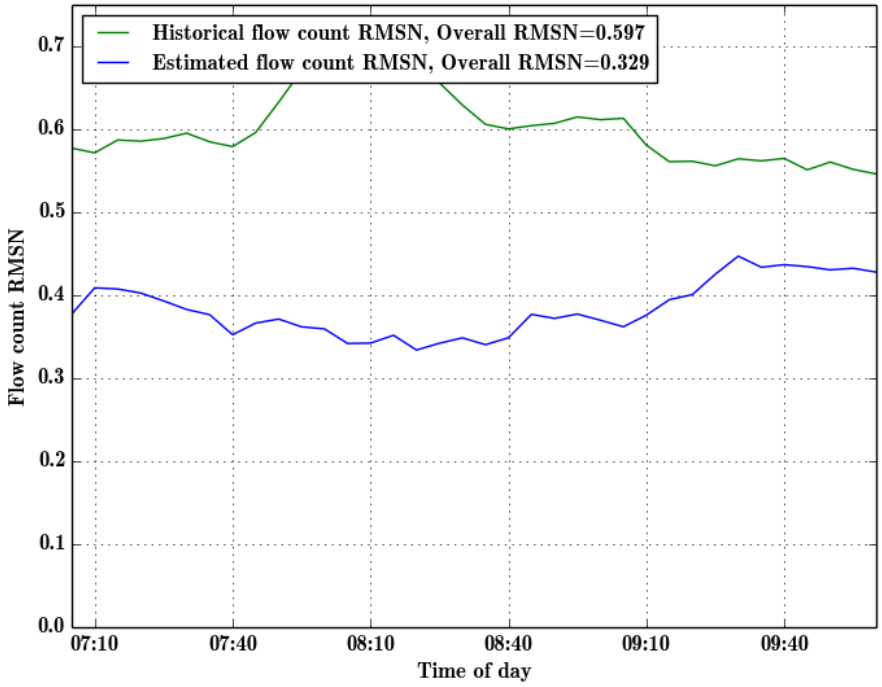


Figure 5-3: RMSN by intervals for FD-CEKF and PSP-CEKF

this to 438 pairs. The computation time is presented in Table 5.2. Note the simulation is run on a server with 40 cores.

Table 5.2: Compuation time comparison for FD-CEKF and PSP-CEKF iterations

Estimation method	# Parameter groups	Calibration time for interval (minutes)				
		6:00-6:05	7:00-7:05	8:00-8:05	9:00-9:05	Average
FD	4121	12.2	22.3	32.6	48.3	28.8
PSP	438	2.5	3.9	5.3	7.2	4.7

In summary, we make two conclusions. In terms of accuracy, the PSP approach attains a similar accuracy as FD. In terms of computational complexity, the PSP reduces significantly the number of computations needed by FD, with the extent of reduction depending on the sparsity of the gradient structure.

5.5 Practical Considerations for Gradients with Flow Counts vs OD

5.5.1 Random Order of Coloring

As mentioned in the graph coloring algorithm, the order of parameters affects the optimality of the partitioning. So a random ordering may be helpful. Here we demonstrate another benefit of random ordering in terms of reducing unobservable common impacts from system ordering.

The partitioning is based on avoiding common impacts on sensors. However, unobservable common impacts may also affect the gradient accuracy. As an example, Figure 5-4 presents a network with 3 OD pairs. It is obvious that these OD pairs affect different sensors. According to the PSP algorithm, they can be in the same partition. However, when we perturb them at the same time, the link without the sensor will be affected by all perturbations. For instance, if we increase the demand of all OD pairs by 30 vehicles each, the demand of the shared link increases by 90 vehicles! In such cases, unexpected congestion may occur. Thus, the simultaneous perturbations may lead to unexpected change in traffic status due to the large perturbation due to

aggregation of small individual perturbations.

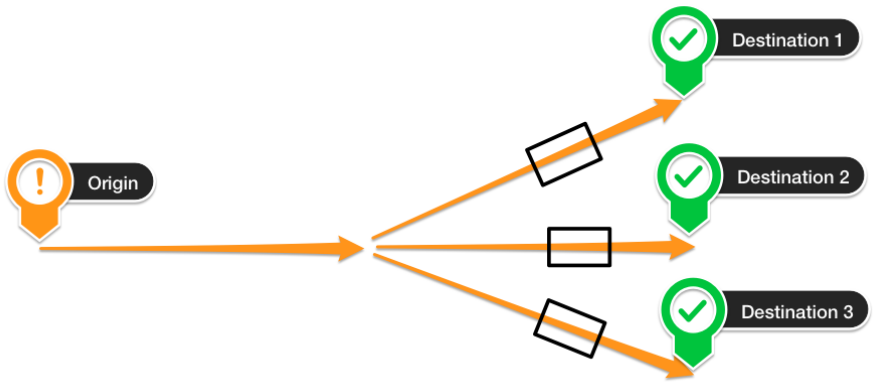


Figure 5-4: Three OD pairs sharing the same link but not sensors (in black rectangles)

This issue is more likely to happen if the OD pairs are specified in order as opposed to being random, because OD pairs generated systematically usually either start from the same origin or end with the same destination. This again necessitates the usage of a random order for partitioning. Figures 5-5 and 5-6 presents the heat map of PSP gradient differences from FD, for both cases. The matrix in the former case is very sparse, as seen from the little dots in the figure. But the matrix for the latter is strictly a zero matrix. The presented interval is 15 minutes after simulation starts. With random ordering, the PSP gradient is identical to the FD gradient.

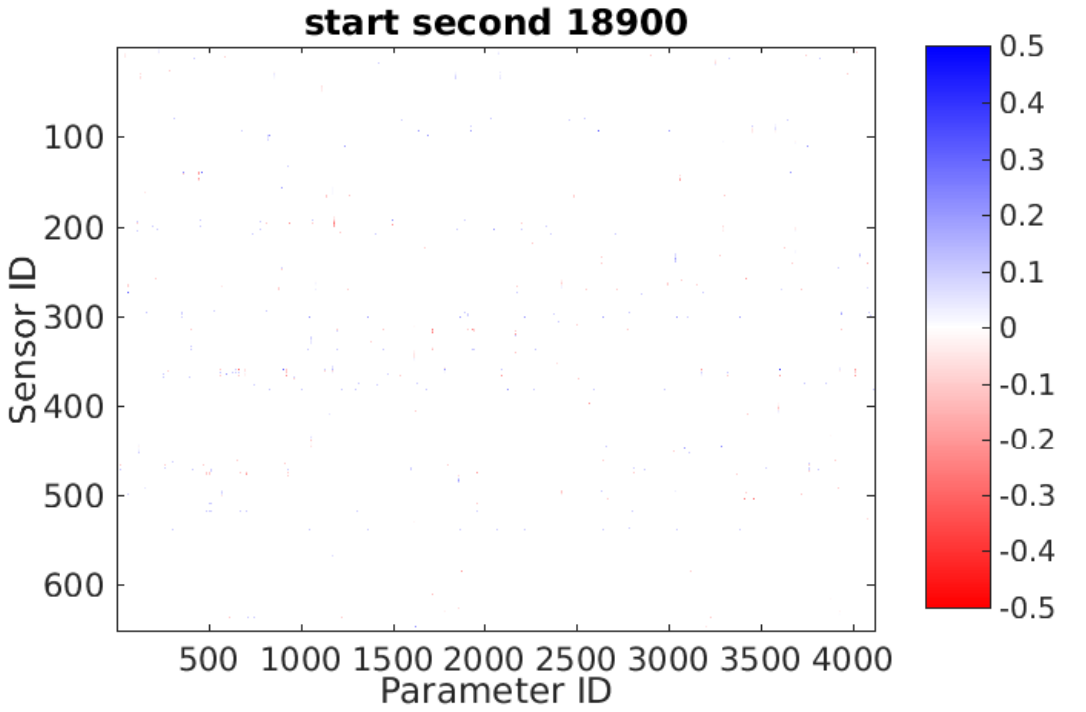


Figure 5-5: PSP gradient difference with FD gradient estimation, original ordering

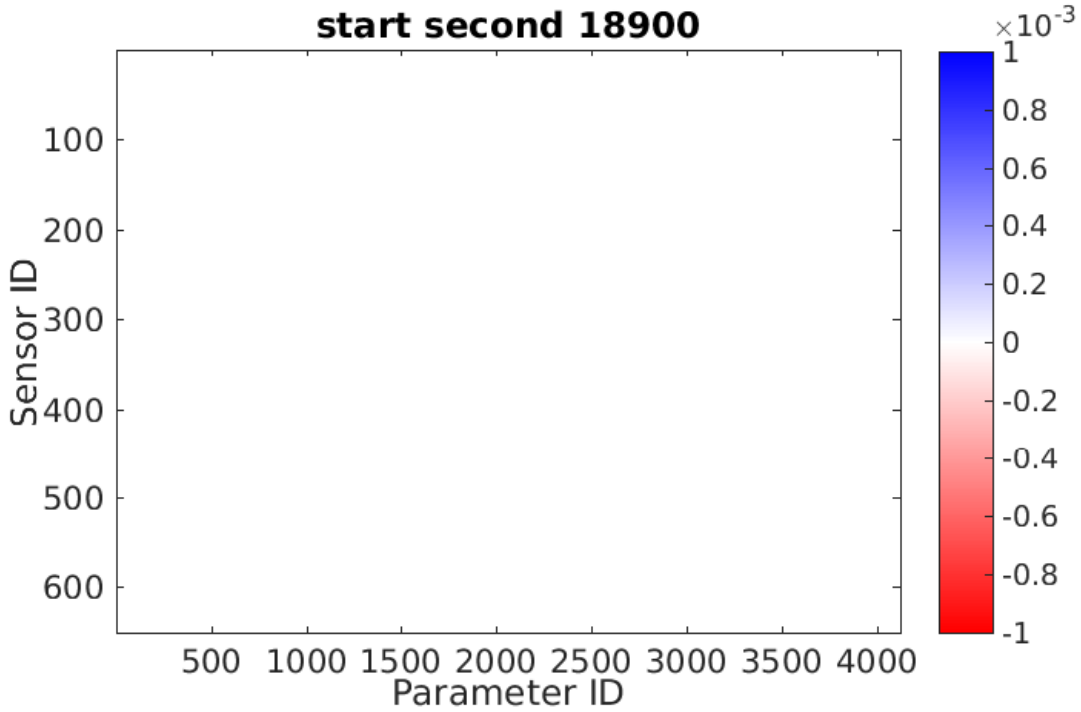


Figure 5-6: PSP gradient difference from FD gradient estimation, random ordering

5.5.2 Gradient Structure for Flow Counts vs ODs

For the issue mentioned previously, there is a remedy for flow counts vs ODs. From the simulator we can generate the *path sets* for all OD pairs. Thus, it is possible to record all the links each OD pair could traverse and construct an incidence matrix for links vs OD pairs. Then based on the incidence matrix, we can follow the same procedure and partition the OD pairs.

One important comment is that using the link incidence matrix is likely to result in more partitions. This is because we consider any two OD pairs sharing the same link as *conflicting*, while they may not affect the link within the same interval. Thus, such a dense incidence matrix will likely yield unnecessary partitions. For the case of Singapore expressway network with 4121 OD pairs, the best coloring result is 2370 partitions, which comes from 30 runs of random ordering. Compared with 438 in Table 5.2, it is not advisable to apply this idea directly to the PSP approach.

5.5.3 A Universal Gradient Structure

We have shown that the link incidence matrix is not a good idea for PSP. Next we look into how the *path sets* may be helpful for a universal gradient structure.

The assumed gradient structure relies on previous runs of calibration with FD. In our experiment, we handle this with an *OR* operation for all the available gradients that were calculated offline. However, we cannot guarantee the gradients are generalizable for all cases. In addition, it is unacceptable to run FD for each scenario whenever the traffic state changes. Thus, it is best to identify a gradient structure that is universal to all traffic state scenarios.

For the gradient of flow counts with respect to ODs, good news is that the gradient structure obtained with an empty network covers the case for congested networks. Given fixed supply parameters, all traffic demand scenarios will yield travel times no less than empty networks. Thus, compared with an empty network, perturbations based on a fully-loaded network are captured by a small number of sensors. Once a gradient incidence matrix is obtained for the empty network, the gradient will

only be more sparse for following intervals. This feature satisfies our need for the *universal* gradient perfectly, however, on one condition: all paths will be chosen when obtaining the empty structure in simulation. The gradient from an empty network is not universal when all drivers only choose a particular route in case of congestion. Hence, checking if all the paths are traversed is necessary to claim a gradient structure is indeed universal.

We propose the following procedure to check for a universal gradient structure. For each OD pair, we claim a path is traversed if any of its *unique* subpath is traversed. A subpath of a path is defined *unique* if and only if it is not in any other paths for the same OD pair. If all the paths for an OD are traversed, the corresponding column of the gradient is universal. There is one exception: when the traversed part of two paths is the same, we treat both paths as traversed since within one interval, the traffic flows have not bifurcated yet.

With this approach, we can calculate the percentage of route choice coverage. The result we got for Singapore expressways is 76.1%. Which means the coverage is not exhaustive, but decent enough for real applications. Further research should continue to increase the coverage for OD estimation in order to obtain a universal gradient structure.

5.6 Conclusion

We close this chapter with the following comments. We investigated a gradient estimation method named partitioned simultaneous perturbation. It is an existing method, but seldom used in DTA calibration context. The computational performance of this method is superior over the traditional finite difference. To employ this method, a predefined sparse gradient structure is necessary.

Chapter 6

Case Study

In this chapter, we present a case study to demonstrate the performance of the solution approaches in Chapters 4 and 5. Compared with the synthetic case studies under full control, real case studies usually suffer from various sources of uncertainty. The objective is to apply the proposed approaches to a real-world scenario and report the performance under uncertainty. In such a case, practical considerations to mitigate the uncertainty are extremely useful as guidelines for similar applications in the real world.

This chapter is structured as follows: first the data source and the Singapore expressway network is briefly introduced. Second, the preparations and calibration settings are summarized. Third, we conduct the experiments with the proposed solution approaches in this thesis and present the results along with discussions. Finally, we draw conclusions for this case study.

6.1 Data Description

6.1.1 Singapore Expressway Network

The Singapore expressway network is a large-scale city-wide urban network shown in Figure 6-1 (dark orange). The corresponding representation of the network used in DynaMIT is shown in Figure 6-2. It includes all the expressways and some selected

arterials. The network consists of 939 nodes, 1157 links and 3906 segments. There are 4121 origin destination (OD) pairs on the network, where on-ramps serve as origin nodes and off-ramps serve as destination nodes. These 4121 OD pairs have 18532 routes in total, thus, on average, each OD pair has 4.5 routes to choose from. On the measurement side, there are 650 sensors distributed across the network that capture traffic flow volumes. In this case study, we run the online calibration from 6AM to 10PM to model the morning peak.

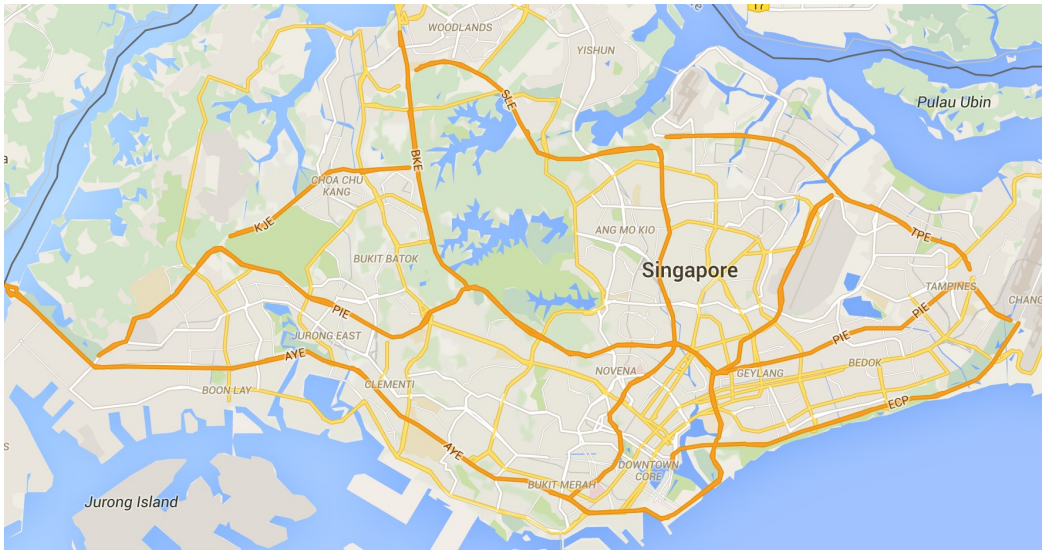


Figure 6-1: Singapore expressway network (Google Maps, 2016)

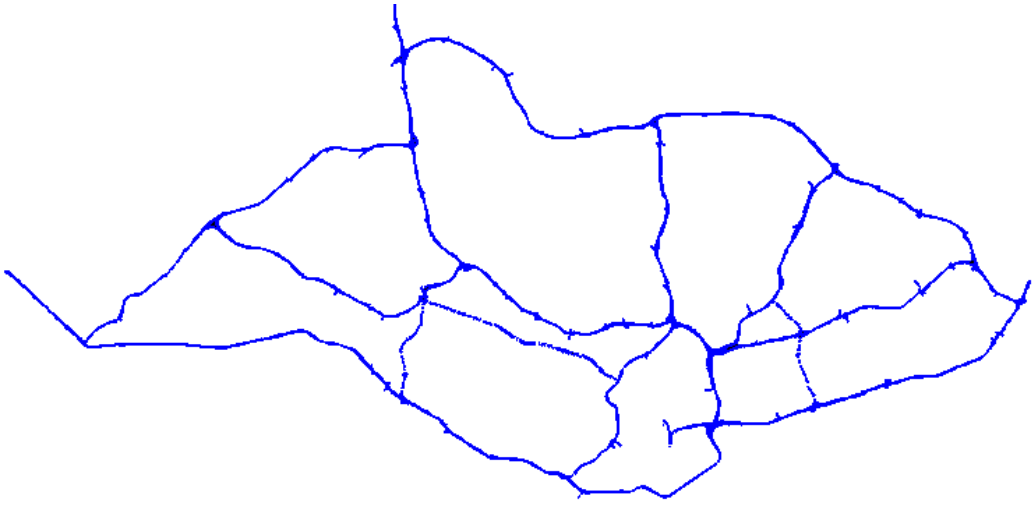


Figure 6-2: Singapore expressway network in the DTA model

6.1.2 Surveillance Data

Following the introduction of the Singapore expressway network, we briefly summarize the real-time traffic flow data from the 650 sensors. The sensors detect traffic flow for all the lanes on the segments. The real-time flow volumes are collected and aggregated into each batch of data roughly every 5 minutes. The real-time data are provided by the Land Transport Authority (LTA) in Singapore for 14 weekdays in December 2015.

However, as with any field data, the flow volumes are prone to errors. The sensors are unreliable in the following ways:

(1) Some sensors are absent constantly for hours, and some are absent occasionally.

(2) The measurement errors are excessive on numerous sensors. The flow conservation law is violated for some sensors on the same expressway without ramps in between. For example, the aggregated flow volume from 5am to 12pm for a particular sensor is 15542 vehicles, while the flow for its downstream sensor is 13049 vehicles. Observations suggests that similar cases exist in an extensive area of the Singapore expressway network.

To address the first issue, we remove the sensors that are absent over half of the experiment period. For the remaining sensors that occasionally disappear, we modify the Kalman filter update rule to handle this. In Equation (2.36), the corresponding rows of the missing sensors for interval h are deleted. Similarly, the corresponding rows and columns are deleted for $\boldsymbol{R}_h$ in Equation (2.37). The same deletion rule is applied to the elements of $\boldsymbol{M}_h$ and $\boldsymbol{g}(\cdot)$ in Equation (2.38).

The second issue is more difficult to handle. Thus, we have to carefully set the measurement error covariance $\boldsymbol{R}_h$. This is addressed in the preparation for the calibration.

6.2 Preparation and Experiment Settings

6.2.1 Overview

In this case study, the online calibration task is OD estimation and prediction using the real-time flow data. The goal is to test the accuracy improvement with the augmented SSM. The partitioned simultaneous perturbation is also applied for all the experiments to speed up the calculation.

The parameters in this case study are 4121 OD pairs for each 5-minute departure interval. At the same time, route choice and supply parameters such as speed-density relationships and capacity for each segment are set to offline calibrated values.

6.2.2 Preparations of Kalman Filtering

Before performing the experiments, the Kalman filtering framework needs several inputs to work properly, namely:

- **Time-dependent OD matrices**. To employ the deviation as the state, calibrated time-dependent ODs are needed as historical values.
- **The autoregressive (AR) model**. Needs to be estimated for the transition equation.
- **The tuning parameters for Kalman filters**. The tuning parameters include the transition and measurement error covariances $\boldsymbol{Q}$ and $\boldsymbol{R}$.

The workflow in Figure 6-3 indicates how to obtain the mentioned inputs. We explain the workflow with the following procedure.

(1) Divide the weekdays into training set (10 days), validation set (3 days) and test set (1 day);

(2) Perform calibration for the training set and validation set. The inputs used in this stage of calibration are based on heuristics;

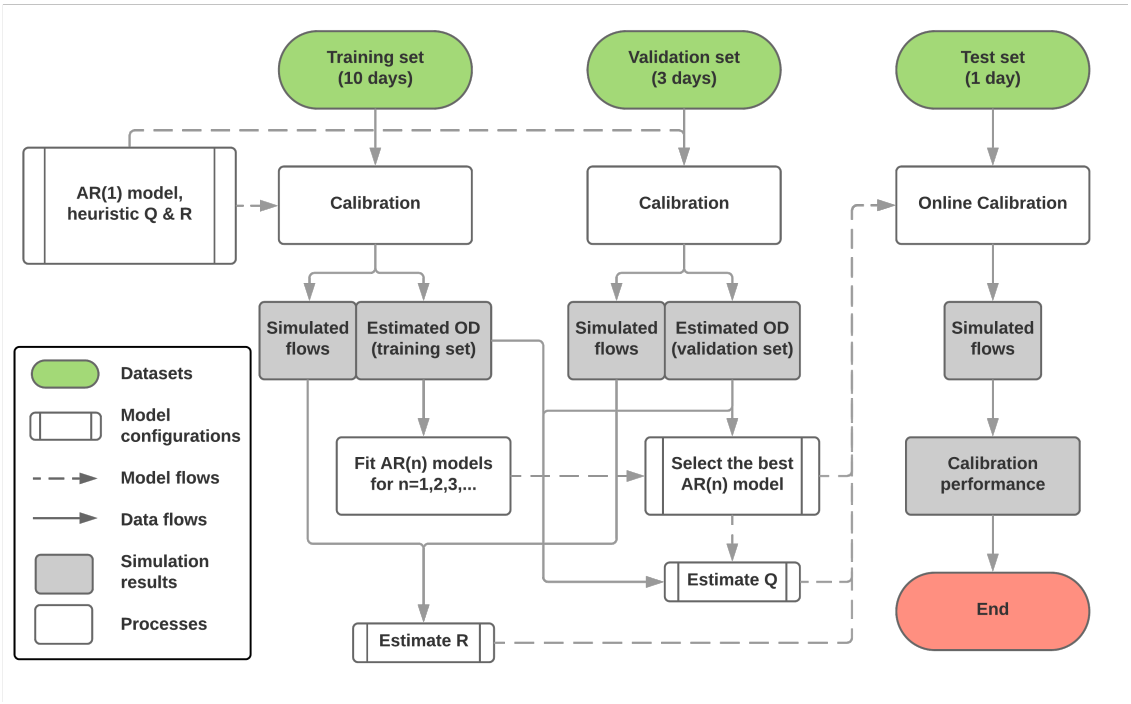


Figure 6-3: Workflow of the preparation for the Kalman filter

(3) Calculate the residuals between the estimated flows and the data in the training set and validation set. We then compute the variance of the residuals for each sensor across time intervals, and these variances serve as diagonal elements of $\boldsymbol{R}$. The calculation is given by:

$$\boldsymbol{R} = \begin{bmatrix} r_1 & 0 & \cdots & 0 \\ 0 & r_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_m \end{bmatrix} \tag{6.1}$$

$$r_i = \frac{1}{DN} \sum_{d=1}^{D} \sum_{h=1}^{N} \left(M_{h,i}^{(d)} - g_{h,i}^{(d)}(\cdot) \right)^2 \tag{6.2}$$

where, $M_{h,j}^{(d)}$ is the observed measurement and $g_{h,i}^{(d)}(\cdot)$ is the simulated counterpart for the jth sensor at time interval h on day with index d. m is the number of sensors, and d ranging from 1 to D is the day index in the training set.

(4) Fit an AR(n) model to calibrated time-dependent OD matrices in the training set.

n takes multiple values: 1,2,3,... Each n results in a different fitted AR model. Then we test the models and select the *best* model based on their prediction performance on the validation set.

(5) Calculate the residuals between each estimated OD and predicted values given by the best model. We compute the variance of the residuals for all ODs across time intervals. It serves as a universal variance for all the diagonal elements of $\boldsymbol{Q}$, which is given by:

$$\boldsymbol{Q} = \begin{bmatrix} q & 0 & \cdots & 0 \\ 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & q \end{bmatrix} \tag{6.3}$$

$$q = \frac{1}{DnN} \sum_{d=1}^{D} \sum_{i=1}^{n} \sum_{h=1}^{N} \left(\left(\boldsymbol{X}_h^{(d)} \right)_i - \left(\boldsymbol{\Phi} \boldsymbol{X}_{h-1}^{(d)} \right)_i \right)^2 \tag{6.4}$$

where, $\boldsymbol{X}_h^{(d)}$ is the state vector in interval h on the day with index d. $\left(\boldsymbol{X}_h^{(d)} \right)_i$ is the ith calibrated OD in interval h on day d, and $\left(\boldsymbol{\Phi} \boldsymbol{X}_{h-1}^{(d)} \right)_i$ is the ith predicted OD with the AR model, parameterized by $\boldsymbol{\Phi}$. n is the number of OD pairs. D is the number of days in the training set.

(6) The computed $\boldsymbol{Q}$ and $\boldsymbol{R}$, together with the selected AR model serve as inputs for the Kalman filter in online calibration. When these preparations finish, we can perform online calibration on the test set and report the accuracy and speed, as performance measures of the calibration results.

(7) The mean of the calibrated demand over the training and validation set for each interval serves as the time-dependent historical values to construct deviations for the test set.

Some explanations would be helpful to show the validity of the procedure. Here we explain the details in steps (3)-(5) and some practical considerations in the following paragraphs.

Obtaining $\boldsymbol{R}$

We now explain how step (3) addresses the uncertainty in measurement. A large error variance indicates a poor fit to the data. The fault is not entirely in the model, because of the excessive noise in the measurements (Section 6.1.2). Thus, a large error variance may imply an enormous noise in the measurement. The large variances in $\boldsymbol{R}$ indicate high uncertainty for their corresponding measurements, potentially due to large measurement noise. Then Kalman filters will give less weights to the uncertain measurements and focus more on fitting others. This mechanism will alleviate the issue of large measurement noise.

However, there is a risk of *overfitting* in this approach: the obtained $\boldsymbol{R}$ may not be generalizable to other model specifications (e.g., different AR model or augmented SSM). The reason lies in the contribution of modeling errors to the variances. A measurement poorly fitted with one model may be fine with another. Thus, when the modeling error is predominant over the measurement noise, the obtained error variance will prevent the Kalman filter to fit the measurements, resulting in suboptimal solutions for other models. In a word, the $\boldsymbol{R}$ is "overfitted" to the model specification that generates it. Hence, researchers should consider the tradeoff between suboptimal solutions due to overfitted $\boldsymbol{R}$ and the potential erroneous fit caused by large noise in data. From our observation, the traffic flow inconsistency is severe and widely present in the dataset. Also, 650 sensors are significantly smaller than 4121 OD pairs as parameters, which implies an under-determined system that fits measurements with enormous degree of freedom. Hence, it is likely that error covariances capture more measurement noise than the modeling error.

Obtaining the AR model

In step (4), we attempt to find the evolution of ODs by selecting the best AR model to fit the calibrated ODs. To avoid overfitting, we force the same AR model for all the OD pairs, instead of one model for each OD pair. The reason lies in the validity of the fitted models. Due to the enormous degree of freedom (4121 OD pairs to fit

650 sensors), the estimated ODs are zeros for most of the time, which results in high uncertainty of the fitted AR models. They will not generalize well if the estimated ODs are non-zero in test set. Thus, we include the calibrated ODs with more than 60 vehicles per hour into the dataset to which we fit the AR models. The dataset contains 2142 OD pairs for all the 10 days.

Step (4) to choose the best AR model follows a typical machine learning setting: holdout a validation set before training/fitting the models. The underlying reason is that a more complex model will always achieve a better fit on the training set, however it may not generalize well on the validation set, which was not used when training the model. In our experiment, n ranges from 1 to 10, because we believe a tenth-order AR process should suffice to describe the transition trend in OD. Next, we select the AR(n) model with the best prediction power on the validation set. Note that depending on whether to use deviations, the data for model fitting are different. Finally we will apply the selected AR model as the transition equation for online calibration experiments on test set.

Note that depending on whether to use deviations, the data for model fitting are different. Thus, we would result in two sets of AR models.

Obtaining $\boldsymbol{Q}$

Similar to obtaining $\boldsymbol{R}$, we calculate the variance for $\boldsymbol{Q}$ from residuals of the selected AR model with Equations (6.3) and (6.4). The diagonals have the same value q to avoid overfitting, with the similar logic for a universal AR model. Otherwise, the ODs that were calibrated to zeros lead to small variances, with which the subsequent Kalman filter will always give near-zero ODs. In other words, the same q for the diagonals of $\boldsymbol{Q}$ would allow calibrating insignificant ODs that are previously obtained. Thus, it seems reasonable to assume a universal diagonal structure for $\boldsymbol{Q}$.

Model Configurations

Following the procedure mentioned above, the calibration results (using all available sensors) for the training and validation sets are given by Table 6.1, measured in root

Table 6.1: Calibration result (RMSN and RMSE) for training set

Dataset	Day of Dec 2015	Estimation RMSE	Estimation RMSN	Prediction RMSN		
				1 step	2 step	3 step
Training	1	125.1	49.0%	49.9%	50.8%	51.4%
	2	123.3	48.0%	49.0%	49.9%	50.4%
	10	122.5	50.5%	51.6%	52.7%	53.0%
	17	120.5	47.9%	48.9%	49.7%	50.1%
	18	117.1	47.6%	48.8%	49.5%	50.3%
	21	115.4	47.2%	48.2%	49.6%	50.5%
	23	119.1	48.0%	49.4%	50.6%	51.6%
	24	118.8	50.2%	51.6%	52.7%	53.8%
	28	113.4	48.6%	49.8%	51.4%	51.8%
	29	112.9	46.8%	48.2%	49.1%	49.7%
Validation	7	123.2	50.1%	51.1%	51.9%	53.0%
	14	117.2	47.7%	49.0%	50.4%	51.0%
	30	117.3	49.6%	50.9%	52.5%	53.0%

mean square error (RMSE) and root mean squared normalized error (RMSN), with formula given in Equation (3.2).

Following the procedure in step (4), the best model with calibrated OD from the training and validation sets as historical values is an AR(2) model, given by:

$$\boldsymbol{x}_h = 0.884\boldsymbol{x}_{h-1} + 0.0967\boldsymbol{x}_{h-2} + \epsilon \tag{6.5}$$

$$\epsilon \sim \mathcal{N}(\mathbf{0}, 17.6\boldsymbol{I}) \tag{6.6}$$

Figure 6-4 illustrates the RMSE for each sensor across all intervals for Day 1. It is also the standard error for each estimated measurement, i.e.,

$$\sqrt{\frac{1}{N}\sum_{h=1}^{N}\left(M_{h,i}^{(1)} - g_{h,i}^{(1)}(\cdot)\right)^2} \tag{6.7}$$

for the ith sensor.

The measurement errors are with different magnitudes, which implies the different noise levels for sensors. Similar to the insight discussed when obtaining $\boldsymbol{R}$, this figure indicates the structure of the diagonals in $\boldsymbol{R}$. It is also noticeable that some sensors have zero RMSEs, which correspond to the sensors missing for the whole

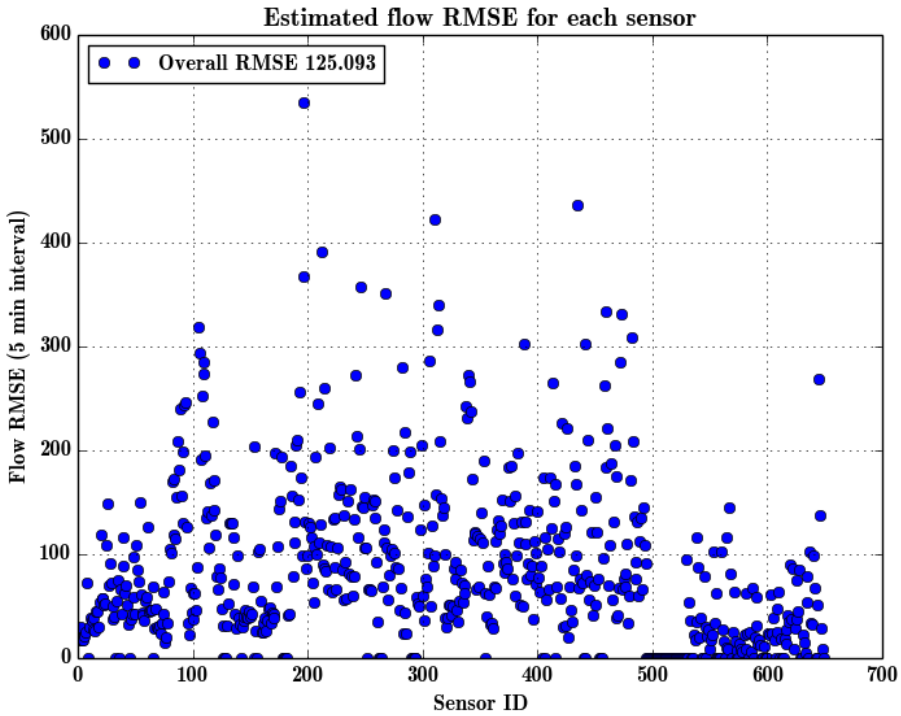


Figure 6-4: Estimated flow RMSE (standard error) for each sensor for Day 1

simulation period. Those small variances in $\boldsymbol{R}$ will lead to an overstated certainty in corresponding sensors. This improper certainty is then adjusted by setting the diagonals of $\boldsymbol{R}$ to be no less than 10. There were 89 elements affected by this rule, while 82 of them are zero (Figure 6-5). Observe that we only have around 430 sensors with variance less than 10000 (standard error < 100).

6.2.3 Experiment Settings

Given the real-time flow data, the online calibration task is OD estimation and prediction. The goal of this case study is to test the accuracy improvement with the state augmentation technique compared to the original case. To test the augmented SSM, we first need to determine the degrees of augmentation. To elucidate this determination process, we first analyze the gradients with transition step t, which indicates the gradient for measurements after t intervals.

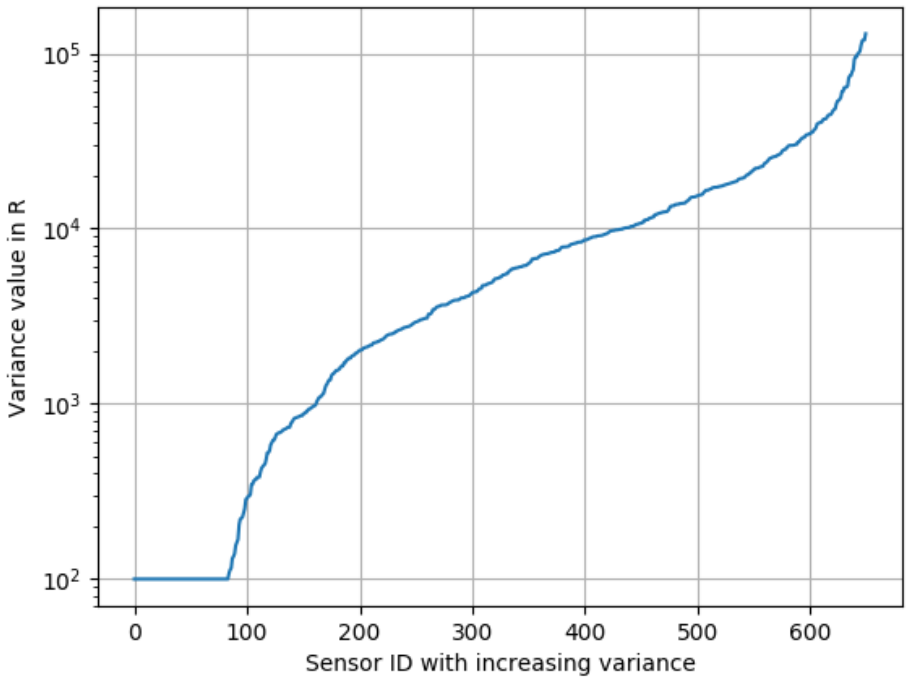


Figure 6-5: Estimated measurement variance in increasing order

Degrees of Augmentation

Table 6.2 exhibits the statistics of gradients with different transition steps, calculated with finite differences. A row or column is only zero if all elements in it are zero. The rank indicates the dimensions of parameter space determined by each matrix $\boldsymbol{H}^1_{t+1}$, while the cumulative rank for t is calculated from the vertically concatenated $\boldsymbol{H}^1_1$ to $\boldsymbol{H}^1_{t+1}$. The cumulative rank shows the dimensions of parameter space if we consider its impact on future measurements to a degree t.

The growing cumulative rank demonstrates the benefit of augmenting measurement equation. The increment of cumulative rank slows down for $t > 7$ (Table 6.2). Thus, the benefit of augmentation to more than 7 steps may be marginal. This conclusion is also verified by the nonzero elements in gradient matrices. The nonzero elements for step 8-11 are considerably less than step 7.

Note that we obtained these gradients by perturbing ODs in the first interval in an empty network. Therefore, when there is congestion, it is likely that the transition

Table 6.2: Statistics of gradients $\boldsymbol{H}_{t+1}^{1}$ for transition step t

Transition step	Nonzero elements	Nonzero rows	Nonzero columns	Rank	Cumulative rank
0	41635	635	4119	628	628
1	78183	609	4089	607	1210
2	66676	562	3545	562	1736
3	52401	548	2822	541	2242
4	33026	517	2036	511	2705
5	17588	491	1299	459	3064
6	7737	437	675	390	3262
7	2969	360	274	236	3284
8	1045	213	115	99	3284
9	286	95	45	44	3285
10	52	34	12	12	3285
11	0	0	0	0	3285

steps greater than 7 is beneficial in later intervals. This conjecture implies that it may be beneficial to determine the degree of augmentation based on current traffic conditions.

While a higher degree of augmentation is favorable, another consideration is the limited computational power. A model of degree 6 already consumes enormous computational power: the model needs 6 times more simulations for gradient estimation than the non-augmentation case. Due to the matrix operations of $O(n^3)$ in Kalman filters, the model needs $6^3 = 216$ times of computations. Thus, we choose the maximum degree of augmentation as 6.

Experiment Specifications

The experiments report the performance for the original SSM and the augmented SSM of various degrees. The partitioned simultaneous perturbation is applied for all the experiments to improve computation time. The constrained extended Kalman filter (CEKF) is also applied to model the non-negativity constraint for OD flows (H. Zhang et al., 2017). Based on the model configurations in Section 6.2.2, we propose the following experiments. Each experiment is conducted with the calibrated time-dependent OD matrices as historical values.

(1) **CEKF(1)**: constrained extended Kalman filter without state augmentation;

(2) **CEKF(3)**: constrained extended Kalman filter with state augmented to degree 3;

(3) **CEKF(6)**: constrained extended Kalman filter with state augmented to degree 6.

6.3 Results and Discussions

In this section, we present the results and discuss the performance of each model. We first show the overall performance for DTA estimation and prediction. As for performance measures, we select root mean square error (RMSE), weighted sum of squared error (WSSE) and root mean squared normalized error (RMSN) criteria.

6.3.1 Performance Metrics

As a metric to address the sensors that are assumed highly uncertain, the WSSE utilizes the inverse of $\boldsymbol{R}$ as weights for the squared errors of each measurement. In our case of a diagonal $\boldsymbol{R}$, each squared error is divided by its assumed variance and then summed up. Thus, the WSSE, as an objective function, discounts the impact of the uncertain measurements. Also, note that the WSSE is a component in the Kalman filter's objective function (Sorenson, 1970), and thus a lower bound of it.

Similarly for RMSN, we removed some of the erroneous sensors. The condition of removal is that a sensor satisfies: (1) assumed variance in $\boldsymbol{R}$ greater than 10000, and (2) fitted mean square error greater than 10000. This rule removes 122 sensors poorly fitted with both the training set (216 sensors) and the test set (186 sensors) while keep the sensors whose fitted variance is less than 10000 with any model.

6.3.2 Results and Discussions

Table 6.3 summarizes the performance of experiments. We start with the estimation results. The results imply a strict increase of estimation and prediction performance

as we augment the states. The benchmark is directly using the calibrated demand as historicals with which we perform calibration for training and validation sets. All CEKF experiments significantly improve over the benchmark. CEKF(3) obtains the lowest error for RMSE and RMSN. The RMSN values show that CEKF(3) improves over CEKF(1) by around 13%. The CEKF(6) has a worse performance in RMSN for estimation than CEKF(3), but still a 4% improvement over CEKF(1). RMSE, as an overall goodness-of-fit for all the sensors, suggests a marginal improvement of CEKF(3) over CEKF(1) by around 3%. However, the WSSE of augmented models shows a decrease. Since WSSE is a lower bound of the Kalman filter objective, it is possible that the RMSN does not precisely reflect the decrease in the objective function, especially when erroneous sensor measurements are assigned large variance in $\boldsymbol{R}$.

Table 6.3: Performance of all experiments on test day, simulation period 6:20-7:20

Index	Description	Estimation			Prediction RMSN		
		RMSE	WSSE	RMSN	1 step	2 step	3 step
0	Historical	112.6	18047	36.6%	36.3%	36.2%	35.9%
1	CEKF(1)	109.7	13664	33.1%	33.9%	34.9%	34.4%
2	CEKF(3)	106.8	13995	28.7%	30.1%	31.1%	30.1%
3	CEKF(6)	111.0	16409	31.7%	30.7%	31.9%	30.8%
Index	Description	Prediction RMSE			Prediction WSSE		
		1 step	2 step	3 step	1 step	2 step	3 step
0	Historical	116.4	120.6	124.1	19064	20022	20924
1a	CEKF(1)	114.6	119.8	123.2	17428	19133	21013
2a	CEKF(3)	109.7	115.0	118.2	16498	18165	20007
3a	CEKF(6)	110.3	116.3	119.1	16969	18632	20716

For the prediction results in Table 6.3, the overall RMSN also suggests a same amount of improvement by around 13% for augmented models. The prediction performances in RMSE and WSSE are also improved with the augmented models. If we compare the prediction results with estimation in WSSE, both augmented models perform worse in estimation but better in prediction. However, the overall criteria may not be a conclusive proof that the state augmentation improves greatly on prediction. Next we examine the results in more detail.

To address the issue of different variances in sensors, we divide them into groups

regarding their assumed variances in $\boldsymbol{R}$ and report the RMSE for each group (Table 6.4). The best RMSE for each group is bolded. Note here we do not remove any sensors. We have two major observations. First, the CEKF(3) and CEKF(6) have similar prediction performances. The reason may lie in the fact that there are 4121 OD pairs and 650 sensors. Thus, a large degree of freedom exists in the non-augmented model, which may already imply a high model complexity. While augmenting the states further increases the model complexity, the benefit may be marginal when the degree of freedom is already large. The marginal improvement also implies a degree of 3 for augmentation should be enough for our case study. The second observation about Table 6.4 is that the majority of improvement by the augmented models lies in sensors with large assumed variances. Recall that they were estimated from residuals of non-augmented models on the training set. Thus, this observation indicates that augmented models may improve the sensors that were poorly fitted in non-augmented models. These improvements are clear and significant.

Table 6.4: Estimation and 3 step prediction of sensors in variance groups

r	range from range to	1 500	500 2500	2500 5000	5000 10000	10000 20000	20000 $+\infty$
Number of sensors		118	111	90	115	109	107
Estimation	CEKF(1)	35.48	53.86	**76.72**	**132.1**	125.1	157.2
	CEKF(3)	**35.28**	**52.67**	81.99	134.6	**119.2**	**146.7**
	CEKF(6)	38.87	56.49	88.60	140.4	123.6	149.5
3 step prediction	CEKF(1)	48.32	60.76	86.42	148.4	142.9	170.0
	CEKF(3)	**47.36**	**58.11**	86.67	**143.0**	**134.4**	**163.0**
	CEKF(6)	48.59	58.39	**86.20**	143.8	136.1	165.5

Following the discussion of model complexity in augmented models, one may ask if they may overfit to the measurements. The concern is valid because the dimension of parameter space is multiplied through state augmentation. However, this is not true if only the latest updated parameters are reported in our estimation RMSN. Thus, by state augmentation, we actually adjust our previously estimated parameters to fit the current interval better, which may result in worsened estimations in previous intervals, but should lead to better future predictions. Thus, the comparison between augmented and non-augmented models is fair. This observation explains the fact that

estimation results are worse than predictions in WSSE and RMSE.

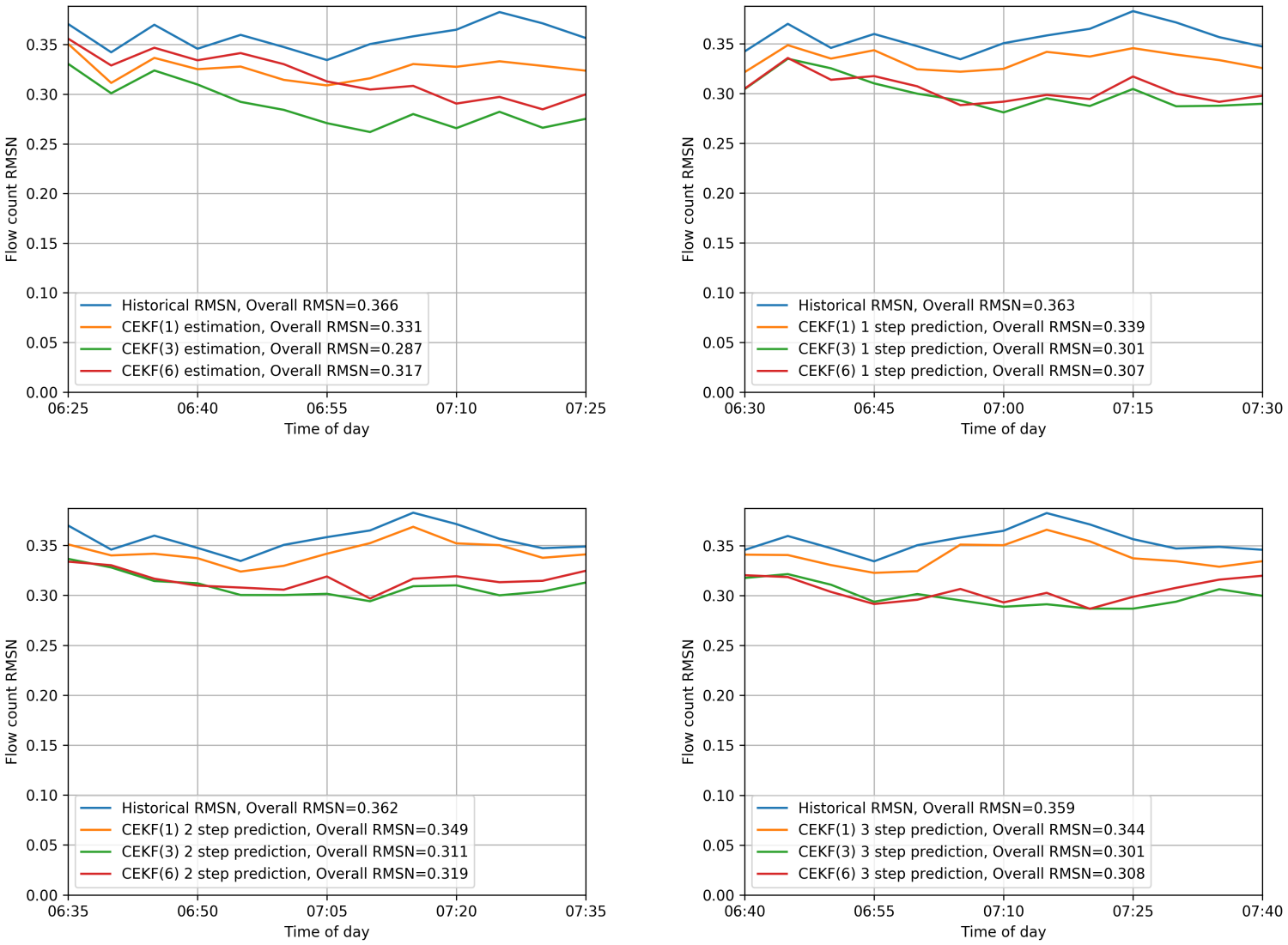


Figure 6-6: Flow volume RMSN for estimation (top left) and predictions, simulation period 6:20-8:20

Besides, we present the performances in each interval by showing the time-dependent RMSN values (Figure 6-6). It is noticeable that augmented models give the better predictions than CEKF(1) by a small but clear margin. One explanation of the CEKF(1)'s performance is the lack of delay modeling in the measurement equation. When using the non-augmented model, it ignores the correlation between parameters and measurements across intervals. Thus, previously estimated ODs cannot be adjusted. On the other hand, for augmented models with degree q, each state is estimated/updated q times. The previous traffic states are re-simulated with updated parameters through the "rollback" feature.

Following the discussion about modeling delay, we make a comment on the drawbacks of using the non-augmented model (CEKF(1)). It omits important independent variables (ODs in previous intervals) when the true model contains delay. The non-augmented model is forced to explain measurements with parameters in the same interval. Thus, the error term absorbs the effect caused by omitting variables, which results in a less accurate model. On the contrary, when we use an augmented model, part of the error in measurements are "explained away" by modeling the delay for parameters of previous intervals. Thus by applying such a model, the model does not force all the unexpected results in measurements be explained by parameters in the current interval. As a result, longer trips are captured in later intervals, and hence can be estimated better with augmented models. Therefore, we are likely to recover the true parameters. Suppose the AR model is good, predicted parameters will also be more accurate, which yield better traffic predictions.

As for the prediction performance, the augmented models improve over CEKF(1), but the improvement is less than the synthetic case study in Section 4.4. This observation may come from two reasons. First, as mentioned before, there are much more parameters (4121) than observations (less than 650 due to missing sensors) in each interval. Given the large degree of freedom in the problem, non-augmented models will perform well enough in terms of goodness-of-fit for sensors. Second, as discussed in Section 6.1.2, the excessive noise in measurements makes severe violation to the flow conservation law. Such erroneous surveillance data determines the lower bound

of the error rate.

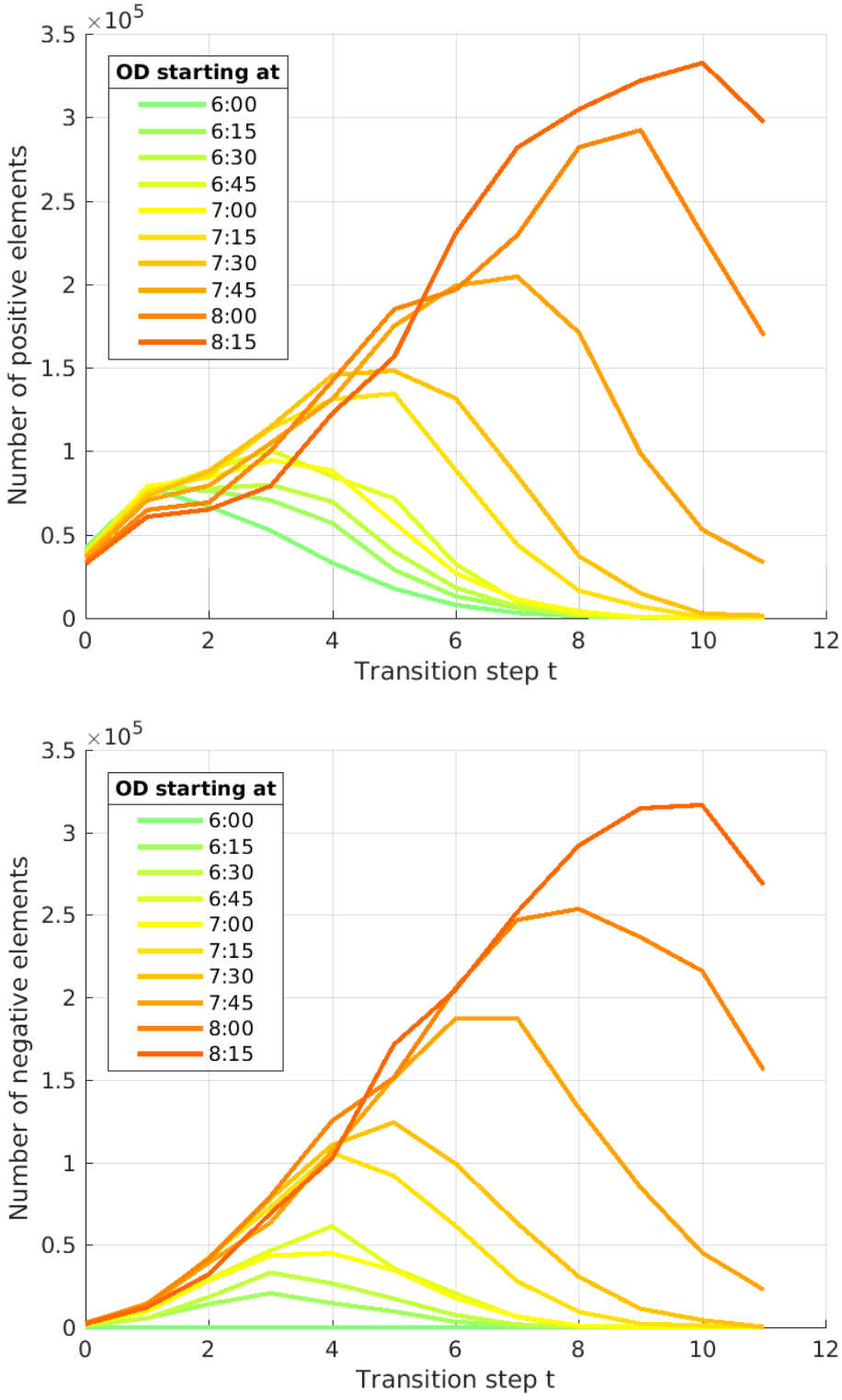


Figure 6-7: Number of positive and negative elements for the gradient $\boldsymbol{H}^h_{h+t}$ in each transition step t

Another interesting observation is that at later intervals, the prediction performance of CEKF(3) and CEKF(6) begins to deteriorate after 7:30 (Figure 6-6). To explain this observation, it is helpful to show some details of the H matrices used in the augmented models. Figure 6-7 shows the number of nonzero elements in gradient matrices. At 6:00, starting from an empty network, there are no negative elements for all transition step t. If we only look at transition step $t = 0$, as time interval increases from 6:00 to 8:15, the number of negative elements increases, while positive elements decrease. The negative elements are likely due to the congestion formed in the network, where assigning more vehicles will only increase the congestion level and reduce flows. However, the increase of negative elements in later intervals for the transition steps $t > 0$ is suspicious. The increase may be because the perturbations in a previous interval change the random number sequence in vehicle movement/queue dissipation in later intervals. Similar to the case in Chapter 3, gradients in later transition steps are prone to the simulation stochasticity due to more chance of interactions with the DTA simulator. Thus, a slight change will cause large variations in the realization of simulation. This conjecture explains the close number of positive and negative elements for higher transition steps.

We support the conjecture with Figures 6-8 and 6-9, which presents the distribution of nonzero elements for the gradient used in SSM and augmented models at 7:00. A uniform distribution of $[0, 1]$ can approximate reasonably the gradient elements in CEKF(1) (Figure 6-8). The uniform distribution conforms with the fact that the sensors capturing different fractions of ODs are equally likely. On the other hand, the distribution of gradients in augmented models is very different (Figure 6-9). There are numerous small values in the gradient, which can be approximated with Gaussian distribution. It is probable that the gradient is affected largely by the stochasticity in simulations. A large number of elements have noisy gradients, and thus, the augmented model that rely on them will yield worse estimation and prediction results. The noise in the gradients is an additional reason of limited improvement of CEKF(3) over CEKF(1). We believe the results with less noisy gradient estimations for state augmentation will further improve the performance of augmented models.

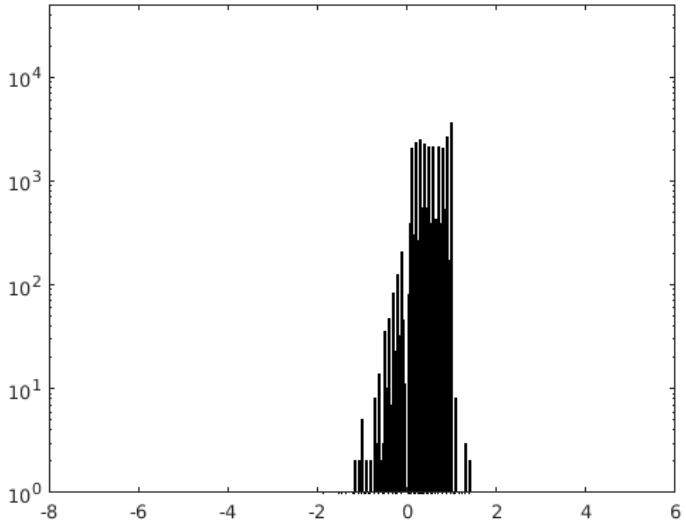


Figure 6-8: Distribution of nonzero elements of all gradients $\boldsymbol{H}_{h+t}^{h}$ for CEKF(1) (transition step $t = 0$) at 7:00

6.3.3 The Computation Performance

Lastly, we make some comments on the computational performance and the PSP, as it is extensively used in the experiments. First, the PSP method largely reduced the computational time for gradient estimation. The CEKF(1) has near real-time performance: the calibration for each interval takes around 5 minutes, as shown in Table 5.2. CEKF(3) takes around 20 minutes on average and CEKF(6) takes around 1 hour. Second, the computation complexity increases with the degree of augmentation, because of more conflicts introduced when quantifying the impact of parameters on future measurements. Nevertheless, the noisy gradient in later intervals is also responsible for the excessive conflicts introduced (after 7:00 in Figure 6-7), which results from the fact that excessive noise makes the gradient less sparse. Thus, a sparse and accurate gradient is beneficial for both calibration accuracy and efficiency. Lastly, the computational complexity of Kalman filtering operations increases in a cubic manner with the parameter dimension, which is also a reason for augmented models to take much longer time.

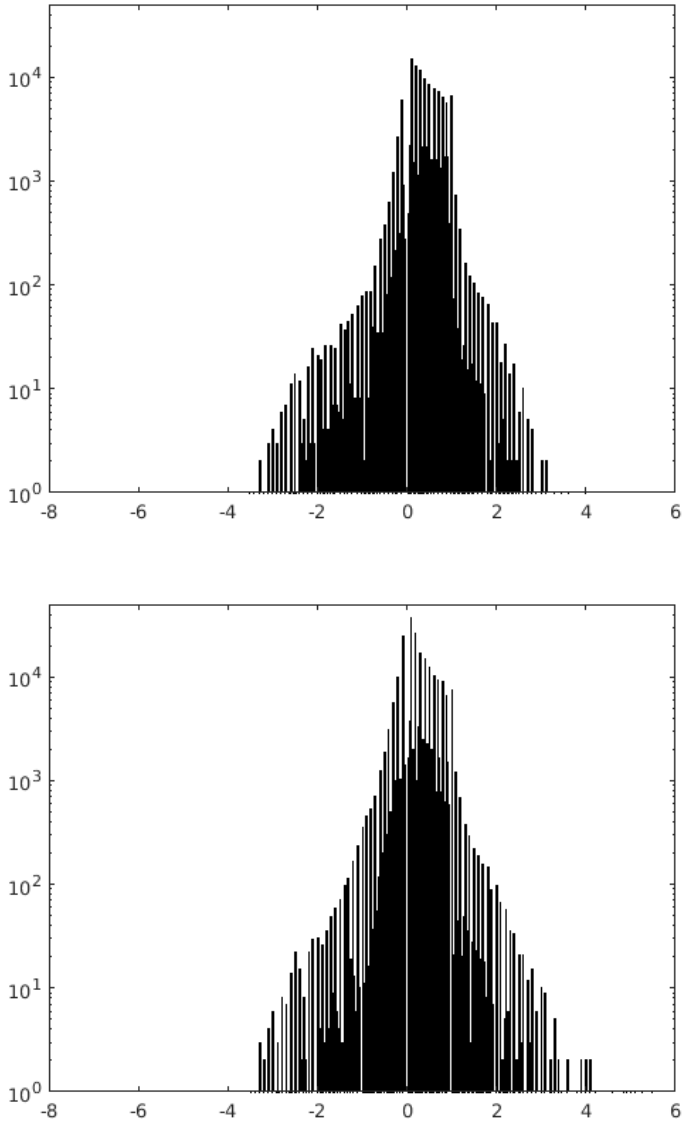


Figure 6-9: Distribution of nonzero elements of gradient $\boldsymbol{H}_{h+t}^{h}$ for CEKF(3) ($t = 0, 1, 2$, top) and CEKF(6) ($t = 0, 2, ..., 5$, bottom) at 7:00

6.4 Conclusion

From our discussion of results, we conclude that state augmentation is a useful technique that handles system delay, especially when sensors capture parameters in different time intervals on the traffic network. In such cases, it is erroneous to force the non-augmented (non-delayed) model to fully explain the discrepancy in simulated and observed measurements, while it may also be caused by incorrectly estimated pa-

rameters in previous intervals. Thus, by applying an augmented model, we consider important independent variables and their impact in current and subsequent intervals. This is beneficial to accurate estimation and prediction of traffic conditions. The case study with Singapore expressways illustrated that state augmentation improves over the non-augmented case by 13% for estimation and prediction accuracy. We expect the improvement to be greater, if we can address the issue of noisy gradient matrices. Thus, the noise in gradients should be a future direction as an extension for this case study.

Chapter 7

Conclusion

In this chapter, we first identify the contributions of this thesis to the state-of-the-art of online calibration for simulation-based Dynamic Traffic Assignment (DTA) systems. We then summarize the detailed findings in this research and discuss directions for future research.

7.1 Research Contributions

This research contributes to the field of online calibration for simulation-based DTA. The approaches developed in this thesis are generic to all simulation models. Specifically, this work contributes to the existing literature in the following respects:

- Proposing an error decomposition framework to account for the simulator randomness in online calibration. Two approaches are proposed to mitigate simulation stochasticity: (1) characterizing the error covariance for simulated measurements and (2) enforcing a sparse gradient structure.

- Applying an extension of the State Space Model to deal with the delayed measurement issue on large-scale networks and provide more accurate predictions. The approach is also helpful on small-scale and congested networks.

- Presenting a sparse gradient estimation procedure that significantly improves the computational performance of online calibration and facilitates real-time

performance.

7.2 Summary of Findings

The main findings from this thesis in the context of simulation stochasticity, generalization to large-scale networks, and computational performance are as follows:

- Considering simulation stochasticity
  - The stochasticity arises from the random numbers drawn in the simulation. Thus its stochasticity can be quantified by the statistics from simulations starting with different random seeds.
  - The simulated traffic measurements are *significantly* affected by the stochasticity (particularly speeds and link travel times in our case), and longer aggregation intervals reduce the randomness.
  - The transition between free flow and congestion scenarios are particularly prone to stochasticity.
  - Multiple realizations of the random measurement vector can be obtained from simulations with different seeds for the same interval. The *covariance matrix* calculated from these realizations describes the simulation error (under certain input parameters).
  - An error analysis suggests that the *covariance matrix* is crucial in online calibration. A synthetic test demonstrates an improvement in supply calibration after applying the covariance matrix.
  - The simulation stochasticity also affects the *gradient estimation* procedure. For the finite difference method, larger perturbation sizes give a less stochastic gradient estimation.
  - The Holm-Bonferroni test is helpful to identify the gradient elements that are not significantly different from zeros. Thus, the noise of those gradient elements can be eliminated by enforcing a *sparse gradient* structure.

  - A synthetic test demonstrated that enforcing the sparse gradient structure improves the calibration accuracy.

- State augmentation with the State Space Model (SSM) for large-scale networks
  - As network area grows, surveillance sensors and origin/destination nodes grow linearly, while OD pairs grow quadratically. Thus large-scale networks are likely to be unobservable.
  - A graphical representation for the SSM is introduced to intuitively explain the *violation* of the Markovian assumption.
  - The augmented SSM is presented to mitigate the violation and the graphical representation is utilized to show its compliance with the Markovian assumption.
  - Synthetic tests on a congested small-scale network demonstrates that the augmented SSM model yields more accurate and less biased predictions for online calibration.
  - A case study with the Singapore Expressway Network demonstrates its accuracy improvement on a large-scale network.

- Acceleration of the gradient estimation procedure for real-time performance
  - A partitioned simultaneous perturbation algorithm is presented to speed up gradient estimation. It utilizes the sparse gradient structure to group parameters and perturb them together; thus the number of calculations is reduced to the number of groups.
  - Finding the minimum grouping of parameters is a NP-hard problem. Finding the optimum may not be necessary because a heuristic solution may reduce most of the computational complexity compared with the finite difference method.
  - The application of the heuristic algorithm reduced by nearly 90% the computations necessary for OD estimation on the Singapore Expressway Network.

– The state estimation and prediction accuracy is comparable to the online calibration result using the finite difference method.

7.3 Future Research Directions

There are several potential future research topics in each of the three directions addressed in this thesis which are described next.

7.3.1 Considering simulation stochasticity

Since the covariance matrix changes with the simulated traffic scenarios, it may be beneficial to identify a dynamic measurement covariance matrix $\mathbf{\Sigma}_h$ to account for simulation stochasticity based on the current traffic conditions. For instance, during the transition between free flow and congestion traffic scenario, the covariance matrix has a higher magnitude on the diagonals. If the task to determine the appropriate covariance matrix is difficult, a simplification could be applied with a diagonal matrix.

As suggested in Section 6.3.2, the noisy gradients also play an important role in state augmentation. Thus, the stochasticity within traffic simulators should always be examined and addressed where applicable. Future research should also include the stochasticity in traffic flow measurements.

7.3.2 Online calibration for large-scale networks with real-time performance

The large-scale applicability and real-time performance are closely related in the context of online calibration. The extension to large-scale networks is challenging due to issues of accuracy and real-time computational constraints. This research addresses the accuracy issue with augmented State Space Model (SSM) on large-scale networks. Besides, the partitioned simultaneous perturbation (PSP) approach accelerates the gradient estimation procedure. However, the computational complexity largely increases with the augmented State Space Model. If the state dimension is n, the

matrix operations of $O(n^3)$ dominate the complexity of the Kalman filter. Polynomial as the algorithm is, the cubic increase of computation time makes the augmented SSM difficult to generalize.

A recent improvement termed Localized EKF (L-EKF) was proposed to address this (van Hinsbergen et al., 2012). The algorithm utilized an assumed sparse structure in $\boldsymbol{P}_{h|h-1}$ and decomposed the original EKF update into smaller and faster updates. The small updates are then collected and used to reconstruct the original EKF update. The main idea is to decompose the n parameters into collectively exhaustive groups, where each group includes the parameters closely related to each other. Then for each group, an EKF with a smaller dimension is executed. Thus, the computational complexity is reduced at the expense of enforcing the covariance structure. The complexity of L-EKF is controlled by $O(n_{group}n_{max}^3)$, where n_{group} is the number of groups, n_{max} is the size of the largest group. If groups are divided in similar sizes, then $n_{group}n_{max} \simeq n$. Therefore, the complexity will roughly decrease from $n_{group}^3 n_{max}^3$ to $O(n_{group}n_{max}^3)$. It is a promising approximation of the original EKF and the computational performance should be reported in the future research.

References

Ananthasayanam, M., Mohan, M. S., Naik, N., & Gemson, R. (2016). A heuristic reference recursive recipe for adaptively tuning the kalman filter statistics part-1: formulation and simulation studies. *Sādhanā*, *41*(12), 1473–1490.

Antoniou, C. (2004). *On-line calibration for dynamic traffic assignment* (Doctoral dissertation, Massachusetts Institute of Technology). Retrieved from `http://dspace.mit.edu/`

Antoniou, C., Ben-Akiva, M., & Koutsopoulos, H. (2004). Incorporating automated vehicle identification data into origin-destination estimation. *Transportation Research Record: Journal of the Transportation Research Board*(1882), 37–44.

Antoniou, C., Ben-Akiva, M., & Koutsopoulos, H. N. (2006). Dynamic traffic demand prediction using conventional and emerging data sources. In *Iee proceedings-intelligent transport systems* (Vol. 153, pp. 97–104).

Antoniou, C., Ben-Akiva, M., & Koutsopoulos, H. N. (2007). Nonlinear kalman filtering algorithms for on-line calibration of dynamic traffic assignment models. *IEEE Transactions on Intelligent Transportation Systems*, *8*(4), 661–670.

Antoniou, C., Koutsopoulos, H. N., & Yannis, G. (2007). An efficient non-linear kalman filtering algorithm using simultaneous perturbation and applications in traffic estimation and prediction. In *Intelligent transportation systems conference, 2007. itsc 2007. ieee* (pp. 217–222).

Ashok, K. (1996). *Estimation and prediction of time-dependent origin-destination flows* (Doctoral dissertation, Massachusetts Institute of Technology). Retrieved from `http://dspace.mit.edu/`

Ashok, K., & Ben-Akiva, M. E. (1993). Dynamic origin-destination matrix estimation and prediction for real-time traffic management systems. In *International symposium on the theory of traffic flow and transportation (12th: 1993: Berkeley, calif.). transportation and traffic theory.*

Ashok, K., & Ben-Akiva, M. E. (2000). Alternative approaches for real-time estimation and prediction of time-dependent origin–destination flows. *Transportation Science*, *34*(1), 21–36.

Ben-Akiva, M., Koutsopoulos, H. N., Antoniou, C., & Balakrishna, R. (2010a). Traffic simulation with DynaMIT. In *Fundamentals of traffic simulation* (pp. 363–398). Springer.

Ben-Akiva, M., Koutsopoulos, H. N., Toledo, T., Yang, Q., Choudhury, C. F., Antoniou, C., & Balakrishna, R. (2010b). Traffic simulation with MITSIMLab. In *Fundamentals of traffic simulation* (pp. 233–268). Springer.

Bierlaire, M., & Crittin, F. (2004). An efficient algorithm for real-time estimation and prediction of dynamic od tables. *Operations Research*, *52*(1), 116–127.

Chang, G.-L., & Wu, J. (1994). Recursive estimation of time-varying origin-destination flows from traffic counts in freeway corridors. *Transportation Research Part B: Methodological*, *28*(2), 141–160.

Chiu, Y.-C., Bottom, J., Mahut, M., Paz, A., Balakrishna, R., Waller, T., & Hicks, J. (2011). Dynamic traffic assignment: A primer. *Transportation Research E-Circular*(E-C153).

Coleman, T. F., & Moré, J. J. (1983). Estimation of sparse jacobian matrices and graph coloring blems. *SIAM journal on Numerical Analysis*, *20*(1), 187–209.

Cremer, M., & Keller, H. (1987). A new class of dynamic methods for the identification of origin-destination flows. *Transportation Research Part B: Methodological*, *21*(2), 117–132.

Daganzo, C. F. (1995). The cell transmission model, part ii: network traffic. *Transportation Research Part B: Methodological*, *29*(2), 79–93.

FHWA. (2009). *2009 urban congestion trends: How operations is solving congestion problems.*

FHWA. (2016). *2016 urban congestion trends: Using technology to measure, manage, and improve operations.*

Frederix, R., Viti, F., Corthout, R., & Tampère, C. (2011). New gradient approximation method for dynamic origin-destination matrix estimation on congested networks. *Transportation Research Record: Journal of the Transportation Research Board*(2263), 19–25.

Fu, M. C. (2002). Optimization for simulation: Theory vs. practice. *INFORMS Journal on Computing*, *14*(3), 192–215.

Fu, M. C. (2015). Stochastic gradient estimation. In M. C. Fu (Ed.), *Handbook of simulation optimization* (pp. 105–147). New York, NY: Springer New York. Retrieved from `https://doi.org/10.1007/978-1-4939-1384-8_5` doi: 10.1007/978-1-4939-1384-8_5

Google Maps. (2016). *Singapore road network.* Retrieved from `http://www.google.com/maps/@1.3482868,103.756796,12z` ([Online; accessed May 13, 2016])

Hegyi, A., Girimonte, D., Babuska, R., & De Schutter, B. (2006). A comparison of filter configurations for freeway traffic state estimation. In *Intelligent transportation systems conference, 2006. itsc'06. ieee* (pp. 1029–1034).

Hegyi, A., Mihaylova, L., Boel, R., & Lendek, Z. (2007). Parallelized particle filtering for freeway traffic state tracking. In *Control conference (ecc), 2007 european* (pp. 2442–2449).

Huang, E. (2010). *Algorithmic and implementation aspects of on-line calibration of dynamic traffic assignment* (Master's thesis, Massachusetts Institute of Technology). Retrieved from `http://dspace.mit.edu/`

Huynh, N., Mahmassani, H., & Tavana, H. (2002). Adaptive speed estimation using transfer function models for real-time dynamic traffic assignment operation. *Transportation Research Record: Journal of the Transportation Research Board*(1783), 55–65.

Lighthill, M. J., & Whitham, G. B. (1955). On kinematic waves ii. a theory of traffic flow on long crowded roads. *Proc. R. Soc. Lond. A*, *229*(1178), 317–345.

Mahmassani, H. S. (2001). Dynamic network traffic assignment and simulation methodology for advanced system management applications. *Networks and spatial economics*, *1*(3-4), 267–292.

Murphy, K. P. (2002). Dynamic bayesian networks. *Probabilistic Graphical Models, M. Jordan*, *7*.

Okutani, I., & Stephanedes, Y. J. (1984). Dynamic prediction of traffic volume through kalman filtering theory. *Transportation Research Part B: Methodological*, *18*(1), 1–11.

Osorio, C., & Bierlaire, M. (2013). A simulation-based optimization framework for urban transportation problems. *Operations Research*, *61*(6), 1333–1345.

Peeta, S., & Ziliaskopoulos, A. K. (2001). Foundations of dynamic traffic assignment: The past, the present and the future. *Networks and Spatial Economics*, *1*(3-4), 233–265.

Richards, P. I. (1956). Shock waves on the highway. *Operations research*, *4*(1), 42–51.

Schneider, R., & Georgakis, C. (2013). How to not make the extended kalman filter fail. *Industrial & Engineering Chemistry Research*, *52*(9), 3354–3362.

Schrank, D., Eisele, B., Lomax, T., & Bak, J. (2015). 2015 urban mobility scorecard.

Sorenson, H. W. (1970). Least-squares estimation: from gauss to kalman. *IEEE spectrum*, *7*(7), 63–68.

Spall, J. C. (1992). Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *Automatic Control, IEEE Transactions on*, *37*(3), 332–341.

St-Pierre, M., & Gingras, D. (2004). Comparison between the unscented kalman filter and the extended kalman filter for the position estimation module of an integrated navigation information system. In *Ieee intelligent vehicles symposium* (pp. 831–835).

Tavana, H., & Mahmassani, H. (2000). Estimation and application of dynamic speed-density relations by using transfer function models. *Transportation Research Record: Journal of the Transportation Research Board*(1710), 47–57.

van Hinsbergen, C. P., Schreiter, T., Zuurbier, F. S., Van Lint, J., & Van Zuylen, H. J. (2012). Localized extended kalman filter for scalable real-time traffic state estimation. *IEEE transactions on intelligent transportation systems*, *13*(1), 385–394.

Wang, Y., & Papageorgiou, M. (2005). Real-time freeway traffic state estimation based on extended kalman filter: a general approach. *Transportation Research Part B: Methodological*, *39*(2), 141–167.

Zhang, C., Osorio, C., & Flötteröd, G. (2017). Efficient calibration techniques for large-scale traffic simulators. *Transportation Research Part B: Methodological*, *97*, 214–239.

Zhang, H. (2016). *Constrained extended kalman filter: an efficient improvement of calibration for dynamic traffic assignment models* (Master's thesis, Massachusetts Institute of Technology). Retrieved from `http://dspace.mit.edu/`

Zhang, H., Seshadri, R., Prakash, A. A., Pereira, F. C., Antoniou, C., & Ben-Akiva, M. E. (2017). Improved calibration method for dynamic traffic assignment models: Constrained extended kalman filter. *Transportation Research Record: Journal of the Transportation Research Board*(2667), 142–153.

Zhou, X., & Mahmassani, H. S. (2007). A structural state space model for real-time traffic origin–destination demand estimation and prediction in a day-to-day learning framework. *Transportation Research Part B: Methodological*, *41*(8), 823–840.