Title
Studies on Complex and Connected Vehicle Traffic Networks

Author
Wright, Matthew Abbott

Publication Date
2019

Peer reviewed|Thesis/dissertation

Studies on Complex and Connected Vehicle Traffic Networks

by

Matthew Abbott Wright

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering- Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Roberto Horowitz, Chair
Professor Kameshwar Poolla
Professor Scott Moura

Summer 2019

Studies on Complex and Connected Vehicle Traffic Networks

Abstract

Studies on Complex and Connected Vehicle Traffic Networks

by

Matthew Abbott Wright

Doctor of Philosophy in Engineering- Mechanical Engineering

University of California, Berkeley

Professor Roberto Horowitz, Chair

Transportation networks such as road networks are well-known for their complexity. Its users make choices of route, which mode to take, etc.; these users then interact with each other, producing emergent dynamics such as traffic jams on roads. These localized multi-user emergent physical phenomena then interact with similar group movements occurring in other locations, creating more complex network-scale dynamics. These patterns of hierarchical levels of organization and emergent phenomena at each level are typical of so-called "complex systems." In addition, the increasing adoption of information-technology systems like connected and autonomous vehicles is creating new challenges in modeling transportation networks, as new emergent behaviors become possible, but also provide new sources of information and possibilities for traffic operations management.

The complexity of transportation networks precludes the use of a single all-encompassing theory for all situations at all scales. This dissertation describes several analyses into understanding and controlling emergent dynamics on road traffic networks. It is broken into three parts. The first part proposes models for several new phenomena at the "macroscopic," group-of-vehicles to group-of-vehicles, level. In particular, we solve a problem of modeling arbitrary road junctions with populations of behaviorally-heterogenous vehicles, where the vehicle flows are modelled by a continuum-approximation, partial-differential-equation-based model. We also present several new modeling constructions for a particular complex road network topology: freeways with managed lanes. It has been noted that these managed lane-freeway networks induce new emergent behaviors that are not present in traditional freeways; we propose modeling techniques for several of them, and fit them into traditional modeling paradigms.

The second part presents several contributions for estimating the state of the macro-scale traffic dynamics on the road network, based on the micro-scale data of global navigational satellite system readings of the speed and position of individual vehicles. These contributions are extensions of the particle filtering mathematical framework. First, we demonstrate the use of a Rao-Blackwellized particle filter in assimilating vehicle-local speed measurements to better estimate the macroscopic density state of a freeway. Then, we propose new "hypothesis-

testing" particle filters that can be used to reject outlier or otherwise malign measurements in a principled statistical manner.

The third and final part presents two items on applying deep neural networks to transportation system problems at smaller scales. Both items make use of neural attention, which is a neural network design technique that allows for the integration of structural domain knowledge. First, we demonstrate the applicability of this technique towards estimating aggregate traffic states at the lane level, and present evidence that designing the neural network architecture to encode different types of lane-to-lane relationships (e.g., upstream lane vs neighboring lane) greatly benefits statistical learning. Then, we apply similar methods to an autonomous vehicle coordination problem in a deep reinforcement learning framework, and show that an attention-based neural network that allows each vehicle to attend to the other vehicles enables superior learning compared to a naive, non-attention-based architecture, and also allows principled generalization between varying numbers of vehicles.

To my family and friends

Contents

List of Figures

Acknowledgments

There are so many people who I am deeply indebted to, either for their mentorship or their friendship, and I dread the feeling of realizing I had neglected to thank just one of them. To anyone who I did miss, I apologize profusely.

First, I of course must thank Roberto Horowitz, my advisor. His mentoring has been a valuable resource over the past years.

I must thank several of the people who I have had the chance to work closely on various projects. My labmates Negar Mehr, Ruolin Li, and Cheng-Ju Wu have been wonderful colleagues for many years. My frequent coauthor Alex A. Kurzhanskiy deserves particular mention. Some of my work with Alex appears in chapters 4 and 5 here. Simon Ehlers, who visited Berkeley for several months in 2018, helped immensely with the work that is presented in chapter 8. Thanks also to the other people who I have had the pleasure to coauthor with: Gabriel Gomes, Alexandre Bayen, and Anthony Patire. Thank you also to Gabriel for giving me the opportunity to mentor several of the Masters of Engineering capstone project teams.

I had the opportunity to make many friends among my classmates and colleagues during my studies here. Just among the people in my program or who I worked with, thank you to Drew Sabelhaus, Daniel Pineda, Colin Ho, Matt Powell-Palm, Lily Hu, David Fernández, Rachael Hager, Claire Funke, Ashwin Carvalho, Spencer Frank, Louis Malito, Abdulrahman Jbaily, Geoff Wehmeyer, Rishi Ganeriwala, Ugo Rosalia, Eric Kim, Sam Coogan, Stan Smith, Tuomas Haarnoja, Chris Meissen, Ean Hall, and everyone else for being wonderful colleagues. Thank you to my colleagues and neighbors at PATH, including Qijian Gan, Yi Zhou, Thomas Schreiter, Yiguang "Ethan" Xuan, Samitha Samaranayake, Jack Reilly, Leah Anderson, Greg Merritt, Gary Gremaux, and Brian Peterson, for helping foster a welcoming environment.

Thanks are also due to the many professors and teachers I have had the privilege to learn from. In particular, thank you to Andy Packard, Kameshwar Poolla, Scott Moura, Francesco Borrelli, Murat Arcak, Pravin Varaiya, Ching-Yao Chan, Kurt Keutzer, and all other the faculty and researchers I've had the chance to discuss research with at PATH, BDD, and BAIR, including of course Joey Gonzalez, who I am excited to be joining as a postdoc in a few short weeks.

Many of the administrative and departmental staff that have helped me at Berkeley have been true saints. Thank you to Donna Craig, Yawo Akpawu, Emily Higgins, Shareena Samson, and Isabel Blanco at the ME Department; and Rosita Alvarez-Croft, Jessica Rojas, and Angie Abbatecola at PATH and BDD.

I had the chance to work in two very educational industry internships during my studies. Thank you to Tony Haro, Jane Macfarlane, Mohammadreza Mostajabi, Davide Pietrobon, and everyone else at HERE, as well as Juan Pablo Soulier, Anmol Garg, Ben Rouif, Dylan Diamond, and everyone else at Tesla, for being great colleagues and mentors.

Lastly, I would be remiss to not acknowledge the California Department of Transportation, Berkeley DeepDrive, the National Science Foundation, the Anselmo J. Macchi Fellowship Fund, and the UC Berkeley Graduate Division for, at various points, supporting my PhD research.

Chapter 1

Introduction

1.1 The Transportation System

The transportation system is a central component in the modern built environment and economy. Many economic trends and business practices considered core parts of contemporary economic development, such as just-in-time industrial practices, the business models of many Wall Street darlings, and, indeed, the modern era's globalized trade networks in general, are inherently dependent on an easily-accessible, low-marginal-cost, and highly-reliable transportation network (Small and Verhoef, 2007; Carrion and Levinson, 2012). It has been classically argued by economists and policymakers that transportation infrastructure can serve as a catalyst for economic growth (Munnell, 1992; Lakshmanan, 2011). The expansion of road and rail networks to gain their trade benefits has also been core components of countries' geopolitical strategies, notably by the U.S.'s postwar Marshall Plan in Europe, and by China in, e.g., the widely-discussed contemporary "belt and road" plan to extend Chinese transportation networks throughout greater Asia and beyond.

Gains in social utility from transportation are not just derived from infrastructure construction. Given the centrality of the transportation system to economic activity and peoples' everyday lives, marginal gains in efficiency or service quality from operations improvements or technological developments can diffuse broadly and provide great social benefits. Today, speculative technologies like autonomous vehicles are assigned incredibly large monetary values of social benefits from promised efficiency gains and safety improvements (Fagnant and Kockelman (2015), for example, project tens of billions to hundreds of billions of dollars of annual economic benefits to the U.S. economy annually).

1.2 Mathematical Features of Infrastructure Networks

Transportation networks, and other built-environment systems like the electrical grid and computer/information networks, have been characterized as *complex systems* (Yerra and Levinson, 2005; Vespignani, 2010; Pagani and Aiello, 2013). While there is no universally-agreed-upon definition for a "complex system" or "complexity," two central descriptors common to many definitions are *self-organization* and *emergence* (Waldrop, 1992; Mitchell, 2009; Holland, 2014). Both characteristics are evident in transportation systems: while individual users have a set of rules they must act within, both their end-goals (i.e., their destination) and their choices taken to reach these goals are self-determined. The self-organization of individual travelers through the transportation system, taken in aggregate, generates emergent, macroscopically-coherent, phenomena, such as organizational route hierarchies (Yerra and Levinson, 2005) and dynamic behaviors like transitory traffic jams (Helbing, 1998; Treiber and Kesting, 2013).

Other features attributed to complex systems (Miller and Page (2007); Mitchell (2009); Northrop (2011); Holland (2014), etc.) that are exhibited in transportation systems (with road networks as a specific example system) include

- *Chaotic behavior:* Many authors over the years have proposed the labeling of traffic flow dynamics as "chaotic" (or at least having chaotic components) (Dendrinos (1994); Nagel and Rasmussen (1994); Frazier and Kockelman (2004); Adeli and Jiang (2008), etc.). Chaotic behaviors are often described as having "sensitive dependence on initial conditions," and, as a consequence, being inherently difficult to predict, as a small error when measuring the initial state of the system can result in a large prediction error. In a road network, an obvious example of this sensitive dependence arises in the formation of traffic jams – it has been observed both theoretically (Nagel and Rasmussen, 1994) and empirically (Sugiyama et al., 2008) that, in a smooth traffic flow that qualitatively appears stable, a small perturbation of the inter-vehicle spacing can generate a jam, bringing the flow to a halt.

- *Nonlinearity:* Nonlinearity is closely related to chaotic behaviors. More precisely, the amplification of initial uncertainties that is central to chaotic systems emerge from system nonlinearities. The critical mathematical feature of nonlinearity in this context is the irreducibility of a nonlinear phenomenon into its constituent parts via superposition. Nonlinearity is ubiquitous in transportation systems; even the most simplified models of road traffic networks, that abstract away individual travelers or individual vehicles entirely, are highly nonlinear (Lighthill and Whitham, 1955b; Richards, 1956; Treiber and Kesting, 2013).

- *Hierarchical Compositions:* A complex system's emergent phenomena are caused by interacting discrete entities or agents (Simon, 1996). Typical of these interactions are a

hierarchy of groupings, with multiple individual agents forming a somewhat coherent group, that interacts with other such groupings, and these group-group interactions being more easily described at this group level, rather than the individual agent level (Holland, 2012). Such a hierarchy is evident in transportation networks: individual travelers group themselves with other travelers with whom they share a vehicle, road, etc., then interactions like traffic jams emerge between interactions of different vehicles, and their transmission through the network is visible at the road-to-road level of hierarchy as well.

- *Integration with networks across domains:* Relatively recently in the history of the study of networks and complexity, authors have recognized the importance of how large complex networks that may at first be considered as distinct – for example, the road network and the internet – actually interrelate with each other in significant ways (Rosvall et al., 2005; Kurant and Thiran, 2006; Vespignani, 2010). In the engineering context, the awareness of the interrelations between the "physical" domain (e.g., the actual, physical cars on the road) and the "informational" domain (e.g., the availability of information about the traffic network state on the internet), and how these interrelations can permit phenomena that would not be predicted otherwise (for example, drivers obtaining information about the physical state of far-off regions of the road network over the internet, and then reacting to it) is exemplified by the widespread use of the term "cyber-physical system" (Rajkumar et al. (2010); Lee and Seshia (2017), etc.), to promote the co-importance of the informational and physical domains. More recently, engineered systems that directly interact with people like infrastructural networks, have also begun to be referred to as "cyber-physical-social systems" (Cassandras, 2016), to promote the human factor to equal importance as the engineered system(s).

The presence of these many complexities means that there is no single theory or discipline to fully describe an infrastructure system. Rather, an effort to engineer, study, or improve a transportation system will call on, to varying degrees, civil engineering, electrical engineering/computer science, economics (Small and Verhoef, 2007; Redding and Turner, 2015), sociology (Jacobs, 1961), psychology (Legrain et al., 2015; Sadigh et al., 2016) (and behavioral economics (Walker et al., 2011; Abou-Zeid et al., 2013)), and others; not to mention various mathematical tools and methods.

The content presented in this dissertation features a similar breadth. We do not attempt to integrate most of the fields just mentioned, and instead remain in the context of systems and control engineering. Even then, we find that considering a road network at different levels in the hierarchical organization of a complex system requires entirely distinct mathematical tools. This dissertation's material naturally divides into three distinct parts, each making use of different mathematical tools and drawing from different disciplines.

The unifying theme of the three parts is the modeling, analysis, estimation, and control of the particular emergent features that appear at different levels of the hierarchical organization of the complex system of the road network. Much of the work also leverages

new technological connectivities – the "cyber" in the cyber-physical system. The growth of information technology into the road network has been long recognized as bringing new challenges and opportunities to the study of transportation systems. Hence, in referring to "complex and connected" transportation systems in this dissertation's title, we promote the connectivity to an equal mathematical importance as the physical complexity itself, echoing the systems-engineering view of the cyber-physical system.

1.3 Outline of this dissertation

As mentioned, this dissertation is structured into three parts. Each part, and each part's component chapters, are written with the goal of being able to stand alone, so that readers interested only in a subset can skip the remainder of the dissertation.

1.3.1 Part I

Part I discusses advances in macroscopic road traffic network modeling. By macroscopic, we mean models that explicitly attempt to capture the *aggregate* behavior of the traffic flow system *ab initio*. This is in contrast to so-called *microscopic* models, which model individual vehicles and their relations, and instead allow the macroscopic behavior to emerge from the encoded vehicle-local and vehicle-to-vehicle rules. Macroscopic models, by bypassing the "emergence" step entirely of vehicle-vehicle (or driver-driver) interactions, are by definition simpler than a microscopic model as applied to macroscopic phenomena, but model design at the macro level requires careful analysis to ensure that they realistically capture the complex phenomena of interest. The body of work in this part concerns models for emergent phenomena that occur at the *group of vehicles to group of vehicles* level in the organizational hierarchy. This part contributes the first quantitative models for many such emergent phenomena that had previously been observed in data and described qualitatively.

Chapter 3 analyzes rules for modeling the macroscopic dynamic behavior at road junctions. The culminating result of this chapter is a new method for solving for the dynamics of road junctions, when the junctions are modeled by a so-called "second order" macroscopic system. The second-order system, as we will discuss, is one in which the traffic flows on each link are described by two coupled partial differential equations (PDEs), one a continuity equation with a nonlinear flux function, and the other an advection equation where the advected variable governs the first PDE's flux function. Junctions where flows governed by this two-PDE system meet must generalize these scalar (that is, scalar in space) PDEs to arbitrary couplings, in a manner that also conserves drivers' desired destination (i.e., their specified path through the junction), as well as realistic queueing and lane-blockage behavior. This chapter is directly related to the contemporary study of mixed-autonomy traffic flows (Mehr and Horowitz, 2019; Mehr, 2019), providing a new physical model to advance the study of mixed-autonomy traffic systems. This chapter relates to material that was presented in Wright et al. (2016, 2018); Wright and Horowitz (2017a).

The remainder of Part I discusses modeling constructions for a particular topologically-complex traffic flow system of interest: road networks consisting of a mainline, general-purpose lane, and an adjacent managed lane reserved only for a subset of vehicles. In the macroscopic world, such a network topology means a system with two adjacent and interacting, but nevertheless physically distinct, PDEs. Chapter 4 proposes modeling constructions, both *topological* and *phenomological* (in the sense that they are novel models for a raft of empirically-observed emergent macro-scale phenomena that are thought to be particular to this two-adjacent-flows setting) for this situation. In particular, we define explicitly network topologies for the two dominant managed lane-freeway network configurations (continous or gated-access), as well as macroscopic models for aggregate behaviors known as the *friction*, *inertia*, and *smoothing* effects. To the best of our knowledge, our smoothing effect model is the first proposed macroscopic model for the smoothing effect, and those for others are the first to be integrated into the greater macroscopic framework. This chapter is based on material that was previously reported in Wright et al. (2019b).

Chapter 5 is a further discussion of the managed lane-freeway network modeling framework, but explains how to actually put it into practice. It is well-known that macroscopic traffic models are difficult to calibrate (Treiber and Kesting, 2013, Chapter 16), since they have many parameters, and, being nonlinear and highly networked by construction, minor parameter tweaks can have large effects on the macroscopic behavior of interest. In particular, we are motivated by the use case of these models for civil authorities as planning tools, where an *accurate* model of the base case, present-day scenario is required, that can be used to generate predictions of the effects of proposed network developments. This chapter thus presents a calibration (model calibration being the usual term for parameter identification for these models) methodology for our expanded macroscopic modeling framework that acts as a superset for the existing macroscopic calibration loop, as well as several case studies on real-world managed lane-freeway networks in California. This chapter is based on material that was previously reported in Wright et al. (2019c).

1.3.2 Part II

Part I's discussion of *ab-initio* macro-scale models serves as a natural jumping-off point for Part II. As mentioned earlier in this introduction, macro-scale dynamics that emerge from unmodeled or unobserved micro-scale dynamics and interdynamics are inherently stochastic, which suggests the need for state estimation schemes in any application. In the particular problems discussed here, we present results on problems of estimating the macroscopic state of the system using measurements taken from the microscopic constituent elements (i.e., the vehicles in the road networks) (Seo et al., 2017). This is an instance of a central problem in systems of controls engineering called sensor or data fusion. In our applications, we need to integrate micro-scale data (i.e., measurements from individual vehicles) into our macroscopic models (as well as fusing same with macro-scale sensor readings from traffic fixtures).

Chapter 6 discusses an approach for data fusion based on a particular estimation method known as a *Rao-Blackwellized* Particle Filter. A particle filter (Gordon et al., 1993) is a

popular state estimation scheme, and the "Rao-Blackwellized" particle filter (Doucet et al., 2000) is one that samples only a subset of the state variables when moving forward in time and explicitly computes a distribution for the rest (as opposed to a vanilla particle filter that samples all state variables). In our application, we make use of a Rao-Blackwellized particle filter where our explicitly-computed state variable subset are found via fundamental relations from the macroscopic traffic theory. This construction lets us easily compute likelihoods of microscopic measurements (and thus use them for state estimation). This chapter's material was previously reported in Wright and Horowitz (2016).

Chapter 7 is an extension of this macro-scale esimation from micro-scale measurements setting. In this chapter, we describe a method for outlier rejection of these micro-scale data. This chapter's work was motivated by the fact that in the study reported in chapter 6, the received micro-scale data were sourced from GNSS transponders controlled by end-users, and upon manual examination we found instances of clearly erroneous data (data that were not matched to the correct road, data that displayed a zero velocity near many nonzero velocities, etc.). In particular, this chapter develops a particle filtering scheme with a hypothesis-testing step, to reject these outlier measurements prior to fusing them into the state estimator. Of particular note is that we derive and demonstrate methods for both if there exists a proper model of faulty data, and one for if the characteristics of faulty data are unknown. This chapter's material was previously reported in Wright and Horowitz (2018), and an expanded version is currently in preparation.

1.3.3 Part III

The last part of this dissertation presents efforts on applying deep neural networks to transportation system problems. Deep neural networks have seen their popularity grow at a meteoric rate in recent years and have been applied towards solving a wide variety of problems. The expressive power of deep neural networks lies in their inherent nonlinearities and complexity, such that they can be used to approximate arbitrarily complex functions with surprising effectiveness. These characteristics suggest that they can be a natural fit for problems in the transportation system, where every problem is nonlinear, networked and complex. In particular, the two studies reported in this part make use of a neural network architecture known as *attention* (Bahdanau et al., 2015; Vaswani et al., 2017), with the guiding principle of encoding prior knowledge about network relations within the transportation network to guide and enhance statistical learning (Battaglia et al., 2018).

Chapter 8 discusses statistical learning of a lane-level dynamics model in transportation networks via a deep neural network with these attentional architectures. This scale of resolution is far too fine to be able to accurately use macroscopic models à la the freeway-scale case studies presented in part I (specifically, there are not enough vehicles to aggregate over to obtain reliable average dynamics trends), but real-time estimation and prediction are just as necessary for control. We consider as a case study a problem of estimating the number of vehicles occupying a lane in a timeseries manner, using data from in-lane fixed loop detectors and signal timing information as regressors. This chapter's work considers the network of

lanes in a road network as a multidimensional lane graph, and uses a particular attentional construction based on the "graph attention network" neural network layer of Veličković et al. (2018) to learn how to aggregate information from adjacent lanes, in different manners based on the actual lane-to-lane relationship (self, upstream, downstream, or neighbors). This work presents evidence that the deep neural network is able to meaningfully use the physical and topological information of the road network encoded via the lane graph to learn a powerful statistical dynamics model. This chapter draws from material in Wright et al. (2019a).

Chapter 9 goes even smaller-scale, to the problem of control of individual vehicles using deep reinformcent learning. In particular, we consider the problem of multi-agent reinforcement learning, with an application towards autonomous vehicle coordinative control. Our neural network learning controller interacts with a microscopic traffic simulator to coordinate several vehicles sharing a road with non-controlled vehicles (governed by a behavioral model) to mitigate the effects of naturally-occurring congestion, in an interventative manner to counteract the congestion worsening as it cascades through the agent-to-agent network's nonlinearities. Our particular contribution of this chapter is the proposal of using attention-based neural networks as the deep neural network controller in these multi-agent setups, which have inbuilt architectural benefits that allow them to relate to other agents in relationship-type-dependent ways (similar to how the attentional network used in chapter 8 allowed the statistical learning to learn to view data on upstream, neighboring, etc. lanes differently), and also generalizes naturally to *variable numbers* of agents. The particular well-suitedness of the attentional neural architecture to this multi-agent reinforcement learning problem is exhibited by the decentralized attentional controller's superior performance to a non-attentional but fully-centralized one. This chapter's material is also reported in Wright and Horowitz (2019a).

Chapter 2

Mathematical Preliminaries

In this chapter, we will give some background and preliminaries that become relevant in more than one later chapter. Background information that is relevant only to one chapter will be discussed at the beginning of that chapter.

2.1 Discrete-time stochastic systems and particle filtering

We summarize the basic definition of a discrete-time stochastic system. The material in this section is needed in part II.[1]

2.1.1 Basic Definitions

Let $x_k \in \mathbb{R}^N$ be the state of our system at time k. The system state is not fully observed; instead what is observed at time k is a measurement vector $y_k \in R^{M_k}$. The state and observation vectors' temporal behavior are governed by stochastic update and output equations,

$$\begin{aligned} x_k &= \mathcal{F}_\theta\left(x_{k-1}\right) \\ y_k &= \mathcal{G}_\theta\left(x_k\right) \end{aligned} \tag{2.1}$$

with θ a parameter vector describing the randomness or process/measurement noise of $\mathcal{F}$ and $\mathcal{G}$. An equivalent probabilistic notation may rewrite (2.1) as

$$X_k|\left(X_{k-1} = x_{k-1}\right) \sim f_\theta\left(x_k|x_{k-1}\right) \tag{2.2a}$$

$$Y_k|\left(X_k = x_k\right) \sim g_\theta\left(y_k|x_k\right) \tag{2.2b}$$

[1] In chapter 9, we discuss *Partially-Observed Markov Decision Problems* (POMDPs), which are nearly equivalent to the items presented in this section, but in that chapter we use the POMDP notation because it is the standard for the reinforcement learning field of literature in which that chapter fits.

where, following conventions of probability, a capital letter (e.g., X_k) denotes a random vector and a lower-case letter (e.g., x_k) denotes the value of a particular realization. The functions on the RHS's of (2.2) are probability density functions (PDFs). More precisely, $f_\theta(\cdot)$ and $g_\theta(\cdot)$ are the PDFs of the conditional distributions for the random variables X_k given X_{k-1} and Y_k given X_k, respectively. Also, X_0, the initial state of the system, is distributed according to some other distribution, $X_0 \sim p_\theta(x_0)$.

Oftentimes the parameter θ is omitted when writing the above distributions to reduce notational clutter.

2.1.2 The Filtering Problem

The model-based *filtering problem*, a classic problem in stochastic systems, is the problem of estimating the unknown system state $X_k(\forall k)$ from the known observations $Y_{0:k}$ (Doucet and Johansen, 2011), where where $Y_{0:k}$ is shorthand for the set $\{Y_0, Y_1, \ldots, Y_{k-1}, Y_k\}$. This is often done iteratively forward in time, repeating a two-step process at each successive time k.

The first step is called the **prediction step**. Assuming that we have an estimate of the PDF of the random variable $X_{k-1}|Y_{0:k-1}$ from the previous timestep, we can use (2.2a) to obtain

$$p_\theta(x_k|y_{0:k-1}) = \int f_\theta(x_k|x_{k-1}) p_\theta(x_{k-1}|y_{0:k-1}) dx_{k-1}. \tag{2.3}$$

The second step is called the **filtering step** or **update step**. Here, we use the obtained measurements y_k and (2.2b) to compute

$$p_\theta(x_k|y_{0:k}) = \frac{p_\theta(x_k|y_{0:k-1}) g_\theta(y_k|x_k)}{p_\theta(y_k|y_{0:k-1})} \tag{2.4}$$

where

$$p_\theta(y_k|y_{0:k-1}) = \int p_\theta(x_k|y_{0:k-1}) g_\theta(y_k|x_k) dx_k. \tag{2.5}$$

Note that (2.4) is a particular statement of Bayes' rule, with $p_\theta(x_k|y_{0:k-1}), g_\theta(y_k|x_k)$, $p_\theta(y_k|y_{0:k-1})$, and $p_\theta(x_k|y_{0:k})$ playing the role of the prior, likelihood, marginal likelihood, and posterior PDFs, respectively. Because of this, the iterative predict-update approach to filtering is sometimes called *recursive Bayesian estimation* (Gordon et al., 1993).

For some simple classes of systems $f_\theta(\cdot)$ and measurement distributions $g(\cdot)$, the computations in (2.3)-(2.5) are computable in closed form (the most well known example being that if both $f_\theta(\cdot)$ and $g_\theta(\cdot)$ are affine in the state x_k with additive white Gaussian noise, all PDFs in the recursion (2.3)-(2.5) can be computed exactly through matrix algebra, and is known as the Kalman Filter (Kalman, 1960)). In more general settings with more complex system and noise behaviors, some numerical approximation is required.

2.1.3 Particle Filter

One popular approximation method when the integrals in (2.3) and (2.5) are difficult or computationally intractable is the *particle filter* (Gordon et al., 1993; Arulampalam et al.,

2002; Doucet and Johansen, 2011). A particle filter may be used even when there is no closed-form expression for $f_\theta(\cdot)$ (precluding many classic numerical integration schemes), but the PDF may be sampled repeatedly, such as by running a stochastic simulation many times.

A particle filter is constructed by replacing the PDFs for X_k in the filtering equations (2.3)-(2.5) with approximate PDFs, which we will denote with a hat (e.g., $\hat{p}_\theta(\cdot)$ for $p_\theta(\cdot)$). These approximate PDFs are made up of many discrete samples (also called particles) from the continuous PDF. The particles are generated by repeatedly sampling from $f_\theta(\cdot)$.

In other words, a particle filter can approximate continuous PDFs via discrete probability mass functions (PMFs). For example, a particle filter approximation of the posterior PDF $p_\theta(x_k|y_{0:k})$ from equation (2.4) may be written as

$$p_\theta(x_k|y_{0:k}) \approx \hat{p}_\theta(x_k|y_{0:k}) = \sum_{p=1}^{P} p_\theta(x_k^p|y_{0:k})\delta_{x_k^p}(x_k) \tag{2.6}$$

where $p \in \{1, \ldots, P\}$ denotes individual particles, or atoms of the discrete PMF, and $\delta_{x_k^p}(x_k)$ denotes a Dirac delta that places a unit mass on the point x_k^p (we use the subscript as a notational shorthand for the usual notation, $\delta_{x_k^p}(x_k) \triangleq \delta(x_k - x_k^p)$, where x_k denotes the argument to the "function" $\delta(\cdot)$ and x_k^p is the offset that moves the unit mass from $x_k = 0$).

Reviewing the two items in the summand of (2.6), we see that individual particles have an atom of probability mass placed in the state space of the system, x_k^p (where the superscript p denotes the pth particle), and an associated probability $p_\theta(x_k^p|y_{0:k})$. Summing up these particles results in a PMF with P discrete points, each with an associated probability.

Much like in the theoretical, closed-form version of recursive filtering (2.3)-(2.5), the particle filter proceeds in an iterative predict-then-update manner. As before, to estimate the system state at timestep k, we assume that we start with an approximate PDF from the previous timestep, $\hat{p}_\theta(x_{k-1}|y_{0:k-1})$ (note the hat indicating it is an approximation). This approximation has P individual particles. We can obtain a particle filter estimate of the prior PDF, $\hat{p}_\theta(x_k|y_{0:k-1})$, by plugging each particle's state value x_{k-1}^p into the stochastic system equation $\mathcal{F}_\theta(x_{k-1})$ (2.1) (Gordon et al., 1993)

$$x_k^p = \mathcal{F}_\theta(x_{k-1}^p)$$

where the randomness of $\mathcal{F}_\theta(\cdot)$ means

$$\mathcal{F}_\theta(x_{k-1}^p) \sim f_\theta(x_k|x_{k-1}^p).$$

Then, a particle filter approximation for the prior PDF is

$$\begin{aligned} p_\theta(x_k|y_{0:k-1}) &= \int f_\theta(x_k|x_{k-1})p_\theta(x_{k-1}|y_{0:k-1})dx_{k-1} \\ &\approx \sum_{p=1}^{P} p_\theta(x_{k-1}^p|y_{0:k-1})\delta_{\mathcal{F}_\theta(x_{k-1}^p)}(x_k) \end{aligned}$$

$$= \sum_{p=1}^{P} p_\theta(x_k^p | y_{0:k-1}) \delta_{x_k^p}(x_k) \tag{2.7}$$
$$= \hat{p}_\theta(x_k | y_{0:k-1})$$

and the particle filter approximation for the posterior PDF is found by plugging (2.7) into (2.4),

$$\begin{aligned} p_\theta(x_k | y_{0:k}) &= \frac{p_\theta(x_k | y_{0:k-1}) g_\theta(y_k | x_k)}{p_\theta(y_k | y_{0:k-1})} \\ &\approx \frac{\hat{p}_\theta(x_k | y_{0:k-1}) g_\theta(y_k | x_k)}{p_\theta(y_k | y_{0:k-1})} \\ &= \frac{\sum_{p=1}^{P} p_\theta(x_k^p | y_{0:k-1}) \delta_{x_k^p}(x_k) g_\theta(y_k | x_k^p)}{p_\theta(y_k | y_{0:k-1})} \qquad (2.8a) \\ &= \frac{\sum_{p=1}^{P} p_\theta(x_k^p, y_k | y_{0:k-1}) \delta_{x_k^p}(x_k)}{p_\theta(y_k | y_{0:k-1})} \qquad (2.8b) \\ &= \hat{p}_\theta(x_k | y_{0:k}). \end{aligned}$$

This posterior approximate PDF is thus made up of the same collection of Dirac deltas as the prior approximate PDF, $\hat{p}_\theta(x_k | y_{0:k-1})$, but with updated weights to reflect each point's posterior probability, after assimilating the measurement y_k through the likelihood.

As has been mentioned, the use of the particle filter avoids having to explicitly calculate difficult integrals. Of particular relevance is the calculation of the marginal likelihood $p_\theta(y_k | y_{0:k-1})$. Instead of using (2.5), we can compute the marginal likelihood $p_\theta(y_k | y_{0:k-1})$ via a marginalization of the PMF $p_\theta(x_k^p, y_k | y_{0:k-1})$,

$$p_\theta(y_k | y_{0:k-1}) \approx \sum_{p=1}^{P} p_\theta(x_k^p, y_k | y_{0:k-1}). \tag{2.9}$$

In implementations of a particle filter, (2.8b)-(2.9) make up the filtering step that is used in practice.

As an important side note, we have omitted discussion of the particle filter's post-update resampling step because it is not immediately relevant to the work presented in the remainder of this thesis. See, e.g., Doucet and Johansen (2011), for details.

2.2 Macroscopic traffic modeling basics

Several later chapters will study a stochastic system (as defined in section 2.1) that represents a road network at a macroscopic scale. In this section, we give some background definitions that apply to all subsequent discussions. This section's material is preliminary to chapters 4, 5, 6, 7, and to a lesser extent 3.

2.2.1 Partial Differential Equation (PDE) models of traffic

As mentioned in Chapter 1, a complex system like the road network exhibits observable phenomena at multiple scales of resolution. For the road network, these scales are often defined by the level of aggregation applied to vehicles: models that consider each vehicle as a discrete unit are usually called "microscopic"[2] and models that lump together all the vehicles in a spatial region are called "macroscopic."

A very popular class of model that we use extensively is the so-called first-order "kinematic wave" macroscopic model. These are usually traced back to Lighthill and Whitham (1955a,b) and Richards (1956) and are also called "Lighthill-Whitham-Richards" (LWR) types of models. An LWR model describes aggregate traffic flows as fluids following a one-dimensional conservation law,

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho v)}{\partial x} = 0 \tag{2.10}$$

where $\rho(x,t)$ is the density of vehicles, t is time, x is the lineal direction along the road, and $v(\rho)$ is the flow speed.

2.2.2 Traffic flux functions

The total flow in (2.10), ρv, is often expressed in terms of a flux function, $f(\rho) = \rho v$ (the flux function on a long straight road is often called the *fundamental diagram*). The earliest flux function was estimated by Greenshields et al. (1935), who proposed a linear relationship between density and speed, with speed decreasing linearly as density increases, up to a maximum density value where the speed becomes zero. This extremal point in the $\rho - v$ (or, equivalently, $\rho - f$) plane represents deadlocked or jammed traffic.

Another fundamental diagram used in many analyses is the so-called *triangular* fundamental diagram, also called the Newell-Daganzo fundamental diagram after Newell (1993) and Daganzo (1994) shown in Figure 2.1. The explicit formula for this fundamental diagram is

$$f = \min\left(v_f \cdot \rho,\ w \cdot (\rho_j - \rho)\right) \tag{2.11}$$

where ρ is the density, f is the flow, v_f is the freeflow speed (the speed at which unobstructed cars will travel), w is the congestion wave speed (the maximum speed at which shockwaves move backward along the road), and ρ_j is the jam density (the maximum possible density at which point the flow becomes zero).

We will use this fundamental diagram in many of our simulations in this thesis.

[2]Sometimes, models that break down the vehicle unit even further (e.g., modeling the driver and vehicle separately, or individual vehicle components separately) are sometimes called microscopic and sometimes called "sub-microscopic" traffic models (Hoogendoorn and Bovy, 2001; Treiber and Kesting, 2013)

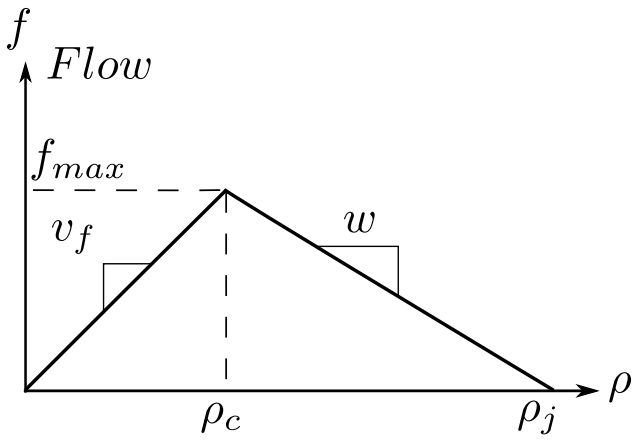


Figure 2.1: Schematic illustration of the Newell-Daganzo triangular fundamental diagram. The horizontal axis ρ is vehicle density (typically veh/m) and the vertical axis f is vehicle flow (typically veh/s).

2.2.3 Finite-volume discretization

The LWR PDE model of traffic (2.10) is defined in continuous time t and continuous space x. For simulation purposes, it is required to discretize its domain. The most common discretization scheme for the LWR model is Godunov's discretization scheme (Godunov, 1959). The resulting discretized traffic model is also called the Cell Transmission Model (CTM) due to Daganzo (1994, 1995). [3]

In the CTM discretization, a long, straight road is divided into small spatial cells (also called links) with uniform density. The flow between two subsequent links ℓ and $\ell + 1$ on a flow is computed as

$$f_\ell = \min\left(S_\ell\left(\rho_\ell\right), R_{\ell+1}\left(\rho_{\ell+1}\right)\right) \tag{2.12}$$

where ρ_ℓ (resp. $\rho_{\ell+1}$) is the density in link ℓ (resp. $\ell + 1$), $S_\ell(\cdot)$ is the "sending function" (also called the demand) for link ℓ, and $R_{\ell+1}(\cdot)$ is the "receiving function" (also called the supply) for link $\ell + 1$. The supply and demand functions $S(\cdot)$ and $R(\cdot)$ are obtained from the continuous flux function $f(\rho)$. For the Newell-Daganzo flux function, they are

$$S_\ell(\rho) = \min\left(F_\ell, v_{f,\ell}, \rho_\ell\right) \tag{2.13a}$$

$$R_\ell(\rho) = \min\left(F_\ell, w_\ell \cdot \left(\rho_{j,\ell} - \rho_\ell\right)\right) \tag{2.13b}$$

where F_ℓ is the maximum possible flow of link ℓ, also called the link's capacity, and $v_{f,\ell}$ (resp. w_ℓ and $\rho_{j,\ell}$) is the value of the parameter v_f (resp. w and ρ_j) from (2.11) in link ℓ.

Note that if the units of density ρ are vehicles per meter (or vehicle per meter per lane) and the units of flow f are vehicles, when computing inter-link flows between two cells one needs to standardize the outputs of the flux function supply and demand functions so that

[3] Historically, Daganzo (1994) proposed the CTM discretization independent of the Godunov scheme, and they were shown to be equivalent by Lebacque (1996).

they are in units of whole vehicles that are transmitted between cells. For example, if a density ρ_ℓ is given in units of vehicles per meter,[4] then the total number of vehicles in a given link ℓ is $\rho_\ell \cdot L_\ell$, where L_ℓ is the length of link ℓ.

So far, we have just discussed computation of flows between two consecutive cells. The problem of finding inter-link flows at junctions where multiple roads meet (as opposed to the setting here, where link ℓ and link $\ell + 1$ are subsequent volumes of the same road) is the topic of much discussion in chapter 3 of this thesis.

[4]This abuse of terminology, where a "density" is defined per unit length rather than per unit volume, is typical in the traffic literature.

Part I

Nonlinear Macroscopic Traffic Models: Dynamical, Topological, and Phenomological

Modeling the Macro Scale *ab initio*

Chapter 3

Generic second-order macroscopic traffic node model for general multi-input multi-output road junctions via a dynamic system approach

3.1 Introduction

In section 2.2, we introduced partial differential equation (PDE) models of traffic, such as the "kinematic wave" or "Lighthill-Whitham-Richards" model. For many traffic studies, the LWR-type formulation (2.10) can be used to model a number of traffic dynamics of interest. For example, in dynamic traffic assignment studies, modeled vehicle demands are fed into a road network model, and a first-order dynamical simulation of the resulting network flows can estimate – among other items of interest that result from the nonlinear dynamics – where traffic jams will appear and how long they will last, and quantify the general efficiency of the road network as a function of the input demands and the network geometry and topology.

However, there are some macroscopic traffic phenomena that an LWR-type formulation cannot capture. Three particular examples that have been discussed in the literature are hysteresis loops in the $(\rho - v)$ plane (e.g., Treiterer and Myers (1974); Zhang (1999)), drops in capacity when roads become congested (Hall and Hall, 1990), and the occasional emergence of congestion behavior in regions of low density and no bottlenecks (that is, in conditions where we should expect free-flow behavior) (Sugiyama et al., 2008). These phenomena cannot be captured in solutions to the LWR-type model, so, in situations where these smaller-scale (that is, relative to "larger-scale" phenomena like bottlenecks) macroscopic phenomena are of importance to the modeler, more expressive models that can produce them are appropriate. See, e.g., Wong and Wong (2002); Zhang (2002) for some examples where the LWR-type

models' lack of expressiveness are used as justifications for extensions.

One extension of the LWR model that can express a richer variety of dynamics is the so-called Aw-Rascle-Zhang (ARZ) (Aw and Rascle, 2000; Zhang, 2002) family of models. These models fit into the so-called "generic second order"[1] or "extended ARZ" class of traffic models (Lebacque et al., 2007b), which can be written as

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho v)}{\partial x} = 0 \tag{3.1a}$$

$$\frac{\partial w}{\partial t} + v \frac{\partial w}{\partial x} = 0 \tag{3.1b}$$

$$\text{where } v = V(\rho, w) \tag{3.1c}$$

where $w(x,t)$ is a property or invariant that is conserved along trajectories (Lebacque et al., 2007b). In words, (3.1a) is a continuity equation of ρ, (3.1b) is an advection equation of w, and (3.1c) defines the velocity field that governs both the flux of ρ and the speed at which w moves through space.

The property w in (3.1) can be described as a characteristic of vehicles that determines their density-velocity relationship. Members of the generic second order model (GSOM) family are differentiated by the choice of w and its relationship on the ρ-v behavior. Examples of properties modeled by chosen w's include the difference between vehicles' speed and an equilibrium speed (Aw and Rascle, 2000; Zhang, 2002), driver's desired spacing (Zhang, 2002), or the flow's portion of autonomous vehicles (Wang et al., 2017).[2] An intuitive way of describing the effect of the property w in (3.1) is that it parameterizes a family of flow models, $f(\rho, w) = \rho V(\rho, w)$, with different flow models for different values of w (Lebacque et al., 2007b; Fan et al., 2017). In particular, different classes of vehicles (e.g., autonomous vs. human-driven) are assumed to have different equations governing their driving behavior: these varying dynamics are aggregated to create an averaged dynamics in the macroscopic flow equation (3.1a). The aggregate dynamical behavior (parameterized by w) of course then tracks the vehicles themselves, as reflected in (3.1).

The models covered thus far describe traffic along single roads. For application to multiple roads with intersections, these road networks are often modeled as directed graphs. Edges that represent individual roads are called links, and junctions where links meet are called nodes. Typically, the flow model $f(\cdot)$ on links is called the "link model," and the flow model at nodes is called the "node model." Development of accurate link and node models have been areas of much research activity in transportation engineering for many years.

[1] As seen in (3.1) and previously pointed out by many authors, the "second-order" model actually consists of two first-order partial differential equations (that is, they only contain first derivatives). In a case of overloaded mathematical terminology, the name "second order" here comes from a system-theoretic view, where a second-order system is one that has two state variables: in this case, ρ and w (or, equivalently, ρ and v).

[2] The use of autonomous vehicles for control has been proposed and argued as very effective by, e.g., Ge and Orosz (2014); Cui et al. (2017); Wu et al. (2017b); Stern et al. (2018), but only on simple road networks like a single road. Enabling model-based control involving a *controllable* property w for more complex networks is a major motivation of this work.

This chapter focuses on node models for first- and second-order macroscopic models. The node model resolves the discontinuities in ρ and/or w between links and determines a flux boundary condition at each link's exit (for incoming links) or entrance (for outgoing links). In other words, the node model takes as input the Dirichlet boundary condition for each link at the junction, and outputs a resulting Neumann boundary condition. For nodes with merges, diverges, or both, this Riemann problem becomes multidimensional. Through this, the node model determines how the state of an individual link affects and is affected by its connected links, their own connected links, and so on through the network. As a result, it has recently been recognized that the specific node model used can have a very large role in describing the network-scale congestion dynamics that emerge in complex and large networks (for more on this, see the discussions in, e.g., the introduction sections of Tampère et al. (2011) and Jabari (2016)).

In Wright et al. (2016), we introduced a novel characterization of node models as dynamic systems. Traditional studies of node models (see, e.g., Tampère et al. (2011); Flötteröd and Rohde (2011); Corthout et al. (2012); Smits et al. (2015); Jabari (2016); Wright et al. (2017)) usually present the node model as an optimization problem (where the node flows are found by solving this problem) or in algorithmic form (where an explicit set of steps are performed to compute the flows across the node). In contrast, the dynamic system characterization describes the flows across the node as themselves evolving over some period of time (in application, this means that the dynamic system characterization presents time-varying dynamics that are said to occur during the simulation timesteps of the link PDEs). The dynamic system characterization can be thought of as making explicit the time-varying behavior of the flows at nodes of many algorithmic node models: it was shown in Wright et al. (2016) that the dynamic system characterization produces the same solutions as the algorithm introduced in Wright et al. (2017), which also reduces to the one introduced in Tampère et al. (2011) as a special case.

The dynamic system characterization has proven useful in imparting an intuition as to what physical processes over time are implicit in these algorithmic node models (see the discussions referring to Wright et al. (2016) in Wright et al. (2017) for some examples). In this chapter, we develop a dynamic system characterization of a second-order node model, and use it to solve the general node problem for second-order models.

This chapter has several main contributions. The first is an extension of the dynamic system characterization of first-order node models as introduced in Wright et al. (2016) to a simple, closed-form solution algorithm. This represents the completion of an argument began in Section 4.1 of that reference. The second contribution is the extension of the dynamic system characterization to the generic second-order models. As we will see, the dynamic system characterization lends itself to an intuitive incorporation of the second PDE in (3.1) that is not obvious in the traditional, optimization-problem presentation of node models. The third contribution, and the principal contribution of this chapter, parallels the first by using the second-order dynamic system node model to derive an intuitive, closed-form algorithm for computing node flows for second-order flow models for general, multi-input multi-output nodes. To the best of our knowledge, this represents the first proposed generic (applicable to

multi-input multi-output nodes with arbitrary numbers of input and output links) node flow solver for second-order traffic flow modeling.[3] Finally, our fourth contribution is an argument that the second-order node problem is not well-suited to the pervasive "supply and demand" CTM-like discretization that is highly prevalent in macroscopic modeling (see Section 3.4.3).

The remainder of this chapter is organized as follows. Section 3.2 reviews the first-order node flow problem, the first-order dynamic system characterization introduced in Wright et al. (2016), and presents the aforementioned closed-form solution algorithm (contribution one in the above paragraph). Section 3.3 reviews the link discretization of the GSOM (3.1) as presented in, e.g., Lebacque et al. (2007b); Fan et al. (2017), which produces the inputs to our second-order node model, and the standard one-input one-output second-order flow problem and its solution. Section 3.4 presents the extension of the second-order flow problem to the multi-input multi-output case, the dynamic system characterization to the GSOM family (3.1) and the solution algorithm for the general node problem (contributions two and three). Finally, Section 3.5 concludes and notes some open problems.

3.2 First-order node model

In this section, we review the general first-order node problem and a particular node model (and its solution algorithm). This node model will be extended to the second-order node problem in section 3.3.

The traffic node problem is defined on a junction of M input links, indexed by i, and N output links, indexed by j. We further define C "commodities" of vehicle, indexed by c. Each commodity c denotes a different kind of vehicle (e.g., cars, trucks, autonomous vehicles, etc.). [4] The first-order node problem takes as inputs the incoming links' per-commodity demands S_i^c, split ratios $\beta_{i,j}^c$ (which define the portion of vehicles of commodity c in link i that wish to exit to link j), and outgoing links' supplies R_j, and gives as outputs the set of flows from i to j for commodity c, $f_{i,j}^c$. We denote as a shorthand the per-commodity directed demand $S_{i,j}^c \triangleq \beta_{i,j}^c S_i^c$. Nodes are generally infinitesimally small and have no storage, so all the flow that enters the node must exit the node.

The rest of this section is organized as follows. Section 3.2.1 defines our first-order node problem as an optimization problem defined by explicit requirements, following the example set by Tampère et al. (2011). Section 3.2.3 reviews the dynamic system of Wright et al. (2016) whose executions produce solutions to the node problem. Finally, section 3.2.4 uses

[3] A note on naming: as we will see in Section 3.2.1, we build off the so-called "generic class of first-order node models" to develop our second-order node model. Given that the relevant second-order model used (3.1) is itself called the "generic second order model," it might be accurate to describe this chapter's results as the "genericization of the generic class of node models to the generic second-order model," but this description likely loses in comprehensibility what it might gain in accuracy.

[4] Commodities are sometimes called vehicle classes. In this work, we purposefully use the term "commodity" when referring to different types of vehicles to differentiate from the usage of the term "class" with respect to node models.

the dynamic system formulation as a base to develop a node model solution algorithm. This algorithm represents the completion of an argument began in Wright et al. (2016).

3.2.1 "Generic Class of Node Model" requirements

The node problem's history begins with the original formulation of macroscopic discretized first-order traffic flow models Daganzo (1995). There have been many developments in the node model theory since, but we reflect only some more recent results.

We can divide the node model literature into pre- and post-Tampère et al. (2011) epochs. They drew from the literature several earlier-proposed node model requirements to develop a set of conditions for first-order nodel models they call the "generic class of first-order node models" (GCNM). These set of conditions give an excellent starting point for our discussion of the mathematical technicalities of node models, and have been used as a starting point by many subsequent papers, such as Flötteröd and Rohde (2011); Corthout et al. (2012); Smits et al. (2015); Jabari (2016); Wright et al. (2017). In the following list, we present the variant of first-order GCNM requirements used in Wright et al. (2017), which includes a modification of the first-in-first-out (FIFO) requirement (item 6 below) to Wright et al. (2017)'s "partial FIFO" requirement.

1. Applicability to general numbers of input links M and output links N. In the case of multi-commodity flow, this also extends to general numbers of commodities c.

2. Maximization of the total flow through the node. Mathematically, this may be expressed as $\max \sum_{i,j,c} f^c_{i,j}$. According to Tampère et al. (2011), this means that "each flow should be actively restricted by one of the constraints, otherwise it would increase until it hits some constraint." When a node model is formulated as a constrained optimization problem, its solution will automatically satisfy this requirement. However, what this requirement really means is that constraints should be stated *correctly* and not be overly simplified (and thus, overly restrictive) for the sake of convenient problem formulation. See the literature review in Tampère et al. (2011) for examples of node models that inadvertently do not maximize node throughput by oversimplifying their requirements.

3. Non-negativity of all input-output flows. Mathematically, $f^c_{i,j} \geq 0$ for all i, j, c.

4. Flow conservation: Total flow entering the node must be equal to total flow exiting the node. Mathematically, $\sum_i f^c_{i,j} = \sum_j f^c_{i,j}$ for all c.

5. Satisfaction of demand and supply constraints. Mathematically, $\sum_j f^c_{i,j} \leq S^c_i$ and $\sum_i f^c_{i,j} \leq R_j$.

6. Satisfaction of the (partial) first-in-first-out (FIFO) constraint: if a single destination j' for a given i is not able to accept all demand from i to j', then all other flows from i are constrained by the queue of j'-destined vehicles that builds up in i.

We will spend some time outlining the specifics of the partial FIFO constraint and developing an intuition for it in section 3.2.1.1.

7. Satisfaction of the invariance principle. This principle was introduced by Lebacque and Khoshyaran (2005) and specifies that "under constant demand and supply constraints, flows should be invariant during an infinitesimal timestep." (Tampère et al., 2011). In particular, this requirement means that the portioning of a supply R_j among the input links i not be proportional to the demands S_i.

8. Supply restrictions on a flow from any given input link are imposed on commodity components of this flow proportionally to their per-commodity demands. Mathematically, $f^c_{i,j}/(\sum_c f^c_{i,j}) = \beta^c_{i,j} S^c_i/(\sum_c \beta^c_{i,j} S^c_i)$.

   This assumes that the commodities are mixed isotropically. This means that all vehicles attempting to take movement i, j will be queued in roughly random order, and not, for example, having all vehicles of commodity $c = 1$ queued in front of all vehicles of $c = 2$, in which case the $c = 2$ vehicles would be disproportionally affected by spillback. We feel this is a reasonable assumption for situations where the demand at the node is dependent mainly on the vehicles near the end of the link (e.g., in a small cell at the end).

9. A "supply constraint interaction rule" (SCIR), the idea of which was explicitly defined by Tampère et al. (2011). We discuss this in more detail in section 3.2.1.2.

3.2.1.1 Details of the Partial FIFO constraint

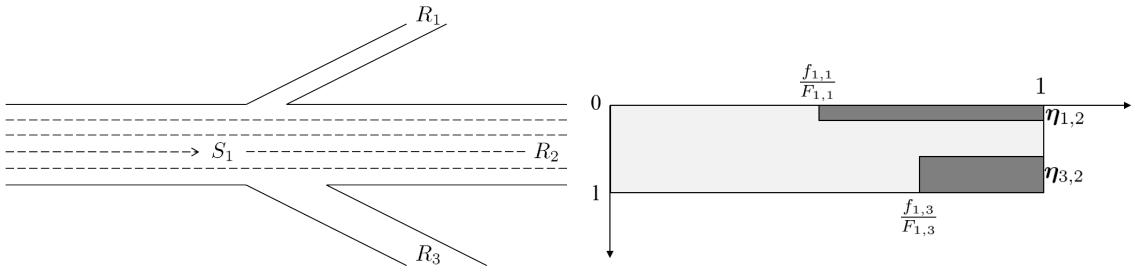
In this chapter, we use the "partial" FIFO constraint, item 6 above, which was introduced in Wright et al. (2017). A classic (non-partial) FIFO constraint (also referred to as a "conservation of turning fractions" constraint by Tampère et al. (2011)) has the mathematical form $\beta^c_{i,j} = f^c_{i,j}/(\sum_j f^c_{i,j})$, and means that when some vehicles that want to go to a particular j are unable to take this movement, they queue and block all vehicles behind them, regardless of whether the blocked vehicles want to go to that particular j or another one.

In Wright et al. (2017), we argued that this full FIFO requirement can be unrealistic if like i has multiple lanes, since it necessarily implies that all lanes are blocked, by vehicles queueing for output j even if the movement (i, j) is only accessible by a subset of the lanes (supposing of course that vehicles queueing for j will not be queueing on lanes that they cannot use to reach j). The partial FIFO requirement encodes how vehicles queueing for a movement (i, j') will only partially block another movement (i, j), with the degree of blockage being related to the degree that (i, j)'s lanes are blocked by this queue. In our model, we will say that one of these partial blockages affects the blocked flow by reducing the movement's capacity $F_{i,j}$, where the nominal (un-reduced) capacity is defined as the maximum possible

flow for the particular movement.[5] Following, e.g., (Tampère et al., 2011), we will define the movement capacity $F_{i,j}$ as a portion of the input link capacity F_i weighted by the vehicles that are making use of that capacity,

$$F_{i,j} = \frac{\sum_c S^c_{i,j}}{\sum_c S^c_i} F_i. \tag{3.2}$$

To move to specifics, suppose a particular link j' is not able to receive all the vehicles from i that want to enter it, and the denied vehicles queue up. The degree to which this queue restricts the other flows $f^c_{i,j}$ is partially defined by *restriction intervals* $\boldsymbol{\eta}^i_{j',j} = [y, z] \subseteq [0, 1]$. This interval means that a queue in the i, j' movement will block the portion of i, j-serving lanes in i with leftmost extent y and rightmost extent z (e.g., if i, j is a through movement that uses two lanes and i, j' is a right-turn movement that uses the right of those two lanes, then $\boldsymbol{\eta}^i_{j',j} = [1/2, 1]$.[6] The traditional, full FIFO behavior, where any queue in i blocks all of i's lanes, can be recovered by setting all $\boldsymbol{\eta}^i_{j',j} = [0, 1]$. [7]



(a) (b)

Figure 3.1: Example application of partial FIFO. (a): A one-input, three-output road junction. (b): Pictorial representation of restriction intervals on $f_{1,2}$: this flow is upper-bounded by the nominal capacity of movement $(1, 2)$ multiplied by the lightly-shaded portion of the rectangle. The dark-shaded portions represent the capacity that is blocked by queueing vehicles in the partial FIFO construction.

Figure 3.1 illustrates the application of partial FIFO in a slightly more complicated example. This example was previously discussed in Wright et al. (2016, 2017). In Figure 3.1(a), a five-lane road diverges into three output links. For the sake of this example, we say that only the leftmost lane can access movement $(1, 1)$, all five lanes can access movement $(1, 2)$, and only the two rightmost lanes can access movement $(1, 3)$. So, the restriction

[5]The constraint $\sum_c f^c_{i,j} \leq F_{i,j}$, that the flow be no more than the maximum possible flow (before any capacity reductions due to blockages) is generally not given in node model problem statements because it is implied by demand constraint, since by definition $\sum_c \beta^c_{i,j} S^c_i \leq F_{i,j}$.

[6] Continuing this example, we will have $\boldsymbol{\eta}^i_{j,j'} = [0, 1]$ since the only lane in i that serves movement i, j' (the right lane) will be blocked by a queue for the through movement, which will queue on both lanes).

[7]To help keep the meaning of $\boldsymbol{\eta}^i_{j',j}$ clear, we find it helpful to read it as "the restriction interval of j' onto j for i"

intervals for this example, for the restrictions onto $(1,2)$'s flow, are $\boldsymbol{\eta}^1_{1,2} = [0, 1/5]$, $\boldsymbol{\eta}^1_{2,2} = [0,1]$, and $\boldsymbol{\eta}^1_{3,2} = [3/5, 1]$.[8]

Figure 3.1(b) shows a graphical computation of the effects of restriction intervals on movement $(1,2)$'s capacity, given particular flows for the other movements $f_{1,1}$ and $f_{1,3}$. In this particular example, we are supposing that both $f_{1,1}$ and $f_{1,3}$ are supply-constrained (i.e., $f_{1,1} = R_1$ and $f_{1,3} = R_3$). The dark-shaded portions of the figure show the effects of the vehicles queueing for these two movements on the $(1,2)$ movement's capacity. The extent of the dark-shaded regions on the vertical axis are exactly the restriction intervals $\boldsymbol{\eta}^1_{j',2}$ for $j' \in \{1,3\}$.

We now explain the meaning of the horizontal-axis extent of the dark-shaded regions. By definition, the capacity is the maximum possible amount of vehicles that can pass through a movement in some time period. For some movement i,j with capacity $F_{i,j}$, suppose we have a supply-limited downstream j, and therefore an actual flow $f_{i,j} < Fi,j$. It was proposed in section 5.1.1 of Tampère et al. (2011) to behaviorally model the drivers taking movement i,j to attempt the movement as quickly as they can, i.e., selfishly competing against drivers from other inbound links for the open space in j. This maximum speed that they can claim supply is already defined as the capacity $F_{i,j}$. So, if $F_{i,j}$ specifies that N vehicles can take the movement in T seconds, and $f_{i,j}$ is the amount of vehicles that were able to take the movement over a particular T-second time period,[9] then $f_{i,j}/F_{i,j}$ is the portion of T at which vehicles took the movement i,j (at rate $F_{i,j}/T$) before all supply in j was filled and they were forced to queue. Therefore, one minus this ratio, $(1 - f_{i,j}/F_{i,j})$, is then the portion of T that the movement (i,j) is blocked by a queue. So, returning to Figure 3.1(b)'s example, the restriction intervals reduce the capacity of the $(1,2)$, proportional both to the intervals' length and to the "portion of time" that their queue exists.

Putting together the pieces, the (partial) FIFO requirement can be stated mathematically as

$$f^c_{i,j} \leq F_{i,j} \frac{S^c_{i,j}}{\sum_c S^c_{i,j}} \left(1 - \mathcal{A}\left(\bigcup_{j' \neq j} \left\{ \boldsymbol{\eta}^i_{j',j} \times \left[\frac{f_{i,j'}}{F_{i,j'}}, 1\right] \right\}\right)\right) \tag{3.3}$$

where $\mathcal{A}(\cdot)$ denotes the area of a two-dimensional object, $\times$ denotes a Cartesian product, $F_{i,j}$ is given by (3.2), and $f_{i,j} \triangleq \sum_c f^c_{i,j}$. Note that we take the union of the two-dimensional objects that define the extent of the partial FIFO restriction (the dark-shaded regions in Figure 3.1(b)) to obtain the effective partial FIFO restriction of *all* blocking queues.

The formulation in (3.3) is complicated in order to state it as an optimization constraint. The time-period intuition developed above will be used explicitly in the dynamic system

[8] To further explain the restriction interval concept via example, these three intervals mean, in order, that when movement $(1,1)$ (the left turn) is blocked by a queue, then the leftmost $1/5$ of the lanes serving $(1,2)$ (the through movement) is blocked by that queue; when the through movement $(1,2)$ is by a queue, all of its lanes must be blocked by that queue (that is, we need that $\boldsymbol{\eta}^i_{j,j} = [0,1]$ by definition); and that when the right turn $(1,3)$ is blocked by a queue, the rightmost $2/5$ of lanes serving the trough movement $(1,2)$ are also blocked by that queue.

[9] Recall the units of capacity F are $^{\text{veh}}/_{\text{time}}$ and the units of flow f are veh.

definition of the node models. A major contribution of the dynamic system approach to node modeling is the explicit encoding of this more intuitive description.

Remark 3.1. In prior work on the partial FIFO constraint (Wright et al., 2016, 2017), our construction was slightly different than as stated in (3.3) above. In particular, the quantity that was brought down as restriction intervals activate in the prior version was the *demand* rather than the capacity. As we discuss in Wright and Horowitz (2019b), this prior version leads to unintentional unrealistic constraints on the flow that are rectified by tightening the capacity constraint instead of the demand constraint.

3.2.1.2 The "Supply Constraint Interaction Rule"

The term "supply constraint interaction rule" (SCIR), introduced by Tampère et al. (2011), refers to two other elements needed to define a node model. The first element is a rule for the portioning of output link supplies R_j among the input links. Following Gentile et al. (2007), in Tampère et al. (2011) it was proposed to allocate supply for incoming flows proportionally to input link capacities F_i. In this chapter, we allocate supply proportionally to the links' "priorities" p_i (in the spirit of Daganzo (1995); Ni and Leonard (2005); Flötteröd and Rohde (2011); Wright et al. (2017)). In the dynamic system view, priorities represent the relative rate at which vehicles exit each link i to claim downstream space (one reasonable formulation might be to follow the capacity-proportional example, $p_i = F_i$, if, as in Tampère et al. (2011), it assumed that vehicles exit a link at rate F_i). In this chapter, we assume for simplicity that all $p_i > 0$.[10]

The second necessary element is a redistribution of "leftover supply." Following the initial partitioning of supplies R_j, if one or more of the supply-receiving input links does not fill its allocated supply, some rule must redistribute the difference to other input links who may still fill it. This second element is meant to model the selfish behavior of drivers to take any space available, and ties in closely with requirement 2 above. Tampère et al. (2011) referred to these two elements collectively as a "supply constraint interaction rule" (SCIR). For some discussion of choices of SCIRs in recent papers, see Wright et al. (2017, Section 2.1), and for more background discussion of the need for an SCIR, see Tampère et al. (2011, Sections 2 and 4).

In this chapter, we consider a SCIR of the form (equation (3.37) in Wright et al. (2017),

[10] In a more general formulation, a priority of $p_i = 0$ makes sense in a "staged" scheme, where a zero-priority link or movement only gets to send its vehicles after other input links have exhausted their demand. As mentioned in (Wright et al., 2017, Section 4.2), this is equivalent to a common assumption in freeway onramp junction modeling (particularly models for control schemes like ALINEA (Papageorgiou et al., 1991)) the onramp gets to fill all its demand, and the freeway gets the remainder. This staging of priorities is also related to Jabari (2016)'s suggestion of modeling signalized junctions such that protected movements get to claim all the supply they can use before non-protected movements. One can fit these staged-priority problems into the node problem scheme presented here by breaking them into sub-problems that are solved sequentially, with different sets of links having nonzero priority for each sub-problem.

but rearranged slightly)

$$\sum_j f_{i,j} < \sum_c S_i^c \implies W_i \neq \emptyset \quad \forall\, i \tag{3.4a}$$

$$f_{i,j} \geq \frac{p_{i,j}}{\sum_{i'=1}^{M} p_{i',j}} R_j \quad \forall\, j \in W_i \quad \forall i \tag{3.4b}$$

where

$$W_i = \left\{ j^* : \sum_c \beta_{i,j^*}^c S_i^c > 0 \text{ and } \nexists\, i' \neq i \text{ s.t. } \frac{f_{i,j^*}}{p_{i,j^*}} < \frac{f_{i',j^*}}{p_{i',j^*}} \right\} \tag{3.4c}$$

and

$$p_{i,j} = \frac{\sum_c S_{i,j}^c}{\sum_c S_i^c} p_i \tag{3.5}$$

is the "oriented priority," which distributes input links' priority proportionally to the actual vehicles using that priority to claim downstream supply, and $f_{i,j} \triangleq \sum_c f_{i,j}^c$ (as before). In the case of capacity-equivalent priorities $p_i = F_i$, the oriented priority (3.5) is of course the same as the movement capacity (3.2).

The set W_i denotes all output links that restrict the flow from i.[11] The first condition for membership in W_i ($\sum_c \beta_{i,j^*}^c S_i^c > 0$) can be read as "there is some nonzero demand for the movement i, j^*."

The second condition ($\nexists\, i' \neq i$ s.t. $f_{i,j^*}/p_{i,j^*} < f_{i',j^*}/p_{i',j^*}$) communicates that there is not *some other* input link i' that is able to send more (priority-normalized) flow to j^*: that is, that j^* is either a) restricting to both i and i' (in which case we will have $f_{i,j^*}/p_{i,j^*} = f_{i',j^*}/p_{i',j^*}$ and $j^* \in W_i \cap W_{i'}$), or b) that i''s flow to j^* was demand-constrained. If the opposite was true (that is, that i' was able to send more (priority-normalized) flow to j^* than i), then j^* was not restricting after all (i.e., it was not the link that ran out of supply) and some other output link is what restricted the flow from i. More discussion of a physical meaning for the second condition in (3.4c) will be given later, in Remark 3.3.

Constraint (3.4a) says that if a link i is not able to fill its demand, then there is at least one output link in W_i that restricts i, and that i's movements claim at least as much as their oriented-priority-proportional allocation of supply. Constraint (3.4b) captures the reallocation of "leftover" supply, which states that a link i that cannot fulfill all of its demand to the links in W_i will continue to send vehicles after links $i' : j \notin W_{i'}$ have fulfilled their demands to the $j \in W_i$.

Remark 3.2. In (3.5), we distribute the link priority among the movements proportionate to the demand for each movement. Consider the case where the link priority is chosen to be the link capacity, $p_i = F_i$, as we have mentioned as an example, and as suggested by Tampère et al. (2011) (they argued the capacity makes sense as a priority value because vehicles

[11] We defined the set W_i that consisted of the output links restricting i slightly differently in Wright et al. (2017, Definition 3.2) Specifically, there we wrote the second condition as $\exists\, i' \neq i$ s.t. $p_{i',j^*} f_{i,j^*} \geq p_{i,j^*} f_{i',j^*}$. We have inverted the conditional in the present definition so that the definition is still valid for $M = 1$.

leaving i in a discharging queue will be claiming downstream supply as fast as possible, i.e., at the link's capacity). If the capacity is chosen based on a multiple of i's number of lanes, note that this means that, in (3.5), each movement will theoretically have available priority proportional to all lanes rather than only the lanes the movement can actually use. This will not be too unrealistic, however, if the demands $S_{i.j}$ are proportional to the number of lanes. A more refined SCIR where the oriented priorities $p_{i,j}$ are assigned in a manner aware of both the relative demands between the movements (as in (3.5)), and the spatial extent of the lane facilities the movements have available (similar to the partial FIFO construction), is an avenue for future work.

3.2.1.3 Our first-order node model optimization problem

Putting together the pieces, we have

Definition 3.1 (First-order node model problem)**.**

$$\max \left(\sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{c=1}^{C} f_{i,j}^{c} \right) \tag{3.6a}$$

subject to:

$$f_{i,j}^{c} \geq 0 \qquad \forall i, j, c \tag{3.6b}$$

$$f_{i,j}^{c} \leq S_{i,j}^{c} \qquad \forall i, j, c \tag{3.6c}$$

$$\sum_{i} \sum_{c} f_{i,j}^{c} \leq R_j, \qquad \forall j \tag{3.6d}$$

$$\sum_{i} f_{i,j}^{c} = \sum_{j} f_{i,j}^{c} \qquad \forall c \tag{3.6e}$$

$$\frac{f_{i,j}^{c}}{\sum_c f_{i,j}^{c}} = \frac{\beta_{i,j}^{c} S_i^{c}}{\sum_c \beta_{i,j}^{c} S_i^{c}} \quad \forall i, j, c \tag{3.6f}$$

$$\text{(Partial) FIFO and SCIR constraints (in this work, (3.3) and (3.4)).} \tag{3.6g}$$

A solution will have flows that are constrained by at least one of the constraints outlined above. An algorithm to solve this problem and proof of optimality was given in Wright et al. (2017). Below, we will present a new (simpler) algorithm for the same problem.

3.2.1.4 Other first-order node model requirements

Note that the list of first-order node requirements presented so far 3.2.1 (which is the particular node problem of interest for the remainder of this chapter) is not an exhaustive list of all "reasonable" node model requirements. Since the statement of the GCNM requirements in Tampère et al. (2011), several authors have proposed extensions or modifications (as we

have in the "partial FIFO" relaxation). Beyond what we have covered here, one of the most discussed are nodal supply constraints. These supply constraints, as their name suggests, describe supply limitations at the node rather than in one of the output links. They are meant to describe restrictions on traffic that occur due to interference between flows in the junction (rather than vehicles being blocked in the input link), or the exhaustion of some "shared resource" such as green light time at a signalized intersection. Each movement through the node may or may not consume an amount of a node supply proportional to its throughflow.

The node supply constraints in the GCNM framework were originally proposed in Tampère et al. (2011). In Corthout et al. (2012) it was noted that these node supplies may lead to non-unique solutions. Jabari (2016) revisited the node supply constraints (mostly in the context of distribution of green time) to address Corthout et al. (2012)'s critique of non-uniqueness of solutions and suggested that non-uniqueness can be resolved by properly accounting for signal phasing (i.e., which green time allocations are active at the same time).

We do not explicitly include the node supply constraints in the dynamic system node models and resulting solution algorithms in this chapter. The path towards their inclusion in the first- and second-order cases is similar to the partial FIFO construction but notationally cumbersome and somewhat beyond this chapter's scope of fusing the GCNM and second-order link models.

3.2.2 Other approaches to road junction modeling

In this section so far, we mostly reviewed node models that pose the node flow problem as an optimization problem (e.g., Tampère et al. (2011); Flötteröd and Rohde (2011); Corthout et al. (2012); Smits et al. (2015); Jabari (2016); Wright et al. (2017) and their references). As mentioned in section 3.1, this type of problem setup can be interpreted as taking as input the adjacent links' supply and demand (i.e., their state at the boundary) and producing as output in- and out-flows (i.e., Neumann boundary conditions) for those links. We will use this framework for the remainder of this chapter.

Beyond the solving-for-node-flows approach, another class of methods have seen recent development and should be mentioned. As described in Jin (2017), these methods instead resolve the multidimensional Reimann problem by breaking it into one Riemann problem for each link, then solving each one. Methods of this type tend to explicitly use the terminology of "Riemann solvers" (e.g., Herty and Rascle (2006); Garavello and Piccoli (2006); Haut and Bastin (2007); Jin (2017)) rather than "node models." Compared to the node-flow framework, the Riemann-solver framework could be interpreted as setting up and solving a traditional PDE boundary value problem for each link. One work of particular interest *vis-à-vis* the node-flow framework is Jin (2017), which claims to bring these two classes of methods closer together by incorporating the concepts of supply and demand that the node-flow framework inherited from the one-dimensional discretization (Daganzo, 1995).

More related to this chapter's concepts, though, are the works of Herty and Rascle (2006); Garavello and Piccoli (2006); Haut and Bastin (2007), which applied this second approach to junctions of roads with second-order dynamics. These "Riemann-solver-framework" methods

make a point to *not* use the the terminology of demand S and supply R. As we will discuss in more detail in section 3.4.3, the particular differences of the second-order node flow problem relative to the first-order problem make the approach (as done in (3.2.1)) of treating supply and demand as exogenous constraints not suitable to the second-order case.

The second-order generalization of the node problem, then, needs to take inspiration from the "Riemann-solver-framework" approach of not isolating the node flows from the surrounding links. We will make this statement more concrete later.

3.2.3 Review of first-order node dynamic system

This section reviews the node dynamic system characterization of node models presented in Wright et al. (2016). This dynamic system is a hybrid system, which means that it contains both continuous and discrete states (also called discrete modes). Here, the continuous states evolve in time according to differential equations, the differential equations themselves change between discrete states, and the discrete state transitions are activated when conditions on the continuous states are satisfied.

Definition 3.2 (Generic first-order node hybrid system)**.**

- Let there be $N \cdot M \cdot C$ time-varying continuous states $f^c_{i,j}(t)$, each representing the number of vehicles of commodity c that have taken movement i, j through the node. The continuous state space is denoted X.
- Let $\mathcal{J}$ be the set of all output links j. Let there be 2^M discrete states $q_\nu, \nu \in 2^{\mathcal{J}}$ (recall $2^{\mathcal{J}}$ refers to the power set of $\mathcal{J}$), the index ν representing the set of downstream links that have become congested. A downstream link j is said to "become congested" at time t if $\sum_i \sum_c f^c_{ij}(t) = R_j$. The discrete state space is denoted Q.
- Init $\subseteq Q \times X$ defines the set of permissible initial states of the system at $t = 0$.
- Dom: $Q \to X$ denotes the domain of a discrete state, which is the space of permissible continuous states while the discrete state is active.
- $\Phi : Q \times X \to Q \times X$ is a reset relation, which defines the transitions between discrete states and the conditions for those transitions.
- The hybrid system execution begins at time $t = 0$ and discrete state index $k = 0$. We will say that k increments by one every time there is a transition between discrete states.
- Each link is given a "time limit" $T_i \triangleq {}^{F_i}/_{p_i}$. As we discussed in section 3.2.1.1, the capacity F_i is defined as the maximum possible flow of a given time interval, and as shown in (3.7d) below, the maximum possible flow rate (when there is no partial FIFO

blockage) for a link i is the priority p_i, the partial-FIFO constraint is enforced in this manner.

Our hybrid system $(Q, X, \text{Init}, \dot{f}^c_{i,j}, \text{Dom}, \Phi)$ is

$$Q = \{q_\nu\},\ \nu \in 2^{\mathcal{J}} \tag{3.7a}$$

$$X = \mathbb{R}^{M \cdot N \cdot C} \tag{3.7b}$$

$$\text{Init} = Q \times \{f^c_{i,j}(t=0) = 0 \quad \forall i, j, c\} \tag{3.7c}$$

$$\dot{f}^c_{i,j}(q, f) = \begin{cases} p_{i,j} \dfrac{S^c_{i,j} - f^c_{i,j}(t)}{\sum_c \left(S^c_{i,j} - f^c_{i,j}(t)\right)} \left(1 - \left|\bigcup_{\substack{j' \in \nu(k), \\ \exists\, c:\, f^c_{i,j'} < S^c_{i,j'}}} \boldsymbol{\eta}^i_{j',j}\right|\right) & \text{if } f^c_{i,j}(t) < S^c_{i,j} \text{ and } t < T_i \\ 0 & \text{otherwise} \end{cases} \tag{3.7d}$$

$$\text{Dom}(q_\nu) = \left\{ f : \sum_i \sum_c f^c_{i,j} = R_j \quad \forall j \in \nu \text{ and } \sum_i \sum_c f^c_{i,j} \le R_j \quad \forall j \notin \nu \right\} \tag{3.7e}$$

$$\Phi(q_\nu, f) = (q_{\nu'}, f) \text{ if } \sum_i \sum_c f^c_{i,j^*} = R_{j^*} \tag{3.7f}$$

$$\text{where } \nu' = \nu \cup j^*.$$

When $\dot{x}^c_{ij} = 0$ for all i, j, c, the execution is complete and the f^c_{ij} take their final values.

It was shown in Wright et al. (2016) that the hybrid system (3.7) produces the same solutions as Wright et al. (2017)'s algorithm (again, for the prior version where the partial FIFO constraint was defined differently, which changed the dynamic system definition in the definition of the link time limit T_i). In the following section, we show how to quickly compute executions of the hybrid system, which, since it is based on the continuous-time dynamics of (3.7), presents a more intuitive algorithm than the one in Wright et al. (2017).

3.2.4 Execution of the first-order node dynamic system as a simple algorithm

Evaluating continuous-time or hybrid systems typically involves forward integration of the differential equation(s) with fixed or varying step sizes. However, in the case of (3.7), evaluation can be performed in a much simpler manner. This is due to the particular dynamics of the system - since the continuous-time dynamics and the condition for discrete mode switching are very simple, the time that the next mode switch will occur can be found in closed form. Equations (3.7e) and (3.7f) say that a mode switch where link j enters ν will occur when

$$\sum_i \sum_c f^c_{ij} = R_j. \tag{3.8}$$

Say we are currently at time t_0. Combining (3.8) with (3.7d), we can find the time that the mode switch occurs, which we denote t_j.

$$R_j = \sum_i \sum_c f_{ij}^c(t_0) + \int_{t_0}^{t_j} \sum_i \sum_c \dot{f}_{ij}^c dt. \tag{3.9}$$

Solving the integral in (3.9),

$$\begin{aligned}
\int_{t_0}^{t_j} \sum_i \sum_c \dot{f}_{i,j}^c dt &= \int_{t_0}^{t_j} \sum_i \sum_c p_{i,j} \frac{S_{i,j}^c - f_{i,j}^c(t)}{\sum_c \left(S_{i,j}^c - f_{i,j}^c(t)\right)} \left(1 - \Big|\bigcup_{\substack{j' \in \nu, \\ \exists\, c:\, x_{i,j'}^c < S_{i,j'}^c}} \boldsymbol{\eta}_{j',j}^i\Big|\right) dt \\
&= \int_{t_0}^{t_j} \sum_i p_{i,j} \left(1 - \Big|\bigcup_{\substack{j' \in \nu, \\ \exists\, c:\, f_{i,j'}^c < S_{i,j'}^c}} \boldsymbol{\eta}_{j',j}^i\Big|\right) \\
&= (t_j - t_0) \sum_i p_{i,j} \left(1 - \Big|\bigcup_{\substack{j' \in \nu, \\ \exists\, c:\, f_{i,j'}^c < S_{i,j'}^c}} \boldsymbol{\eta}_{j',j}^i\Big|\right).
\end{aligned} \tag{3.10}$$

Then, plugging (3.10) into (3.9),

$$t_j = t_0 + \frac{R_j - \sum_i \sum_c f_{i,j}^c(t_0)}{\sum_i p_{i,j} \left(1 - \Big|\bigcup_{\substack{j' \in \nu, \\ \exists\, c:\, f_{i,j'}^c < S_{i,j'}^c}} \boldsymbol{\eta}_{j',j}^i\Big|\right)}. \tag{3.11}$$

This value can be computed for each output link j. Then, the j with the smallest t_j will be the first link to fill and join ν. We had used j^* for this output link, so let $t_{j^*} \triangleq \min t_j$. However, one of the input links may have its time limit T_i expire. This would also change the dynamics, as it stops sending vehicles at that time.

Therefore, evaluation of the system trajectory beginning from t_0 can be done by (i) evaluating (3.11) for each output link, (ii) identifying t_j^*, and (iii) checking whether any of the time limits T_i occur before t_{j^*}. This is an event-triggered simulation: it is only necessary to determine when the next event will occur. The equations for $\dot{x}_{ij}^c$ over $[t_0, \min(\{T_i\}, t_{j^*})]$ can then be evaluated in closed form under q_ν.

Note that the $\dot{x}_{i,j}^c$'s for an i may change to zero from nonzero without a change in the discrete state q_ν, if the conditional of $x_{i,j}^c(t) < S_{i,j}^c$ in (3.7d) is broken. This can be understood as the i running out of vehicles that it is able to send. This may happen if $p_i > S_i$ for that i, and some (partial) FIFO constraint becomes active on i. In the following algorithm, we introduce a new set, μ, that was not present in the dynamic system definition and contains the i's that either exhaust their supply or have their time limits expire (i.e., those i's whose $\dot{x}_{i,j}^c$ become zero without j necessarily entering ν).

These steps are summarized in the Algorithm below. This algorithm represents the completion of an argument began in Wright et al. (2016).

Definition 3.3 (First-order node model solution algorithm)**.**

List of inputs:

- Per-commodity input demands $S_i^c \; \forall \; i, c$
- Per-output supplies $R_j \; \forall \; j$
- Per-commodity split ratios $\beta_{i,j}^c \; \forall \; i, j, c$
- Per-input priorities $p_i \; \forall \; i$
- Per-movement restriction intervals $\eta_{j',j}^i \; \forall \; i, j, j'$

1. **Initialize.**

$$
\begin{aligned}
k &= 0 && \text{Set iteration counter to 0} \\
T_i &= \frac{\sum_c S_i^c}{p_i} && \text{Compute input ``time limits''} \\
S_{i,j}^c &= \beta_{i,j}^c S_i && \text{Compute directed demands} \\
p_{i,j} &= p_i \frac{\sum_c S_{i,j}^c}{\sum_c S_i^c} && \text{Compute oriented priorities} \\
\nu(k=0) &= \{j : R_j = 0\} && \text{Note initial } (k=0) \text{ zero-supply output links} \\
f_{i,j}^c(k=0) &= 0 && \text{Initialize all throughflows to 0}
\end{aligned}
$$

2. **Compute flow rates.**

$$
\dot{f}_{i,j}^c(k) = \begin{cases} p_{i,j} \dfrac{S_{i,j}^c - f_{i,j}^c(t)}{\sum_c \left(S_{i,j}^c - f_{i,j}^c(t) \right)} \left(1 - \left| \bigcup_{\substack{j' \in \nu(k), \\ \exists \, c: \, f_{i,j'}^c < S_{i,j'}^c}} \boldsymbol{\eta}_{j',j}^i \right| \right) & \text{if } f_{i,j}^c(k) < S_{i,j}^c \text{ and } t(k) < T_i \\ 0 & \text{otherwise} \end{cases} \quad \forall \; i, j, c \tag{3.12}
$$

3. **If** $\dot{f}_{i,j}^c(k) = 0 \; \forall \; i, j, c$**, then the algorithm ends. Otherwise, continue.**

4. **Compute the length of this iteration's timestep.**

$$
\begin{aligned}
t_j(k) &= \frac{R_j - \sum_i \sum_c f_{i,j}^c(k)}{\sum_c \sum_i \dot{f}_{i,j}^c(k)} && \forall \; j \\
dt(k) &= \min \left\{ \{ t_j(k) - t(k) \}_{j \notin \nu(k)}, \; \{ T_i - t(k) \}_{i \notin \mu(k)} \right\}
\end{aligned}
$$

5. **Advance forward in time.**

$$\begin{aligned} \Delta f^c_{i,j}(k) &= \dot{f}^c_{i,j}(k) \cdot dt(k) && \forall\, i,j,c \\ f^c_{i,j}(k+1) &= f^c_{i,j}(k) + \Delta f^c_{i,j}(k) && \forall\, i,j,c \\ t(k+1) &= t(k) + dt(k) \end{aligned}$$

6. **Update set of "completed" links.**

$$\nu(k+1) = \nu(k) \cup \left\{ j : R_j - \sum_i \sum_c f^c_{i,j}(k+1) = 0 \right\}$$

7. **Set $k \leftarrow k+1$, return to step 2 and repeat.**

We consider the algorithm given in Definition 3.3 to be quite a bit more intuitive than previous forms of solution algorithms to the node model problem that do not follow the dynamic system approach. Key to this is that all the quantities have a clear physical meaning.

Remark 3.3. Now that we have introduced the concept of continuous-time timesteps in this context (Step 4 above), we can briefly revisit our specific SCIR (3.4) to provide a physical intuition for the second condition in (3.4c). Recalling the second condition in (3.4c),

$$\frac{f_{i,j^*}}{p_{i,j^*}} < \frac{f_{i',j^*}}{p_{i',j^*}}$$

and recalling (3.12), an un-restricted flow from i to j will have flow rate $\dot{f}^c_{i,j} = p_i \cdot (S^c_{i,j} / (\sum_c S^c_{i,j}))$. So, (3.4c) can be rewritten as

$$\frac{f_{i,j^*}}{\dot{f}_{i,j^*}} < \frac{f_{i',j^*}}{\dot{f}_{i',j^*}}.$$

In other words, the condition that states that i cannot be restricted by j^* (and must be restricted by some other constraint) if another link i' is able to send more priority-normalized flow to j^* also means that i' cannot send flow to j^* for a *longer time* than i.

3.3 Review of second-order flow modeling

3.3.1 Introduction

The formulation of the GSOM seen in (3.1) has been called the "advective form" (Fan et al., 2017). In this form, the property w is advected with the vehicles at speed v. That is, it is constant along trajectories. This form makes the statement that the property w is a property of vehicles. This understanding is useful because it communicates how the forms of w and $V(\rho, w)$ are often meant to model some microscopic, per-driver behavior.

For example, it has been shown (Zhang, 2002; Aw et al., 2002) that the original ARZ model can be characterized as a "coarsening" to a macro-scale of simple car-following models of the form

$$\begin{aligned} \ddot{x}_n(t) &= \frac{\dot{x}_{n-1}(t) - \dot{x}_n(t)}{\tau(s_n(t))} \\ s_n(t) &= x_{n-1}(t) - x_n(t) \end{aligned} \tag{3.13}$$

where $x_n(t)$, $\dot{x}_n(t)$ and $\ddot{x}_n(t)$ are the position, velocity, and acceleration, respectively, of the nth car, and $\tau(s_n(t))$ is a driver's response time, which is stated to be a function of their distance from the car ahead of them. This equivalence is shown by setting the velocity function as $V(\rho, w) = V_{\text{eq}}(\rho) + (w - V_{\text{eq}}(0))$, with V_{eq} defined as an *equilibrium* velocity function. The advected property w is thought of as the driver's distance from equilibrium velocity (Aw et al., 2002; Zhang, 2002; Lebacque et al., 2007b).[12]

Each individual vehicle can be thought of as having its own w, and the macroscopic w in the PDE form (3.1b) then equals the average of the vehicle w's. To apply a discretization to this PDE formulation, it is useful to consider the *total property* ρw, and rewrite (3.1) in "conservative form" (Lebacque et al., 2007b; Fan et al., 2017),

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial (\rho v)}{\partial z} &= 0 \\ \frac{\partial (\rho w)}{\partial t} + \frac{\partial (\rho w v)}{\partial z} &= 0 \\ \text{where } v &= V(\rho, w). \end{aligned} \tag{3.14}$$

We will review the relevant finite-volume discretization using the Godunov scheme of (3.14) (Lebacque et al., 2007b) in the next section. For a deeper analysis on the physical properties of (3.14), see, e.g., Lebacque et al. (2007b).

We make one note on constraints imposed on the form of $v(\rho, w)$ in (3.14). It has been stated (Lebacque et al., 2007b, (19)) that, to apply the Godunov discretization to (3.14), one is restricted to choices of $V(\rho, w)$ for which there is a unique ρ for every $(v, w), v \neq 0$ and a unique w for every $(v, \rho), v \neq 0$. That is, $V(\rho, w)$ must be invertible in both its arguments for nonzero values of velocity.

Remark 3.4. In this chapter, the only further assumption we make on the form of $V(\cdot)$ is that $v = 0$ occurs only for a specific maximum value of density, ρ_{max}, and that at that density, $V(\rho_{max}, w) = 0$ for any value of w. Further, we assume that the converse is not true: that no value of w exists that induces $v = 0$ for $\rho < \rho_{max}$.

[12] Of potential interest to the reader may be the analysis of Aw et al. (2002), where a particular form of $\tau(\cdot)$ is shown to be equivalent to a form of (3.1) with an inhomogeneous (3.1b): that is, with w allowed to be created or destroyed. We do not consider inhomogeneous forms of (3.1) in this chapter.

3.3.2 Godunov discretization of the GSOM

The Godunov discretization of the first-order (LWR) model (2.10), first introduced as the Cell Transmission Model (Daganzo, 1994) is well-known. The Godunov scheme discretizes a conservation law into small finite-volume cells. Each cell has a constant value of the conserved quantity, and inter-cell fluxes are computed by solving Riemann problems at each boundary. The solution to each Riemann problem is a flux that describes the amount of the conserved quantity that is sent from the upstream cell to the downstream cell. The Godunov scheme is a first-order method, so it is useful for simulating solutions to PDEs with no second- or higher-order derivatives like the LWR formulation. In the CTM, the Godunov flux problem is stated in the form of the demand and supply functions.

Since (3.14) is also a conservation law with no second- or higher-order derivatives, the Godunov scheme is applicable as well (Lebacque et al., 2007b). However, due to the second PDE for ρw, an *intermediate state* arises in the Riemann problem and its solution (Aw and Rascle, 2000; Zhang, 2002; Lebacque et al., 2007a,b). This intermediate state has not always had a clear physical meaning, and this lack of clarity likely inhibited the extension of the Godunov discretization to the multi-input multi-output node case. In our following outline of the discretized one-input one-output flow problem, we make use of a physical interpretation of the intermediate state due to Fan et al. (2017).

A final note: in the first-order node model, we were able to ignore the first-order demand and supply functions that generated the supplies R_j and per-commodities demands S_i^c. That is, we were agnostic to the method by which they were computed (and to the input and output link densities), as they did not change during evaluation of the node problem. As we will see shortly, this is not the case for the second-order flow problem (due to the intermediate state and its interactions with the downstream link). Therefore, our explanation below makes use of the second-order demand and supply functions $S(\rho, w)$ and $R(\rho, w)$, respectively.

3.3.2.1 Preliminaries

In this work, we say that each vehicle commodity c has its own property value w^c. The net (averaged over vehicle commodities) property of a link ℓ, denoted w_ℓ, is

$$w_\ell = \frac{\sum_c w^c \rho_\ell^c}{\rho_\ell} \tag{3.15}$$

where $\rho_\ell \triangleq \sum_c \rho_\ell^c$ is the total density of link ℓ.

In the second-order model, the fundamental diagram of a link is a function of both net density and net property as defined above. This carries over to the demand and supply functions in the Godunov discretization (Lebacque et al., 2007b; Fan et al., 2017). This means that the supply and demand are defined at the link level with the net quantities ρ_ℓ and w_ℓ. For an input link i,

$$S_i = S_i(\rho_i, w_i) = \begin{cases} \rho_i v_i & \text{if } \rho_i \leq \rho_c(w_i) \\ F(w_i) & \text{if } \rho_i > \rho_c(w_i) \end{cases} \tag{3.16}$$

where $\rho_c(w_i)$ is the critical density for property value w_i and $F(w_i)$ is the capacity for property value w_i.

The demand from (3.16) is split among the commodities and movements proportional to their densities and split ratios,

$$S_i^c = S_i \frac{\rho_i^c}{\rho_i}$$
$$S_{i,j}^c = \beta_{i,j}^c S_i^c.$$

The oriented priorities $p_{i,j}$ are computed according to (3.5), as before.

3.3.2.2 Computing supply

Compared to computing the demand, solving for an output link's supply in the multi-input-multi-output second-order case is a much more complicated problem. We will begin our discussion with a review of the one-input-one-output case (Fan et al., 2017, Sections 3.3, 3.4).

The supply R of the output link in this one-to-one case, where i is the input link and j is the output link, is

$$R_j = R_j(\rho_M, w_i) = \begin{cases} F(w_i) & \text{if } \rho_M \leq \rho_c(w_i) \\ \rho_M v_M & \text{if } \rho_M > \rho_c(w_i). \end{cases} \tag{3.17}$$

We see that the supply of the downstream link is actually a function of the upstream link's vehicles' property, and the density and speed of some "middle" state, M. The middle state is given by (Fan et al., 2017, (16))

$$w_M = w_i \tag{3.18a}$$
$$v_M = \begin{cases} V(0, w_i) & \text{if } V(0, w_i) < v_j \\ v_j & \text{otherwise} \end{cases} \tag{3.18b}$$
$$\rho_M \text{ s.t. } v_M = V(\rho_M, w_M) \tag{3.18c}$$

where $v_j = V(\rho_j, w_j)$ is the velocity of the downstream link's vehicles and $V(\cdot)$ is the velocity function as given by the fundamental diagram.

It is worth emphasizing that the speed of the flow between two links, v_M, is equal across all vehicles that are moving, despite each individual vehicle possibly having different w's. This is, again, due to the isotropic commodity mixing assumption. The particular value at which the flow moves is dependent on the net w of this isotropic mixture (3.15), as shown in (3.18).

In Fan et al. (2017), the intuition behind the meaning of the middle state is given as follows: the middle state vehicles are actually those that are leaving the upstream link i and entering the downstream link j. As they leave i and enter j, they clearly carry their own property w (3.18a), but their velocity v is upper-bounded by the velocity at which that the downstream vehicles exit link j and free up the space that the i-to-j vehicles enter (3.18b).

The middle density, ρ_M (and therefore the downstream supply R), is then determined by both the upstream vehicles' characteristics (i.e., w_i) and the downstream link's flow characteristics (through v_j). In other words, the number of vehicles that can fit into whatever space is freed up in the downstream link is a function of the drivers' willingness to pack together (defined by w_i). Since the meaning of supply R_j is "the number of vehicles that j can accept," this means that R_j is dependent on w_i (3.18c).

Note that (3.18b) is also the equation by which congestion spills back from j to i: if j is highly congested, then v_j will be low. This then makes ρ_M large in (3.18c), which in turn leads to a small R_j in (3.17).

Now that we have reviewed the 1-to-1 case, we can consider how to generalize this to a multi-input-multi-output node when we determine supply for several links.

3.4 Second-order node model problem and solution

3.4.1 Multi-input-multi-output extension of the second-order Godunov discretization

We saw that the reasoning behind the dependence of R_j on w_i was that the spacing tendencies of i's vehicles determine the number of vehicles that can fit in j. Therefore, in generalizing to a multi-input-multi-output node, it makes sense to define a link j's "middle state" as being dependent on the vehicles *actually entering* link j. That is, if w_{j^-}, the w just upstream of j, is the "middle state" of link j, then we say

$$w_{j^-} = \frac{\sum_i \sum_c w^c \dot{f}^c_{i,j}}{\sum_i \sum_c \dot{f}^c_{i,j}}. \tag{3.19a}$$

The "j-upstream middle state" velocity and density, v_{j^-} and ρ_{j^-}, are then

$$v_{j^-} = \begin{cases} V(0, w_{j^-}) & \text{if } V(0, w_{j^-}) < v_j \\ v_j & \text{otherwise} \end{cases} \tag{3.19b}$$

$$\rho_{j^-} \text{ s.t. } v_{j^-} = V(\rho_{j^-}, w_{j^-}) \tag{3.19c}$$

and the supply R_j is

$$R_j = R_j(\rho_{j^-}, w_{j^-}) = \begin{cases} F(w_{j^-}) & \text{if } \rho_{j^-} \leq \rho_c(w_{j^-}) \\ \rho_{j^-} v_{j^-} & \text{if } \rho_{j^-} > \rho_c(w_{j^-}). \end{cases} \tag{3.20}$$

Note that in (3.19a), we defined w_{j^-} as a function of $\dot{f}^c_{ij}$. Recall from the first-order node model that the $\dot{f}^c_{ij}$'s can change as (i) upstream links i exhaust their demand or (ii) downstream links j run out of supply. These two events correspond to discrete state changes in our hybrid system. This, of course, carries over to the second-order node model. This

means that the j^- quantities, and thus the supply R_j, change as $\dot{f}_{ij}^c$'s change. Therefore, at each discrete state transition, we need to determine the new supply for each output link j for the new mixture of vehicles that will be entering j in the next discrete state.

We will explain how this is done through the following example. Suppose that at time t_0, we compute some $w_{j^-}, v_{j^-}, \rho_{j^-}$, and R_j with (3.19)-(3.20). Then, at time t_1, one of the $\dot{f}_{i,j}^c$ for that j changes. At that point, we recompute ρ_j and w_j,

$$\begin{aligned} \rho_j^c(t_1) &= \rho_j^c(t_0) + \frac{1}{L_j}\sum_i f_{ij}^c(t_1) \\ \rho_j(t_1) &= \sum_c \rho_j^c(t_1) \\ w_j(t_1) &= \frac{\sum_c w^c \rho_j^c(t_1)}{\rho_j(t_1)} \end{aligned} \tag{3.21}$$

where L_j is the length of j. Then, we recompute all the "middle state" variables and R_j using (3.19)-(3.20). Critically, note that in this recomputation, the new v_j at t_1 is $v_j(t_1) = V(\rho_j(t_1), w_j(t_1))$. This means that $v_{j^-}(t_1)$ will also be different than $v_{j^-}(t_0)$. This will carry through to create a $R_j(t_1)$ that is different from $R_j(t_0)$, and takes into account both the vehicles that have moved into j between t_0 and t_1, and the difference in properties $w_{j^-}(t_0)$ and $w_{j^-}(t_1)$.

Note that if $w_{j^-}(t_1)$ leads to significantly tighter packing (i.e., smaller inter-vehicle spacing) than $w_{j^-}(t_0)$, it is conceivable that we will have $R_j(t_1) > R_j(t_0)$ (especially if $\rho_j(t_0)$ is not that much smaller than $\rho_j(t_1)$). [13]

Of course, the description above assumes isotropic mixing of all vehicle commodities in the link j (recall we stated this assumption for input links i in item (8) of the first-order GCNM requirements in section 3.2.1).

Unlike supply, demand does not need to be recomputed since we assume the mixture of vehicles *demanding* each movement remains the same (due to our isotropic mixture assumption)

In summary, the second-order multi-input-multi-output case is distinct from both the first-order multi-input-multi-output case and the second-order single-input-single-output case in that we cannot define a single supply R_j at the time of problem statement. Instead, the eventual R_j, the maximum amount of vehicles that j can accept, will be dependent on w_{j^-},

[13] As an illustrative example, suppose that we had an output link j with two input links. One input link has high priority and emits large trucks that have high inter-vehicle spacing, while the other input link has low priority and emits low-spacing cars. Suppose at time t_0 both input links are sending vehicles, which would lead to a $w_{j^-}(t_0)$ that is weighted towards the (high-priority) trucks. But then suppose the high-priority input link exhausts its demand of trucks quickly (relative to $R_j(t_0)$). If we call this time of exhaustion t_1, then we will have a new $w_{j^-}(t_1)$ that reflects the low-spacing cars. It is possible that in this situation $R_j(t_1) > R_j(t_0)$, since the units of $R_j(t)$ are in *vehicles*, which is weighted by the *type* of vehicles entering: at time t_1, a slightly smaller amount of road is available than at time t_0, but since the mixture of vehicles at t_1 are themselves are smaller and will pack tighter, we can fit more into that amount of road.

which is itself dependent on the $f^c_{i,j}$'s for that j. This circular dependency greatly exacerbates the complexity of the node problem, as we will see in the next section.

3.4.2 Generic Class of Second-order Node Models: Problem Statement

We now present our proposed statement of the second-order node problem.

Definition 3.4 (Generic second-order node model problem)**.**

$$\max \left(\sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{c=1}^{C} f^c_{i,j} \right) \tag{3.22a}$$

subject to:

$$f^c_{i,j} \geq 0 \qquad \forall i, j, c \tag{3.22b}$$

$$f^c_{i,j} \leq S^c_{i,j} \qquad \forall i, j, c \tag{3.22c}$$

$$\sum_{i=1}^{M} \sum_{c=1}^{C} f^c_{i,j} \leq R_j(\rho_{j^-}, w_{j^-}) \qquad \forall j \tag{3.22d}$$

$$\sum_{i} f^c_{i,j} = \sum_{j} f^c_{i,j} \qquad \forall c \tag{3.22e}$$

$$\frac{f^c_{i,j}}{\sum_c f^c_{i,j}} = \frac{S^c_{i,j}}{\sum_c S^c_{i,j}} \qquad \forall i, j, c \tag{3.22f}$$

$$\text{(Partial) FIFO and SCIR constraints} \tag{3.22g}$$

where $R_j(\rho_{j^-}, w_{j^-})$ is given by (3.20), and where, as before,

$$w_i = \frac{\sum_c w^c \rho^c_i}{\rho_i}$$

$$\rho_i = \sum_c \rho^c_i$$

$$S^c_i = S_i(\rho_i, w_i) \frac{\rho^c_i}{\rho_i}$$

$$S^c_{i,j} = \beta^c_{i,j} S^c_i$$

and where the "j-upstream middle state" is a modified version of (3.19) in that (3.19a) is integrated over time,

$$w_{j^-} = \frac{\sum_i \sum_c w^c f^c_{i,j}}{\sum_i \sum_c f^c_{i,j}} \qquad \forall j \tag{3.22h}$$

$$v_{j^-} = \begin{cases} V(0, w_{j^-}) & \text{if } V(0, w_{j^-}) < v_j \\ v_j & \text{otherwise} \end{cases} \qquad \forall j \tag{3.22i}$$

$$\rho_{j^-} \text{ s.t. } v_{j^-} = V(\rho_{j^-}, w_{j^-}) \qquad \forall j \tag{3.22j}$$

Remark 3.5. Conservation of w is enforced through (3.22h).

Remark 3.6. Examining (3.22f) and the definition of $S^c_{i,j}$, we see that, again, following the isotropic mixing assumption, all vehicles in a flow $(i.j)$ are equally impeded if the movement's downstream link j is congested, even if individual commodities have drastically different w's. The way that w's affect the flow (i, j) is through their contribution to the "$j - upstream$" state in (3.22h). That is, in the macroscopic model, we do not model effects of w variability (e.g., in-movement overtaking) *within* an (i, j) flow.

The functions $S_i(\rho, w)$ and $R_j(\rho, w)$ are the particular demand and supply functions for those links.

In section 3.4.1, we noted that a circular dependency exists between the supplies R_j and the flows $f^c_{i,j}$. The form that this dependency takes in the problem statement above is in (3.22d) and (3.22h)-(3.22j). Plugging the definitions of w_{j^-} and ρ_{j^-} into (3.22d), we see that the $f^c_{i,j}$'s are constrained by another quantity that is a nonlinear function (through the fundamental diagram) of $f^c_{i,j}$. We conjecture that this makes the optimization problem (3.22) nonconvex for general fundamental diagrams. In addition, this construction makes clear that the downstream supplies in the second-order multi-input-multi-output cannot be stated *a priori*. We discuss some implications of this fact in the next section.

3.4.3 Generic Class of Second-order Node Models problem: Discussion

In the first-order case as reviewed in section 3.2, the node problem is usually stated in terms of supply and demand instead of the actual conserved quantity ρ. This is done to separate the node and link models from each other: once the link model(s) have been used to produce demand and supply, the node model problem is decoupled. In addition, it is possibly more intuitive to think of flows as a function of supplies and demands (which have the same units as flow), rather than as a function of density discontinuities (in other words, we can abstract away the Riemann problem in the first-order case).

However, we have seen that a construction that abstracts away the link states in the node model is not possible for the general second-order case. This is because the downstream

supplies are dependent on the upstream w_i's of the flows that enter each downstream link.[14]

In our view, this difficulty in applying the extremely-useful "supply-and-demand-based" node problem setup to the second-order setting has likely hampered efforts towards defining and solving this extension. Compare, for example, the several papers that have made progress towards the second-order junction problem in a "traditional" PDE, non-Godunov-based approach (e.g., (Herty and Rascle, 2006; Garavello and Piccoli, 2006; Haut and Bastin, 2007)), that explicitly do not decouple the junction flow problem from the PDE model on links via the Godunov supply and demand functions.

In sum, when discussing the node model problem in the second order, one should take care to not think of a downstream link's "supply" in an unqualified sense. Rather, the downstream link has a supply R_j for each pair (ρ_{j^-}, w_{j^-}) (3.22d), one of which is compatible with the solution of the optimization problem.

However, our dynamic system construction for the node model solution generalizes to this case rather well. This is described next.

3.4.4 Dynamic system definition

Since we have defined the downstream supplies R_j as depending on the net property of the flows *actually entering* link j (3.19)–(3.20), and the final R_j of the node problem solution (3.22) as depending on the integrated-over-time flows into link j (3.22h), all we have to do to generalize the first-order node model dynamic construction to the second-order case is add a recomputation of the downstream supplies R_j whenever we also recompute the $\dot{x}^c_{i,j}$'s due to a discrete mode switch.

In this dynamic system model and solution algorithm, we use as the partial FIFO and SCIR constraints referred in (3.22g) second-order generalizations of (3.3) and (3.4) such that supply is now a function of the "j-upstream middle state." Specifically,

$$f^c_{i,j} \le F_{i,j}\left(1 - \mathcal{A}\left(\bigcup_{j' \neq j}\left\{\boldsymbol{\eta}^i_{j',j} \times \left[\frac{f_{i,j'}}{F_{i,j'}},\, 1\right]\right\}\right)\right) \tag{3.23}$$

and

$$\sum_j f_{i,j} < \sum_c S^c_i \implies W_i \neq \emptyset \quad \forall\, i \tag{3.24a}$$

$$f_{i,j} \ge \frac{p_{i,j}}{\sum_{i'=1}^M p_{i',j}} R_j(\rho_{j^-}, w_{j^-}) \quad \forall\, j \in W_i \quad \forall i \tag{3.24b}$$

where

$$W_i = \left\{ j^* :\ \sum_c \beta^c_{i,j^*} S^c_i > 0 \text{ and } \nexists\ i' \neq i \text{ s.t. } \frac{f_{i,j^*}}{p_{i,j^*}} < \frac{f_{i',j^*}}{p_{i',j^*}} \right\} \tag{3.24c}$$

[14] In the case of the one-to-one junction described in section 3.3.2, we *could* abstract away the supplies' dependency on the upstream w_i and find an *a priori* statement of the downstream R_j, since there was only one upstream w_i to influence the supply in (3.19a)

and

$$p_{i,j} = \frac{\sum_c S^c_{i,j}}{\sum_c S^c_i} p_i \tag{3.25a}$$

$$F_{i,j} = \frac{\sum_c S^c_{i,j}}{\sum_c S^c_i} F_i. \tag{3.25b}$$

In the second-order extension of the dynamic system formulation, most of the symbols remain the same, with a few changes:

Definition 3.5 (Generic second-order node dynamics system)**.**

- Let $\mu \in 2^{\mathcal{I}}$ (where $\mathcal{I}$ is the set of all input links i) be the set of all exhausted input links (this is the same set that was introduced in the first-order solution algorithm in section 3.2.4). This is necessary to state the recalculations of supply according to the steps in section 3.4.1 when a link exhausts its demand and the net property of a j^- changes.
- Paralleling j^*, let i^* denote an exhausted input link. An input link is said to be exhausted at time t if $S^c_{i,j} - f^c_{i,j}(t) = 0 \ \forall j, c$. Note that the formula for the time of demand exhaustion remains the same as in the first-order case, $T_i = {}^{F_i}/_{p_i}$.
- In the first-order dynamic system, we had one continuous state, the node flows $f^c_{i,j}$. In the second-order construction, we will need more states to account for the more complicated dynamics. We introduce new continuous states $\bar{f}^c_{i,j}(t)$, which are continuous states that are initialized to 0 and have the same continuous dynamics as $f^c_{i,j}$, and $\rho^c_j(t)$, which are the densities of each commodity c in the downstream links j. To be more specific, we are breaking $f^c_{i,j}(t)$ into two components: $\bar{f}^c_{i,j}(t)$ is the amount of c-vehicles that have taken movement i, j since the last mode switch, and $\rho^c_j(t)$ is the (cumulative) density of c vehicles that have entered link j up until the last mode switch. We need to do this because the downstream supplies $R_j(t)$ depend on both the commodity mixture of vehicles entering j at time t and the commodity mixture of vehicles that are already in j as of time t. So, $\bar{f}^c_{i,j}$ will be reset to 0 after each mode switch. This also means that the input flow vehicles that will count against the supplies $R_j(t)$ will be the $\bar{f}^c_{i,j}$.
- We assume we have the initial $\rho^c_j(0)$ for all j, c.

Our hybrid system $(Q, X, \text{Init}, \dot{f}^c_{i,j}, \dot{\bar{f}}^c_{i,j}, \dot{\rho}^c_j, \text{Dom}, \Phi)$ is

$$Q = \{q_{\mu,\nu}\},\ \mu \in 2^{\mathcal{I}},\ \nu \in 2^{\mathcal{J}} \tag{3.26a}$$

$$X = \mathbb{R}^{M \cdot N \cdot C} \times \mathbb{R}^{M \cdot N \cdot C} \times \mathbb{R}^{M \cdot C} \tag{3.26b}$$

$$\text{Init} = Q \times \left\{ \begin{array}{ll} f^c_{i,j}(t=0) = 0 & \forall i, j, c; \\ \bar{f}^c_{i,j}(t=0) = 0 & \forall i, j, c; \\ \rho^c_j(t=0) = \rho^c_j(0) & \forall j, c \end{array} \right\} \tag{3.26c}$$

$$\dot{f}^c_{i,j} = \begin{cases} p_{i,j} \dfrac{S^c_{i,j} - f^c_{i,j}(t)}{\sum_c \left(S^c_{i,j} - f^c_{i,j}(t)\right)} \left(1 - \left|\bigcup_{\substack{j' \in \nu, \\ \exists\, c:\, f^c_{i,j'} < S^c_{i,j'}}} \boldsymbol{\eta}^i_{j',j}\right|\right) & \text{if } i \notin \mu \\ 0 & \text{otherwise} \end{cases} \tag{3.26d}$$

$$\dot{\bar{f}}^c_{i,j} = \dot{f}^c_{i,j} \tag{3.26e}$$

$$\dot{\rho}^c_j = \frac{\sum_i \dot{f}^c_{i,j}}{L_j} \tag{3.26f}$$

$$\mathrm{Dom}(q_{\mu,\nu}) = \left\{ \begin{array}{ll} f : t \geq T_i \vee\ \forall j, c :\ \bar{f}^c_{i,j} = S^c_{i,j} & \forall i \in \mu, \\ \quad t \leq T_i,\ \exists j, c :\ \bar{f}^c_{i,j} \leq S^c_{i,j} & \forall i \notin \mu, \\ \quad \displaystyle\sum_i \sum_c f^c_{i,j} = R^{q_{\mu,\nu}}_j & \forall j \in \nu, \\ \quad \displaystyle\sum_i \sum_c f^c_{i,j} \leq R^{q_{\mu,\nu}}_j & \forall j \notin \nu \end{array} \right\} \tag{3.26g}$$

$$\Phi(q_{\mu,\nu}, x) = \begin{cases} (q_{\mu,\nu'}, x') & \text{if } \sum_i \sum_c \bar{f}^c_{i,j^*} = R^{q_{\mu,\nu}}_{j^*} \\ & \text{where } \nu' = \nu \cup j^* \\ (q_{\mu',\nu}, x') & \text{if } t = T_i \\ & \quad \vee\ \forall j, c\ f^c_{i,j} = S^c_{i,j} \\ & \text{where } \mu' = \mu \cup i^*. \end{cases} \tag{3.26h}$$

$$\text{where } x' \triangleq (\{f'^c_{i,j}\}, \{\bar{f}'^c_{i,j}\}, \{\rho'^c_j\}) = (\{f^c_{i,j}\}, \{0\}, \{\rho^c_j\})$$

$$\text{and } R^{q_{\mu,\nu}}_j \text{from (3.19), (3.20), with } w_j = \frac{\sum_c w^c \rho^c_j}{\sum_c \rho^c_j}$$

When $\dot{f}^c_{ij} = 0$ for all i, j, c, the execution is complete and the node flows are given by the final values of the $f^c_{i,j}$.

Unsurprisingly, the second-order dynamic system is more complicated than the first-order one. The reader will note that the discrete dynamics, as discussed before, are triggered by links j^* filling and links i^* emptying. The filling of a j^* and its entering into ν remains the same as the first-order system. The emptying or time-expiry of input links, rather than being encoded in the continuous dynamics as was done in the first-order system's (3.7d), is now in the discrete dynamics in (3.26g) and (3.26h). While it was possible to reduce the number of discrete states in the first-order system by including i-emptying in the continuous dynamics in the first-order system, in the second order system, any change in the continuous dynamics changes the output links' w_{j^-}, so all continuous dynamics changes must trigger a recomputation of R_j, which, in (3.26), we do when μ or ν change.

Although the second-order system seems much more complex than the first-order system, the second-order solution algorithm is thankfully not that much more complicated than the solution method of the first-order system. We will see why in the next section.

3.4.5 Solution algorithm

Note that, just as in the first-order system, the second order system has constant continuous dynamics in each discrete state. This means that, just as in the first-order case, we can easily compute the time that the next discrete state transition occurs. Like in section 3.2.4, this is the smallest of the t_j's and T_i's. As we said, the input link "time limits" remain the same as before, $T_i = (\sum_c S_i^c)/p_i$. The time that an output link runs out of supply and is filled under the discrete state $q_{\mu,\nu}$, if t_0 is the time that the discrete state switched to $q_{\mu,\nu}$ and j's supply was recomputed, is similar to (3.11),

$$t_j = t_0 + \frac{R_j^{q_{\mu,\nu}}}{\sum_i \sum_c \dot{x}_{i,j}^c} = t_0 + \frac{R_j^{q_{\mu,\nu}}}{\sum_{i \notin \mu} p_{i,j} \left(1 - \left|\bigcup_{\substack{j' \in \nu, \\ \exists\, c:\, f_{i,j'}^c < S_{i,j'}^c}} \boldsymbol{\eta}_{j',j}^i\right|\right)} \tag{3.27}$$

but differs in two key ways. First, the term for supply is the recomputed $R_j^{q_{\mu,\nu}}$ from (3.19) and (3.20) (this also accounts for why the numerator in (3.27) does not have a subtracted quantity as in (3.11), as that subtraction of already-filled supply is accounted for in the recomputed supply). Second is that the denominator is summed over $i \notin \mu$ rather than all i, as the set μ is not in the definition of the first-order dynamic system as stated in section 3.2.3.

We now state the solution algorithm for the second-order dynamic system. It follows the same logic as the first-order case: identifying the next T_i or t_j to occur, finding the constant continuous-time dynamics that the system will evolve under until that time, integrating forward in time, a new step of recomputing supply, and repeating.

Definition 3.6 (Second-order node model solution algorithm)**.**

List of inputs:

- Per-commodity input demands $S_i^c \; \forall \; i, c$
- Per-commodity split ratios $\beta_{i,j}^c \; \forall \; i, j, c$
- Per-input priorities $p_i \; \forall \; i$
- Per-input link capacities $F_i \; \forall \; i$
- Per-movement restriction intervals $\boldsymbol{\eta}_{j',j}^i \; \forall \; i, j, j'$
- Per-commodity properties $w^c \; \forall \; c$
- Output link fundamental diagram velocity function $v_j = V(\rho_j, w_j) \; \forall \; j$
- Initial downstream link per-commodity densities $\rho_j^c(0) \; \forall \; j, c$

1. **Initialize.**

$$k = 0 \qquad \text{Set iteration counter to 0}$$

$$T_i = \frac{F_i}{p_i} \qquad \text{Compute input "time limits"}$$

$$S_{i,j}^c = \beta_{i,j}^c S_i \qquad \text{Compute directed demands}$$

$$p_{i,j} = p_i \frac{\sum_c S_{i,j}^c}{\sum_c S_i^c} \qquad \text{Compute oriented priorities}$$

$$\mu(k=0) = \{i : S_i^c = 0 \; \forall \; c\} \qquad \text{Note initial } (k=0) \text{ empty input links}$$

$$\nu(k=0) = \{j : V_j(\rho_j, w) = 0\}; \text{ where } \rho_j = \sum_c \rho_j^c \qquad \text{Note initial } (k=0) \text{ zero-supply output links}$$

$$f_{i,j}^c(k=0) = 0 \qquad \text{Initialize all throughflows to 0}$$

$$\rho_j^c(k=0) = \rho_j^c(0) \qquad \text{Initialize all downstream densities to input values}$$

2. **Compute flow rates.**

$$\dot{f}_{i,j}^c(k) = \begin{cases} p_{i,j} \dfrac{S_{i,j}^c - f_{i,j}^c(t)}{\sum_c \left(S_{i,j}^c - f_{i,j}^c(t)\right)} \left(1 - \left|\bigcup_{\substack{j' \in \nu(k), \\ \exists\, c:\, f_{i,j'}^c < S_{i,j'}^c}} \boldsymbol{\eta}_{j',j}^i\right|\right) & \text{if } i \notin \mu(k) \\ 0 & \text{otherwise} \end{cases} \quad \forall \; i,j,c \qquad (3.28)$$

3. **If $\dot{f}_{i,j}^c(k) = 0 \; \forall \; i,j,c$, then the algorithm ends. Otherwise, continue.**

4. **Compute output link supplies.**

   a) Compute intermediate states:

$$w_{j^-}(k) = \frac{\sum_c \sum_i w^c(k) \dot{f}_{i,j}^c(k)}{\sum_i \sum_c \dot{f}_{i,j}^c(k)} \qquad \forall \; j$$

$$v_{j^-}(k) = \begin{cases} V(0, w_{j^-}(k)) & \text{if } V(0, w_{j^-}(k)) < V_j(\rho_j(k), w_j(k)) \\ V_j(\rho_j(k), w_j(k)) & \text{otherwise} \end{cases} \qquad \forall \; j$$

$$\rho_{j^-}(k) \text{ s.t. } v_{j^-}(k) = V(\rho_{j^-}(k), w_{j^-}(k)) \qquad \forall \; j$$

   where

$$w_j(k) = \frac{\sum_c w^c \rho_j^c(k)}{\sum_c \rho_j^c(k)} \qquad (3.29)$$

b) Compute supplies:

$$R_j(k) = \begin{cases} F(w_{j^-}(k)) & \text{if } \rho_{j^-}(k) \leq \rho_c(w_{j^-}(k)) \\ \rho_{j^-}(k) v_{j^-}(k) & \text{if } \rho_{j^-}(k) > \rho_c(w_{j^-}(k)) \end{cases} \qquad \forall\, j$$

5. **Compute the length of this iteration's timestep.**

$$dt_j(k) = \frac{R_j(k)}{\sum_c \sum_i \dot{f}^c_{i,j}(k)} \qquad \forall\, j \tag{3.30}$$
$$dt(k) = \min\left\{ \{dt_j(k)\}_{j \notin \nu(k)},\ \{T_i - t(k)\}_{i \notin \mu(k)} \right\}$$

6. **Advance forward in time.**

$$\begin{aligned} \Delta f^c_{i,j}(k) &= \dot{f}^c_{i,j}(k) \cdot dt(k) && \forall\, i,j,c \\ f^c_{i,j}(k+1) &= f^c_{i,j}(k) + \Delta f^c_{i,j}(k) && \forall\, i,j,c \\ t(k+1) &= t(k) + dt(k) \\ \rho^c_j(k+1) &= \rho^c_j(k) + \frac{\sum_i \Delta f^c_{i,j}(k)}{L_j} && \forall j,c \end{aligned}$$

7. **Update sets of "completed" links.**

$$\begin{aligned} \mu(k+1) &= \mu(k) \cup \{i : T_i \leq t(k+1)\} \\ \nu(k+1) &= \nu(k) \cup \{j : V_j(\rho_j(k+1), w) = 0\}\,; \text{ where } \rho_j(k+1) = \sum_c \rho^c_j(k+1) \end{aligned}$$

8. **Set $k \leftarrow k+1$, return to step 2 and repeat.**

Remark 3.7. Note that in the above, the extra bookkeeping state $\bar{f}^c_{i,j}$ that appeared in our definition of the dynamic system (Definition 3.5 is not necessary. This is because, in (3.30), we compute directly the time that the current $R_j(k)$ is exhausted under the flow rates found in Step 2. Since the dynamics of f are constant, in practice, when we combine the dynamics (3.26d) and the switching condition (3.26g) (as in (3.30)) we can plug the dynamics equation into place and solve for the switching time, eliminating the $\bar{f}$ variable.

Remark 3.8. The final downstream link net properties can be computed with (3.29) using the final (output) values of $\rho^c_j(k)$.

3.5 Conclusion

This chapter presented a generalization of the widely-used "Generic Class of Node Model" macroscopic traffic junction models to the so-called "General Second Order Model" flow model.

This chapter's results allow the extension of macroscopic modeling of variable-behavior flows (i.e., the macroscopic flow behavior depends on the mixture of behaviorally-different vehicle types) to complex general networks. Many of these flows and networks had been only able to be modeled by microscopic models that consider the behavioral variability on a per-car level, but macroscopic models that can capture the aggregate features of a more granular model can greatly increase the scale of problems that we are able to study. As stated before, the second-order flow models have been used to represent flows of great contemporary interest, such as mixtures of human-driven and autonomous vehicles (Wang et al., 2017). Researchers and practitioners will need to use every tool available to understand and predict the system-level changes that will arise from the traffic demand changing not just in size, but in characteristics.

Some immediate avenues for future refinement of second-order macroscopic models presented themselves during this chapter. As mentioned in Section 3.2.1.4, we do not address node supply constraints in this chapter's node models. However, the immediate application of a general, multi-input-multi-output second-order node model, macroscopic simulation of mixed-human-driven-and-autonomous traffic on complex networks, is of particular concern in scheduling problems involving green light timing. Future work, then, should incorporate the node supply constraints into the general second-order node problem so that they may be used in signal optimization and the still-developing potential that connected and automated vehicles bring to traffic control.

We mentioned briefly in Section 3.3.1 that the ARZ model was originally developed as a spatial coarsening of a particular simple car-following model (Aw and Rascle, 2000; Zhang, 2002). More sophisticated car-following models exist. Some popular ones include the Gipps model (Gipps, 1981), the so-called General Motors family of car-following models (Gazis et al., 1959), and the Intelligent Driver Model (Treiber et al., 2000) and "Improved" Intelligent Driver Model (Shen and Jin, 2012). The Intelligent Driver Model family in particular form the base for many automated cruise control systems in production vehicles today. That is, the microscopic behavior of many vehicles on the road are well-understood – either through human-behavioral models or because their behaviors are already (partially) computer-controlled. There is a need to analytically derive second-order fundamental diagrams in the form of spatial discretizations of contemporary microscopic models, especially in light of recent analyses (e.g., (Mehr and Horowitz, 2019)) that non-controlled deployment of autonomous vehicles can actually worsen traffic network equilibria (i.e., the familiar Braess/Jevons Paradox, but due to changes in the vehicle fleet rather than changes in the road network itself). In other words, some of the technological development and deployment already exists for macroscopic mixed-autonomy modeling and control, but some of the theoretical understanding is not yet developed (an uncommon situation in engineering, indeed).

Chapter 4

Macroscopic Modeling, Calibration, and Simulation of Managed Lane-Freeway Networks, Part I: Topological and Phenomenological Modeling

4.1 Introduction

This chapter and the next discuss modeling techniques that we have developed to describe a particular class of road networks: freeways with managed lanes. Managed lanes are implemented on freeways by restricting the use of one or more lanes to certain vehicles. Compared to the life of freeways and freeway modeling, managed lane-freeway networks are a relatively recent development (Obenberger, Nov/Dec 2004).

Managed lanes are some of the most visible realizations of recent trends of trying to handle increased freeway demand through improving the performance of *existing* infrastructure (rather than continually building more roads), both through improved road operations and *demand management*, which seeks to lower the number of vehicles on the road (Kurzhanskiy and Varaiya, 2015). As an example, high-occupancy-vehicle (HOV) lanes are intended to incentivize carpooling, which reduces the total number of cars on the road as a demand management outcome (Chang et al., 2008).

In addition to demand management, managed lanes provide an opportunity for improved road operations through real-time, responsive traffic control. For example, tolled express lanes give drivers the opportunity to pay a toll to drive parallel to the general-purpose lanes on a (presumably less-congested) express lane. Traffic management authorities here have an opportunity to adjust the toll amount in response to the real-time state of traffic on the network. The potential for managed lanes as instruments for reactive, real-time traffic

operations–in addition to their demand-management purpose–has made them popular among transportation authorities (Kurzhanskiy and Varaiya, 2015).

However, the traffic-operational effects of managed lanes are not always straightforward or as rehabilitative as expected, as their presence can create complex traffic dynamics. Even in a freeway with simple geometry, the dynamics of traffic flow are complex and not fully understood, and adding managed lanes alongside the non-managed, general-purpose (GP) lanes only exacerbates this. In effect, adding a managed lane creates two parallel and distinct, but coupled, traffic flows on the same physical structure. When used as intended, managed lanes carry flows with different density-velocity characteristics and vehicle-type (e.g., strictly HOVs) compositions than the freeway. When vehicles move between the two lane flows, these two heterogeneous flows mix, and complex phenomena that are unobserved in GP-only freeways can emerge (see, e.g., Menendez and Daganzo (2007); Daganzo and Cassidy (2008); Cassidy et al. (2010, 2015); Liu et al. (2011); Jang and Cassidy (2012); Thomson et al. (2012); Ponnu and Coifman (2015, 2017); Fitzpatrick et al. (2017); Kim and Park (2018), and others).

Making better use of managed lanes requires an understanding of the macroscopic behavior they induce. One widely-used tool for understanding macroscopic traffic flow behavior is the macroscopic traffic flow model. A rich literature exists on macroscopic models for flows on long roads, and at junctions where those roads meet, but an extension to the parallel-flows situation created by placing a managed lane in parallel with a freeway (a "managed lane-freeway network") is not straightforward. For example, the GP/managed lane interface has been observed to exhibit unique phenomena, such as e.g., 1) a "friction effect" (Liu et al., 2011; Jang et al., 2012), where vehicles in a GP lane adjacent to the managed lane(s) move slower than what would be expected based solely on their density of cars (Liu et al., 2011; Fitzpatrick et al., 2017) (a common theory is that these GP lane vehicles move slowly out of fear that vehicles may suddenly move out of the managed lane in front of them), or 2) a "smoothing effect" at bottlenecks (Menendez and Daganzo, 2007; Cassidy et al., 2010; Jang and Cassidy, 2012) that leads to an *increased* flow in the GP lanes closest to the managed lane(s) by reducing lane changes (briefly, fewer vehicles being eligible for the fastest lane means that fewer vehicles will change lanes - and in the process of changing lanes, slow down surrounding traffic - to enter it. See, e.g., Zheng (2014) for more information on the effects of lane changes on macroscopic flow characteristics). In the present chapter, we propose simple models for these emergent phenomena that fit within the classic macroscopic kinematic wave theory. To the best of our knowledge, we present the first macroscopic modeling technique for the smoothing effect, and the first proposed model of the friction effect that considers junctions between the managed and GP lanes (we will discuss the differences between our proposed friction effect model and ones previously proposed in the literature below).

This chapter presents macroscopic flow modeling tools we have used for simulation of managed lane-freeway networks. We begin in section 4.2 with a discussion of relevant modeling tools from the literature, and how we make use of them. Section 4.3 describes network structures for the two common managed lane configurations: gated-access and full-access (ungated) lanes. Section 4.4 describes simple models for the emergent phenomena that are particular to managed lane-freeway networks: the friction, smoothing, and inertia

effects. In chapter 5, we will put all of these pieces together, discuss how to calibrate the full model, and present modeling case studies of two freeways with managed lanes in California.

4.2 Managed Lane Modeling

The modeling techniques presented in this work are based on the first-order "kinematic wave" macroscopic traffic flow model. These models describe aggregate traffic flows as fluids following a one-dimensional conservation law. We briefly introduce our notation here, but do not discuss the basics of this class of models. Detailed reviews are available in many references.

4.2.1 Modeling basics

In this simulation framework, a road is divided into discrete cells, which we refer to as *links*. Links are drawn between *nodes*: a link begins at one node and ends at another. Many links may begin and end at each node. Each link l is characterized by density ρ_l, the number of cars in the link. In a first-order model, the traffic flows are fully prescribed by the density. From timestep t to $t+1$, link l's density updates according to the equation

$$\rho_l(t+1) = \rho_l(t) + \frac{1}{L_l}\left(\sum_{i=1}^{M} f_{il}(t) - \sum_{j=1}^{N} f_{lj}(t)\right), \tag{4.1}$$

where L_l is the length of link l, $f_{il}(t)$ is the flow (number of vehicles) leaving link i and entering link l at time t, $f_{lj}(t)$ is the flow leaving link l and entering link j at time t, M is the number of links that end at link l's beginning node, and N is the number of links that begin at link l's ending node.

Computing the inter-link flows requires the use of two intermediate quantities for each link. These are the link demand $S_l(t)$, which is the number of vehicles that wish to exit link l at timestep t; and the link supply, $R_l(t)$, which is the number of vehicles link l can accept at time t. Both S_l and R_l are functions of the density ρ_l. The model that computes S_l and R_l from ρ_l is often called the "fundamental diagram" or "link model," and the model that computes the flows from all links' supplies and demands is often called the "node model."

A brief outline of how first-order macroscopic simulation of a road network (sometimes called a *dynamic network loading* simulation) is performed could be:

1. At time t, use the link model for each link l to compute the link's demand $S_l(t)$ and supply $R_l(t)$ as a function of its density $\rho_l(t)$.

2. Use the node model for each node to compute the inter-link flows f_{ij} for all incoming links i and outgoing links j as functions of $S_i(t)$, $R_j(t)$, and information about vehicles' desired movements ij.

3. Update the state of each link using (4.1).

4. Increment t and repeat until the desired simulation end time is reached.

The tools described in this work are compatible with any such link model. We make use of a particular node model that we have studied in Wright et al. (2016, 2017). An aspect of this node model of particular relevance to managed lane modeling is our "relaxed first-in-first-out (FIFO) rule" construction (Wright et al., 2017). This is necessary for modeling the flows between the GP and managed lanes. Without a FIFO relaxation, congestion in one of the two lane groups could block traffic in the other when that may be unrealistic (see Wright et al. (2017, Section 2.2) for a detailed discussion).

So far, we have presented ingredients for a model that, while able to express many simple network topologies by joining links and nodes, does not capture several important behaviors in managed lane-freeway networks. The next three Sections briefly overview the additions to the standard model that will be explained in greater detail in the remainder of the work.

4.2.2 Multiple classes of vehicles and drivers

In (4.1), we describe the number of vehicles in a link as a single number, ρ_l. In this formulation, all vehicles are treated the same. However, for simulation in a managed lane-freeway network, it makes sense to break ρ_l into different classes of vehicles and/or drivers. For example, for a freeway with an HOV lane facility, we might consider two classes: HOVs and non-HOVs. To this end, (4.1) can be rewritten as

$$\rho_l^c(t+1) = \rho_l^c(t) + \frac{1}{L_l}\left(\sum_{i=1}^{M} f_{il}^c(t) - \sum_{j=1}^{N} f_{lj}^c(t)\right), \tag{4.2}$$

where $c \in \{1, \ldots, C\}$ indexes vehicle classes (often called "commodities" in the traffic literature).

Extending the density update equation to multiple classes means that the link and node models must also be extended to produce per-class flows f_{ij}^c. In this work, we will not specify a particular link model, but assume use of one that produces per-class demands S_i^c and overall supplies R_j (the node model, in computing the f_{ij}^c, is responsible for splitting the available supply R_j among the different demanding vehicle classes). Examples of this type of link model include those considered in Wong and Wong (2002), Daganzo (2002), and van Lint et al. (2008) (examples of multi-class link models of second- or higher-order include those of Hoogendoorn and Bovy (2000). These types of models have higher-order analogs of supply and demand).

4.2.3 Toplogical expression of managed lane-freeway networks

In both (4.1) and (4.2), we describe a link in terms of its total density ρ_l and its breakdown into per-commodity portions, ρ_l^c. By discretizing the road into these one-dimensional links,

we lose information about differences between vehicle proportions across lanes, as well as inter-lane and lane-changing behavior. This becomes a problem if such unmodeled behavior is of interest. In our setting, this means that modeling a freeway with a managed lane should not be done with a single link following (4.1) or (4.2), as it would be impossible to study the managed lane-freeway network behavior of interest.

Modeling differences in vehicle density across lanes is natural in microscopic and mesoscopic (see, for example, Treiber et al. (1999); Hoogendoorn and Bovy (1999); Ngoduy (2006), and others) models, but macroscopic models, in their simplicity, have less readily-accessible avenues for including these differences. One straightforward method is to model each lane as a separate link, as in, e.g. Bliemer (2007) or Shiomi et al. (2015). However, this method has a few drawbacks. First, it requires the addition of some lane-assignment method to prescribe the proportions of each vehicle class c for each lane (such as a logit model as used in Farhi et al. (2013) and Shiomi et al. (2015)), which requires not-always-accessible data for calibration. Second, drastically increasing the number of links in a macroscopic model will necessarily increase the size of the state space and model complexity, which is, in a sense, incompatible with the overall goal of selecting a macroscopic model over a micro- or mesoscopic model: some of the "macro" in the macroscopic model is lost.

Instead, in this work we choose to model the GP lanes (or "GP lane group") as one link and the parallel managed lanes (or "managed lane group") as another link. Applied to an entire length of road, this creates a network topology of two "parallel chains" of links - one GP and one managed. The two chains will share nodes, but cross-flows between the chains are permitted only in locations where there is physical access (i.e., no physical barriers) and policy access (i.e., no double solid lines under U.S. traffic markings). Where cross-flows are possible, we do not use a logit model, but instead a driver behavior model first introduced in Wright et al. (2017). This two-chain model is similar to the one described in Liu et al. (2012), though in this reference, GP-managed lane crossflows (i.e., physical joining of the chains) were not considered.

4.2.4 Modeling emergent phenomena particular to managed lane-freeway networks

Our techniques for modeling the friction, smoothing, and inertia effects are rooted in a full-picture view of the macroscopic modeling framework. As we will see, we propose isolating each of these "effects" to one component of the traditional macroscopic modeling framework: the friction effect model is based on a principled feedback mechanism of the fundamental diagram, the inertia effect model is included in modeling of driver choices for which of the two lane groups they will take, and the smoothing effect model emerges from a particular recently-developed node model construction.

4.3 Full- and Gated-Access Managed Lane-Freeway Network Topologies

We will consider two types of managed lane-freeway network configurations: *full access* and *separated with gated access*. In a a full-access configuration, the managed lane(s) are not physically separated from the GP lane(s), and eligible vehicles may switch between the two lane groups at any location. Often, full-access managed lane(s) are special-use only during certain periods of the day, and at other times they serve as GP lane(s) (e.g., HOV lanes are often accessible to non-HOVs outside of rush hour). On the other hand, in a gated-access configuration, traffic may switch between the managed lane(s) and GP lane(s) only at certain locations, called *gates*; at non-gate locations, the two lane groups are separated by road markings (i.e., a double solid line in the U.S.) or a physical barrier. Usually, gated-access managed lanes are special-use at all times. The implemented managed lane access scheme depends on jurisdiction. For example, full-access lanes are common in Northern California, and separated lanes are common in Southern California.

The differences in physical geometry and access points between the two access types requires two different types of topology in constructing a network for a macroscopic model.

4.3.1 Note on link and node models used in this section

As discussed in the previous Section, we attempt to be agnostic with regards to the particular link model (e.g., first-order fundamental diagram) used in our implementations. However, in our model of the friction effect in Section 4.4.1, we parameterize friction being in effect on a particular link l for a particular vehicle class c at time t by adjusting that link's demand $S_l(t)$. For that discussion only, we specify a particular link model. This link model is reviewed in Appendix A.

The node model used here and for the remainder of this work, when a particular form is necessary, is the one discussed in Wright et al. (2016, 2017). We use this node model because it handles multi-commodity traffic, optimizes the utilization of downstream supply, makes use of input link priorities and has a relaxation of the "conservation of turning ratios" or "first-in-first-out" (FIFO) constraint of most node models. This last feature allows us to describe a set of GP lanes just upstream of an offramp with one link, and handle a condition of a congested offramp by having the congestion spill back onto only the offramp-serving lanes of the GP link (as opposed to the entirety of the GP link). See Wright et al. (2017, Section 3) and Wright et al. (2016) for more discussion.

4.3.2 Full-access managed lanes

A full-access managed lane configuration is presented in Figure 4.1: GP and managed links (recall, as discussed above, all GP lanes and all managed lanes are collapsed into one link each) are parallel with the same geometry and share the same beginning and ending node

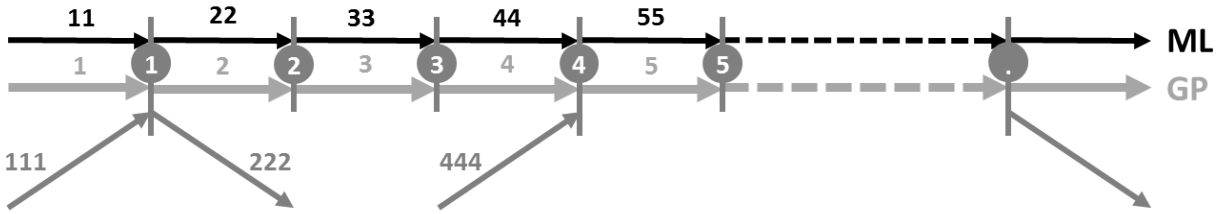


Figure 4.1: Freeway with full-access managed lane. ML = Managed Lane.

pairs; traffic flow exchange between GP and managed lanes can happen at every node. Note that in Figure 4.1, we use a slightly irregular numbering scheme so that it is clear whether a link is a GP link, managed lane link, or ramp link. Parallel links in the graph have numbers made up of the digit of their terminating node, with GP links having one digit (i.e., link 1), managed lane links having two (i.e., link 11), and ramp links having three (i.e., link 111). Note also that we use a U.S.-style, driving-on-the-right convention here, with the managed lane(s) on the left of the GP lanes and the ramps on the far right.

Links that are too long for modeling purposes (i.e., that create too low-resolution a model) may be broken up into smaller ones by creating more nodes, such as nodes 2 and 3 in Figure 4.1. Fundamental diagrams for parallel GP and managed lane links may be different (Liu et al., 2011).

We introduce two vehicle classes ($C = 2$): $c = 1$ corresponds to the GP-only traffic and $c = 2$ corresponds to the special traffic. When the managed lane(s) is (are) active, $c = 1$-traffic is confined to the GP link, whereas $c = 2$-traffic can use both the GP and managed lane links. We denote the portion of vehicles of class c in link i that will attempt to enter link j as $\beta^c_{i,j}$. This quantity is called the split ratio.

4.3.2.1 Split ratios for full-access managed lanes

We make an assumption that both vehicle classes take offramps at the same rate. For example, for node 1 in Figure 4.1, we might say that $\beta^1_{i,222} = \beta^2_{i,222} \triangleq \beta_{i,222}$. Strictly speaking, it is not necessary to assume that the $\beta^c_{i,222}$ are equal for all c. However, in practice the offramp split ratios are typically estimated from flow count data taken from detectors on the offramp and freeway. Generally speaking, these detectors cannot identify vehicle type, so the only quantity estimable is a flow-weighted average of the quantities $\beta^c_{i,222}$. To estimate the class-specific split ratios, one needs some extra knowledge of the tendency of each class to take each offramp (for example, that GP-only vehicles are half as likely as special traffic to take a certain offramp). Assuming that each class exits the freeway at the same rate is a simple and reasonable-seeming assumption.

This same problem of unidentifiability from typical data appears in several other split ratios. First, it may not be possible to tell how many vehicles taking an offramp link come from the upstream GP, managed lane, or (if present) onramp link. In this case, some assumptions must then be made. For example, three different assumptions that may be

reasonable are (1) that vehicles in each link i take the offramp at the same rate; or (2) that no vehicles from the managed lane(s) are able to cross the GP lanes to take the offramp at this node, and that no vehicles entering via the onramp, if one is present, exit via the offramp at the same node; or (3) that vehicles in GP and managed links take the offramp at the same rate, while no vehicles coming from the onramp are directed to the offramp. Looking back at node 1 in Figure 4.1 again, assumption (1) would say the $\beta_{i,222}$ are equal for all i; assumption (2) would say $\beta_{11,222} = \beta_{111,222} = 0$; and assumption (3) would say $\beta_{1,222} = \beta_{11,222}$ and $\beta_{111,222} = 0$. The best assumption for each node will depend on the road geometry for that particular part of the road (how near the offramp is to any onramps, how many GP lanes a vehicle in a managed lane would have to cross, etc.).

Second, the crossflows between the GP and managed lane links are not observable. Even if there exist detectors immediately upstream and downstream of the node where traffic can switch between GP and managed lanes, it is impossible to uniquely identify the crossflows. In a simulation, these crossflows must be governed by some driver choice model.

Putting together these assumptions and the special-traffic-only policy for the managed lane, we can summarize most of the necessary split ratios needed for computing flows in a node model. For example, for node 1 in Figure 4.1,

$$\beta^1_{i,j} = \begin{cases} & \boldsymbol{j=2} & \boldsymbol{j=22} & \boldsymbol{j=222} \\ \boldsymbol{i=1} & 1-\beta_{1,222} & 0 & \beta_{1,222} \\ \boldsymbol{i=11} & \text{n/a} & \text{n/a} & \text{n/a} \\ \boldsymbol{i=111} & 1-\beta_{111,222} & 0 & \beta_{111,222} \end{cases}$$

$$\beta^2_{i,j} = \begin{cases} & \boldsymbol{j=2} & \boldsymbol{j=22} & \boldsymbol{j=222} \\ \boldsymbol{i=1} & - & - & \beta_{1,222} \\ \boldsymbol{i=11} & - & - & \beta_{11,222} \\ \boldsymbol{i=111} & - & - & \beta_{111,222} \end{cases}$$

where "n/a" means that the split ratios $\beta^1_{11,j}$ are not applicable, as there should be no vehicles of class $c = 1$ in the managed lane. The split ratios marked with a dash are those above-mentioned flows that are unobservable and come from some driver choice model. Of course, whatever method is chosen to compute these unknown split ratios, we must have $\beta^2_{i,2} + \beta^2_{i,22} = 1 - \beta^2_{i,222}$.

Similarly, for node 2, which does not have an onramp or an offramp,

$$\beta^1_{i,j} = \begin{cases} & \boldsymbol{j=3} & \boldsymbol{j=33} \\ \boldsymbol{i=2} & 1 & 0 \\ \boldsymbol{i=22} & \text{n/a} & \text{n/a} \end{cases}$$

$$\beta^2_{i,j} = \begin{cases} & \boldsymbol{j=3} & \boldsymbol{j=33} \\ \boldsymbol{i=2} & - & - \\ \boldsymbol{i=22} & - & - \end{cases}$$

with "n/a" and the dash meaning the same as above.

As previously mentioned, full-access managed lanes often have certain time periods during which nonspecial ($c = 1$) traffic is allowed into the managed lane. We can model this change in policy simply by changing the split ratios at the nodes. For node 1, for example, the nonrestrictive policy is encoded as

$$\beta^c_{i,j} = \begin{cases} & \boldsymbol{j=2} & \boldsymbol{j=22} & \boldsymbol{j=222} \\ \boldsymbol{i=1} & - & - & \beta_{1,222} \\ \boldsymbol{i=11} & - & - & \beta_{11,222} \\ \boldsymbol{i=111} & - & - & \beta_{111,222} \end{cases} \qquad \text{for } c = \{1, 2\},$$

and for node 2 as

$$\beta^c_{i,j} = \begin{cases} & \boldsymbol{j=3} & \boldsymbol{j=33} \\ \boldsymbol{i=2} & - & - \\ \boldsymbol{i=22} & - & - \end{cases} \qquad \text{for } c = \{1, 2\}.$$

In other words, the managed lane link is treated as additional GP lane(s), and the split ratios governing the crossflows between the two links should be found from the driver choice model for both vehicle classes.

4.3.2.2 Node model for full-access managed lane-freeway networks

As mentioned in Section 4.3.1, we make use of the node model discussed in Wright et al. (2016, 2017) to describe a freeway network with managed lanes. This node model differentiates itself from others in that it deals with multi-commodity traffic flow, optimally utilizes the available supply, makes use of input link priorities, and has a relaxation of the common FIFO constraint. By default, link priorities can be taken proportional to link capacities. To explain relaxed FIFO, say that some link i has vehicles that wish to enter both links j and j'. If link j' is jammed and cannot accept any more vehicles, a strict FIFO constraint would say that the vehicles in i that wish to enter j' will queue at i's exit, and block the vehicles that wish to enter j. In a multi-lane road, however, only certain lanes may queue, and traffic to j may still pass through other lanes. The relaxation is encoded in so-called "mutual restriction intervals" $\boldsymbol{\eta}^i_{j'j} \subseteq [0, 1]$. This interval partly describes the overlapping regions of link i's exit that serve both links j and j'. For $\boldsymbol{\eta}^i_{j'j} = [y, z]$, a $z - y$ portion of i's lanes that serve j also serve j', and will be blocked by the cars queueing to enter j' when j' is congested. For example, if j is served by three lanes of i, and of those three lanes, the leftmost also serves j', we would have $\boldsymbol{\eta}^i_{j'j} = [0, 1/3]$.

As an example, we consider again node 1 in Figure 4.1. Say that the GP links (1 and 2) have four lanes, that the managed lane links (11 and 22) have two lanes, and that the onramp merges into and the offramp diverges from the rightmost GP lane. Further, we say that when the managed lane link is congested, vehicles in the GP lanes that wish to enter the managed lanes will queue only in the leftmost GP lane. On the other hand, when the GP link is congested, vehicles in the managed lanes that wish to enter the GP lanes will queue only

in the rightmost managed lane. Finally, we suppose that jammed offramp (222) will cause vehicles to queue only in the rightmost GP lane. Taking together all of these statements, our mutual restriction intervals for this example are:

$$\boldsymbol{\eta}^{1}_{j'j} = \begin{cases} & \boldsymbol{j=2} & \boldsymbol{j=22} & \boldsymbol{j=222} \\ \boldsymbol{j'=2} & [0,1] & [0,1] & [0,1] \\ \boldsymbol{j'=22} & [0,1/4] & [0,1] & \emptyset \\ \boldsymbol{j'=222} & [3/4,1] & \emptyset & [0,1] \end{cases}$$

$$\boldsymbol{\eta}^{11}_{j'j} = \begin{cases} & \boldsymbol{j=2} & \boldsymbol{j=22} & \boldsymbol{j=222} \\ \boldsymbol{j'=2} & [0,1] & [0,1/2] & \emptyset \\ \boldsymbol{j'=22} & [0,1] & [0,1] & \emptyset \\ \boldsymbol{j'=222} & \emptyset & \emptyset & [0,1] \end{cases}$$

$$\boldsymbol{\eta}^{111}_{j'j} = \begin{cases} & \boldsymbol{j=2} & \boldsymbol{j=22} & \boldsymbol{j=222} \\ \boldsymbol{j'=2} & [0,1] & [0,1] & [0,1] \\ \boldsymbol{j'=22} & \emptyset & [0,1] & \emptyset \\ \boldsymbol{j'=222} & [0,1] & [0,1] & [0,1] \end{cases}.$$

To read the above tables, recall that as written, j' is the congested, restricting link, and j is the restricted link. These chosen restriction intervals allow for expected behavior in this network, such as a congested GP link causing possible queueing in the right managed lane (if some drivers are trying to enter the GP link), but no spillback into the left managed lane.[1]

For a detailed discussion on how mutual restriction intervals are included in the node model's flow calculations and solution algorithms, see Wright et al. (2016, 2017).

4.3.3 Separated managed lanes with gated access

A separated, gated-access managed lane configuration is presented in Figure 4.2. Unlike the full-access configuration, the GP and managed lane link chains do not necessarily meet at every node. Instead, they need only meet at a few locations, where vehicles can move into and out of the managed lane(s). Note that, unlike the full-access configuration, there is no need for GP and managed lane links to be aligned.

As labeled in Figure 4.2, the nodes where the two link chains meet are called *gates* (as an aside, one way to describe the full-access managed lane configuration would be that every node is a gate). Similar to our construction of excluding GP-only traffic from the managed lane in the full-access case, we can disable flow exchange at a given gate by fixing split ratios so that they keep traffic in their lanes. For example, to disable the gate (the flow exchange between the two lanes) at node 2 in Figure 4.1, we set $\beta^{c}_{2,3} = 1$ and $\beta^{c}_{22,33} = 1$ (this means that $\beta^{c}_{2,33} = 0$ and $\beta^{c}_{22,3} = 0$), $c = 1, 2$. Thus, the full-access managed lane can be easily converted into the separated managed lane by setting non-exchanging split ratios everywhere but designated gate-nodes.

[1] In this example, we assume that the managed lane has two sublanes — left and right.

In practice, a gate is stretch of freeway that may be a few hundreds of meters long (Cassidy et al., 2015), and, potentially, we can designate two or three sequential nodes as gates. In this work, however, we model a gate as a single node.

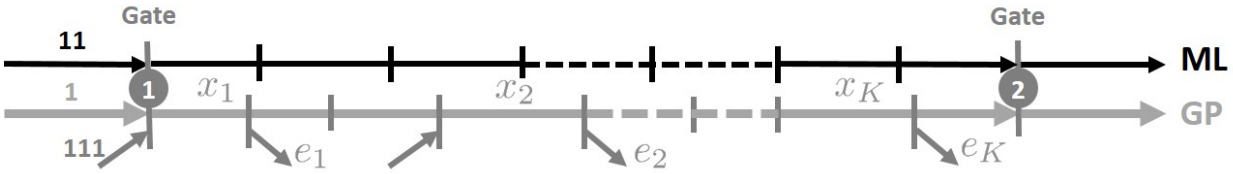


Figure 4.2: Freeway with separated managed lane and gates. ML = Managed Lane.

For the gated-access configuration, we suggest setting mutual restriction coefficients in the same manner as full-access managed lanes, in Section 4.3.2.2.

Compared to the full-access managed lane configuration, the gated-access configuration has a smaller friction effect (Jang et al., 2012): drivers in the separated managed lane feel somewhat protected by the buffer, whether it is virtual (double solid line) or real (concrete), from vehicles changing abruptly from the slow moving GP lane and, therefore, do not drop speed as dramatically. The degree to which the friction effect is mitigated is disputed (e.g., see Footnote 3 in Cassidy et al. (2015)), but overall the bottlenecks created by the gates are much greater instigators of congestion (Cassidy et al., 2015). Inclusion of the friction effect in modeling separated managed lane configurations is thus not as essential as in modeling the full-access case.

By similar logic, the smoothing effect should be expected to be less prominent. If a physical barrier prevents lane changing into the managed lane, then there will be no lane changing into the managed lane - the flow is already as "smoothed" as possible. It will become apparent in section 4.4.3, when we discuss our smoothing effect model, that in our model the degree of traffic smoothing is indeed maximized when the inter-link split ratios are zero (as they are in the non-gate nodes).

4.3.3.1 Modeling a flow of vehicles from the managed lanes to the offramps

Recall from Section 4.3.2.1 that, in the full-access managed lane model, we can model vehicles moving from the managed lane link to offramps in a straightforward manner, by setting corresponding split ratios (for example, $\beta^c_{11,222}$, $c = 1, 2$, for node 1 in configuration from Figure 4.1). For the gated-access configuration, however, modeling traffic as moving from the managed lanes to offramps is more complicated: generally, gates do not coincide with offramp locations. In fact, there are typically between two and five offramps between two gate locations. These offramps cannot be accessed directly from the managed lane. To model a vehicle flow that originates at the managed lane link, goes through a gate to the GP lane link, and then takes the correct offramp (and doing so at some rate that matches external origin-destination data), requires a more involved modeling and bookkeeping construction.

To resolve this challenge, our gated-access model introduces new vehicle classes in addition to the $c = 1$ (GP-only) and $c = 2$ (special) traffic used in the full-access model of Section 4.3.2. These additional classes will be used to distinguish subsets of the special traffic population by its destination offramp. If K is the largest number of offramps between two adjacent gates, then altogether we have $C = K + 2$ vehicle classes: $c = 1, 2, e_1, \ldots, e_K$, where e_k indicates the class of vehicles that will exit through the k-th offramp after leaving the managed lane through the gate. By definition, traffic of type $c = e_k$ *may* exist in the GP lane segment between gate 1 and offramp e_k, but there is *no traffic of this type* either in the GP link segment between offramp e_k and gate 2 or in the managed lane link. This movement pattern is ensured by setting *constant* split ratios:

$$\begin{array}{lll} \beta^{e_k}_{i\,x_1} = 1, \quad i = 1, 11, 111, & \text{direct all } e_k\text{-type traffic to the GP link at gate 1;} & \\ \beta^{e_k}_{x_k e_k} = 1, & \text{direct all } e_k\text{-type traffic to offramp } e_k; & (4.3) \\ \beta^{e_k}_{x_{k'} e_{k'}} = 0, \quad k' \neq k, & \text{do not send any } e_k\text{-type traffic to other offramps,} & \end{array}$$

where $k = 1, \ldots, K$, and x_k denotes the input GP link for the node that has the output link e_k (see Figure 4.2).

Vehicles of class $c = e_k$ do not enter the network via onramps or the upstream boundary, but instead are converted from special $c = 2$ traffic as it leaves the managed lane(s) through the gate. We perform this conversion as part of the link model computation, such that the total demand for the switching link, $\sum_{c=1}^{C} S_l^c$ remains the same before and after the switch. The exact link on which this switching takes place is the managed lane link immediately upstream of the gate (e.g., link 11 in Figure 4.2).

We say that the amount of traffic that should change from vehicle class $c = 2$ to vehicle class e_k at time t is (using link 11 as an example) $\rho^2_{11}(t)\beta^2_{x_k e_k}(t)v_{11}(t)$, where $v_{11}(t)$ is in units of vehicles per simulation timestep (this factor is included so that the switching done is proportional to link 11's outflow, rather than its density), and $\beta^2_{x_k e_k}(t)$ is the split ratio from x_k, the GP link immediately upstream of exit e_k, and the exit e_k at time t. This statement is based on the assumption that the vehicles in the managed lane link will exit the freeway through exit e_k at the same rate as special ($c = 2$) vehicles that happened to stay in the GP lanes. That is, if a $\beta^2_{x_k e_k}$ portion of $c = 2$ vehicles intend to leave the GP lanes through exit e_k, then a $\beta^2_{x_k e_k}$ portion of the $c = 2$ vehicles in the managed lane(s) will leave the managed lane link at the closest upstream gate and leave the network at exit e_k when they reach it. Note that if there are $K' < K$ offramps between two particular gates, then no vehicles should switch to type $c = e_k, k \in \{K' + 1, \ldots, K\}$ at the upstream gate, as they would have no ramp to exit through.

We now explain how e_k-type traffic appears in the system. The original demand $d^c_l(\cdot)$ is specified at origin links l for commodities $c = 1, 2$, and $d^{e_k}_l(\cdot) \equiv 0$, $k = 1, \ldots, K$. Destination specific traffic appears in the managed lane links that end at gate-nodes by assigning destinations to portions of the type-1 (GP-only) and type-2 (managed lane-eligible) traffic in those links. We propose using offramp split ratios $\beta^c_{x_k, e_k}$, $c = 1, 2$, $k = 1, \ldots, K$, to determine portions of managed lane traffic to be assigned particular destinations. The destination assignment algorithm at a given time t, for a given HOV link ending with a gate-node, is

described next. Without loss of generality, we will refer to Figure 4.2 and managed lane link 11 ending at the gate-node 1 in this description. Using Figure 4.2 as a reference, we can now formally describe the procedure for destination assignment to traffic in the managed lane link.

1. Given are vehicle counts per commodity ρ_{11}^c, $c = 1, 2, e_1, \ldots, e_K$; free flow speed v_{11}; and offramp split ratios β_{x_k,e_k}^1 and β_{x_k,e_k}^2, $k = 1, \ldots, K$.[2]

2. Initialize:

$$\begin{aligned} \tilde{\rho}_{11}^c(0) &:= \rho_{11}^c, \quad c = 1, 2, e_1, \ldots e_K; \\ k &:= 1. \end{aligned}$$

3. Assign e_k-type traffic:

$$\tilde{\rho}_{11}^{e_k}(k) = \tilde{\rho}_{11}^{e_k}(k-1) + \beta_{x_k,e_k}^1 v_{11} \tilde{\rho}_{11}^1(k-1) + \beta_{x_k,e_k}^2 v_{11} \tilde{\rho}_{11}^2(k-1); \tag{4.4}$$

$$\tilde{\rho}_{11}^1(k) = \tilde{\rho}_{11}^1(k-1) - \beta_{x_k,e_k}^1 v_{11} \tilde{\rho}_{11}^1(k-1); \tag{4.5}$$

$$\tilde{\rho}_{11}^2(k) = \tilde{\rho}_{11}^2(k-1) - \beta_{x_k,e_k}^2 v_{11} \tilde{\rho}_{11}^2(k-1). \tag{4.6}$$

4. If $k < K$, then set $k := k + 1$ and return to step 3.

5. Update the state:

$$\rho_{11}^c = \tilde{\rho}_{11}^c(K), \quad c = 1, 2, e_1, \ldots, e_K.$$

After the switches to $c = e_k$ class traffic have been done, there may be unresolved split ratios for both classes $c = 1$ and $c = 2$ at the gates (similar to the dashed split ratios in the tables in Section 4.3.2.1). These split ratios should be filled in with the same tools as those in Section 4.3.2.1: some sort of driver lane choice behavior.

4.4 Modeling Emergent Phenomena Particular to Managed Lane-Freeway Networks

As previously discussed, managed lane-freeway networks exhibit macroscopic phenomena that do not arise in situations where the managed lanes may act as traditional GP lanes [3]. These unique behaviors are caused by the interactions between the qualitatively different the flows in the GP lanes and the neighboring managed lanes. In this section, we describe simple physical models for three such behaviors: the friction effect, the smoothing effect, and the inertia effect.

[2] If a given GP segment connecting two adjacent gates has K' offramps, where $K' < K$, then assume $\beta_{x_k,e_k}^1 = \beta_{x_k,e_k}^2 = 0$ for $k \in (K', K]$.

[3] One of these effects, the smoothing effect, was even observed in a previous study (Cassidy et al., 2010) to be present in a managed lane-freeway network, but *not present on the same section of road* in another period of the day when the managed lane policy was not enforced (in that particular case, carpool/HOV lane enforcement during the peak hours)

4.4.1 Friction effect

The *friction effect* is an empirically-observed phenomenon in situations where managed lanes are relatively uncongested, but the managed-lane traffic will still slow down when the adjacent GP lanes congest and slow down (see Daganzo and Cassidy (2008); Liu et al. (2011); Thomson et al. (2012); Fitzpatrick et al. (2017), etc.). It has been hypothesized (Jang and Cassidy, 2012) that this phenomenon arises from the managed-lane drivers' fear that slower-moving vehicles will suddenly and dangerously change into the managed lane ahead of them.

We suggest modeling the friction effect based on a *feedback mechanism* that uses the difference of speeds in the parallel GP and managed lane links to scale down the flow (and therefore the speed) out of the managed lane link if necessary.

To explain the concept, we again refer to Figure 4.1 and consider parallel links 1 (GP) and 11 (managed lane). Recall that, under a first-order model (4.2), the speed of traffic in link l at time t is

$$v_l(t) = \begin{cases} \frac{\sum_{c=1}^{C}\sum_{j=1}^{N} f_{lj}^c(t)}{\sum_{c=1}\rho_l^c(t)} & \text{if } \sum_{c=1}^{c}\rho_l^c(t) > 0, \\ v_l^f(t) & \text{otherwise,} \end{cases} \tag{4.7}$$

where $v_l^f(t)$ is the theoretical free flow speed of link l at time t.

We say that the friction effect is present in managed lane link 11 (following the notation of Figure 4.1) at time t if

$$v_1(t-1) < \min\left\{v_1^f,\ v_{11}(t-1)\right\}, \tag{4.8}$$

which means that (1) the GP link is in congestion (its speed is below its current free flow speed), and (2) the speed in the GP link is less than the speed in the managed lane link. We denote this speed differential as:

$$\Delta_{11}(t) = v_{11}^f - v_1(t-1). \tag{4.9}$$

It has been observed (Jang et al., 2012) that the magnitude of the friction effect — the degree to which managed-lane drivers slow down towards the GP lane's traffic speed — depends on the physical configuration of the road. For example, less of a friction effect will be present on managed lanes that are separated from the GP lanes by a buffer zone than those that are contiguous with the GP lanes (Thomson et al., 2012; Jang et al., 2012; Fitzpatrick et al., 2017), and the presence of a concrete barrier would practically eliminate the friction effect. Other factors that may affect this magnitude include, for example, whether there is more than one managed lane, or whether there is a shoulder lane to the left of the managed lane that drivers could swerve into if necessary.

To encode this variability in the magnitude of the friction effect in managed lane link 11, we introduce $\sigma_{11} \in [0, 1]$ the *friction coefficient* of this link. The friction coefficient reflects the strength of the friction. Its value depends on the particular managed configuration and is chosen by the modeler. A value of $\sigma_{11} = 0$ means there is no friction (which may be appropriate if, perhaps, the managed lane(s) are separated from the GP lanes by a concrete

barrier), and $\sigma_{11} = 1$ means that the managed lane link speed tracks the GP link speed exactly.

When the friction effect is active (i.e., when (4.8) is true), we adjust the fundamental diagram of the managed lane link by scaling down its theoretical free flow speed $v_l^f(t)$, and propagate that change through the rest of the fundamental diagram parameters. The exact mathematical changes will of course be different for every different form of fundamental diagram. For the particular fundamental diagram discussed in Appendix A, this means adjusting the free flow speed and capacity as follows:

$$\hat{v}_{11}(t) = v_{11}^f(t) - \sigma_{11}\Delta_{11}(t); \tag{4.10}$$

$$\hat{F}_{11}(t) = \hat{v}_{11}(t)\rho_{11}^+, \tag{4.11}$$

where ρ_{11}^+ is the high critical density (see Appendix A for its definition), and using these adjusted values in the calculation of the sending function (A.1),

$$S_{11}^c(t) = \hat{v}_{11}(t)\rho_{11}^c(t)\min\left\{1, \frac{\hat{F}_{11}(t)}{\hat{v}_{11}(t)\sum_{c=1}^{C}\rho_{11}^c(t)}\right\}. \tag{4.12}$$

For the fundamental diagram of Appendix A, we must also check whether

$$\sum_{c=1}^{C}\rho_{11}^c(t) < \frac{\hat{F}_{11}(t)}{v_{11}^f - \Delta_{11}(t)} = \frac{\hat{v}_{11}(t)\rho_{11}^+(t)}{v_{11}^f - \Delta_{11}(t)}. \tag{4.13}$$

If not, then applying friction will lead to the managed lane link speed falling below the GP link speed, and the unadjusted sending function should be used (the possibility of this happening is due to the use of two critical densities to create Appendix A's fundamental diagram's "backwards lambda" shape).

Again, the exact form of (4.12), the sending function with friction, will depend on the link's original fundamental diagram model.

4.4.1.1 Comparison with related work

We are not the first to proposed modeling the friction effect by adjusting the fundamental diagram of the managed lane as a function of the state of the GP lane(s). In particular, Liu et al. (2012) consider a stretch of an HOV-equipped freeway where the two classes of lanes are separated by a buffer, and propose to fit two different sets of flow parameters for the HOV lane based on whether or not the GP lanes are congested.

Our method differs in formulation and potential uses. First, we propose a principled modification to the fundamental diagram that is based on observations of how the friction effect actually affects the managed lane's fundamental diagram, via the idea of lowering the theoretical managed lane free-flow speed. Examining (4.10) again, this formulation has an effect on the fundamental diagram's free-flow speed that is linear in the speed differential.

This formulation was chosen based on prior studies: evidence of a friction effect reducing managed lane speed in a form that is linear in the speed differential was reported in, e.g., Jang et al. (2012). This is in contrast to earlier models that switch between two distinct and discrete sets of fundamental diagram parameters based on a GP congestion threshold (i.e., a piecewise rather than linear relationship).

Our theoretically-based fundamental diagram adjustment is also motivated by the common use for macroscopic simulation as a forecasting tool. Traffic simulation tools are often used to study potential effects of proposed, but not yet developed, infrastructure construction (Kurzhanskiy and Varaiya, 2015). Moreover, the particular infrastructure investment of managed lane construction introduces the potential for additional policy choices (e.g., whether to restrict an HOV lane to vehicles with at least 2 occupants or those with at least 3 occupants, the tolling prices in a tolled express lane, etc.), each with their own feedback effects of the flow on the road, and underlying demand management. These complexities have motivated the development of new simulation tools specifically for new managed lane types (e.g., Guohui et al. (2009); Michalaka et al. (2013)). In the business and policy worlds, these complex investments with many feedback effects are evaluated through Monte Carlo simulation of principled models (Kurzhanskiy and Varaiya, 2015). Our friction effect model is motivated in part by this type of use case, where a theoretically-justified model is to be used in situations where data is unavailable due to the construction having not occurred yet. On the other hand, of course, if a traffic engineer is studying an already-existing managed lane-freeway network, where data measuring the friction effect can be collected, it makes sense to fit a friction effect model based on those data.

4.4.2 Inertia effect

At the end of Section 4.3.3.1, we mentioned that some split ratios will likely be undefined after switching $c = 2$ class vehicles to the $c = e_k$ classes. In particular, $c = 2$ class vehicles in the upstream GP link (e.g., link 1 in Figure 4.2) and remaining $c = 2$ vehicles in the upstream managed lane link (e.g. link 11 in Figure 4.2) will need to decide whether to pass through the gate or remain in their current lane. Sample tools for modeling driver lane choices such as these include the class of "logit" logistic regression models (McFadden, 1973), or dynamic split ratio solvers such as the one presented in Wright et al. (2018).

When applied to lane choice (e.g., as in Farhi et al. (2013) or Shiomi et al. (2015)), logit models produce a set of portions in $[0, 1]$, one for each lane, that sum to one. Each lane's value is the equilibrium portion of vehicles that will travel on that lane. In a dynamic simulation context, such as considered in this article, the differences between the logit model's equilibrium portions and the current distribution of vehicles across lanes at time t are used to select split ratios at time t such that the actual distribution approaches the logit equilibrium. We argue, though, that use of this sort of split ratio solver in unmodified form might be inappropriate for computing gate split ratios in the gated-access managed lane configuration. Unmodified, a value function for either the GP or managed lane link might include terms such as the link's speed of traffic, density, etc. However, often a gated-access managed lane is

separated from the GP lanes by some buffer zone or visibility-obstructing barrier that makes switching between the two links more hazardous than switching between two contiguous lanes. Therefore, if one uses a logit-based model, it would be appropriate to modify the logit model's value function such that staying in the current link (e.g., the movements (1,2) and (11,22) in Figure 4.3) has some positive value for drivers, and the gains (e.g., in travel time) for ingress/egress movements (e.g., (1,22) and (11,2) in Figure 4.3) must be of more value than the staying-in-the-lane value. We refer to this model as the *inertia effect*.

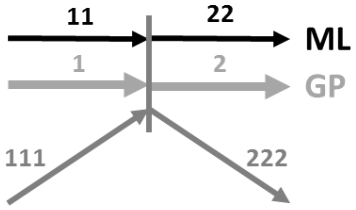


Figure 4.3: A node where some of the input links form travel facilities with some of the output links. ML = Managed Lane.

We may also incorporate the inertia effect into dynamic split ratio solvers, such as the one introduced in Wright et al. (2018). That dynamic split ratio solver is reviewed in Appendix B. At time t, this particular algorithm selects split ratios in an attempt to balance the density ratio at the next timestep $t+1$, $\sum_c \rho_l^c(t+1)/\rho_l^J$, where ρ_l^J is the jam density, or the maximum number of vehicles that link l can hold. For example, if applied to the node in Figure 4.3, this algorithm would attempt to make $\sum_c n_2^c(t+1)/\rho_l^J$ and $\sum_c \rho_{22}^c(t+1)/\rho_l^J$ as equal as possible.

Here, we modify several steps of the solver so that this equality-seeking goal is balanced with a goal towards enforcing the inertia effect. We illustrate these changes with the particular example of the node in Figure 4.3. Ensuring that the split ratio assignment algorithm gives preferences to movements (1,2) and (11,22) over (1,22) and (11,2) can be done in step 5 of the original algorithm, setting of oriented priorities. Specifically, we modify (B.4). For this particular example, the original formula gives us:

$$\begin{aligned} \gamma_{1,2}^c(k) &= \tilde{\beta}_{1,2}^c(k) + \tfrac{\overline{\beta}_1^c(k)}{2} \qquad \gamma_{1,22}^c(k) = \tilde{\beta}_{1,22}^c(k) + \tfrac{\overline{\beta}_1^c(k)}{2}; \\ \gamma_{11,2}^c(k) &= \tilde{\beta}_{11,2}^c(k) + \tfrac{\overline{\beta}_{11}^c(k)}{2} \quad \gamma_{11,22}^c(k) = \tilde{\beta}_{11,22}^c(k) + \tfrac{\overline{\beta}_{11}^c(k)}{2}. \end{aligned}$$

Since for $k=0$ $\tilde{\beta}_{i,j}^c(0)=0$, $i=1,11$, $j=2,22$, we get $\gamma_{1,2}^c(0)=\gamma_{1,22}^c(0)=\frac{\overline{\beta}_1^c(k)}{2}$ and $\gamma_{11,2}^c(0)=\gamma_{11,22}^c(0)=\frac{\overline{\beta}_{11}^c(k)}{2}$, which, according to (B.3), yields $\tilde{p}_{1,2}(0)=\tilde{p}_{1,22}(0)$ and $\tilde{p}_{11,2}(0)=\tilde{p}_{11,22}(0)$.

For each input link i that forms one lane with an output link $\hat{\jmath}$ and class c, such that $\hat{\jmath} \in V_i^c$, (B.4) can be modified as follows:

$$\gamma_{ij}^c(k) = \begin{cases} \beta_{ij}^c, & \text{if split ratio is defined a priori: } \{i,j,c\} \in \mathcal{B}, \\ \tilde{\beta}_{ij}^c(k) + \overline{\beta}_i^c(k)\lambda_i^c, & i \text{ and } j \text{ form one lane: } j=\hat{\jmath}, \\ \tilde{\beta}_{ij}^c(k) + \overline{\beta}_i^c(k)\frac{1-\lambda_i^c}{|V_i^c|-1}, & i \text{ and } j \text{ are in different lanes: } j \neq \hat{\jmath}, \end{cases} \tag{4.14}$$

where the parameter $\lambda_i^c \in \left[\frac{1}{|V_i^c|}, 1\right]$ is called the *inertia coefficient*, and indicates how strong the inertia effect is. With $\lambda_i^c = \frac{1}{|V_i^c|}$, (4.14) reduces to the original formula, (B.4). With $\lambda_i^c = 1$, all the *a priori* unassigned traffic from link i must stay in its lane — be directed to output link $\hat{\jmath}$. The choice of λ_i lies with the modeler. In the case of example from Figure 4.3, the modified formula (4.14) yields:

$$\begin{array}{ll} \gamma_{1,2}^c(k) = \tilde{\beta}_{1,2}^c(k) + \overline{\beta}_1^c(k)\lambda_1^c & \gamma_{1,22}^c(k) = \tilde{\beta}_{1,22}^c(k) + \overline{\beta}_1^c(k)(1-\lambda_1^c); \\ \gamma_{11,2}^c(k) = \tilde{\beta}_{11,2}^c(k) + \overline{\beta}_{11}^c(k)(1-\lambda_{11}^c) & \gamma_{11,22}^c(k) = \tilde{\beta}_{11,22}^c(k) + \overline{\beta}_{11}^c(k)\lambda_{11}^c, \end{array} \tag{4.15}$$

where $\lambda_1, \lambda_{11} \in \left[\frac{1}{2}, 1\right]$, and picking $\lambda_1 > \frac{1}{2}$ ($\lambda_{11} > \frac{1}{2}$) would give preference to movement (1,2) over (1,22) (and (11,22) over (11,2)).

The way of choosing λ_i^c for multiple input links is not obvious and an arbitrary choice may result in an unbalanced flow distribution among output links. Therefore, we suggest picking just one input-output pair $(\hat{\imath}, \hat{\jmath})$, and for that input link setting $\lambda_{\hat{\imath}}^c = 1$, while for other input links i setting $\lambda_i^c = \frac{1}{|V_i^c|}$, $c = 1, \ldots, C$. The input link $\hat{\imath}$ must be from the lane that is expected to have a positive net inflow of vehicles as a result of the split ratio assignment and flows computed by the node model. So,

$$\hat{\imath} = \arg\min_{i \in \hat{U}} = \frac{\sum_{c:\{i,j,c\} \in \overline{\mathcal{B}}} \overline{S}_i^c + \sum_{i:\{i,j,c\} \in \mathcal{B}} \sum_{c:\{i,j,c\} \in \mathcal{B}} \beta_{ij}^c S_i^c}{R_j}, \tag{4.16}$$

where

$$\hat{U} = \{\text{input links } i: \ \exists j, c, \text{ s.t. } j \in V_i^c \text{and } (i,j) \text{ belong to the same lane}\} \tag{4.17}$$

and j denotes the output link that is in the same lane as input link i.

For the particular example node in Figure 4.3, we need to determine whether flow from link 1 will proceed to link 2 or flow from link 11 to link 22, while other *a priori* undefined split ratios will be computed according to the split ratio assignment algorithm. If $\hat{\imath} = 11$, then $\lambda_{11} = 1$, $\lambda_1 = \frac{1}{2}$, and *a priori* unassigned traffic in the managed lane will stay in the managed lane ($\beta_{11,2}^c = 0$), while the *a priori* unassigned traffic coming from links 1 and 111, will be distributed between links 2 and 22 according to the dynamic split ratio solver.

On the other hand, if $\hat{\imath} = 1$, then $\lambda_1 = 1$, $\lambda_{11} = \frac{1}{2}$, and the *a priori* unassigned traffic in the GP lane will stay in the GP lane ($\beta_{1,22}^c = 0$), while the *a priori* unassigned traffic coming from links 11 and 111, will be distributed between links 2 and 22 according to the dynamic split ratio solver.

4.4.3 Smoothing effect

The smoothing effect is the name given to an empirically-observed phenomenon where the presence of a managed lane leads to increased capacity at bottlenecks (Menendez and Daganzo, 2007).

Cassidy et al. (2010) outline the mechanics behind the smoothing effect as follows. They observe two sites in California where freeway bottlenecks regularly occur at peak traffic (at one site, a merge with an onramp and at the other, a curved section in the road). At both sites, a managed lane exists: specifically, the leftmost lane is restricted to only HOVs during rush hour. They find that during the time period when the HOV lane policy is enforced, the discharge rate of the managed lanes at the bottleneck is increased. Specifically, they report that at the two lanes closest to the HOV lane have discharge flow increases of roughly 10-20% (Cassidy et al., 2010, Tables 1 and 2).

Cassidy et al. (2010) suggest that this increase in discharge flow is due to a reduction in lane changing induced by the presence of the managed lane. To summarize their argument, restricting the innermost lane - the furthest from on- and off-ramps and usually the one with the fastest traffic speed - to certain classes of vehicles means that fewer drivers will attempt lane changes to enter that fastest lane. This reduction in lane-changing is likely to be particularly pronounced just downstream of onramps, since, normally, some drivers that enter the freeway will want to enter the fastest lane as soon as possible, which requires multiple lane changes. Lane change maneuvers are known to have a cost to freeway performance, since a) drivers performing a lane change are effectively taking up two lanes for at least some amount of time, and b) the surrounding traffic must slow down to accommodate a lane change, potentially causing a stop-and-go wave (Jin, 2010).

A recent study (Kim and Park, 2018) has also found that on a freeway with a gated-access managed lane, onramp/offramp locations adjacent to a gate counterintuitively have a lower rate of accidents than those with no nearby gate. This result may also be related to the presence of a managed lane somehow reducing the number of unnecessary lane changes.

We propose that the smoothing effect can be elegantly captured within the node model of a macroscopic simulation (recall from Section 4.2 that the node model refers to the mathematical rule used to compute the node throughflows $f^c_{i,j}$ given the link demands and supplies at a junction). In particular, we recall the so-called "general class of node models (GCNM)" introduced by Tampère et al. (2011), that serves as the base for most modern node models. In the present work, we will not review the GCNM node models or their derivatives (e.g., Smits et al. (2015); Jabari (2016); Wright et al. (2017), and others) in depth, but will discuss a particular component of this body of work that neatly captures the same phenomenon described by the smoothing effect.

Our discussion of node models so far in this work has referred to "supply" as a property of links, that defines how many vehicles that link can accept as a function of its current occupancy and the link model. Defining what occurs in situations where there is insufficient supply to accept all upstream demand is a key component of node models, and shortcomings of previous node models in this area was a key point of contrast in the original presentation of the GCNM (Tampère et al., 2011). The scheme for distribution of scarce supply in the node model, and re-distribution of "leftover" supply if some input link is not able to use all of its initially-apportioned supply, was termed the "supply constraint interaction rule" (SCIR) in Tampère et al. (2011), and one area in which node models in the GCNM family vary is the choice of SCIR (Smits et al., 2015; Wright et al., 2017).

In prior work, we have discussed a particular SCIR choice and node model constraint variously referred to as the "conservation of turning fractions" (CTF) or "first-in-first-out" (FIFO) constraint (Wright et al., 2016, 2017). As the name suggests, a node model that includes the FIFO constraint will require that, in congested (supply-constrained) nodes, total input-link queueing occurs when any one of an input links' destination links runs out of supply. As an example to illustrate this idea, a FIFO constraint would mandate queueing in a GP link, if the GP link is not able to send to an adjacent managed lane link due to supply constraints in the managed lane link. Some recent papers (Wright et al., 2016, 2017) discuss a relaxation of the FIFO constraint that we call the "partial FIFO" constraint. It turns out that a node model that encodes the partial FIFO constraint is also one that, in the case of a managed lane-freeway network, can lead to the emergence of the smoothing effect. We will illustrate this through the following example.

4.4.3.1 Emergence of the smoothing effect under partial FIFO constraints

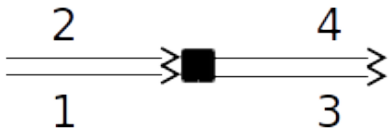


Figure 4.4: A simple merge-diverge node representing a GP lane and managed lane, with the opportunity for vehicles to change between the lanes.

Consider the example node in Figure 4.4. Here, we depict a simple managed lane-GP lane interface (with no onramps or offramps). This may represent any node in a continuous-access managed lane, or a gate node in a gated-access managed lane. For clarity, we will say that the links 1 and 3 represent the GP lane(s), and links 2 and 4 represent the managed lane(s).

As we have noted, lane changes are deleterious to road throughput, with several authors arguing they have a negative effect on *capacity* specifically (e.g., Cassidy et al. (2010); Jin (2010)). This effect, of an input link having its capacity lowered via spillback in congestion, and the degree to which the capacity drop being dependent on the amount of lane-changing, can be captured in a node model of Figure 4.4's node that incorporates the partial FIFO constraint. In this work, we will not go into the technical details of the partial FIFO constraint formulation, but will instead present a schematic example of how the smoothing effect emerges. For more specific implementation details, see Wright et al. (2017).

Table 1 presents some specific numerical values we will use in our example. In this example, we will consider only one vehicle class c, and so we omit the vehicle class index superscript. Suppose the GP lane link (links 1 and 3) has three lanes, and the managed lane link (links 2 and 4) has one lane. This is reflected in our values for the supplies and demands, with R_3 being three times that of R_4 and S_1 being four times S_2 (i.e., the GP lanes have flow at their capacity and the managed lane has flow at ¾ its capacity). We use a per-lane supply

Table 4.1: Parameters and example resulting flows for situations of heavy and no lane-changing for the example node discussed in Section 4.4.3.1

Link Information	$S_1 = 6000$	$R_3 = 6000$		
	$S_2 = 1500$	$R_4 = 2000$		
	Split Ratios		Resulting Flows	
Heavy-lane-changing situation	$\beta_{1,3} = 2/3$	$\beta_{2,3} = 0$	$f_{1,3} = 4500$	$f_{2,3} = 0$
	$\beta_{1,4} = 1/3$	$\beta_{2,4} = 1$	$f_{1,4} = 500$	$f_{2,4} = 1500$
			$\sum_{i,j} f_{i,j} = 6500$	
No-lane-changing situation	$\beta_{1,3} = 1$	$\beta_{2,3} = 0$	$f_{1,3} = 6000$	$f_{2,3} = 0$
	$\beta_{1,4} = 0$	$\beta_{2,4} = 1$	$f_{1,4} = 0$	$f_{2,4} = 1500$
			$\sum_{i,j} f_{i,j} = 7500$	

of 2000, which is a common nominal value for freeway capacity in units of vehicles per hour per lane.

Table 1 also describes two situations of lane-changing (characterized by split ratios), and the resulting flows for each situation. For the purpose of illustration (and to avoid getting bogged down in the details of the node model), we select split ratios to describe two extremes of lane-changing. In the heavy-lane-changing situation, we are supposing that one-third of the GP-lane vehicles want to enter the leftmost lane (i.e., the managed lane). This could be equivalent to saying that each and every vehicle in the leftmost GP lane is trying to enter the managed lane. In the no-lane-changing situation, we are supposing that all GP vehicles and all HOV vehicles remain in their original lane.

One item that must be clarified before we can explain how we arrive at the resulting flows for these two situations is our node model's method for portioning the downstream supply among the input links competing for it. A node model's portioning scheme makes up part of its SCIR (Tampère et al., 2011). In this example, we are trying not to get overwhelmed with the details of the node model, so we choose a very simple (and not terribly realistic) portioning scheme, where the HOV lane link is able to claim all the downstream supply it wants *first*, and then the GP lane link is only able to make use of the leftover supply (c.f. the example in (Wright et al., 2017, Section 4.2)). In a more realistic node model, the two input links' flows would be competing for the downstream supply, but again, our intent here is illustration of the emergence of the smoothing effect.

Consider the heavy-lane-changing situation. To restate what is modeled by the split ratios, all the vehicles in the leftmost GP lane are trying to enter the downstream managed lane, while all the vehicles in the right two lanes are staying in the GP lanes. However, it is clear

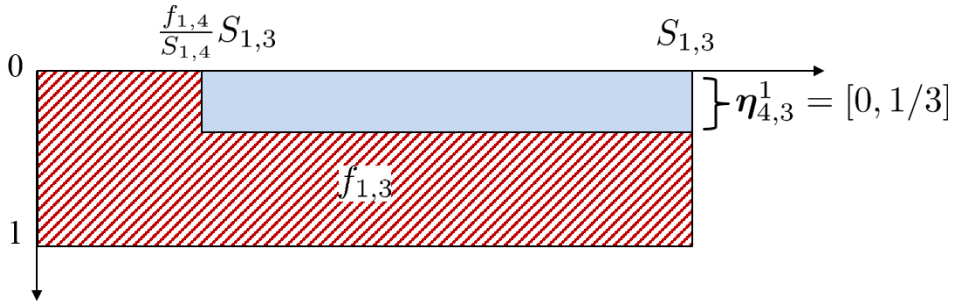


Figure 4.5: Diagram illustrating how the partial FIFO construction reduces the discharge flow of the GP lane under heavy lane changing in Section 4.4.3.1's example.

that between the vehicles trying to change lanes into the managed lane, and the vehicles already in the managed lane, the total demand exceeds the managed lane's supply. This means that the flow from the GP lane to the managed lane, $f_{1,3}$, is supply-constrained and not able to send all of its demand. The implications of this supply constraint via the partial FIFO mechanism is explained next.

Figure 4.5 presents an illustration used to explain the mechanics of the partial FIFO constraint. This type of diagram was introduced in the original presentation of the partial FIFO constraint (Wright et al., 2017, Section 3). In this figure, $\boldsymbol{\eta}^1_{4,3}$ denotes the so-called "restriction interval," that describes the portion of the flow $f_{1,3}$ that becomes blocked (stuck in a queue) when the output link 4 runs out of supply. The value of $\boldsymbol{\eta}^1_{4,3} = [0, 1/3]$ means that the leftmost third (i.e., the left lane of the three) of the GP lanes becomes queued when the managed lane runs out of downstream supply. This value is defined by noting that only this leftmost lane serves the lane-changing movement.

The area of the hashed region labeled $f_{1,3}$ defines that flow. The solid region defines the portion of the demand-limited flow $S_{1,3}$ that is lost due to the partial FIFO constraint and congestion in link 4 [4]. The vertical length of the solid region is defined by $\boldsymbol{\eta}^1_{4,3}$. The horizontal length of the solid region has a more complex definition, and is dependent on the node's supplies, demands, and the node model's supply-portioning scheme (see Wright et al. (2017) for full details). What is important in the present example is that the partial FIFO constraint says that the inability of the downstream managed lane link (link 4) to accept all the vehicles that the upstream GP lane link (link 1) wants to send it, adversely affects the discharge rate of the flow that is staying on the GP lanes. In particular, if we use our earlier-described simple supply-portioning scheme for link 4's supply (the GP lane vehicles trying to change lanes only get the leftover supply after the upstream managed lane link has

[4] To be precise, the area of the total rectangle, $S_{1,3}$, is clearly less than the maximum possible flow for $f_{1,3}$ due to supply constraint $R_3 < S_{1,3}$. The illustration being discussed only covers how much of the original supply becomes stuck in its origin link.

used all it wants), then we can find the leftmost boundary of the solid region as

$$\frac{f_{1,4}}{S_{1,4}} S_{1,3} = \frac{500}{2000} 6000 = 1000$$

which means that the total area of the hashed region, and the GP lane throughflow is

$$S_{1,3}(1) - \left(S_{1,3} - \frac{f_{1,4}}{S_{1,4}} S_{1,3} \right) |\boldsymbol{\eta}_{4,3}^1| = 6000 - \left(6000 - \frac{500}{2000} 6000 \right) \left(\frac{1}{3} \right) = 4500.$$

In other words, the heavy lane changing activated the partial FIFO constraint, which led to spillback into the GP lanes.

We can see the emergence of the smoothing effect by comparing the above heavy-lane-changing situation with the no-lane-changing situation also outlined in Table 4.1. This situation is much simpler to calculate the resulting flows, since both the GP lane vehicles and the managed lane vehicles wholly want to stay in their original link. In this situation, the partial FIFO constraint does not play any role, and both input links are able to satisfy the entirety of their demands. This leads to increased throughflows, both on the GP lanes and the node as a whole.

The dramatic increase in GP lane throughflow (1500 vehicles, or one-and-one-third of the original value of 4500) is much more than the 10-20% increase due to the smoothing effect reported by, e.g., Cassidy et al. (2010). This, though, is partially artifacts of the simplicity of our example. Our two situations had extreme differences in lane-changing split ratios, and, as also mentioned, our supply portioning method (the GP lanes only get any leftover supply in the managed lane link) were both unrealistic but useful to demonstrate that activation of a partial FIFO constraint can be modeled wholly due to lane changing. A more realistic model of either should result in more muted throughflow increases.

To summarize, the partial FIFO construction can describe a bottleneck that is created by heavy lane changing. This, of course, means that a node model with a partial FIFO construction can be used to construct a smoothing effect in macroscopic simulation if lane changes between the GP lane link and the managed lane link are reduced. To the best of our knowledge, this is the first proposed model of the smoothing effect in managed lane-freeway network simulation. The fact that the smoothing effect emerges from a previously-introduced node model is in itself also independently interesting. Future work should focus on more realistic calibration of the partial FIFO construction to empirically-observed patterns in the smoothing effect.

4.4.3.2 Comparison with related work

Other authors (e.g., Jin (2010, 2013)) have considered the problem of modeling lane-changing traffic flow. These other approaches consider how the continuum flow model (i.e., the fundamental diagram) is affected by lane-changing and weaving. In contrast, our approach described here isolates the discharge flow drop due to lane-changing into the node model.

Appealingly, our construction does not mandate a certain link model or the entirety of the node model (recall our discussion that the SCIR is still somewhat a free parameter), which fits well into the macroscopic simulation paradigm. The analysis of discretized forms of the continuum-theory lane-changing-flow models, and comparison with the already-discretized form presented here (as well as the reverse: this link-and-node construction here at a continuous limit) is an interesting area for potential future investigation.

4.5 Conclusion

In this chapter we discussed modeling procedures for two managed lane configurations: (1) full access, where special traffic can switch between the GP and the managed lanes at any node; and (2) separated, where special traffic can switch between the two lanes only at specific nodes, called gates. We have introduced models for the *friction effect* (Section 4.4.1), the *inertia effect* (Section 4.4.2), and the *smoothing effect* (Section 4.4.3). The friction effect reflects the empirically-observed drivers' fear of moving fast in the managed lane while traffic in the adjacent GP links moves slowly due to congestion. We proposed modeling this by making the fundamental diagram parameters of the HOV lanes functions of the GP lanes' traffic state. The inertia effect reflects drivers' inclination to stay in their lane as long as possible and switch only if this would obviously improve their travel condition. We gave an example of how to adjust a driver choice model to account for this effect. The smoothing effect is the name for the fact that implementing a managed lane policy leads to less lane-changing, which in turn increases road capacity. We showed that such an effect of less lane-changing leading to greater throughflows emerges in a class of "partial FIFO" node models.

Models of freeways with managed lanes feature many more parameters than ones with GP lanes, and calibrating them can be difficult. The emergent phenomenological effects create complex feedback loops such that individual parameters cannot be tuned in isolation. In chapter 5, we present an iterative learning approach for estimating some of the harder-to-estimate parameters.

Chapter 5

Macroscopic Modeling, Calibration, and Simulation of Managed Lane-Freeway Networks, Part II: Network-scale Calibration and Case Studies

5.1 Introduction

In chapter 4, we proposed several macroscopic modeling elements for the purpose of mathematically describing a freeway equipped with a managed lane facility (a "managed lane-freeway network"). That chapter discussed the road-topological considerations of modeling two parallel traffic flows, as well as mathematical descriptions for several of the complex emergent (and sometimes counteracting) phenomena that have been observed on managed lane-freeway networks, like the so-called friction (Liu et al., 2011; Jang et al., 2012; Fitzpatrick et al., 2017) and smoothing (Menendez and Daganzo, 2007; Cassidy et al., 2010; Jang and Cassidy, 2012) effects.

The present chapter discusses the actual implementation of these modeling constructions. Transportation authorities considering a significant infrastructure investment like building new roads or installing managed lane facilities will make use of simulation tools during the pre-construction planning phase (Caltrans Office of Project Development Procedures, 2011). Models let transportation planners understand the characteristics of the road network, and predict outcomes of proposed modifications before their undertaking.

The use of models for the predictive study of counterfactuals is especially relevant for managed lane deployments. A major benefit of modern managed lanes to transportation authorities is how they can enable real-time, reactive control (Obenberger, Nov/Dec 2004; Kurzhanskiy and Varaiya, 2015). Managed lanes today may be instrumented with operational

capabilities for real-time traffic control, such as dynamic, condition-responsive toll rates in tolled express lanes (Lou et al., 2011). Real-time control decisions can be studied via a well-tuned predictive model. As the traffic situation changes, multiple potential operational strategies can be evaluated to decide on the best course of action.

Before a traffic model can be used for forecasting or generating counterfactuals, it must be tuned to represent a base-case scenario, usually the existing traffic patterns. This process of model tuning is generally referred to as *calibration*. Calibration of traffic network models is known to be a difficult task (Treiber and Kesting, 2013, Chapter 16): vehicle traffic dynamics exhibit many complex and interacting nonlinear behaviors, and a model must realistically capture these behaviors to be useful for analysis or planning purposes. For a freeway-corridor-scale network, the data available for model calibration may consist of vehicle counts and mean speeds from vehicle detectors, origin-destination survey data, and/or descriptive information about the regular spatiotemporal extent of regular. A calibration process involves tuning a simulation model's parameters such that it accurately reproduces typical traffic behaviors (Treiber and Kesting, 2013). The nonlinear and chaotic nature of traffic means that calibrated parameter values typically cannot be found explicitly, and must be sought after via an iterative tuning-assessment process.

In this chapter, we present a calibration methodology for chapter 4's managed lane-freeway network modeling constructions, that integrates them into the broader macroscopic traffic model calibration loop. Much of our focus is on identifying the split ratios, the portions of vehicles that take each of several available turns between the managed lane(s) and the general-purpose (GP) lane(s). These values are of particular importance when the behavior of the managed lane is of interest, and have major effects on the resulting analysis of managed lane usage and traffic behaviors, but are not directly measurable with traditional traffic sensors. Instead, they must be found as part of the iterative calibration process such that the macroscopic traffic patterns actually observed on the managed lane's instrumentation are reproduced in the model. A managed lane-freeway network calibration procedure, in other words, is a *superset* of the calibration procedure for a typical freeway corridor model.

The remainder of this chapter is organized as follows. Section 5.2 outlines the macroscopic traffic model in question, which augments the traditional kinematic-wave macroscopic traffic model with the managed-lane-specific constructions developed in chapter 4. Calibration procedures for the managed-lane-specific components, and the integration of them into a full calibration loop, is discussed in Section 5.3. Two case studies of macroscopic modeling of managed-lane-equipped freeways in California, one with a full-access managed lane and one with a separated managed lane with gated access, are presented in Section 5.4, followed by concluding points in Section 5.5

5.2 A Managed Lane-Freeway Network Simulation Model

In this Section, we present a macroscopic simulation algorithm for managed lane-freeway networks. This can be considered a fleshing-out of the simplified first-order macroscopic simulation method briefly outlined as context in chapter 4, with extensions made by incorporating the additional items we described in the same chapter.

5.2.1 Background

"Macroscopic" traffic models are those that model traffic flows via the continuum fluid approximation (originally proposed by Lighthill and Whitham (1955a,b); Richards (1956)) as opposed to "microscopic" models that model individual vehicles (Treiber and Kesting (2013) present a relatively recent broad survey of various types of traffic models). While macroscopic models necessarily have a lower resolution than microscopic models, they make up for it in areas like shorter runtimes (making them useful for real-time operational analysis).

The present chapter considers a macroscopic model for managed lane-freeway networks, based on the family of "first-order," "kinematic wave"-type models descending from the Cell Transmission Model (Daganzo, 1994, 1995).

5.2.2 Definitions

- We have a network consisting of set of directed links $\mathcal{L}$, representing segments of road, and a set of nodes $\mathcal{N}$, which join the links.
  - A node always has at least one incoming link and one outgoing link.
  - A link may have an upstream node, a downstream node, or both.
- We have C different vehicle classes traveling in the network, with classes indexed by $c \in \{1, \ldots, C\}$.
- Let $t \in \{0, \ldots, T\}$ denote the simulation timestep.
- In addition, while we have not covered them here, the modeler may optionally choose to define *control inputs* to the simulated system that modify system parameters or the system state. Such control inputs may represent operational traffic control schemes such as ramp metering, changeable message signs, a variable managed-lane policy, etc. In the context of this chapter, we suggest including the parameterization of the friction effect and the class-switching construction of the separated-access managed lane model to be discussed in section 5.3.2 as control actions.

5.2.2.1 Link model definitions

Traditionally, the mathematical equations that govern the traffic flow in the links are called the "link model."

- For each link $l \in \mathcal{L}$, let there be a time-varying C-dimensional state vector $\vec{n}_l(t)$, which denotes the density of the link of each of the C vehicle classes at time t.
  - Each element of this vector, $\rho_l^c(t)$, updates between timestep t and timestep $t+1$ according to (5.1).
  - Also define $\rho_{l,0}^c$ for all l, c, the initial condition of the system.
- We define three types of links:
  - Ordinary links are those links that have both beginning and ending nodes.
  - Origin links are those links that have only an ending node. These links represent the roads that vehicles use to enter the network.
  - Destination links are those links that have only a beginning node. These links represent the roads that vehicles use to exit the network.
- For each link $l \in \mathcal{L}$, define the actual parametric "link model" that computes the per-class demands $S_l^c(t) \leq \rho_l^c(t)$ (the amount of vehicles that want to exit the link at time t) and link supply $R_l(t)$ (the amount of vehicles that link l can accept at time t) as functions of t and $\vec{n}_l(t)$. The particular link model equations is a modeling choice, and many authors have proposed different versions. Appendix A describes a particular example link model that will be used in the example simulations in Section 5.4.
- For each origin link $l \in \mathcal{L}$, define a C-dimensional time-varying vector, $\vec{d_l}(t)$, where the c-th element $d_l^c(t)$ denotes the exogenous demand of class c into the network at link l.

5.2.2.2 Node model definitions

Analogous to the link model, the equations that govern the traffic flows through nodes (i.e., between links) are called the "node model."

- For each node $\nu \in \mathcal{N}$, let $i \in \{1, \ldots, M_\nu\}$ denote the incoming links and $j \in \{1, \ldots, N_\nu\}$ denote the outgoing links.
- For each node, define $\{\beta_{ij}^c(t) : \sum_j \beta_{ij}^c(t) = 1 \ \forall i, c\}$ the time-varying split ratios for each triplet $\{i, j, c\}$. Each split ratio may be fully defined, partially defined, or fully undefined, with the undefined split ratios typically being the ones that specify the crossflows between the managed lane(s) and the GP lane(s).

- For each node with both a managed lane link and a GP link exiting it, define a (state-dependent) split ratio solver for the managed lane-eligible vehicles. We suppose that the decision of eligible vehicles of whether or not to change between the two lane types is dependent on the local traffic conditions. Appropriate methods for determining the desired lane type as a function of the current state may include logit-style discrete choice methods (Farhi et al., 2013) or dynamic-system-based models such as the one proposed in Wright et al. (2018) and reviewed in Appendix B.
- For each node, define a "node model" that, at each time t, takes its incoming links' demands $S_i^c(t)$ and split ratios $\beta_{ij}^c(t)$, its outgoing links' supplies $R_j(t)$, and other nodal parameters, and computes the flows $f_{ij}^c(t)$. As with the link model, the particular node model equations is a modeling choice, and much literature exists defining and analyzing different node models. In this chapter, we refer to a specific node model with a relaxed first-in-first-out (FIFO) construction that has additional parameters $\boldsymbol{\eta}_{j'j}^i(t)$ (mutual restriction intervals) and $p_i(t)$ (the incoming links' priorities). Note also that a node model with the relaxed FIFO construction can also be used to produce the smoothing effect of managed lanes, as discussed in depth in chapter 4.

5.2.2.3 State update equation definitions

These equations define the time evolution of the states from time t to time $t+1$.

- All links $l \in \mathcal{L}$ update their states according to the equation

$$\rho_l^c(t+1) = \rho_l^c(t) + \frac{1}{L_l}\left(f_{l,\text{in}}^c(t) - f_{l,\text{out}}^c(t)\right) \quad \forall c \in \{1,\dots,C\}, \tag{5.1}$$

  which is a slightly generalized form of a traditional multi-class CTM update.

- For all ordinary and destination links,

$$f_{l,\text{in}}^c(t) = \sum_{i=1}^{M_\nu} f_{il}^c(t), \tag{5.2}$$

  where ν is the beginning node of link l.

- For all origin links,

$$f_{l,\text{in}}^c(t) = d_l^c(t). \tag{5.3}$$

- For all ordinary and origin links,

$$f_{l,\text{out}}^c(t) = \sum_{j=1}^{N_\nu} f_{lj}^c(t), \tag{5.4}$$

  where ν is the ending node of link l.

- For all destination links,

$$f_{l,\text{out}}^c(t) = S_l^c(t). \tag{5.5}$$

5.2.3 Simulation algorithm

1. Initialization:

$$\begin{aligned} \rho_l^c(0) &:= \rho_{l,0}^c \\ t &:= 0, \end{aligned}$$

   for all $l \in \mathcal{L}$, $c \in \{1, \ldots, C\}$.

2. Perform all control inputs that have been (optionally) specified by the modeler. For our purposes, this includes:

   a) For each managed lane link, modify the sending function of the link model in accordance with the friction effect model. This was discussed in section 4.4.1. Recall that for the particular link model of appendix A, our friction effect model is to modify the sending function via equations (4.7), (4.8), (4.11), and (4.9).

   b) For each managed lane link whose downstream node is a gate node in a gated-access managed lane configuration, perform class switching to ensure that vehicles realistically leave the managed lane to take a downstream offramp, as detailed in chapter 4.

3. For each link $l \in \mathcal{L}$ and commodity $c \in \{1, \ldots, C\}$, compute the demand, $S_l^c(t)$ using the link's link model.

4. For each ordinary and destination link $l \in \mathcal{L}$, compute the supply $R_l(t)$ using the link model. For origin links, the supply is not used.

5. For each node $\nu \in \mathcal{N}$ that has one or more undefined split ratios $\beta_{ij}^c(t)$, use the node's split ratio solver to complete a fully-defined set of split ratios. Note that if an inertia effect model is being used, the modified split ratio solver, e.g. the one described in Appendix B, should be used where appropriate.

6. For each node $\nu \in \mathcal{N}$, use the node model to compute throughflows $f_{ij}^c(t)$ for all i, j, c.

7. For every link $l \in \mathcal{L}$, compute the updated state $\vec{n}_l(t+1)$:

   - If l is an ordinary link, use (5.1), (5.2), and (5.4).
   - If l is an origin link, use (5.1), (5.3), and (5.4)
   - If l is a destination link, use (5.1), (5.2), and (5.5).

8. If $t = T$, then stop. Otherwise, increment $t := t + 1$ and return to step 2.

5.3 Calibrating the Managed Lane-Freeway Network Model

Typically, a traffic modeler will have some set of data collected from traffic detectors (e.g., velocity and flow readings), and will create a network topology with parameter values that allow the model to reproduce these values in simulation. Then, the parameters can be tweaked to perform prediction and analysis. For our managed lane-freeway networks, the parameters of interest are:

1. Link model parameters for each link (also called "fundamental diagram" parameters, referring to the graph drawn by the sending and receiving functions as functions of density). Calibration of a link model is typically agnostic to the node model and network topology, and there exists an abundant literature on this topic. For the purposes of the simulations in this work, we used the method of Dervisoglu et al. (2009), but any other method is appropriate.

2. Percentage of special (that is, able to access the managed lane) vehicles in the traffic flow entering the system. This parameter depends on, e.g., the time of day and location as well as on the type of managed lane. It could be roughly estimated as a ratio of the managed lane vehicle count to the total freeway vehicle count during periods of congestion at any given location (supposing of course that congestion in the GP lanes will lead the eligible vehicles to select the managed lane to avoid this congestion).

3. Inertia coefficients. These parameters affect only how traffic of different classes mixes in different links, but they have no effect on the total vehicle counts produced by the simulation.

4. Friction coefficients. How to tune these parameters is an open question. In Jang and Cassidy (2012) the dependency of a managed lane's speed on the GP lane speed was investigated under different densities of the managed lane, and the presented data suggests that although the correlation between the two speeds exists, it is not overwhelmingly strong, below 0.4. Therefore, we suggest setting friction coefficients to values not exceeding 0.4.

5. Mutual restriction intervals for the partial FIFO constraint. It is also an open question how to estimate mutual restriction intervals from the measurement data. See the discussion in 4.3.2 for some guidelines. Note also that the choice of restriction intervals govern the magnitude of the smoothing effect.

6. Offramp split ratios.

Calibrating a traffic model, or identifying the best values of its parameters to match real-world data, is typically an involved process for all but the simplest network topologies.

In particular, once we consider more than a single, unbroken stretch of freeway, the nonlinear nature and network effects of these systems means that estimating each parameter in isolation might lead to unpredictable behavior. Instead, nonlinear and/or non-convex optimization techniques such as genetic algorithms (Poole and Kotsialos, 2012), particle swarm methods (Poole and Kotsialos, 2016), and others (Ngoduy and Maher (2012); Fransson and Sandin (2012), etc.) are employed.

In the managed lane-freeway networks we have discussed, the key unknown parameters we have introduced are the offramp split ratios, item 6, which may be particularly hard to estimate as they are typically time-varying and explicitly represent driver behavior, rather than physical parameters of the road. Estimating the values of the other parameters can be done with one of many methods in the literature. The remainder of this section describes iterative methods for identification of the offramp split ratios for both the full-access and gated-access topological configurations, both of which were introduced in chapter 4.

5.3.1 Split ratios for a full-access managed lane

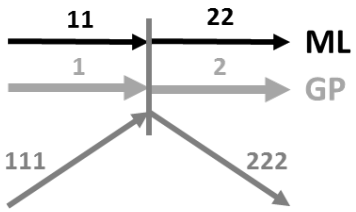


Figure 5.1: A node where some of the input links form travel facilities with some of the output links. ML = Managed Lane.

Consider a node, one of whose output links is an offramp, as depicted in Figure 5.1. We shall make the following assumptions.

1. The total flow entering the offramp, $\hat{f}^{in}_{222}$, at any given time is known (from measurements) and is not restricted by the offramp supply: $\hat{f}^{in}_{222} < R_{222}$.

2. The portions of traffic sent to the offramp from the managed lane and from the GP lane at any given time are equal: $\beta^c_{1,222} = \beta^c_{11,222} \triangleq \beta$, $c = 1, \ldots, C$.

3. None of the flow coming from the onramp (link 111), if such flow exists, is directed toward the offramp. In other words, $\beta^c_{111,222} = 0$, $c = 1, \ldots, C$.

4. The distribution of flow portions not directed to the offramp between the managed lane and the GP output links is known. This can be written as: $\beta^c_{ij} = (1 - \beta)\delta^c_{ij}$, where $\delta^c_{ij} \in [0, 1]$, as well as $\beta_{111,j}$, $i = 1, 11$, $j = 2, 22$, $c = 1, \ldots, C$, are known.

5. The demand S^c_i, $i = 1, 11, 111$, $c = 1, \ldots, C$, and supply R_j, $j = 2, 22$, are given.

At any given time, β is unknown and is to be found.

If β were known, the node model would compute the input-output flows, in particular, $f_{i,222} = \sum_{c=1}^{C} f_{i,222}^{c}$, $i = 1, 11$. Define

$$\psi(\beta) = f_{1,222} + f_{11,222} - \hat{f}_{222}^{in}. \tag{5.6}$$

Our goal is to find β from the equation

$$\psi(\beta) = 0, \tag{5.7}$$

such that $\beta \in \left[\frac{\hat{f}_{222}^{in}}{S_1+S_{11}}, 1\right]$, where $S_i = \sum_{c=1}^{C} S_i^c$. Obviously, if $S_1 + S_{11} < \hat{f}_{222}^{in}$, the solution does not exist, and the best we can do in this case to match $\hat{f}_{222}^{in}$ is to set $\beta = 1$, directing all traffic from links 1 and 11 to the offramp.

Suppose now that $S_1 + S_{11} \geq \hat{f}_{222}^{in}$. For any given $\hat{f}_{222}^{in}$, we assume $\psi(\beta)$ is a monotonically increasing function of β (this assumption is true for the particular node model of Wright et al. (2017)). Moreover, $\psi\left(\frac{\hat{f}_{222}^{in}}{S_1+S_{11}}\right) \leq 0$, while $\psi(1) \geq 0$. Thus, the solution of (5.7) within the given interval exists and can be obtained using the *bisection method*.

The algorithm for finding β follows.

1. Initialize:

$$\begin{aligned} \underline{b}(0) &:= \frac{\hat{f}_{222}^{in}}{S_1 + S_{11}}; \\ \overline{b}(0) &:= 1; \\ k &:= 0. \end{aligned}$$

2. If $S_1 + S_{11} \leq \hat{f}_{222}^{in}$, then are not enough vehicles to satisfy the offramp demand. Set $\beta = 1$ and stop.

3. Use the node model with $\beta = \underline{b}(0)$ and evaluate $\psi(\beta)$. If $\psi(\underline{b}(0)) \geq 0$, then set $\beta = \underline{b}(0)$ and stop.

4. Use the node model with $\beta = \frac{\underline{b}(k)+\overline{b}(k)}{2}$ and evaluate $\psi(\beta)$. If $\psi\left(\frac{\underline{b}(k)+\overline{b}(k)}{2}\right) = 0$, then set $\beta = \frac{\underline{b}(k)+\overline{b}(k)}{2}$ and stop.

5. If $\psi\left(\frac{\underline{b}(k)+\overline{b}(k)}{2}\right) < 0$, then update:

$$\begin{aligned} \underline{b}(k+1) &= \frac{\underline{b}(k) + \overline{b}(k)}{2}; \\ \overline{b}(k+1) &= \overline{b}(k). \end{aligned}$$

Else, update:

$$\begin{aligned} \underline{b}(k+1) &= \underline{b}(k); \\ \overline{b}(k+1) &= \frac{\underline{b}(k) + \overline{b}(k)}{2}. \end{aligned}$$

6. Set $k := k + 1$ and return to step 4.

Here, $\underline{b}(k)$ represents the lower bound of the search interval at iteration k and $\overline{b}(k)$ the upper bound.

5.3.2 Split ratios for the separated managed lane with gated access

The configuration of a node with an offramp as one of the output links is simpler in the case of a separated managed lane, as shown in Figure 5.2. Here, traffic cannot directly go from the managed lane to link 222, and, thus, we have to deal only with the 2-input-2-output node. There is a caveat, however. Recall from the discussion of gated-access managed lanes in chapter 4 that in the separated managed lane case we have destination-based traffic classes, and split ratios for destination-based traffic are fixed due to being determined at an outer loop level.

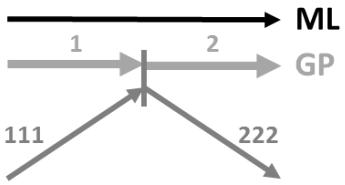


Figure 5.2: A node with a GP link and an onramp as inputs, and a GP link and an offramp as outputs.

We shall make the following assumptions:

1. The total flow entering the offramp, $\hat{f}^{in}_{222}$, at any given time is known (from measurements) and is not restricted by the offramp supply: $\hat{f}^{in}_{222} < R_{222}$.

2. All the flow coming from the onramp (link 111), if such flow exists, is directed toward the GP link 2. In other words, $\beta^c_{111,2} = 1$ and $\beta^c_{111,222} = 0$, $c = 1, \ldots, C$.

3. The demand S^c_i, $i = 1, 111$, $c = 1, \ldots, C$, and supply R_2 are given.

4. We denote the set of destination-based classes as $\mathcal{D}$. The split ratios β^c_{1j} for $c \in \mathcal{D}$ are known. Let the split ratios $\beta^c_{1j} = \beta$ for $c \in \{1, \ldots, C\} \setminus \mathcal{D}$, where β is to be determined (i.e., we assume all non-destination-based classes exit at the same rate).

The first three assumptions here reproduce assumptions 1, 3 and 5 made for the full-access managed lane case. Assumption 4 is a reminder that there is a portion of traffic flow that we cannot direct to or away from the offramp, but we have to account for it.

Similarly to the full-access managed lane case, we define the function $\psi(\beta)$:

$$\psi(\beta) = \sum_{c \in \overline{\mathcal{D}}} f^c_{1,222} + \sum_{c \in \mathcal{D}} f^c_{1,222} - \hat{f}^{in}_{222}, \tag{5.8}$$

where $f^c_{1,222}$, $c = 1, \ldots, C$ are determined by the node model. The first term of the right-hand side of (5.8) depends on β. As before, we assume $\psi(\beta)$ is a monotonically increasing function. We look for the solution of equation (5.7) on the interval $[0, 1]$. This solution exists iff $\psi(0) \leq 0$ and $\psi(1) \geq 0$. The algorithm for finding β is the same as the one presented in the previous section, except that $\underline{b}(0)$ should be initialized to 0, and S_{11} is to be assumed 0.

5.3.3 An iterative full calibration process

For the purposes of the simulations presented in the following Section, we placed the iterative split ratio identification methods of Sections 5.3.1 and 5.3.2 within a larger iterative loop for the remaining parameters. The model calibration follows the flowchart shown in Figure 5.3.

1. We start by assembling the available measurement data. Fundamental diagrams are assumed to be given. Mainline and onramp demand are specified per 5-minute periods together with the special vehicle portion parameter indicating the fraction of the input demand that is able to access the managed lane. Initially, we do not know offramp split ratios as they cannot be measured directly. Instead, we use some arbitrary values to represent them and call these values "initially guessed offramp split ratios". Instead of the offramp split ratios, we have the measured flows actually observed on the offramps, which we refer to as *offramp demand*.

2. We run our network simulation outlined in Section 5.2.3 for the entire simulation period. At this point, in step 5 of the simulation, the *a priori* undefined split ratios between traffic in the GP and in the managed lanes are assigned using a split ratio solver.

3. Using these newly-assigned split ratios, we run our network simulation again, only this time, instead of using the initially guessed offramp split ratios, we compute them from the given offramp demand as described in Sections 5.3.1 and 5.3.2. As a result of this step, we obtain new offramp split ratios.

4. Now we run the network simulation as we did originally, in step 2, only this time with new offramp split ratios, and record the simulation results — density, flow, speed, as well as performance measures such as vehicle miles traveled (VMT) and vehicle hours traveled (VHT).

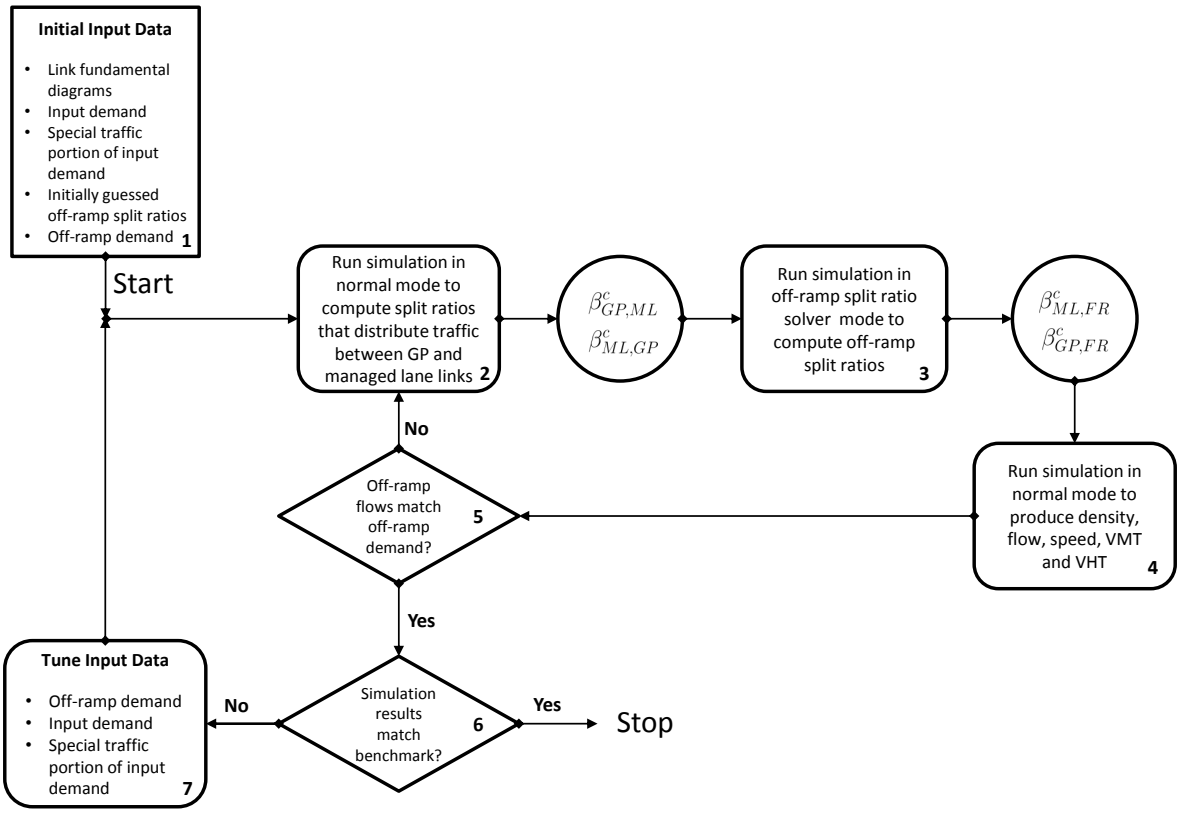


Figure 5.3: Calibration workflow.

5. Check if the resulting offramp flows match the offramp demand. If yes, proceed to step 6, otherwise, repeat steps 2-5. In our experience (i.e., the case studies in the following Section), it takes the process described in steps 2-5 no more than two iterations to converge.

6. Evaluate the simulation results:

   - correctness of bottleneck locations and activation times;
   - correctness of congestion extension at each bottleneck;
   - correctness of VMT and VHT.

   If the simulation results are satisfactory, stop. Otherwise, proceed to step 7.

7. Tune/correct input data in the order shown in block 7 of Figure 5.3.

5.4 Simulation Results

5.4.1 Full-access managed lane case study: Interstate 680 North

We consider a 26.8-mile stretch of I-680 North freeway in Contra Costa County, California, from postmile 30 to postmile 56.8, shown in Figure 5.4, as a test case for the full-access managed lane configuration. This freeway's managed lane facilities are split into two segments whose beginning and end points are marked on the map. The first segment is a high-occupancy-tolled (HOT) lane, which allows free entry to vehicles with two or more passengers and tolled entry to single-occupancy vehicles (Metropolitan Transportation Commission, 2019), The second segment has an HOV lane is 4.5 miles long. There are 26 onramps and 24 offramps. The HOV lane is active from 5 to 9 AM and from 3 to 7 PM. The rest of the time, the HOV lane is open to all traffic, and behaves as a GP lane.

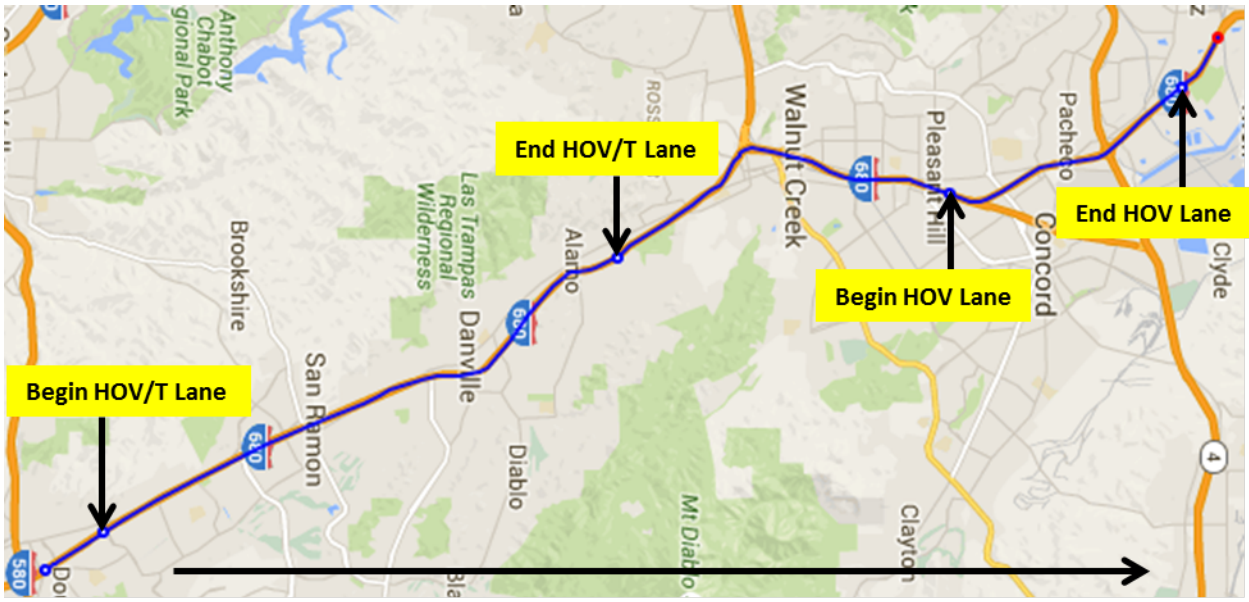


Figure 5.4: Map of I-680 North in Contra Costa County.

To build the model, we used data collected for the I-680 Corridor System Management Plan (CSMP) study (System Metrics Group, Inc., 2015). The bottleneck locations as well as their activation times and congestion extension were identified in that study using video monitoring and tachometer vehicle runs. On- and offramp flows were given in 5-minute increments. For the purposes of our model, we do not consider tolling dynamics, and instead assume that managed lane-eligible vehicles incur no cost to access the managed lane. We assume that the managed lane-eligible portion of the input demand is 15%. The model was calibrated to a typical weekday, as suggested in the I-680 CSMP study.

For this simulation, we used the fundamental diagram described in Appendix A, with parameters as follows:

- The capacity of the ordinary GP lane is 1,900 vehicles per hour per lane (vphl);
- The capacity of the auxiliary GP lane is 1,900 vphl;
- The capacity of the managed lane is 1,800 vphl while active and 1,900 vphl when it behaves as a GP lane;
- The free flow speed varies between 63 and 70 mph — these measurements came partially from the California Performance Measurement System (PeMS) (PeMS, 2019) and partially from tachometer vehicle runs.
- The congestion wave speed for each link was taken as 1/5 of the free flow speed.

The modeling results are presented in Figures 5.5, 5.6 and 5.7 showing density, flow and speed contours, respectively, in the GP and the managed lanes. In each plot, the top contour corresponds to the managed lanes, and the bottom to the GP lanes. In all the plots traffic moves from left to right along the "Absolute Postmile" axis, while the vertical axis represents time. Bottleneck locations and congestion areas identified by the I-680 CSMP study are marked by blue boxes in GP lane contours. The managed lane does not get congested, but there is a speed drop due to the friction effect. The friction effect, when vehicles in the managed lane slow down because of the slow moving GP lane traffic, can be seen in the managed lane speed contour in Figure 5.7.

Figure 5.8 shows an example of how well the offramp flow computed by the simulation matches the target, referred to as *offramp demand*, as recorded by the detector on the offramp at Crow Canyon Road. We can see that in the beginning and in the end of the day, the computed flow falls below the target (corresponding areas are marked with red circles). This is due to the shortage of the mainline traffic in the simulation — the offramp demand cannot be satisfied.

Finally, Table 5.1 summarizes the performance metrics — vehicle miles traveled (VMT), vehicle hours traveled (VHT) and delay in vehicle-hours — computed by simulation versus those collected in the course of the I-680 CSMP study. Delay is computed for vehicles with speed below 45 mph.

Performance Metric	Simulation result	Collected data
GP Lane VMT	1,687,618	-
Managed Lane VMT	206,532	-
Total VMT	1,894,150	1,888,885
GP Lane VHT	27,732	-
Managed Lane VHT	3,051	-
Total VHT	30,783	31,008
GP Lane Delay (hr)	2,785	-
Managed Lane Delay (hr)	6	-
Total Delay (hr)	2,791	2,904

Table 5.1: Performance metrics for I-680 North.

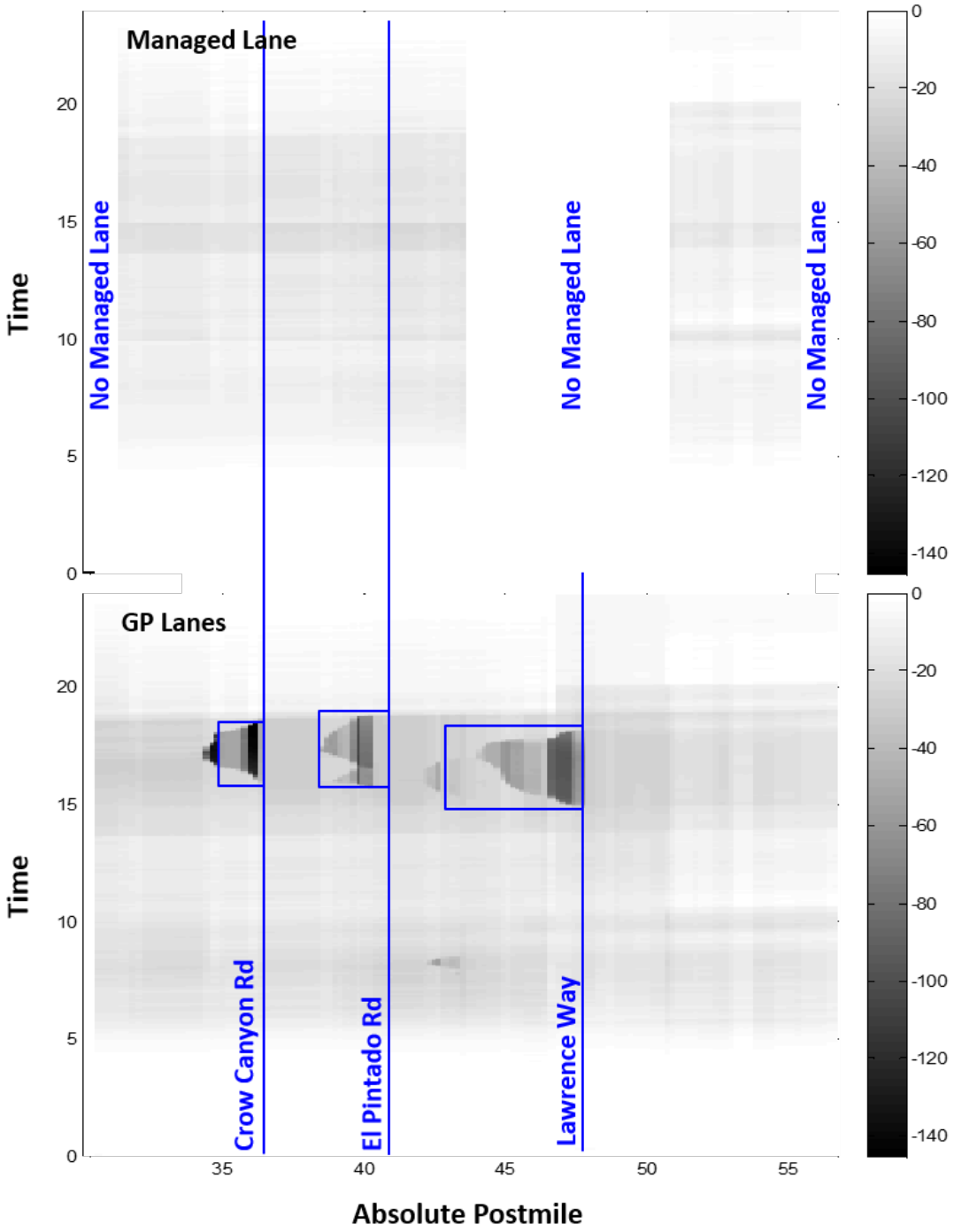


Figure 5.5: I-680 North density contours for GP and managed lanes produced by simulation. Density values are given in vehicles per mile per lane. Blue boxes on the GP lane speed contour indicate congested areas as identified by the I-680 CSMP study.

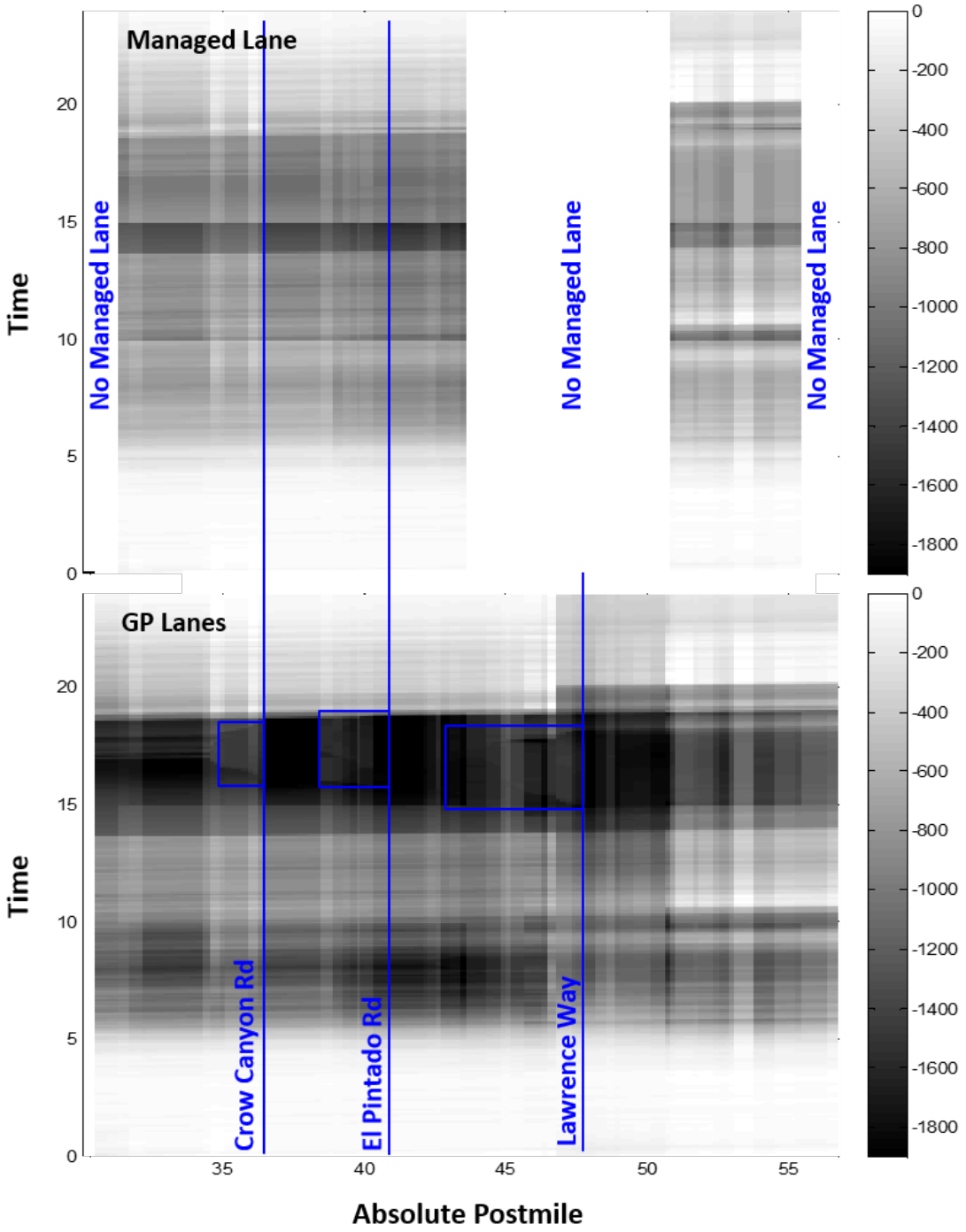


Figure 5.6: I-680 North flow contours for GP and managed lanes produced by simulation. Flow values are given in vehicles per hour per lane. Blue boxes on the GP lane speed contour indicate congested areas as identified by the I-680 CSMP study.

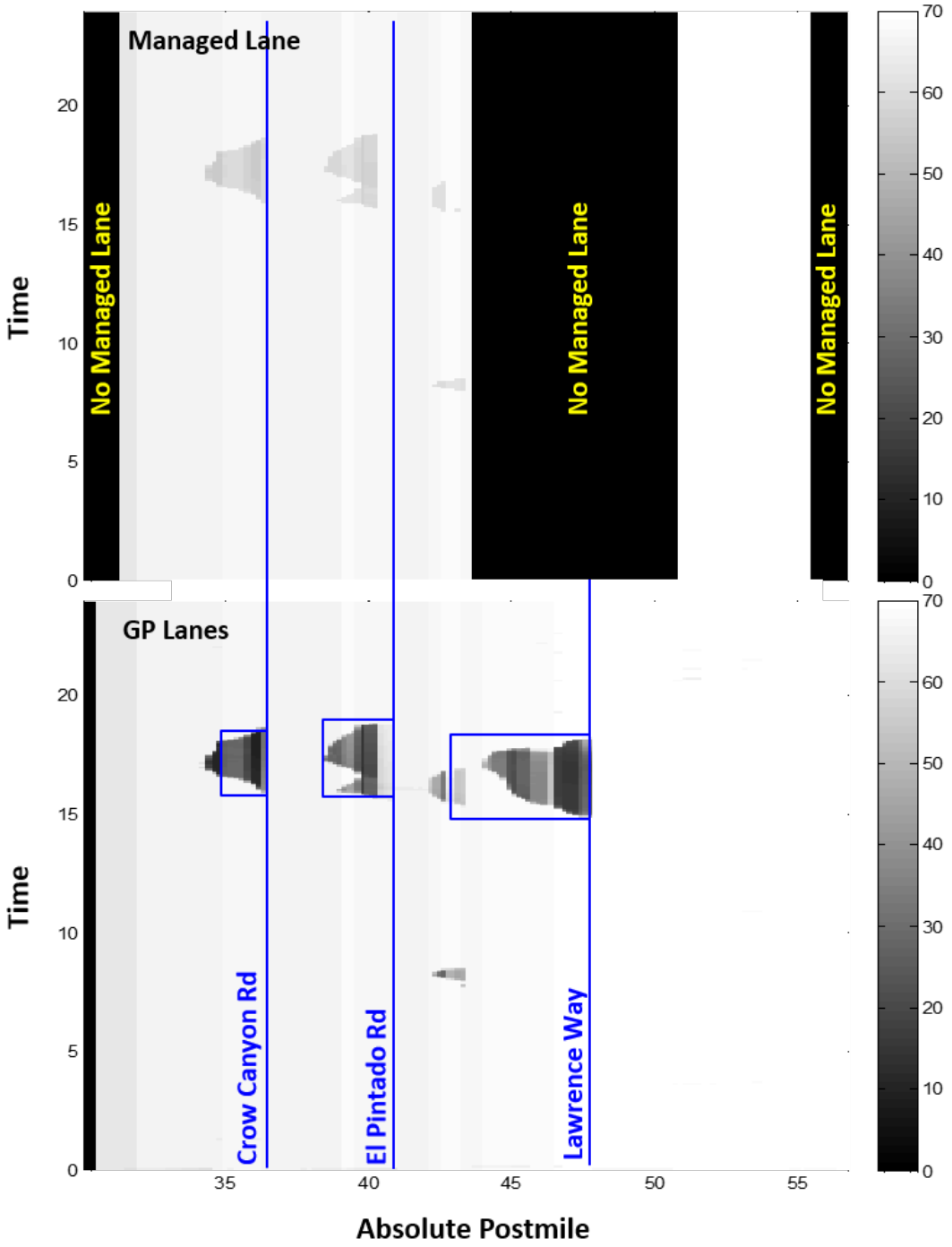


Figure 5.7: I-680 North speed contours for GP and managed lanes produced by simulation. Speed values are given in miles per hour. Blue boxes on the GP lane speed contour indicate congested areas as identified by the I-680 CSMP study.

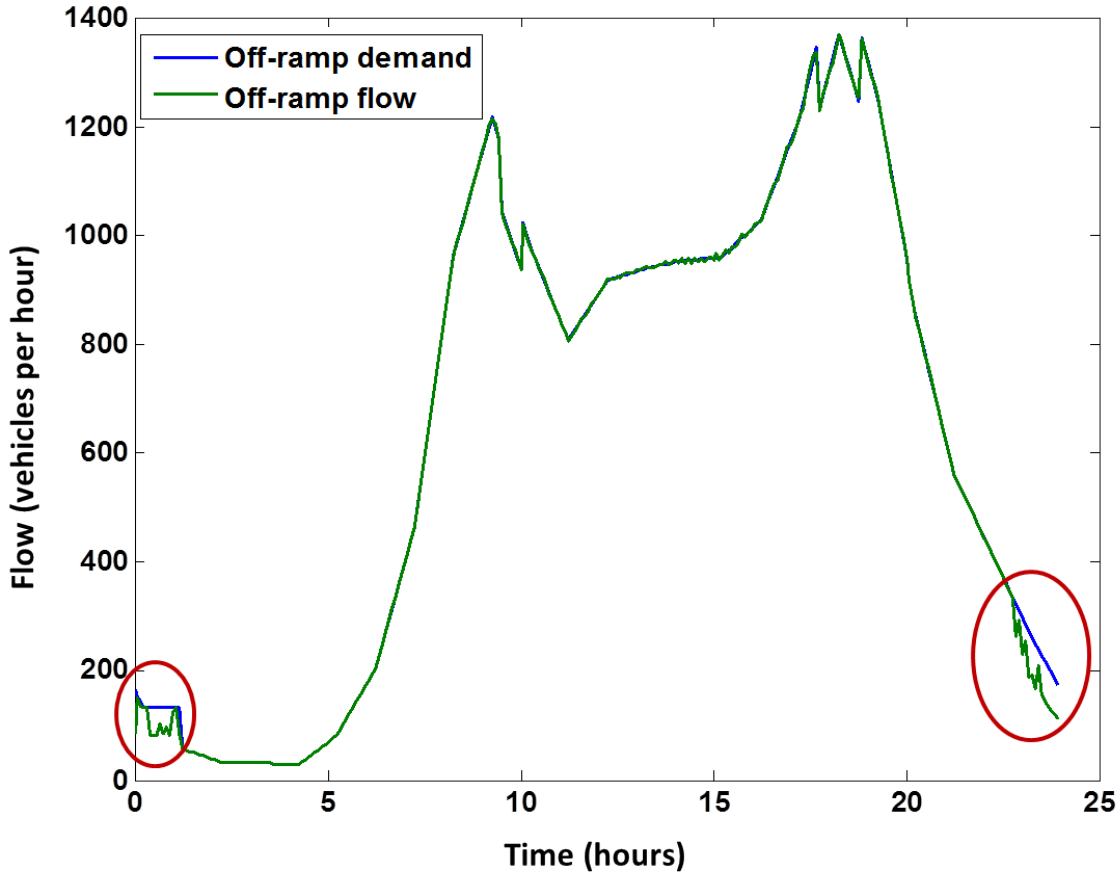


Figure 5.8: Flow at the Crow Canyon Road offramp over 24 hours — collected (offramp demand) vs. computed by simulation (offramp flow).

5.4.2 Gated-access managed lane case study: Interstate 210 East

We consider a 20.6-mile stretch of SR-134 East/ I-210 East in Los Angeles County, California, shown in Figure 5.9, as a test case for the separated managed lane configuration. This freeway's managed lane is also an HOV lane. This freeway stretch consists of 3.9 miles of SR-134 East from postmile 9.46 to postmile 13.36, which merges into 16.7 miles of I-210 East from postmile 25 to postmile 41.7. Gate locations where traffic can switch between the GP and the HOV lanes are marked on the map. At this site, the HOV lane is always active. There are 28 onramps and 25 offramps. The largest number of offramps between two gates is 5. Thus, our freeway model has 7 vehicle classes - LOV (low-occupancy vehicles; not managed lane-eligible), HOV (managed lane-eligible) and 5 destination-based.

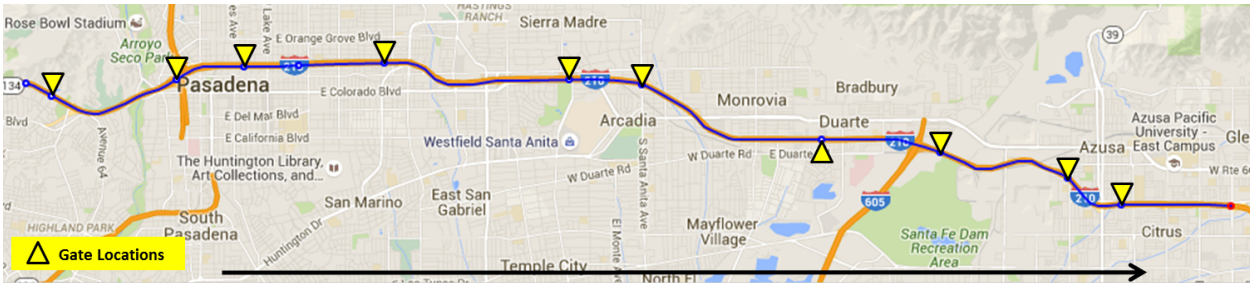


Figure 5.9: Map of SR-134 East/ I-210 East freeway in Los Angeles County.

To build the model, we used PeMS data for the corresponding segments of the SR-134 East and I-210 East for Monday, October 13, 2014 (PeMS, 2019). Fundamental diagrams were calibrated using PeMS data following the methodology of Dervisoglu et al. (2009). As in the I-680 North example, we assume that the managed lane-eligible portion of the input demand is 15%.

The modeling results are presented in Figures 5.10, 5.11 and 5.12 showing density, flow and speed contours, respectively, in the GP and the managed lanes. In each plot, the top contour corresponds to the managed lanes, and the bottom to the GP lanes. As before, in all the plots traffic moves from left to right along the "Absolute Postmile" axis, while the vertical axis represents time. The managed lane does not get congested. Dashed blue lines on the contour plots indicate managed gate locations.

Figure 5.13 shows the PeMS speed contours for the SR-134 East/ I-210 East GP and managed lanes that were used as a target for our simulation model. In these plots, traffic also travels from left to right, with the horizontal axis representing postmiles, while the vertical axis represents time.

Figure 5.14 shows an example of how well the offramp flow computed by the simulation matches the target, referred to as *offramp demand*, as recorded by the detector on the offramp at North Hill Avenue. The simulated offramp flow matches the offramp demand fairly closely. Similar results were found for the other offramps.

Finally, Table 5.2 summarizes the performance metrics — VMT, VHT and delay — computed by simulation versus those values obtained from PeMS. The PeMS data come from both SR-134 East and I-210 East, and VMT, VHT and delay values are computed as sums of the corresponding values from these two freeway sections. Delay values are computed in vehicle-hours for those vehicles traveling slower than 45 mph.

Performance metric	Simulation result	PeMS data
GP Lane VMT	2,017,322	-
Managed Lane VMT	378,485	-
Total VMT	2,395,807	414,941 + 2,006,457 = 2,421,398
GP Lane VHT	33,533	-
Managed Lane VHT	6,064	-
Total VHT	39,597	6,416 + 36,773 = 43,189
GP Lane Delay (hr)	3,078	-
Managed Lane Delay (hr)	584	-
Total Delay (hr)	3,662	1 + 3,802 = 3,803

Table 5.2: Performance metrics for SR-134 East/ I-210 East.

5.4.3 Discussion

As we mention in section 5.3, a common calibration goal for traffic modelers is to create a simulation that can accurately recreate a "base case" of a typical real-world day's traffic patterns; this calibrated "base case" can then be tweaked to predict the system-wide effects of counterfactual events such as a global demand increase, a localized lane closure, etc.

One traditional criterion for evaluating whether a freeway model captures this "base case" is whether the simulation predicts congestion at the same time and the same place as in real-world data (and, if the dynamic behaviors such as the rate of the buildup of the queue and the rate of queue discharge are similar). In this congestion-locality criterion, our simulation case studies perform well. For the case study of I-680 North, examining Figures 5.5 through 5.7 we see that our models predict congestion originating at the bottleneck locations and propagating to the extents identified in the CSMP study (System Metrics Group, Inc., 2015). For the SR-134 East / I-210 East, we can compare the reference PeMS loop data speed contour (Figure 5.13) to our simulations (Figures 5.10-5.12). The PeMS data show and out simulations predict congestion in both the managed lane and the GP lanes that originates at a bottleneck at N. Azusa Avenue at roughly 15:00, propagates backward in space until reaching its maximum spatial extent roughly at postmile 29 at about 17:00, and then recedes until fully dissipating at roughly 19:00. The managed lane data from the PeMS dataset (Figure 5.13) has more identifiable wavefronts propagating backward and receding forward than the GP lanes, and those wavefront's speeds agree somewhat with those predicted by the

simulation (unfortunately, a definitive value for the true wavefront speed is difficult to obtain due to the low resolution of the PeMS data).

Beyond the somewhat qualitative examination of whether the macroscopic congestion matters match reality, we can also evaluate quantitatively whether our simulation fits the available measurements from the site. Figures 5.8 and 5.14 show how our model matched the offramp flow at the two identified bottlenecks of the case study sites. Both figures show agreement between model and data. Note that this "offramp demand" was an explicit calibration target (as described in section 5.3.3). So, close matching of this is an explicit requisite for the calibrated model, and, it is necessary that the offramp flow here be satisfied, and the flow that does not take the offramp still produce the congestion patterns as discussed in the previous paragraph. These two offramp flow figures can be interpreted as stating Neumann (flux) boundary conditions that our freeway PDE model must (and does) fulfill, in addition to the macro-scale congestion requirements.

Finally, Tables 5.1 and 5.2 show macroscopic freeway performance metrics predicted by our simulation and the comparable measured performance metrics. For the three quantities for which measured performance metrics are available (total VMT, total VHT, and total delay), we can see that all percentage errors between our simulation and the measurements are at most roughly 10%. We consider this is a fairly good accuracy, given the inherent high noise of performance metrics computed from stationary sensors like the loop detectors used by PeMS (Jia et al., 2001; Chen et al., 2003). In particular, we find higher errors between the simulated performance metrics and the measured performance metrics for the SR-134 East / I-210 East case study, a location where single-loop detectors are present, than for the I-680 North case study, where mostly dual-loop detectors are present (PeMS, 2019) (single-loop detectors being intrinsically much more noisy than dual-loop detectors at measuring quantities like speed and flow (Jia et al., 2001)).

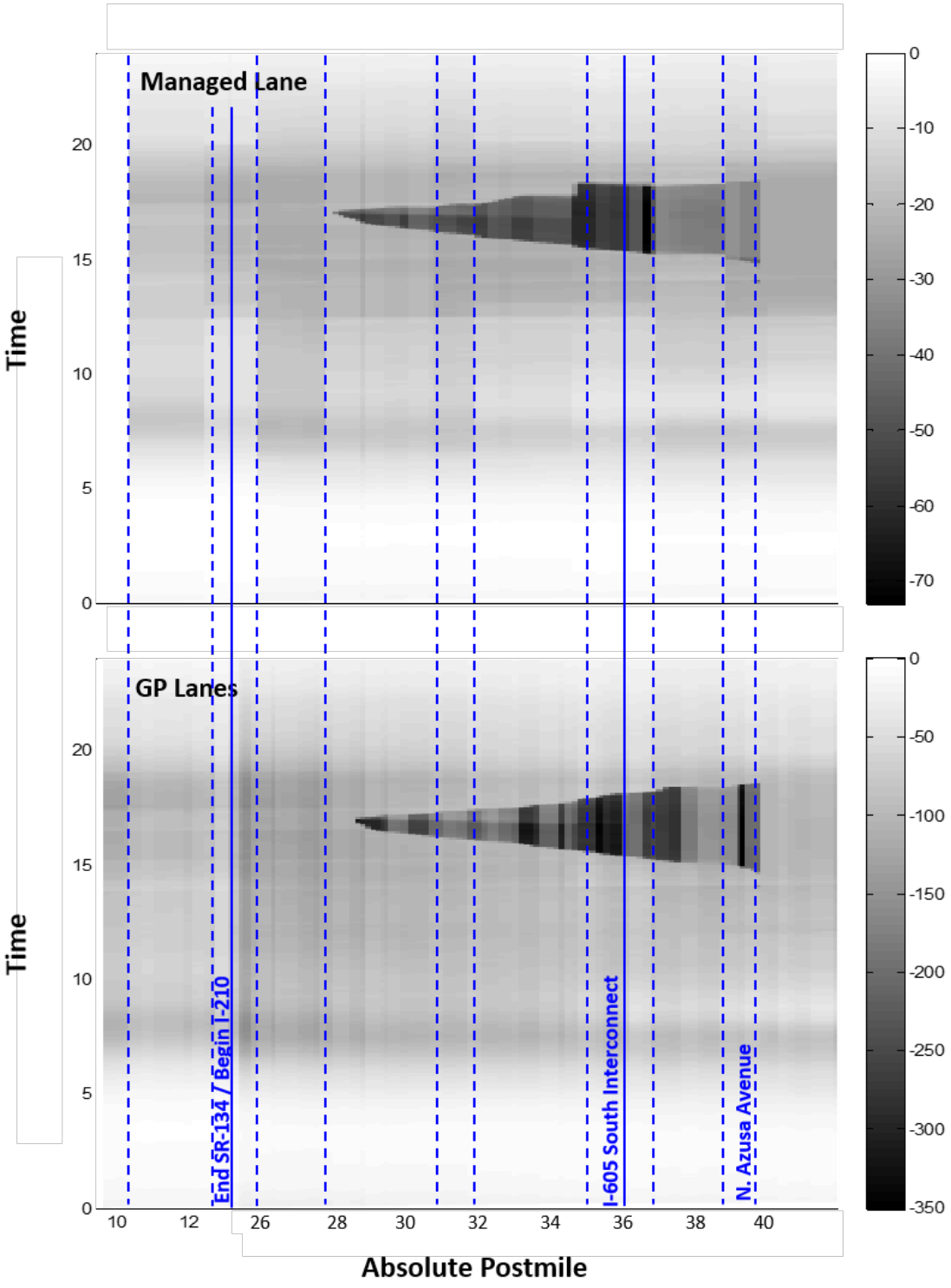


Figure 5.10: SR-134 East/ I-210 East density contours for GP and managed lanes produced by simulation. Density values are given in vehicles per mile per lane.

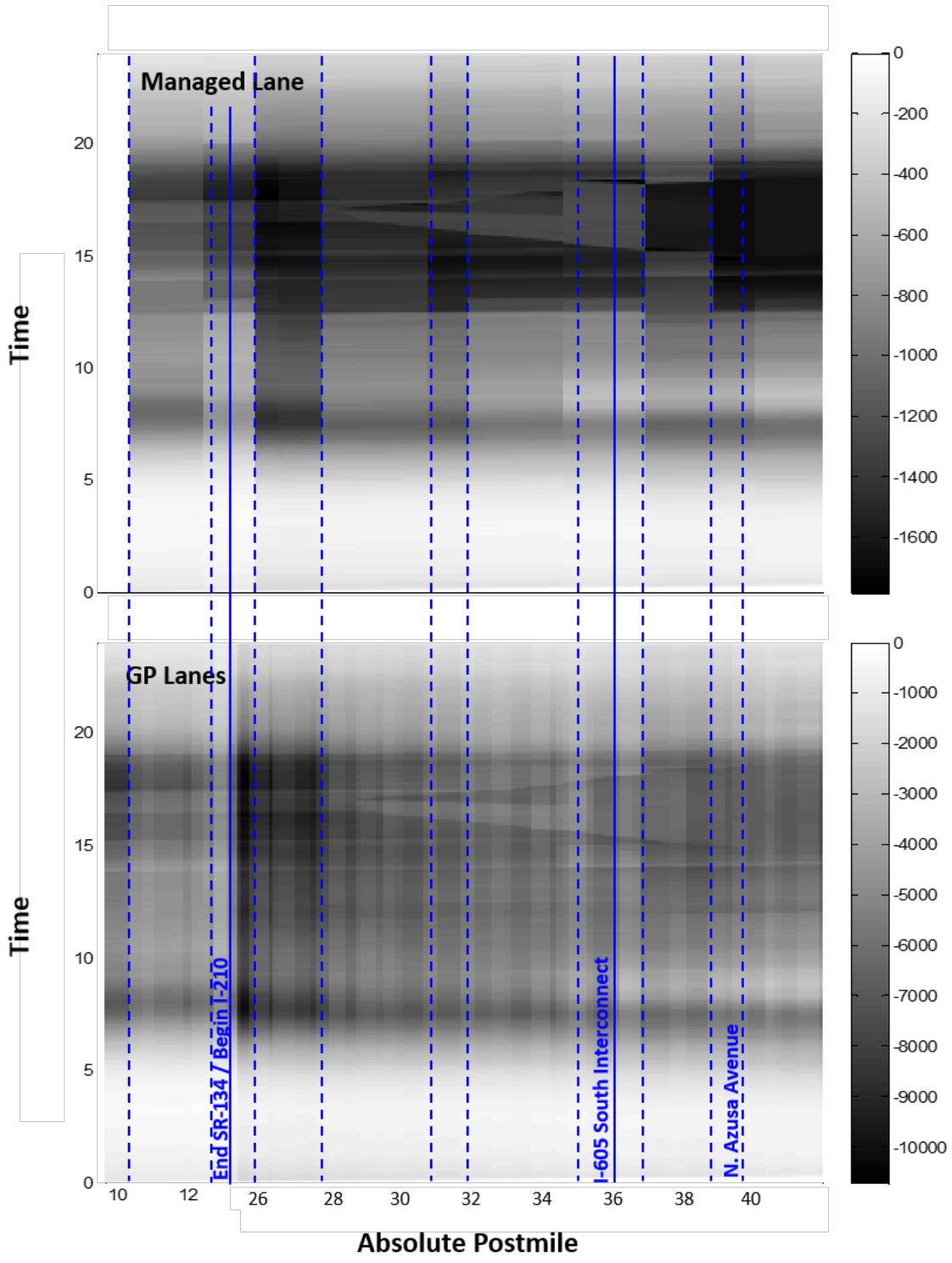


Figure 5.11: SR-134 East/ I-210 East flow contours for GP and managed lanes produced by simulation. Flow values are given in vehicles per hour per lane.

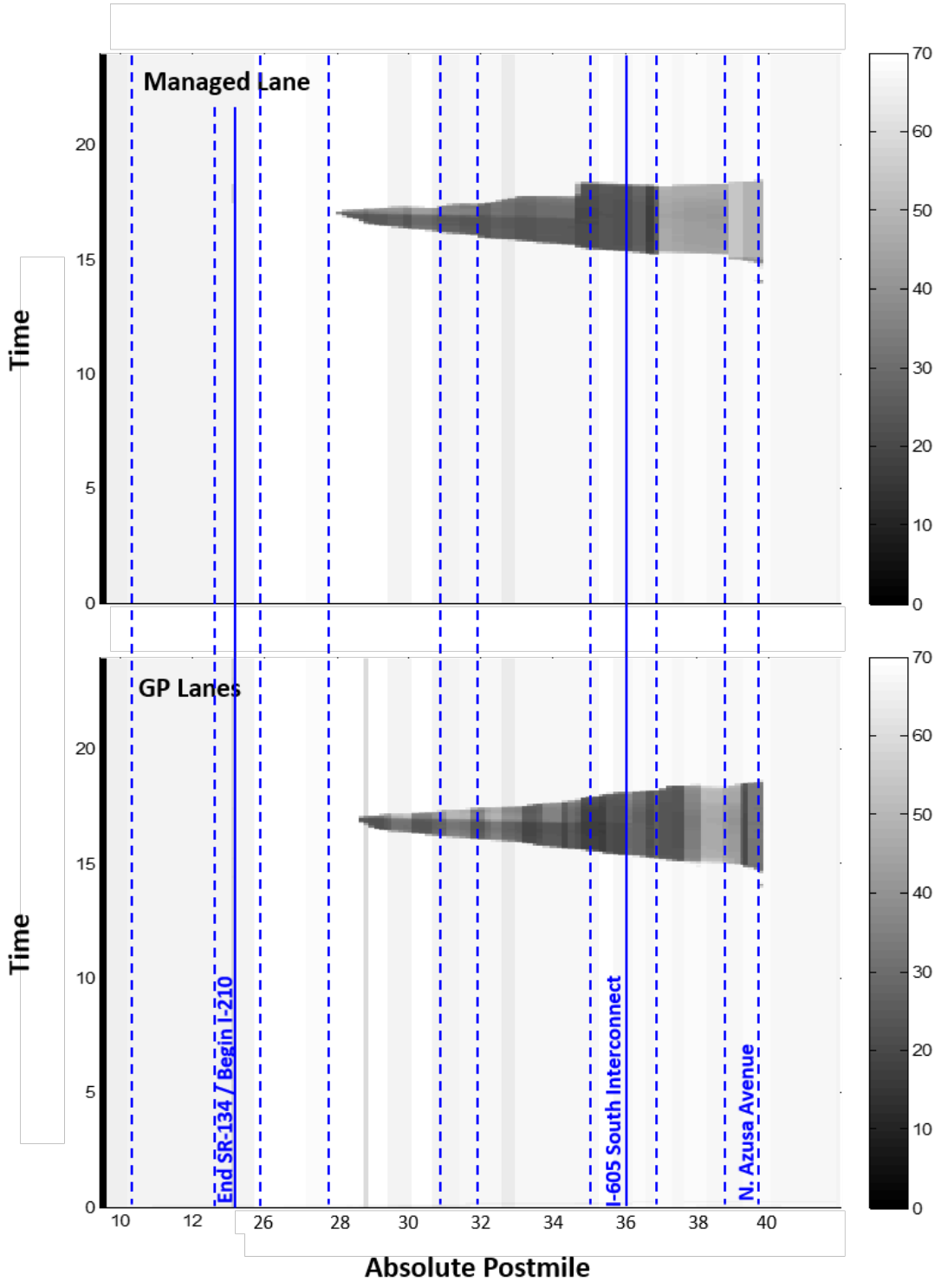


Figure 5.12: SR-134 East/ I-210 East speed contours for GP and managed lanes produced by simulation. Speed values are given in miles per hour.

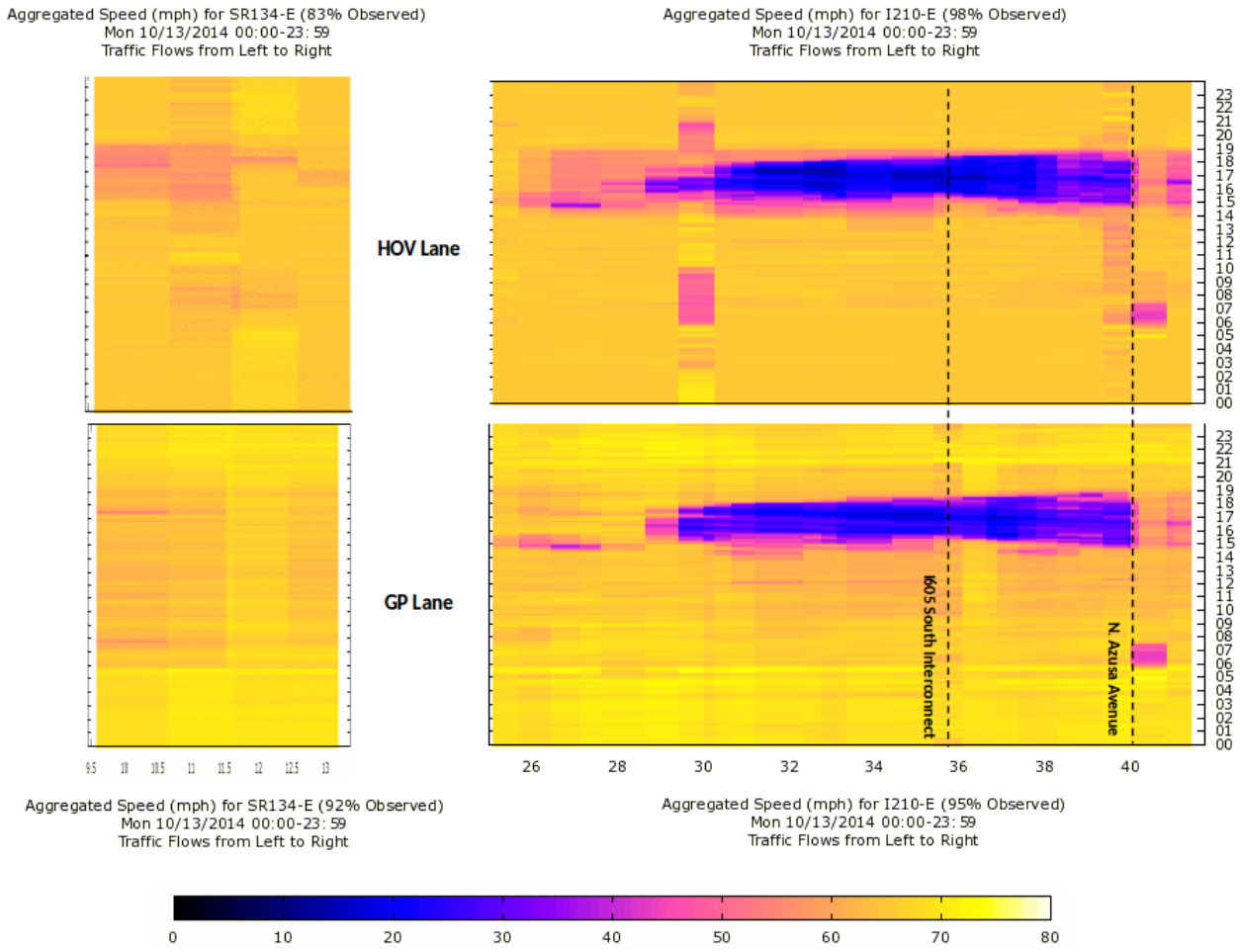


Figure 5.13: SR-134 East/ I-210 East speed contours for GP and managed lanes obtained from PeMS for Monday, October 13, 2014. The horizontal axis represents absolute postmile, and the vertical axis represents time in hours. The four contours share the same color scale.

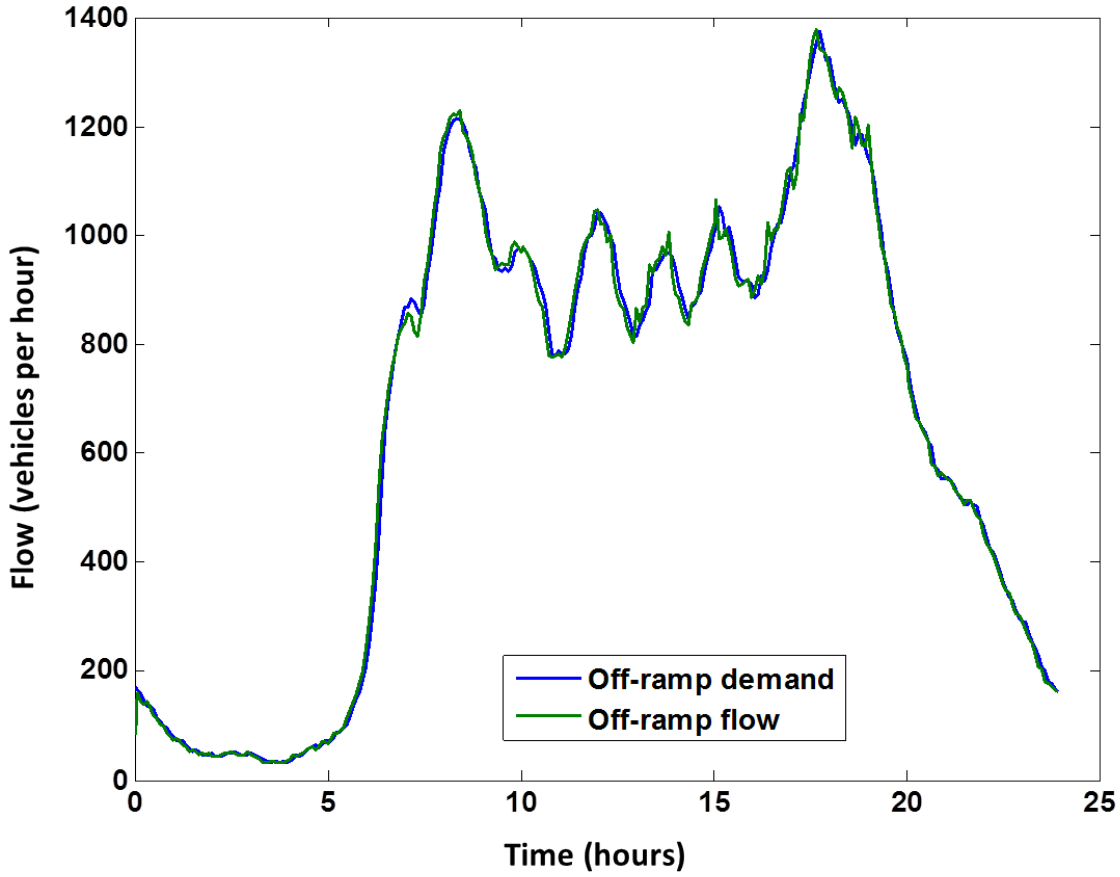


Figure 5.14: Flow at the North Hill Avenue offramp over 24 hours — PeMS data (off-ramp demand) vs. computed by simulation (off-ramp flow).

5.5 Conclusion

This chapter presented the implementation of the modeling components for managed lane-freeway networks originally proposed in chapter 4 This included their integration into both a full macroscopic traffic model suitable for corridor-scale traffic simulation and analysis, as well as fitting in estimation methods for some of the harder-to-estimate parameters into the traditional iterative calibration approach for traffic models. Our simulation results comparing our model and calibration results showed good agreement in two case studies, validating both our full-access and gated-access modeling techniques.

Part II

New Particle Filtering Constructions with Applications to Traffic Network Estimation with Connected Vehicles

Fusing Microscopic Data into the Macroscopic Model

Chapter 6

Fusing Loop and GPS Probe Measurements to Estimate Freeway Density

6.1 Introduction

In the chapter on preliminaries (chapter 2), we introduced the filtering problem and the particle filtering method for (approximately) solving it, as well as the kinematic wave macroscopic traffic model. In this chapter, we discuss combining the two, i.e., using a particle filter to estimate the unobserved macroscopic state of a freeway via partial observations, as well as extend prior filtering results in the literature by filtering across measurement domains: using measurements of velocity to improve the estimate of density.

When compared to many other objects studied in a systems context, road networks exhibit high levels of nonlinear phenomena in congestion shockwaves (Blandin et al., 2012), have relatively low levels of sensor penetration (Patire et al., 2015) (typically fixed sensors that measure vehicle flows), and what sensor measurements are available may exhibit high degrees of bias or noise (Chen et al., 2003). The sparsity and inaccuracy of detection makes estimating the spatiotemporal state of traffic flows difficult.

State estimation methods based on physical traffic models have advantages over those based purely on statistical models. Physical models allow for estimation of conditions at uninstrumented locations, as well as prediction of network behavior in response to previously-unobserved conditions (for example, increased demands on a road network in response to a special event). Extrapolation of this sort with a statistical model is of course risky. Instead, the physical model is used to generate a state trajectory that best fits the sensor data, but the nonlinearity of the traffic PDEs makes this a hard problem to solve. Since the traffic PDEs are nonlinear, finding a solution that best matches he observed data across all space and time is not easy without considerable relaxations (Claudel and Bayen, 2011). More common is to treat the problem in a filtering context, where estimates are propagated forward in time

through traffic models and updated with information from measurements. Nonlinear filters used for traffic state estimation in the literature include the Extended Kalman Filter (Wang and Papageorgiou, 2005) and the Sequential Monte Carlo-based Mixture Kalman (Sun et al., 2004), Ensemble Kalman (Work et al., 2010) and particle filters (Mihaylova et al., 2007).

Traditional sensors for filtering on road networks, such as buried inductive loop detectors or video cameras, have been fixed in location. Recently, though, there has been great interest towards augmenting these fixed data with data collected by other parties as part of the current explosion of sensor availability and data collection (Work et al., 2010). Transportation authorities have been eager to leverage these new data as both a low-cost alternative to increasing penetration of detection and to extend detection to areas for which installation is economically infeasible (Patire et al., 2015).

Works such as Work et al. (2010) have successfully used measurements of individual vehicles' velocity collected from passengers' GPS-enabled mobile devices, or *probe* data, for the filtering problem. This supplementation is principally referred to as *data fusion* in the transportation literature (Patire et al., 2015).

The method of Work et al. (2010) is perhaps the most popular data fusion method, with recent studies examining the marginal gains for varying data quantities in simulation (Bucknell and Herrera, 2014) and deployment at several real-world sites (Allström et al., 2012; Patire et al., 2015). These results and others have shown the availability of probe data makes it useful in estimating the traffic state across a large freeway corridor, on the order of tens of miles.

Unfortunately, the state estimation method in Work et al. (2010) is only intended to estimate the state of traffic's mean velocity (which is then easily converted into travel times, a common metric of road performance), whereas traffic control algorithms tend to rely on estimates of the density of vehicles (Papageorgiou et al., 2003). As we will explain below, estimating density directly from velocity measurements is difficult. A method for supplementing data from fixed detection infrastructure with probe data while retaining the model-based filtering approach has been elusive, but we show that reconsidering the problem in a probabilistic view simplifies the mathematics, and further develop an approach based on Rao-Blackwellization of particle filters as a solution.

The remainder of this chapter is structured as follows. Section 6.2.1 provides context on the idea of filtering on macroscopic traffic models. In Section 6.2.2 we discuss the specific case of filtering on macroscopic flow models from a probabilistic perspective. We also provide a motivation for ensemble methods for the traffic filtering problem and give a formulation of a general particle filtering algorithm for use with any macroscopic flow model. Section 6.3 extends the general model developed in Section 6.2.2 to the specific problem of real-time assimilation of probe velocity measurements to augment traffic density estimates, and Section 6.4 provides some numerical experiments demonstrating the applicability of velocity measurements towards estimating traffic density. Finally, section 6.5 provides some closing thoughts and discusses several immediate uninvestigated problems.

6.2 Traffic flow models and filtering

6.2.1 Macroscopic flow models

In this work, we use as our traffic model the Lighthill-Whitham-Richards (LWR) kinematic wave model and its discretized Cell Transmission Model (CTM) form as introduced in section 2.2, with a few modifications.

Today, simulation methods based on LWR and other macroscopic descriptions are used to model traffic on freeways so that public authorities may estimate congestion for the purposes of traffic control and infrastructure planning (Kurzhanskiy and Varaiya, 2015).

The LWR model of traffic (2.10) is often said to be stated in *Eulerian* coordinates. This designation refers to the Eulerian characterization of a fluid flow field, in which the state of the flow field is parameterized by space z and time k. In contrast is the *Lagrangian* characterization of a flow field, which tracks the position of individual fluid elements. In the context of vehicle traffic, the fluid elements are vehicles, and the flow field is parameterized in terms of individual vehicle number and time (Leclercq et al., 2007).

Along these lines, traffic data taken from fixed sensors are often called Eulerian data, and data that describe individual vehicles (i.e., our probe data) are called Lagrangian data.

A common type of Eulerian sensor is the buried inductive loop detector, which detects the presence of a vehicle as it passes. Later in this article, we will use simulated loop data from the California Performance Measurement System (PeMS) (PeMS, 2019). The PeMS dataset contains measurements from single-loop and double-loop detectors; double-loop detectors measure vehicle flows and speeds directly (from which vehicle densities are inferred), while the single-loop detectors measure vehicle flows and estimate speeds and densities using the PeMS "g-factor" algorithm Jia et al. (2001).

In addition to the values related to the CTM discussed in chapter 2, we consider another key quantity of interest that can be computed from the fundamental diagram: the average velocity of the link, which is simply

$$v(\rho) = \frac{q(\rho)}{\rho} = \frac{\min(R(\rho), S(\rho))}{\rho}. \tag{6.1}$$

Remark 6.1. **Note**: In this chapter and the next, we will refer to vehicle flow with the symbol q rather than with f as was done in the chapter on preliminaries and in part I. We make this change to avoid confusion with the stochastic update equation of a stochastic dynamic system, as was introduced in section 2.1.

Unlike the complex node problems discussed in chapters 3, 4, and 5, here we focus only on freeways. The only relevant junctions are merges of an onramp and the freeway and diverges to an offramp in the freeway. For onramps and offramps, in this chapter we use a simple merge model proposed by Muralidharan et al. (2009). In the Muralidharan et al. (2009) onramp merge model, the flows exiting the upstream mainline link and the onramp link are proportional to the links' demand, which is itself a function of the links' density.

Driver behavior at offramp diverges is parameterized by a *split ratio* coefficient $\beta_\ell \in [0, 1]$, which is the portion of vehicles in link ℓ that wish to enter the offramp. The offramps are assumed to accept all vehicles that wish to enter them (that is, they are assumed to never be in congestion). For more information as to the particularities of this offramp mode, we refer again to Muralidharan et al. (2009).

The original CTM of Daganzo has been characterized as a *first-order* model, as the dynamic equation of state is a function of only one variable, ρ_ℓ.

As mentioned in chapter 2, other authors have proposed alternative parameterizations of the sending and receiving functions, or even higher-order parameterizations of a link's state, where the one-step update calculation is a function of the density and one or more other quantities. We discuss a few second-order models used for filtering in Section 6.2.2.

We have thus far described a deterministic model for traffic flow, but in a filtering context a stochastic model is required. The large number of parameters in the CTM lead to a rich capability to introduce stochasticity into the model. In particular, authors have proposed stochastic models that include uncertainty in the fundamental diagram parameters, upstream (boundary condition) demands, or driver behavior at diverges. For the simulations later in this chaper, we treat the onramp and offramp parameters as stochastic. The onramp flows are contaminated with additive white Gaussian noise (similar to in e.g. Wang and Papageorgiou (2005)), and the split ratios β are taken as independent-across-time beta-distributed random variables. Both distributions were fit to data. Note that the non-correlation across time is a modeling assumption, and a more realistic model would consider a nontrivial autocorrelation of these time-varying random variables.

6.2.2 Past work on filtering on macroscopic flow models

We now consider the problem of traffic state estimation on using recursive filtering. In this chapter, we focus on freeways; a large and separate body of literature exists on state estimation for arterial roads.

Most filtering schemes in the literature, as well as the presented in this chapter (which relies on the CTM), use Eulerian flow models. As mentioned in Section 6.2.1, Eulerian models come in first- and higher-order varieties. Higher-order models add additional PDEs to (2.10) or dynamic equations to the discretization.

Various authors have proposed filtering schemes based on both first- (see for example the work of Sun et al. (2004) and Work et al. (2010)) and second-order models (such as Wang and Papageorgiou (2005) and Mihaylova et al. (2007)). In particular, the second-order models of Wang and Papageorgiou (2005); Mihaylova et al. (2007) add a dynamic equation for the link velocity. While the particular algorithm proposed in Section 6.3 uses a first-order model, we remain agnostic on the question of deciding between first- or higher-order models for filtering; the following discussion is intended to be universal for filtering with Eulerian models of any order.

It will provide clarity to the discussion if we separate out the different state subsets in the state vector x individually; for our discussion we will consider the state vector as potentially

composed of the density, x^ρ, and the velocity, x^v of the links. The vectors of observations of link density and velocity are similarly denoted y^ρ and y^v, respectively.

Examination of (2.3) and (2.4) shows that within the prediction and filtering framework, the only term in which the observations y appear is the likelihood, $g_\theta(y|x)$. Naturally, assimilation of observations, whether density or velocity, requires a model for the likelihood through specification of the observation equation. In particular, the likelihood $g_\theta(y|x) = g_\theta(y^\rho, y^v|x^\rho, x^v)$, which is the joint likelihood of the entire observation vector y, must be posed by the practitioner, and its proper form, particularly in representing the relationship between density and velocity observations, is not obvious. Previous authors have used various methods that exploit the structure of their particular prediction framework to reduce the complexity of the likelihood when dealing with multi-state assimilation.

The filtering schemes of Wang and Papageorgiou (2005) and Mihaylova et al. (2007) use second-order Eulerian models (note that although the model of Mihaylova et al. (2007) used flow and velocity as the state variables, this representation is essentially equivalent to a density-velocity representation through (6.1)). With a second-order model, the link density and velocity vectors are separately predicted with different explicit functions of the current state:

$$\begin{aligned} x_k^\rho &= \mathcal{F}_{\theta,\rho}(x_{k-1}^\rho, x_{k-1}^v) \\ x_k^v &= \mathcal{F}_{\theta,v}(x_{k-1}^\rho, x_{k-1}^v). \end{aligned} \tag{6.2}$$

Under this construction, the following conditional independence assumption has been made explicitly by Wang and Papageorgiou (2005); Mihaylova et al. (2007):

Assumption 6.1 (Second-order traffic model assumption)**.** *The density and velocity states of the network at time k are conditionally independent given the state at time $k-1$. Equivalently, $p_\theta(x_k^\rho, x_k^v|x_{k-1}) = p_\theta(x_k^\rho|x_{k-1})p_\theta(x_k^v|x_{k-1})$*

In our view, Assumption 6.1 is not a good assumption. In particular, it seems to conflict with a construction of a second-order model. Without going into too much detail, adding additional PDEs to (2.10) implies a belief that vehicle traffic dynamics are too complex to be modeled with (2.10) alone, i.e., with local interactions between the state variables ρ and v. Discretization for numerical forward integration as in (6.2) would then lose these additional interactions, except for simulation timescales on the order of the inter-state interactions.

The schemes of Wang and Papageorgiou (2005) and Mihaylova et al. (2007) make another assumption that is unstated in both Wang and Papageorgiou (2005) and Mihaylova et al. (2007) but is reasonable: that density measurements y^ρ are independent of the velocity state x^v (and vice-versa). One may then factor the likelihood in a straightforward manner:

$$g(y_k|x_k) = g(y_k^\rho, y_k^v|x_k^\rho, x_k^v) = p(y_k^\rho|x_k^\rho)p(y_k^v|x_k^v). \tag{6.3}$$

Thus, the ungainly likelihood $g(y_k|x_k)$ need not be specified, and instead only individual observation equations for density and velocity need be specified, the likelihood of the entire

observation vector being their product. By beginning with a second-order model that contains both x^ρ and x^v among its states, this likelihood factorization is natural and elementary.

Despite the prevalence of first-order models for macroscopic simulation, an equivalent operation for first-order models, and thus a "clean" method for model-based filtering of data from other data domains, including Lagrangian data, is not obvious. If, for example, a first-order model had as its only state a link's density, then the factorization of (6.3) is not implementable, as x^v and $p(y_k^v|x_k^v)$ do not exist. As a result, Lagrangian data do not naturally fit into a first-order Eulerian flow model.

We will highlight two works in filtering Lagrangian data in an Eulerian flow model. First, Lovisari et al. (2015), choose to stay entirely in the density regime. While we mentioned methods such as the "g-factor" algorithm Jia et al. (2001) to calculate densities solely from single-loop Eulerian flow data Lovisari et al. (2015) use probe velocity measurements as substitutes for those from double-loop detectors. If the probe velocities are accurate, density calculations at single-loop detectors can be comparable to those at double-loop detectors. Unfortunately, if single-loop-detector coverage is sparse, usefulness would drastically decrease.

In another work, Work et al. (2010), brought Lagrangian data into Eulerian coordinates in a novel manner. The "velocity Cell Transmission Model" (v-CTM) proposed by Work et al. (2010) is a first-order model in which the state vector consists only of the velocity along links. That is, rather than typical first-order models that state the sending and receiving functions as functions of link density, the model of Work et al. (2010) poses them as functions of velocity. Density and velocity measurements are then fused by transforming density measurements to virtual measurements of the equivalent velocity specified by (6.1). While this approach was successful in fusing measurements of different domains (and velocity measurements that are nonfixed), it has its own drawbacks. First, it requires the selection of a fundamental diagram with a bijective relationship between density and velocity (Work et al., 2010), which proscribes many popular fundamental diagrams such as the classic model of Newell (1993) and Daganzo (1994), whose model of traffic having a constant freeflow velocity is intuitively appealing and agrees with many empirical observations on real freeways. In addition, by converting density measurements to velocity measurements, some amount of observed information is lost (specifically, note that (6.1) will equate to the freeflow velocity for any value of $\rho < \rho_c$) or contaminated (due to possible modeling errors in the fundamental diagram) (Fig. 6.1).

As a consequence, while the method of Work et al. (2010) allows estimation of the velocity state trajectory, the density state is not recoverable in general when using the v-CTM. In this chapter, we avoid this problem by estimating density directly with our density and velocity measurements.

Thus far we have focused entirely on filtering with Eulerian flow models, but the (relatively sparse) literature on filtering with Lagrangian flow models is worth mentioning. Yuan et al. (2012) describe one such algorithm. Their model fuses Eulerian (loop detector measurements) and Lagrangian (high-precision individual vehicle traces) to estimate in the Lagrangian coordinates. A state estimator in the Lagrangian coordinates seeks to reconstruct vehicle ordering and location over time, which may be difficult with contemporary Lagrangian data sources such as low-frequency and noisy GPS. However, with higher-precision Lagrangian

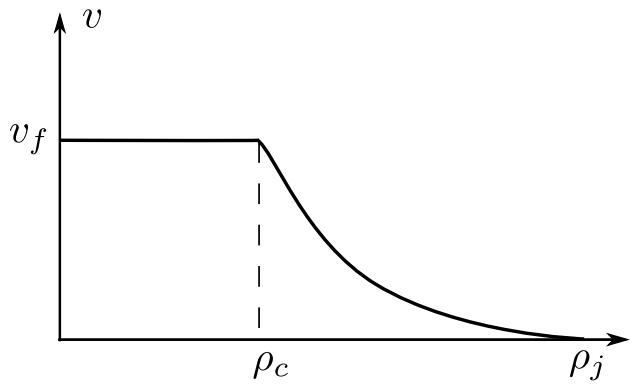


Figure 6.1: Velocity as a function of density with the Newell-Daganzo triangular fundamental diagram (c.f. (6.1)). The function is constant for $\rho < \rho_c$ and has a hyperbolic shape for $\rho > \rho_c$. The function is not injective in the region $\rho < \rho_c$.

data increasing in market penetration, this may soon be feasible.

6.2.3 Ensemble methods for macroscopic flow models

Due to their complex structure, stochastic implementations of macroscopic traffic models create transition kernels $f_\theta(\cdot)$ for which evaluation of the integral in (2.3) is analytically difficult or impossible (Mihaylova et al., 2007). The distribution of $X_t|X_{k-1}$ is thus not able to be expressed in closed form unless restrictive assumptions are made. These difficulties trace back to the fact that macroscopic traffic models are discretizations of the LWR PDE, leading to a system with states that are tightly coupled with commonly-occuring nonlinear behavior such as congestion and shockwaves. These nonlinearities mean that individual links may variously affect their upstream link, downstream link, or both during the next model update, depending on their and their neighbors' current state. For macroscopic traffic models of moderate or large numbers of links, these cascading nonlinearities can cause large multimodality in distributions (Blandin et al., 2012), which are difficult to approximate in a parametric manner. Indeed, Blandin et al. (2012) showed that a natural approximation of $f_\theta(\cdot)$, the linearized estimate as used in the Extended Kalman Filter, produces estimates that diverge quickly from the true traffic state.

Further, macroscopic traffic models of traffic networks are by construction formed of multiple PDE discretizations that interact with each other through junctions, increasing the occurrence of nonlinearities. Stochastic traffic models for which evaluations of (2.3) is a closed-form operation must make restrictive assumptions to control these nonlinearities, namely assuming high levels of instrumentation stochastically simulating only a freeway, leaving onramps, offramps, and arterial roads as known and nonrandom (see for example Sun et al. (2004) or Sumalee et al. (2011)). While this is effective for simulating well-instrumented freeways, the use of GPS data for traffic estimation is most highly desired in locations with relatively low detection infrastructure, making simplified models undesirable.

For these reasons, *ensemble* or *sequential Monte Carlo* methods like the particle filter introduced in chapter 2 have been the major focus of research in traffic state estimation. Here, we describe implementation details for applying the particle filter to the CTM.

Remark 6.2. **Note:** In this chapter, we move the particle index p from the superscript, as we had it in chapter 2 (e.g., x_k^p), to the subscript (e.g., $x_{p,k}$) to clear up notation by isolating the newly-introduced superscript terms ρ and v that form the focus of discussion here.

Recalling the filtering equation (2.8a), note that the likelihood $g_\theta(\cdot)$ is only evaluated for specific values of the conditioned term - namely, the particle-specific value $x_{p,k}$. Let us make the following conditional independence assumption to take advantage of this:

Assumption 6.2. *Given a value of X_k, the state of the entire network at time k, individual measurements in the vector Y_k are conditionally independent. Equivalently, the measurement noises of individual measurements are independent.*

The above assumption allows us to further factor the likelihood:

$$\begin{aligned} g_\theta(y_k|x_{p,k}) &= \prod_{i=1}^{M} p_\theta(y_{k,i}|x_{p,k}) \\ &= \prod_{i=1}^{M} p_\theta(y_{k,i}|x_{p,k,L(i)}), \end{aligned} \tag{6.4}$$

where $i \in \{1, \ldots, M\}$ indexes individual elements of the M-long measurement vector y and $L(i)$ denotes the link where measurement i takes place.

Given an ensemble of particles $\{x_{k,p}|y_{k,p}\}_{p=1}^{P}$, a variety of point estimates of the system state may be obtained, among them the empirical mean

$$\mathbf{E}\left[X_k|Y_k\right] \approx \sum_{p=1}^{P} w_{k,p}\delta\left(x_{p,k}|y_k\right). \tag{6.5}$$

The above construction is implemented to estimate density using only density measurements in Algorithm 6.1. Algorithm 6.1 is itself not novel.

Algorithm 6.1 (Particle Filter for Traffic Density Estimation)**.**
Inputs:

- *PDF of density initial conditions $p_\theta(X_0^\rho)$*
- *Stochastic cell transmission model $f_\theta(x_{p,k}^\rho|x_{p,k-1}^\rho)$*
- *Density measurement likelihood function $p_\theta(y_i|x_{L(i)}^\rho)$, where $L(i)$ indicates the link of the ith measurement*

1. *Initialization: At time $t = 0$:*

   a) *Sample an ensemble of particles of the density state, $x^{\rho}_{p,0} \sim p_\theta(X^{\rho}_0)$, the distribution for the initial condition*
   $\forall\, p \in \{1, \ldots, P\}$.

   b) *Set the initial weights $w_{p,0} = 1/P$*
   $\forall\, p \in \{1, \ldots, P\}$.

2. *Prediction: At time $t > 0$, $\forall$ particle p, sample*
   $x^{\rho}_{p,k} \sim f_\theta(x^{\rho}_{p,k}|x^{\rho}_{p,k-1})$ *by evaluating a one-step stochastic cell transmission model update*

3. *Data assimilation: $\forall$ particle p, at time t*

   a) *For each measurement received at time t, $y_{k,i}$, compute the per-measurement likelihood*
   $g_{p,i} = p(y_{k,i}|x^{\rho}_{p,k,L(i)})$

   b) *Compute the overall particle likelihood,*
   $g_p = \prod_i g_{p,i}$

   c) *Compute the (unnormalized) posterior particle weight, $\tilde{w}_{p,k} = w_{p,k-1} g_p$*

4. *Normalization: Normalize the particle weights,*
   $w_{p,k} = \tilde{w}_{p,k} / \left(\sum_p \tilde{w}_{p,k} \right)$

5. *Resampling: If resampling is desired, resample P particles with replacement from $\{x^{\rho}_{p,k}\}$ with selection probability of particle $p = w_{p,k}$*

6. *If $t = t_{final}$, end, otherwise $t \leftarrow t + 1$ and return to step 2*

6.3 Rao-Blackwellized Particle Filter for Data Fusion

6.3.1 Rao-Blackwellization as an improvement for sequential Monte Carlo

Our algorithm makes use of a modification to standard particle filtering known as a *Rao-Blackwellized Particle Filter* (RBPF). The name refers to the Rao-Blackwell Theorem, a well-known result from mathematical statistics (see e.g. the discussion in Keener (2010, Ch. 3)), which states that an estimate based on data may be improved in terms of expected convex loss, and will never be worsened, by conditioning on a sufficient statistic. In the setting of Monte Carlo methods, *Rao-Blackwellization* refers to a method for improving a Monte Carlo sampler over several random variables. Should some subset of the random

variables have distributions as explicit functions of another subset, gains in computational cost and accuracy can be made by making use of these explicit distributions, rather than approximating them with Monte Carlo (Casella and Robert, 1996). Rao-Blackwellization has gained broad adoption in Sequential Monte Carlo methods in particular (Doucet et al., 2000). If there exists some subset of the state vector, x^B whose distribution is an explicit function of the remaining states, x^A then by analogy x^A is a sufficient statistic for x^B. We may then take a shortcut in evaluating our Monte Carlo prediction step in (2.7),

$$\begin{aligned}
&p_\theta\left(x_k|y_{k-1}\right) \\
&= \int p_\theta\left(x_{k-1}|y_{k-1}\right) f_\theta\left(x_k|x_{k-1}\right) dx_{k-1} \\
&= \int p_\theta\left(x_{k-1}|y_{k-1}\right) f_{\theta,A}\left(x_k^A|x_{k-1}\right) f_{\theta,B}\left(x_k^B|x_k^A\right) dx_{k-1} \\
&\approx \sum_{p=1}^{P} p_\theta(x_{p,k}|y_{k-1}) \left(\delta_{f_\theta,A}\left(x_{p,k}^A|x_{p,k-1}\right) \times \delta_{f_\theta,B}\left(x_{p,k}^B|x_{p,k}^A\right)\right), \qquad (6.6)
\end{aligned}$$

where $f_{\theta,A}\left(\cdot\right)$ is a PDF for the state subset $X_k^A|X_{k-1}$ (that is, a truncated version of $f_\theta(\cdot)$ that only has domain in the space of x^A), $f_{\theta,B}(\cdot)$ is our closed-form PDF for x^B, and the use of the Cartesian product $\times$ is needed due to the individual Dirac deltas residing on disjoint subsets of x. In addition, the second line makes use of the fact that, X^B being an explicit function of X^A, we have that X_k^B is conditionally independent of X_{k-1} given X_k^A.

When implementation is possible, Rao-Blackwellization of a particle filter offers significant advantages. Reducing the size of the state that one must approximate through Monte Carlo brings improvements in computation time and eliminates error in approximating x^B through Monte Carlo approximation. Indeed, Rao-Blackwellization of particle filters were originally proposed for a reduction in the variance of estimated distributions (Doucet et al., 2000). Reducing computation time also allows for improved estimates by freeing additional computational resources to simulating more particles, allowing for richer predictions of $p\left(x_k|y_{k-1}\right)$. For these reasons, RBPFs have gained recent popularity for improving approximations of computationally difficult problems in large state-space settings, such as the simultaneous localization and mapping problem in robotics (Grisetti et al., 2007). We are particularly interested, though, in its immediate application for data fusion.

6.3.2 Implementation for traffic data fusion

As mentioned in Section 6.2.3, first-order traffic models predict one-step model updates only in terms of density, leaving the velocity likelihood term $p_\theta\left(y_k^v|x_k^v\right)$ in (6.3) unspecified. The lack of a plug-in likelihood has been a hindrance for data fusion on first-order models, but by making use of an RBPF, we may overcome this problem.

Recall from (6.1) that under a first-order traffic model, the average velocity of a link may be computed from the link's density and flow (itself a function of the density). Let us then

say that link density is a sufficient statistic in the Rao-Blackwell sense for link velocity. We may then factor the likelihood in an analogous manner to the operation in (6.3),

$$\begin{aligned} g_\theta(y_k|x_k) &= g_\theta(y_k^\rho, y_k^v|x_k^\rho, \bar{x}_k^v) = p_\theta(y_k^\rho|x_k^\rho)p_\theta(y_k^v|\bar{x}_k^v, x_k^\rho) \\ &= p_\theta(y_k^\rho|x_k^\rho)p_\theta(y_k^v|\bar{x}_k^v)p_\theta(\bar{x}_k^v|x_k^\rho), \end{aligned} \tag{6.7}$$

where we introduce $\bar{x}_k^v$, a random variable denoting the average velocity of a link, whose PDF $p_\theta(\bar{x}_k^v|x_k^\rho)$, is an explicit function of the link density under a first order model, and $p_\theta(y_k^v|\bar{x}_k^v)$ is a likelihood denoting the distribution of velocity measurements given an average velocity. This last function will then incorporate information such as the distribution of vehicle velocities about a link's nominal velocity, as well as measurement noise of individual probe measurements around their sensors' true velocity.

The variable $\bar{x}_k^v$ is a time-varying quantity that describes the system, but it is not a state of the system - the only system state of the CTM is density. It is instead an intermediary between the true state x^ρ and observations y^v. We refer to it as a *pseudostate*.

Note that while we have referred to (6.1) as the source of the PDF $p_\theta(\bar{x}_k^v|x_k^\rho)$, which follows the original formulation (6.1) in treating link velocity as a nonrandom function of density, various authors including, notably, Sumalee et al. (2011), have proposed flow and velocity as random functions of link density. A practitioner would be free to incorporate this stochasticity in this function.

By applying assumption 6.2, we may further simplify (6.7):

$$\begin{aligned} g_\theta(y_k|x_k) &= p_\theta(y_k^\rho|x_k^\rho)p_\theta(y_k^v|\bar{x}_k^v)p_\theta(\bar{x}_k^v|x_k^\rho) \\ &= \left(\prod_{i=1}^{M^\rho} p_\theta(y_{k,i}^\rho|x_{k,L(i)}^\rho)\right)\left(\prod_{j=1}^{M^v} p_\theta(y_{k,j}^v|\bar{x}_{k,L(j)}^v)p_\theta(\bar{x}_{k,L(j)}^v|x_{k,L(j)}^\rho)\right) \end{aligned} \tag{6.8}$$

where $i \in \{1, \ldots, M^\rho\}$ indexes the M^ρ-long density measurement vector y_k^ρ, $j \in \{1, \ldots, M^v\}$ performs the same function for y_k^v, and $L(\cdot)$ denotes the link of the associated measurement. Equation (6.8) is the likelihood we use in implementing our RBPF.

An implementation of our RBPF scheme is described in algorithm 6.2. This algorithm is constructed in similar manner to algorithm 6.1, but with (6.8) used for the likelihood $g_\theta(\cdot)$.

Algorithm 6.2 (Rao-Blackwellized Particle Filter for Density Estimation on "Pseudo-Second-Order" Model)**.**
Inputs:

- *PDF of density initial conditions* $p_\theta(X_0^\rho)$
- *Stochastic cell transmission model* $f_\theta(x_{p,k}^\rho|x_{p,k-1}^\rho)$
- *Per-link predicted velocity distribution* $p_\theta(\bar{x}_{k,L(i)}^v|x_{k,L(i)}^\rho)$
- *Density and velocity measurement likelihood functions* $p_\theta^\rho(y_i|x_{L(i)}^\rho)$ *and* $p_\theta^v(y_i|x_{L(i)}^v)$*, where* $L(i)$ *indicates the link of the* i*th measurement*

1. *Initialization: At time $t = 0$:*
   a) *Sample an ensemble of particles of the initial density, $x^{\rho}_{p,0} \sim p_\theta(X^{\rho}_0)$, the distribution for the initial condition*
   $\forall\, p \in \{1, \ldots, P\}$.
   b) *Set the initial weights $w_{p,0} = 1/P$*
   $\forall\, p_\theta \in \{1, \ldots, P\}$.
2. *Prediction: At time $t > 0$, $\forall$ particle p, sample*
   $x^{\rho}_{p,k} \sim f_\theta(x^{\rho}_{p,k}|x^{\rho}_{p,k-1})$ *by evaluating a one-step stochastic cell transmission model update*
3. *Data assimilation: $\forall$ particle p at time k,*
   a) *For each measurement received at time k, $y_{k,i}$*
      i. *If $y_{k,i}$ is a density measurement, compute the per-measurement likelihood $g_{p,i} = p^{\rho}_\theta(y_{k,i}|x^{\rho}_{p,k,L(i)})$*
      ii. *If $y_{k,i}$ is a velocity measurement,*
         A. *Compute the predicted link velocity distribution $p_\theta(\bar{x}^{v}_{k,L(i)}|x^{\rho}_{k,L(i)})$*
         B. *Compute the per-measurement likelihood $g_{p,i} = p^{v}_\theta(y_{k,i}|\bar{x}^{v}_{L(i)})$*
   b) *Compute the overall particle likelihood,*
   $g_p = \prod_i g_{p,i}$
   c) *Compute the (unnormalized) posterior particle weight, $\tilde{w}_{p,k} = w_{p,k-1} g_p$*
4. *Normalization: Normalize the particle weights,*
   $w_{p,k} = \tilde{w}_{p,k} / \left(\sum_p \tilde{w}_{p,k} \right)$
5. *Resampling: If resampling is desired, resample P particles with replacement from $\{x^{\rho}_{p,k}\}$ with selection probability of particle $p = w_{p,k}$*
6. *If $t = T$, end, otherwise $t \leftarrow t + 1$ and return to step 2*

6.3.3 Discussion

We refer to this traffic model construction as a "pseudo-second-order" traffic model, due to the middle ground it occupies between traditional first and second-order models. In the traffic data fusion literature, second-order models are proposed due to a claimed ability to estimate velocity as well as density (see e.g. Mihaylova et al. (2007)). What is meant is that in a second-order model, the traffic velocity undergoes stochastic updates independent of the traffic density (recall (6.2)). Our model also treats velocity as a random variable and seeks to

estimate it, but no Monte Carlo steps are performed in the velocity domain. Instead, velocity state estimation is done in closed form (the Rao-Blackwellization computation of (6.7)-(6.8)). The estimation of velocity with a PF does not in general require that the estimation be done with Monte Carlo. Indeed, this recalls the original justification for Rao-Blackwellization of sampling schemes (Casella and Robert, 1996), in that unnecessary Monte Carlo should be avoided when possible.

Of course, the first-order assumption underlying our use of x^ρ as a "sufficient" estimator for x^v in the "pseudo-second-order" model may reasonably be called into question. There is much debate among traffic theorists as to which PDEs truly govern traffic flow. Our position is that, while it may be the case that the CTM is a simplification that cannot capture some traffic phenomena, its simplicity is a boon in corridor-scale filtering applications. The problem considered in this chapter, fusing data from many loop detectors and probe points presupposes application at the scale of a corridor or larger, where more complex models may be unwieldy. As an example, the CTM parameters can be quickly estimated from loop detector data that is obtained in the form of flow-density pairs, but parameters governing additional complex behavior would require additional data sources or hand-tuning, which may become infeasible for corridor-scale applications. Note that the most successful filtering algorithm for traffic data fusion at a corridor scale has been the first-order v-CTM model (Work et al., 2010; Patire et al., 2015).

Finally, while the development in this section has dealt with a Rao-Blackwellization of a particle filter in particular, note that the derivation only used the PF formalism in specifying the form of the approximation PDFs $\hat{p}_\theta(x_k|y_{k-1})$ and $\hat{p}_\theta(x_k|y_k)$ as weighted sums of Dirac deltas. The Rao-Blackwellization steps can easily be applied to data assimilation with other traffic data filtering schemes by repeating the factorization of the likelihood.

6.4 Experimental Results

We describe two numerical experiments to demonstrate the ability of the RBPF in assimilating velocity measurements to improve density estimates. The first of the experiments uses simulated data, where a "ground truth" state trajectory was created through stochastic simulation, and PFs were used to recover the full trajectory given fixed-location noisy density measurements $\{y^\rho\}$, simulated moving velocity measurements $\{y^v\}$, or both. The second experiment uses real density and velocity measurements collected from the site, and attempts to predict the density observed by a held-out subset of the loop detectors.

6.4.1 An experiment with simulated data

We describe a numerical experiment to demonstrate the ability of the RBPF in assimilating velocity measurements to improve density estimates. We use simulated data, where a "ground truth" state trajectory was created through stochastic simulation, and PFs were used to

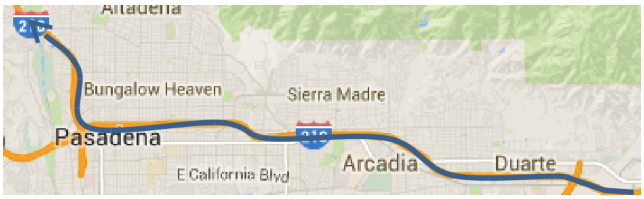


Figure 6.2: Test site of interest: a 19-mile stretch of I-210W near Los Angeles, CA.

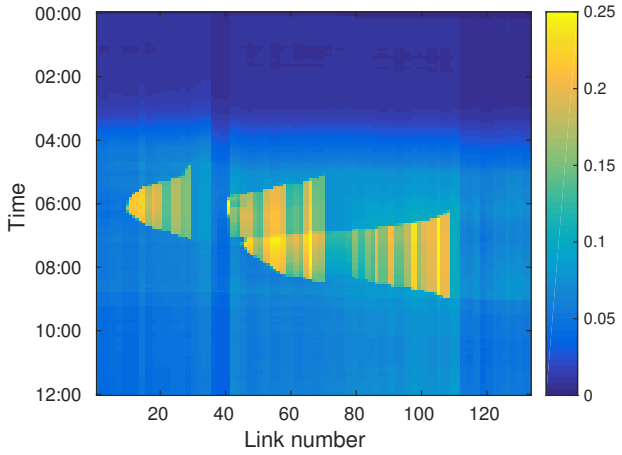


Figure 6.3: Traffic congestion patterns of the model as fit to October 13, 2014.

recover the full trajectory given fixed-location noisy density measurements $\{y^\rho\}$, simulated moving velocity measurements $\{y^v\}$, or both.

We use a CTM scheme as the prediction framework, with the model based on a section of Westbound I-210 (Fig. 6.2). The stretch of freeway of interest has a length of approximately 19 miles, and was divided into 127 model links with length averaging roughly 200 m. In addition, the section of interest has 23 onramps and 21 offramps. The stretch of road has 42 PeMS loop detectors (PeMS, 2019) along the freeway mainline. The model was calibrated against loop data collected during the morning of October 13, 2014. On this date, 8 of the loop detectors were determined to be malfunctioning on Oct 13 through identification as such by the PeMS software or manual checking. The fundamental diagram used in the model was the triangular fundamental diagram due to Daganzo discussed in chapter 2. The fundamental diagram parameters were calibrated according to the procedures described in Munoz et al. (2004) and Dervisoglu et al. (2009), with undetected ramp data imputed according to the method of Muralidharan and Horowitz (2009). Driver behavior at offramp junctions was modeled with turning ratios, calibrated by fitting to the 5-minute mean and variance from the previous month of data. This direction is the peak direction in a typical morning, so the extent of congestion is relatively large (Fig. 6.3).

The fundamental diagrams (and thus the PDF $p_\theta(\bar{x}_k^v|x_k^\rho)$) were taken to be noiseless (i.e. a function that is equal to (6.1) with probability 1) for simplicity. A commonly-used likelihood

in the literature for assimilating density measurements is a Gaussian distribution (Wang and Papageorgiou, 2005), we select $g_\theta\left(y_i^\rho | x_{L(i)}^\rho\right) = \mathcal{N}(x_{L(i)}^\rho, \hat{\sigma^2}^\rho_{L(i)})$ with $\hat{\sigma^2}^\rho_{L(i)}$ being the sample variance of the density of link $L(i)$ across all particles. We also select the likelihood for assimilating velocity measurements as Gaussian, $p_\theta\left(y_i^v | \bar{x}_L^v(i)\right) = \mathcal{N}(x_L^v(i), \hat{\sigma^2}^v_{L(i)})$ with $\hat{\sigma^2}^v_{L(i)}$ defined similarly.

A single simulation of an afternoon period with the stochastic CTM model trained as described above was recorded and taken as the "actual" value of $\{x_k\}, t \in 1, \ldots, k_{final}$. This "actual" value is shown in Fig. 6.4(a), and featured significantly more congestion than the model baseline. We attempted to estimate the full density state across all timesteps using simulated noisy measurements. The point estimates of $\{x_k\}$ described below are the empirical means (6.5) produced by the filter.

The true density state is shown in Fig. 6.4(a). One estimation run used a particle filter as described above, with access to density measurements of freeway links where detectors are located in the real world (Fig. 6.4(b)). These measurements were contaminated with additive noise sampled from a Gaussian distribution with a standard deviation equal to 10% of the measured value. The second estimation run used a RBPF and had access to these same density measurements as well as simulated velocity measurements. The velocity measurements were sampled randomly to simulate various penetration rates: for penetration rate PR, velocity measurements of number equal to $floor(PR \cdot 100)$ were reported to the filter every five minutes. Each link had a probability proportional to its occupancy of reporting a noisy velocity. This random selection was done with replacement, so a link may report multiple measurements. These five-minute bins were used to mimic the typical procedures of data fusion methods, where probe measurements are retained and filtered into the model at the same time that loop data is next received; for PeMS, these data are reported in five-minute intervals. These velocity measurements were also contaminated with additive noise from a Gaussian distribution with a standard deviation of 10% of the measured value.

The state estimate generated by the particle filter with access only to these density measurements is shown in Fig. (c), the estimate produced using only the velocity measurements appears in Fig. 6.4(d), and the estimate generated by the RBPF with access to both the density and velocity measurements is shown in Fig. 6.4(e). We quantified the estimation accuracy by comparing the mean absolute percentage error (MAPE), i.e. the average of the quantities $\left|\hat{x}_{\ell,k}^\rho - x_{\ell,k}^\rho\right| / x_{\ell,k}^\rho$ for all links ℓ and times k, of the two runs. The results are summarized in Table 6.1. Examination of the figures and table show that qualitatively, all estimates compare well to the true state, but data fusion via the RBPF quantitatively outperforms the other two estimates in predicting density.

6.4.2 Real data on a corridor scale

We now present the results of our real-data experiment. As mentioned above, our model was calibrated against loop data from October 13, 2014. The dataset used for the experiment was

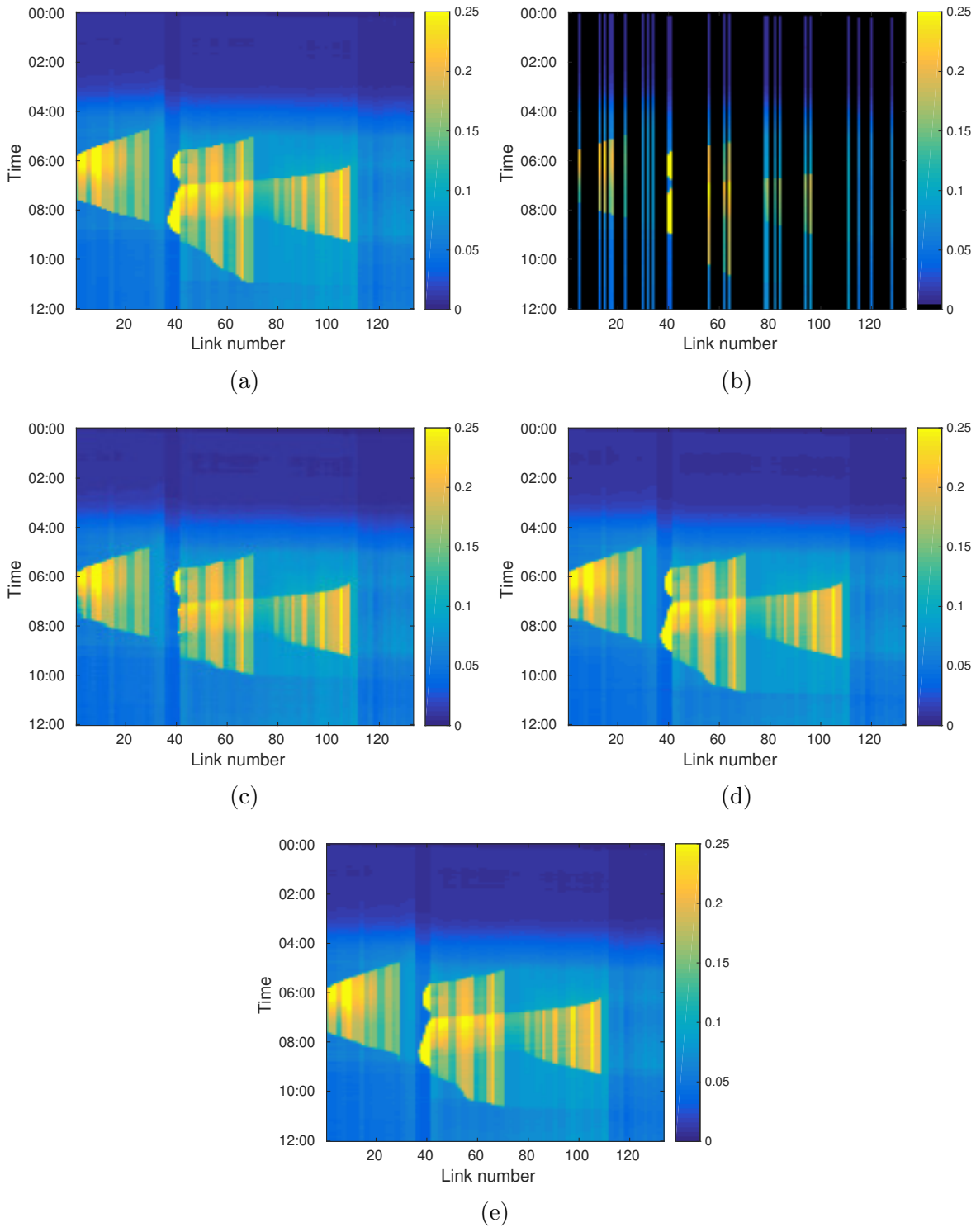
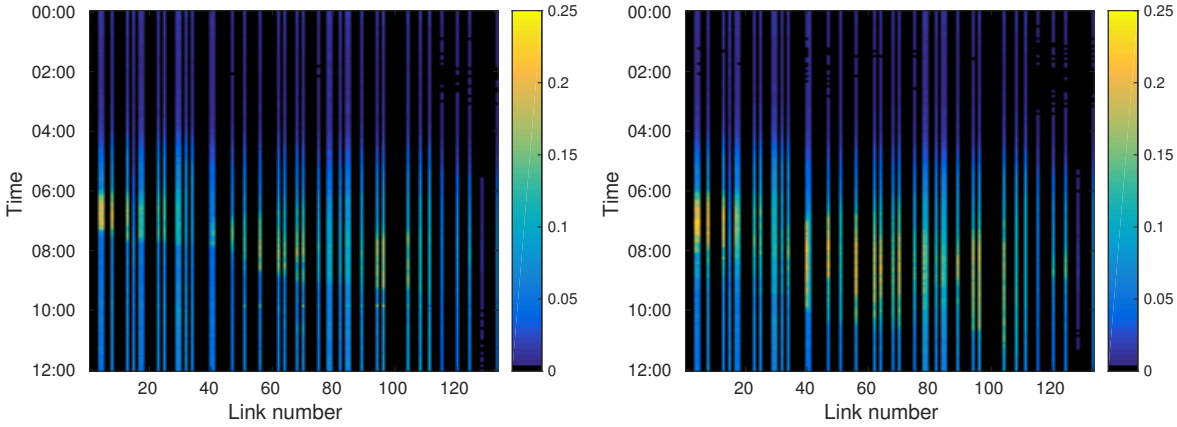


Figure 6.4: Density contour maps from a simulated experiment with loop and probe data (veh/m). Black = no data. (a) "Ground truth" simulation. (b) Simulated density measurements. (c) Estimated w/ density measurements. (d) Estimated w/ velocity measurements of 3% pen. rate. (e) Estimated w/ fused data.

Table 6.1: Density MAPE Metrics from simulation

	MAPE					
	Without Simulated Loops			With Simulated Loops		
Simulated PR[a]	Cong.[b] Links	FF[c] Links	Overall	Cong.[b] Links	FF[c] Links	Overall
None	28.57%	6.54%	11.05%	8.60%	4.00%	4.94%
1%	3.86	5.64	5.28	3.65	3.72	3.71
2%	2.89	5.66	5.09	3.39	3.69	3.63
3%	2.44	5.88	4.93	2.52	3.72	3.47

[a] Penetration Rate [b] Congested [c] Freeflow



(a) (b)

Figure 6.5: PeMS loop data of the site (veh/m). Black = no data. (a) Oct 13, 2014 (calibration date). (b) Oct 22, 2014 (test date). Note that the traffic patterns exhibit broader congestion periods on Oct 22.

recorded on October 22, 2014. The loop data used were obtained from the California PeMS database (PeMS, 2019). The probe data were obtained from a major mapping data provider.

Fig. 6.5 presents the raw loop data used in this procedure. Note that the traffic behavior on October 22 was different from that of October 13, with much larger and longer-lasting areas of congestion (high density).

Of the 35 working detectors on this date, 15 were randomly selected as a "test" set whose density measurements were compared against the estimates, and the measurements of the remaining detectors were provided to the filters. In the CTM simulations, onramps with working detectors had their measured flow input to the simulation on a five-minute delay with a simple zero-order-hold assumption (that is, that the average flow for a given five-minutes would be equal to the average measured flow of the previous five minutes), with per-timestep additive Gaussian noise sampled from a Gaussian distribution with standard deviation equal

to 15% of this nominal value. All other simulation parameters were set up identically to the simulated experiment.

A problem common to attempts to use third-party probe data for traffic estimation is matching noisy GPS points to individual roads, and filtering out those points that are not sent from vehicles in transit in the direction of interest. Map-matching schemes attempt to perform this filtering by reconstructing a vehicle's trajectory along the network and matching individual reports to the mapped links. In the present experiment, though, our network is very simple; we only seek to estimate the state along the mainline, so we used a simpler probe point filtering scheme: non-overlapping rectangular bounding boxes were drawn around each link and probe points were assigned to the link whose bounding box they fell within, if any. Further, probe points with reported headings outside of 15° of a link's end-to-end bearing were discarded. To filter out erroneous data, individual measurements y_i that evaluated to a likelihood $g(y_i|x_{i,p}) \approx 0$ for all particles p were excluded from the calculations.[1] Such measurements may be the result of, e.g., parked cars within the bounding box or faulty equipment.

The probe data were tagged with a hashed device identifier; after our crude map-matching scheme, probe points associated with 2613 unique devices remained. Over this same 12-hour period, the loop detectors along the freeway reported an average cumulative flow of approximately 182,000 vehicles along the mainline, resulting in an estimated penetration rate of roughly 1.42% of our probe data.

Results are shown in Fig. 6.6. It is known that traffic models are limited in open-loop (that is, without data filtering) prediction accuracy due to factors including day-to-day variability in driver and road characteristics and a low signal-to-noise ratio exhibited in freeway offramp and onramp measurements (i.e., PDE boundary conditions); Fig. 6.6(c) shows a baseline simulation using as-detected on- and off-ramp flow values from Oct 22, but no mainline measurements, to demonstrate a base level of error to which filtered estimates may be compared against. Qualitatively, one sees that the use of the probe data (shown in Fig. 6.6(a)) produces a density estimate with larger and longer-lasting congestion periods (compare Figs. 6.6(d), 6.6(f)). As a specific example, we wish to direct the reader's attention to the congestion patterns in the first $\tilde{1}0$ links. Note that in Fig. 6.5(b), a period of high density occurs in the first few detectors during 6-8 AM. These detectors are excluded from the simulation (Fig. 6.6(b)), and accordingly the density-only particle filter does not predict that congestion occurs in these links during this period (Fig. 6.6(d)). There exist probe measurements during that period that show low velocity (Fig. 6.6(a)), and when the PF is run with these measurements available, congestion is predicted (Fig. 6.6(e)).

Note that after 10 AM on the same few detectors, the congestion wholly clears in reality (Fig. 6.5(b)). However, the RBPF still sees a number of probe data that continue to report low velocity (observe the cloud of blueish points in this area in Fig. 6.6(a)). With no loop data

[1]This idea of rejecting faulty or outlier measurements y_i by evaluating their likelihood in the per-particle likelihoods – $g(y_i|x_{i,p})\ \forall p$ – is expanded upon in chapter 7. In particular, the scheme discussed here is the inspiration for the "Fisherian" significance-testing particle filter we discuss in that chapter.

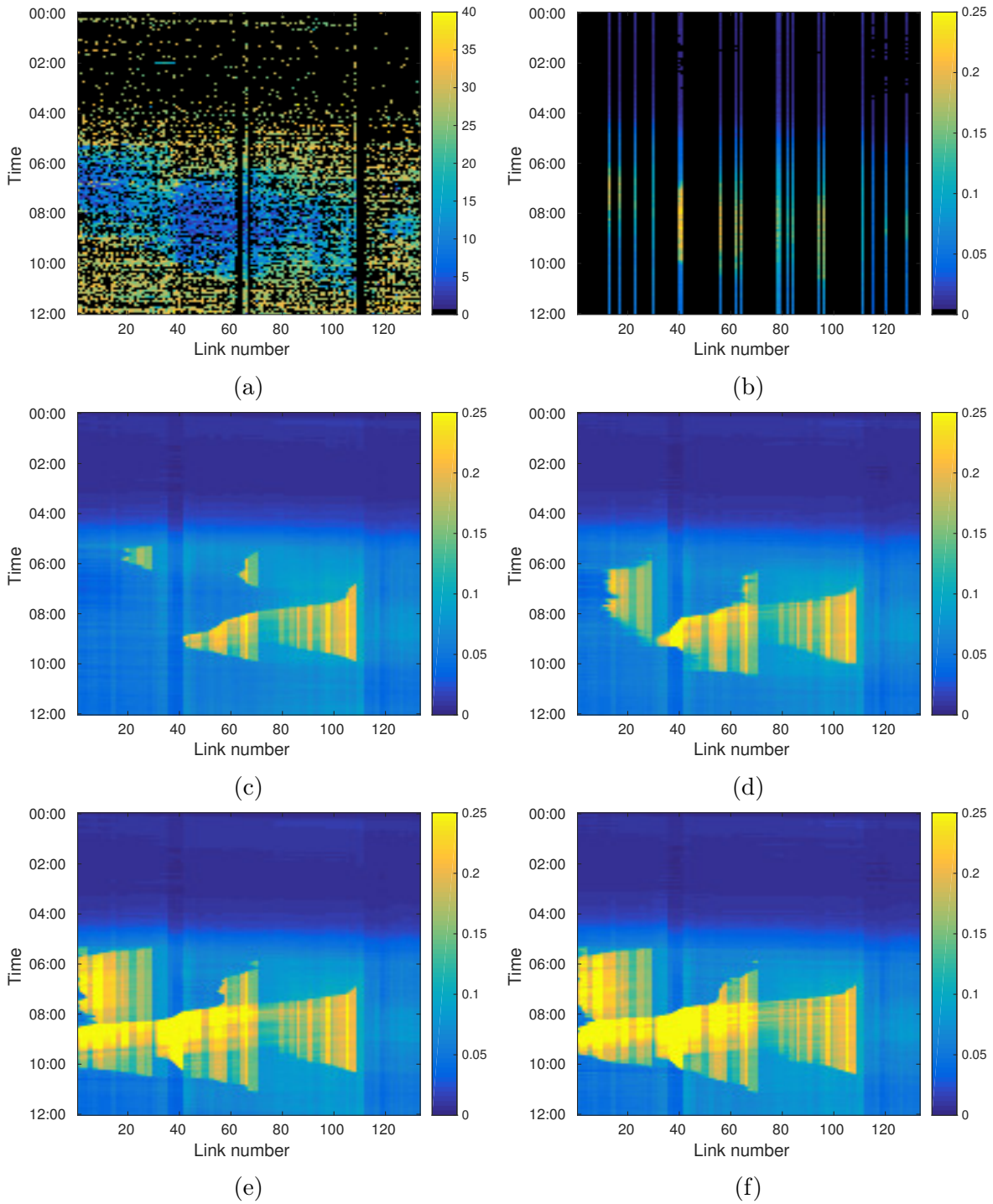


Figure 6.6: Density estimation of traffic density of the corridor on Oct 22, 2014 with real data. Black = no data. (a) Raw probe speeds (m/s). (b) Loop density subset provided to filters (veh/m). (c) "Best-case" open-loop prediction (veh/m). (d) Estimation with loop data only (veh/m). (e) Estimation with probe data only (veh/m). (f) Estimation with data fusion (veh/m). Fusing the data produces an estimate with intermediate-breadth congestion periods (f). All filtering experiments outperform the congestion periods produced with an optimal open-loop prediction (c). C.f. Fig. 6.5(b).

to counteract this, the RBPF continues to estimate congestion until the cloud of low-velocity probe data clears (Fig. 6.6(f)). Recall that in our construction of the RBPF likelihood functions discussed at the beginning of this section, the probe measurements are taken to be distributed symmetrically about the CTM-predicted velocity $\bar{x}_k^v$ with exponentially-decaying outliers (i.e. tail behavior) due to the Gaussian assumption. However, the empirical evidence here shows that, at least in this location, this is not a good assumption. Instead, a large number of probe data reported slow velocities despite the loop detectors in the area reporting low density. The RBPF then estimated a region of high density that would be in agreement with high density that would be in agreement with the low-velocity probe data through (6.1). This suggests the need for investigation into the tail behavior and non-stationarity (in the timeseries sense) of the PDF $p_\theta(y_{k,L(i)}^v|\bar{x}_{k,L(i)}^v)$.

Data fusion generally outperformed use of the disjoint sets in estimating the density measurements for most detectors in terms of MAPE metric, despite oversensitivity to probe data resulting in errors.

Table 6.2: Density MAPE Metrics for real-data experiment

	MAPE			
Detector ID	Open-loop simulation	Loop data only	Probe data only	Fused data
774012	39.20%	28.38%	32.76%	27.20%
717599	39.32	27.54	31.77	27.00
717634	38.76	36.19	34.70	32.87
718210	38.93	35.17	28.39	25.62
717664	46.24	35.29	37.40	31.73
717669	32.56	19.75	22.53	11.43
764146	44.08	35.95	26.82	21.89
717649	18.08	14.72	15.29	11.76
717653	26.85	22.37	21.05	16.48
717637	24.46	25.70	24.03	25.81
717675	38.32	53.57	35.84	32.21
772918	32.05	56.80	33.20	30.16
717688	29.94	59.74	31.29	27.49
761374	52.94	44.44	40.75	35.63
772888	30.89	47.14	29.19	18.49
Mean$\pm\sigma$	35.51 ± 8.96	36.18 ± 13.73	29.67 ± 6.81	25.05 ± 7.55

One may notice the high amounts of error in these results compared to the simulated results in Table 1. This is due to two factors. First, the CTM is a relatively simple model of traffic, and while it can easily capture broad congestion patterns, it cannot reproduce some higher-order traffic phenomena to high accuracy. Second, the "ground truth" density

measurements the estimates are compared against are not actually the true density values, but rather the noisy measurements from the loop detectors. This type of loop detectors is known to be noisy (Chen et al., 2003). We did not attempt to denoise the loop measurements. A low MAPE value would thus require that the model estimate reproduce the loop's sensor noise, which we feel is infeasible.

6.5 Conclusions and future work

This work introduced a filtering scheme for tractably estimating vehicle density while assimilating probe velocity data in the structure of a Rao-Blackwellized particle filter by exploiting some conditional independence assumptions. Use of these assumptions led to a model that implicitly estimated vehicle velocity for the purposes of filtering, which we referred to as a "pseudo-second-order" model. We demonstrated the effectiveness of our model for a long freeway corridor.

Our numerical experiments were based on a freeway, but the particle filter itself is not restricted to traffic networks of this rigid structure, and is applicable to more complex networks. Of particular interest might be interconnected urban networks; these networks typically have lower fixed detection infrastructure then freeways, and our second experimental result of solely velocity data being used to estimate density is encouraging for this application. Any application, though, would have to overcome difficulties in accurately matching probe measurements to individual road segments, which is itself a difficult problem.

Immediate theoretical avenues for investigation that present themselves are estimation of the PDFs $p_\theta(\bar{x}|x^\rho)$ and $p_\theta(y^v|\bar{x}^v)$ in (6.7) (denoting the distribution of a road segment's average velocity about a model-predicted velocity, and the distribution of individual vehicle measurements about a link's average velocity, respectively). Typically these PDFs have taken assumed form (Sumalee et al., 2011), but estimation of these distributions from data would allow for more accurate filtering models. Another item of immediate applicability is the use of a particle smoother to recreate the state trajectory while taking into account future probe measurements, but given the high dimensionality of traffic network models, this would not be a straightforward application either.

As we mentioned in section 6.4.2, working with real data necessitated the use of some rejection scheme to discard measurements that were clear outliers. This need motivated the development of the "hypothesis-testing" particle filters we will present in the next chapter.

Chapter 7

A Framework for Robust Assimilation of Potentially Malign Third-Party Data, and its Statistical Meaning

7.1 Introduction

Intelligent transportation systems (ITS) have long relied on the use of real-time data to enable reactive and proactive operations and control. The widespread and growing use of real-time data, however, brings to ITS a problem that affects many domains in information sciences and engineering: these systems and methods can be fragile when their data are incorrect, either due to faults in the sensors or a feeding-in of malicious data by a hostile attacker ("spoofing").

ITS researchers have shown that existing real-time control schemes are sensitive to errors in data. Some recent research even shows that faulty data can lead to actively harmful control. These vulnerabilities exist at both the small-scale, individual-vehicle level, and the multi-vehicle, infrastructural coordinative level. At the smaller scale, for example, Bhatti and Humphreys (2017) recently demonstrated the capability to drive a ship off-course via spoofed global navigation satellite system (GNSS) signals, evading detection by both the crew and a statistical spoofing detector. At the broader, infrastructural level, Reilly et al. (2016) showed how common road traffic control systems and algorithms (e.g., ramp meters and the programs that control the metering rate in response to observed traffic volumes) can be manipulated into causing complex and costly congestion patterns by taking control of their input data.

In this chapter, we focus more on the larger-scale end of this spectrum. Types of ITS applications at this scale include the above-mentioned road traffic control systems, as well as fleet management and tracking systems in industry. Public and private management entities have both been quick to adopt the use of data from GNSS due to their ubiquitous availability and – especially for public bodies that wish to avoid the need for expensive installation

and maintenance of sensing infrastructure – relatively low cost (Kurzhanskiy and Varaiya, 2015). Many authors in the ITS community have investigated the use of vehicle-carried GNSS transponders for real-time road traffic observation and control (Work et al., 2010; Lovisari et al., 2016; Seo et al., 2017; Ferrara et al., 2018).

As mentioned in footnote 1 of chapter 6, the work described in the present chapter was originally inspired by technical problems we encountered in the work described in chapter 6. In that previous work (and the subsequent paper Wright and Horowitz (2017b), which the present chapter represents a further refinement of), we reported on our efforts to use anonymized *third-party* data from connected vehicles to estimate the state of traffic on a freeway. That is, the assimilation of records consisting of times, positions, and speeds from transponders near the freeway, but without certainty of the correctness of the data. For example, upon manual inspection, several records showed transponders with near-zero speeds in times and spaces we believed were not in congestion (e.g., possibly a stopped car), unrealistically fast movement, or speeds that better matched the congestion patterns on the freeway's opposite direction. Using a standard particle filter (Gordon et al., 1993) for state estimation, when some data are of very low probability, led to divergence of the state estimate from the true state, and in some extreme cases, numerical errors caused by floating-point underflow. In those situations, we want to be able to reject these data that would not improve our state estimate, in a principled manner.

We also sought to develop a method that could reject these malign measurements without having models for all types of faulty data. This work describes two modifications to a familiar estimation algorithm, one applicable to the situation where a model of faults exists; and one where such a model does not exist, and the engineer only has a model for the sensors' correct behavior. These two modifications are based on two different mathematical theories of hypothesis testing.

In the broader picture, we argue that robustness to faulty data is essential to ITS schemes that make use of GNSS. ITS schemes often make use of GNSS position, velocity, and time (PVT) measurements, which are susceptible to many sources of error. These error sources include multipath propagation, non-line-of-sight tracking, signal blockage, tropospheric and ionospheric conditions, and a multiplicity of navigation filter implementations (Jin et al., 2014). In other words, the presence of noise or faults in GNSS data for ITS could be considered the norm, rather than the exception, and system robustness to both modeled and unmodeled faults is desirable.

The rest of this chapter is organized as follows. Section 7.2 briefly reviews the theoretical and historical background of hypothesis testing (which forms the core of our robustification), and introduces the two most common frameworks: *Fisherian* and *Neyman-Pearsonian*. Section 7.3 goes into the mathematical details of the two frameworks, and describes modifications necessary to apply them to a Monte Carlo scheme like the particle filter. Section 7.4 merges the standard particle filter as discussed in chapter 2 with our testing frameworks developed in section 7.3. Section 7.5 recalls our motivating problem of freeway traffic state estimation using third-party data, and presents some simulation results of the two testing-robustified particle filters on this difficult nonlinear estimation problem. Section 7.6 concludes with some

discussion on what we feel is this method's interesting fusion of data and model.

7.2 Statistical Testing: An Introduction

Most readers of scientific literature are familiar with hypothesis testing in the form of reports of "p-values" and "statistical significance" in the context of discussions of, e.g., medical research. The most popular form of hypothesis test is the so-called "null hypothesis significance test" (NHST) (Schneider, 2015). In a NHST, a null hypothesis of, loosely speaking, "no relation" or "no correlation" is proposed. Then, a p-value for the data under this null hypothesis is computed, and if it is less than a hard boundary of, e.g., 5%, the test is said to have shown "statistical significance," and a specified alternative hypothesis is accepted. The NHST is actually a fusion of two distinct theories (Schneider, 2015): *significance testing*, due to Fisher (Fisher, 1925, 1935a,b), and *hypothesis testing*, due to Neyman and Pearson (Neyman and Pearson, 1928, 1933a,b).

It should be noted that concepts that are rooted in one of the two statistical testing frameworks, but do not make sense in the other, are often discussed alongside each other in the NHST presentation. For example, the Fisher framework only considers one hypothesis, the null hypothesis. On the other hand, in the Neyman-Pearson model, multiple hypotheses exist, along with Type I and Type II (also called false positive and false negative, respectively) error rates and statistical power, but p-values are absent (p-values are explicitly defined only in the Fisherian framework) (Schneider, 2015).

The implications of this dichotomy are more than just philosophical and terminological: for some problems, strict adherence to one theory will lead to a different statistical test than would be derived using the other (see Lehmann (1993) for more discussion on the degree of these differences).

7.3 Statistical Testing for Measurement Rejection

7.3.1 Notation

For this section, where we review classical tests for measurement rejection and introduce new Monte-Carlo-based tests, we will use a somewhat simpler notation.

Suppose that we have data $D = \{d_1, \ldots, d_n\}$, which come from a distribution with PDF $p_\theta(D)$. We use D instead of the classical X for data to avoid confusion with our system state variable. The PDF has an unknown parameter (or set of parameters) θ. The testing problem is to evaluate the likelihood of our data for certain values of θ and make decisions about whether those θ values should be used or not.

The notation for our data D will be updated in Section 7.4, when we combine this section's results with the particle filter.

The remainder of this Section deals with the mathematical details of both the Fisherian and Neyman-Pearsonian theories described above. We will begin with the Neyman-Pearson

framework as its basic elements are likely more familiar to a reader with an applied knowledge of statistics.

7.3.2 Neyman-Pearsonian "hypothesis testing"

In this framework, in addition to our data D and PDF $p_\theta(D)$, we have two competing hypotheses: $H_0 : \theta = \theta_0$ and $H_1 : \theta = \theta_1$. In this case, where both hypotheses fully specify the form of the likelihood $p_\theta(D)$ (since each hypothesis consists of only a single point for θ), a ratio of the two hypotheses' likelihoods might take the form

$$\Delta(D) = \frac{p_{\theta_1}(D)}{p_{\theta_0}(D)}. \tag{7.1}$$

A hypothesis test in this case is often called a "simple-vs-simple" hypothesis test (a simple hypothesis is one that fully specifies the model parameters). For the remainder of this discussion, we will focus on the simple-vs-simple tests. The formal extension to compound hypotheses is a part of future work.

A well-known result called the Neyman-Pearson Lemma (Neyman and Pearson, 1933a) states that, for a given simple-vs-simple hypothesis testing problem, the optimal test (in that it minimizes the Type II error rate among all tests with a given Type I error rate[1]) is a *likelihood ratio test*. A likelihood ratio test is one where the likelihood ratio $\Delta(D)$ is the test statistic of interest. A likelihood ratio test using the likelihood ratio given in (7.1) has the form

$$\psi(D) = \begin{cases} 1 & \text{if } \Delta(D) < c \\ 0 & \text{if } \Delta(D) > c \end{cases} \tag{7.2}$$

for some constant c. The test prescribes that, of the two hypotheses, we accept $H_{\psi(D)}$.

The constant c in (7.2) is chosen to set the likelihood ratio test to have a certain Type I error rate. This selected Type I error rate is called the test's *significance level* and usually given the symbol α.

When the integral is computable, the constant c just mentioned is found by solving for it as a function of α in

$$\begin{aligned} E_{\theta_0}\psi(D) &= \int \psi(D) p_{\theta_0}(D)\, dD \\ &= P_{\theta_0}(\Delta(D) < c) = \alpha \end{aligned} \tag{7.3}$$

where we use the fact that the expectation of an indicator function (e.g., $\psi(D)$ (7.2)) is equal to the integral of the PDF p_{θ_0} over the set where the indicator function equals one.

Of critical importance in (7.3) is that the PDF of integration for $\psi(D)$ is the likelihood under H_0. This is because the Type I error rate is defined as a rejection of H_0 *when* H_0 *is*

[1]The Type I ("false positive") error rate, P(Reject $H_0|H_0$ true), is the mathematical probability that H_0 is rejected, conditioned on it being true; and the Type II ("false negative") error rate, P(Accept $H_0|H_0$ false), is the probability that H_0 is accepted, conditioned on it being false.

actually true. This choice is made because, conventionally, H_0 represents the status quo, or a prior belief about θ before any evidence, and practitioners are interested in tests that have a small error probability when H_0 is correct (Keener, 2010).

Once we have selected our α (and therefore our c), we collect the data D, evaluate our likelihood ratio (7.1), and select either H_0 or H_1 depending on the value of $\psi(D)$ (7.2).

7.3.3 Monte-Carlo Neyman-Pearsonian Likelihood Ratio Tests

We now turn to how we must modify the Neyman-Pearson likelihood ratio test framework for use in our Monte Carlo framework. Suppose that rather than a known likelihood $p_\theta(D)$, we only have a set of samples $d_i \sim p_\theta(D)$, $i \in \{1, \ldots, n\}$, as well as their associated likelihoods under the null hypothesis $p_{\theta_0}(d_i)$. Then, while we cannot solve the integral in (7.3) in closed form, we can still approximate it via Monte Carlo simulation,

$$\begin{aligned} E_{\theta_0}\psi(D) &= \int \psi(D) p_{\theta_0}(D)\, dD \\ &\approx \sum_{i=1}^{n} \psi(d_i) p_{\theta_0}(d_i) \\ &= \hat{E}_{\theta_0}\psi(D). \end{aligned} \tag{7.4}$$

Designing our test for a specific significance level (i.e., choosing c) in this case does not make sense, due to a lack of a closed-form integral equation in which to solve for c as a function of α. Instead, we propose to select either H_0 or H_1 based on the observed likelihood ratio statistic directly. The general idea is as follows. Considering (7.3) under the classic Neyman-Pearson framework, we would select c so that an α fraction of the probability mass distributed by $\hat{p}_{\theta_0}(D)$ returns values of D s.t. $\psi(D) = 1$ (this is what is shown in the second line of (7.3)). Under the empirical approximation (7.4), on the other hand, we have a finite amount of points, and can easily compute whether $\Delta(d_i)$ is below or above 1 for every i. The distribution of the probability mass in the empirical distribution is itself defined by our known sample weights $p_{\theta_0}(d_i)$. Therefore, we can determine whether at least an α portion of the probability mass under the empirical distribution recommends selecting H_1 by simply noting which samples have a higher likelihood under H_1 than under H_0, and then adding up those samples' H_0-probabilities $p_{\theta_0}(d_i)$ and seeing whether this sum is greater or smaller than α.

Mathematically, the empirical Neyman-Pearson likelihood ratio test we propose is

$$\hat{\psi}(D) = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} 1_{\{\Delta>1\}}(d_i) p_{\theta_0}(d_i) < \alpha \\ 0 & \text{otherwise} \end{cases} \tag{7.5}$$

where

$$1_{\{\Delta>1\}}(d_i) = \begin{cases} 1 & \text{if } \Delta(d_i) > 1 \\ 0 & \text{otherwise} \end{cases} \tag{7.6}$$

is an indicator function of whether d_i shows a higher likelihood under H_1 than under H_0.

7.3.4 Fisherian "significance testing"

As mentioned above, the Fisherian formulation differs from the Neyman-Pearsonian one in several ways. One important difference is that it specifies the selection of only a single hypothesis, the null hypothesis H_0, which specifies the PDF as $p_{\theta_0}(D)$. Usually, in a significance test, the null chosen is meant to be more interesting than the common "no relationship" hypothesis test, and reflects some *a priori* knowledge. Observing that the data D do not fit well with the null hypothesis H_0 is meant to lead to reconsideration and indication to the practitioner that their prior assumptions used in crafting H_0 should be re-evaluated (Schneider, 2015).

The Fisherian framework calls for the calculation of a test statistic of the data, $T(D)$. Unlike the simple-vs-simple case in the Neyman-Pearson framework (where we can apply the Neyman-Pearson Lemma), the optimal test statistic is not immediately given. Instead, its form depends on the particular form of the likelihood $p_{\theta_0}(D)$.

Here, we will assume that the null hypothesis H_0 is simple in the Neyman-Pearsonian sense, in that it fully specifies the likelihood: $H_0 : \theta = \theta_0$. Then, we want to compute the tail probability of the observed test statistic $T(D)$ under H_0. This quantity is the p-value. Whether this tail probability will be a one-sided or two-sided value again depends on the particular form of the PDF $p_\theta(D)$. For the particular example of a two-sided test, the p-value (assuming $T(D) \in \mathbb{R}$) will be

$$\begin{aligned} &\text{p-value} = P_{\theta_0}\left(T < -|T(d)|\right) + P_{\theta_0}\left(T > |T(d)|\right) \\ &= \int_{-\infty}^{-|T(d)|} p_{\theta_0}(T(D))\, dD + \int_{|T(d)|}^{\infty} p_{\theta_0}(T(D))\, dD \end{aligned} \tag{7.7}$$

where $p_{\theta_0}(T(D))$ is the PDF of the statistic T (also called the statistic's sampling distribution) under H_0 and the lower-case formatting of d in $T(d)$ indicates that it is the actually-observed value of the random statistic $T(D)$.

For some common sampling distributions like the univariate Gaussian, Student's t, and χ^2 distributions, the solutions to the tail probability integrals in (7.7) are available in the familiar statistical testing reference tables, or can be quickly computed via statistical software.

7.3.5 Monte-Carlo Fisherian Significance Tests

We can move from the theoretical framework of continuous integrals with closed-form solutions (7.7) to the Monte Carlo framework using similar arguments as in Section 7.3.3.

Considering (7.7), we see the same type of integral we had in (7.3) (recalling that the test term $\psi(D)$ (7.2) acted as an indicator function for a one-sided interval, effectively performing the same function as the one-sided limits of integration in (7.7)). Therefore, using similar

arguments as before, we can get an approximation of the Fisherian p-value from (7.7) (noting the finite integration bound $T(d)$ is replaced with the approximation $\widehat{T}$) as

$$\widehat{\text{p-value}} = \int_{-\infty}^{-|\widehat{T}|} p_{\theta_0}(T(D))\, dD + \int_{|\widehat{T}|}^{\infty} p_{\theta_0}(T(D))\, dD \tag{7.8}$$

where

$$\widehat{T} = \sum_{i=1}^{n} T(d_i) p_{\theta_0}(d_i). \tag{7.9}$$

Like in the Monte Carlo Neyman-Pearson framework, we have again created a weighted-average statistic using our weighted-average approximation of the PDF for the data.

Fisher himself advocated against the use of fixed levels and hard accept/reject boundaries, but instead suggested reporting the p-values directly (for more on this contrast, see, e.g., the discussions in Lehmann (1993, Sec. 4), Schneider (2015, p. 415)). However, for our current purposes this is not easily implementable because in our filtering context, we are trying to make a decision as to whether to accept or reject a measurement as we receive it. Taking a "soft" view, and considering a range of state space values based on our current range of belief of whether we should accept or reject H_0, while potentially giving us a view of a broader range of possibilities (and separate measurement hypotheses) that we could revisit in light of future data, leads to a blow-up when the number of separate sensors increase (Wright and Horowitz, 2017b). Instead, for expedience, in what follows we adopt the Neyman-Pearson and null hypothesis significance test and select a hard significance level α, and reject or accept the measurement based on whether our estimated p-value (7.8) is larger or smaller. A more "inductive" approach, closer to the spirit of the original Fisherian significance test, that updates H_0 based on repeated tests of measurements from the same sensor, is an avenue for future work.

7.4 A Probabilistic Outlier-Rejecting Particle Filter

This section unifies the particle filtering framework presented in Section 2.1.3 and the hypothesis testing methods developed in Section 7.3. As mentioned in Section 7.3.1, we will unify these ideas in the notation of the state estimation problem. In this work, we will not exhaustively define all PDFs of interest in the interest of readability. See Wright and Horowitz (2017b) for a more lengthy discussion of a precursor to the Fisher-type hypothesis-testing particle filter discussed in the present work.

Recall the filtering or update step in the particle filtering algorithm (2.8). In our prior discussion, we considered a measurement likelihood $g_\theta(y_k|x_k)$ (2.2b). In this notation, we are stating that the random measurement vector $Y_k|X_k$ has a joint distribution across all dimensions. This makes sense if the measurement noises of the different elements of the measurement vector are correlated or otherwise dependent.

Considering our problem of needing to assimilate data from multiple third-party sensors, though, it makes sense to assume a conditional independence of the measurements. If we say that at time k, we receive measurements from M sensors, with sensor j's measurement being the random variable Y_k^j, then by assuming conditional independence of the sensors given X_k, we can write

$$Y_k|(X_k = x_k) \sim g_\theta(y_k|x_k) = \prod_{j=1}^{M} g_\theta^j(y_k^j|x_k). \tag{7.10}$$

This conditional independence assumption is the same one we made in Assumption 6.2 and used in the previous chapter. Examining (7.10), we notice that we have factored our particle filter likelihood (2.8) into per-sensor PDFs.

Considering a sensor likelihood $g_\theta^j(y_k^j|x_k)$, we can parameterize its nonfault, faulty (for one or more known types of fault, if applicable), spoofed, etc. behavior in θ. And, if we have models for one of these behaviors, we can accordingly form hypotheses: $H_0 : \theta = \theta_0$, $H_1 : \theta = \theta_1$, etc. This is how we bring together the particle-filtering and Monte Carlo fault-detection theories. Each individual particle, which has a value of the random variable $Y_k^j|X_k$ and a probability (the particle's weighting in the collection of particles), serves as a sample (a d_i in Section 7.3's terminology). Repurposing these particles as datapoints for the expected behavior of the data under the stated hypotheses lets us reject, in real time, measurements that do not match our prior models for data that would come from a correctly-functioning sensor.

Based on the above discussion, the general framework for the robustified particle filter is given below.

1. Perform a prediction step as normal, using (2.7).

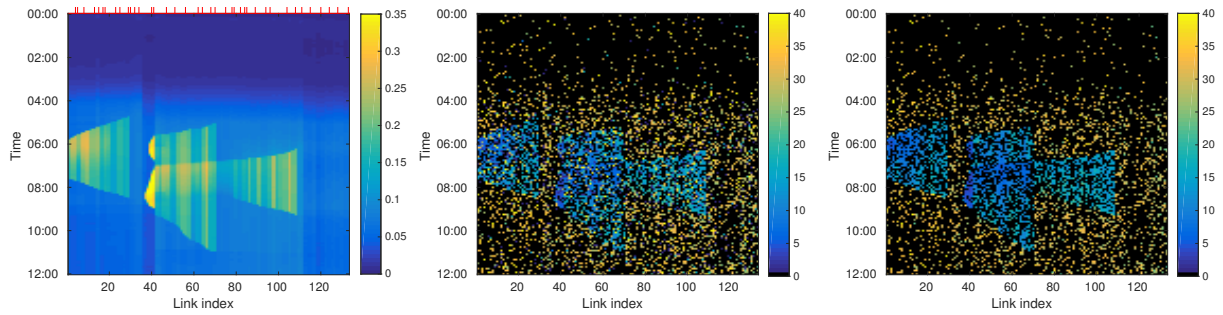
2. For each sensor j at time k, calculate either the likelihood ratio or Fisher test statistic, depending on whether a Neyman-Pearson or Fisher test is used.

3. For each sensor, determine whether to reject it as faulty using the relevant hypothesis test for the actually-observed measurement and selected α.

4. Perform an update step with the non-rejected measurements using (2.8) (and, if desired, a resampling step).

5. Advance in time, $k \leftarrow k + 1$, return to step 1, repeat.

The type of test (Fisherian or Neyman-Pearsonian) to select for each situation and each sensor is dependent on the problem circumstances. We believe that it generally makes sense to favor a Neyman-Pearson-type test when one has trustworthy models for all reasonably-expected types of faults, and a Fisherian test when one does not. Our example case study presented in the next section shows some results for different types of tests and different fault models of varying accuracy.



(a) (b) (c)

Figure 7.1: Simulated true density state trajectory (veh/m) (a), speed measurements (m/s) (b), and non-faulty subset of speed measurements (m/s) (c) used in simulation. Traffic moves to the right, and the time period considered is midnight to noon (as marked on the vertical axis). In (a), the links instrumented with loop detectors that noisily measure density are marked with red ticks. At peak morning demand, bottlenecks near links 30, 70, and 110 lead to traffic jams that propagate upstream (i.e., they extend to the left as time advances), leading to increased density and lower speed. The jams later dissipate as demand falls.

7.5 Example Application

We now present an application of the outlier-rejecting particle filters to the freeway state estimation problem considered in chapter 6. Specifically, we consider a generalization of the simulated-data experiment from section 6.4.1 where some of our simulated velocity measurements are outliers, and we attempt to reject them.

7.5.1 Implementation details

In the case study first introduced in chapter 6, we consider a 19-mile portion of I-210 West in southern California. As our system model $f_\theta(\cdot)$, we make use of the macroscopic Cell Transmission Model (CTM) (Daganzo, 1994, 1995), which approximates traffic as compressible fluid flows. This type of model can capture important nonlinear emergent features in traffic flows like traffic jams and congestion waves.

For more details on the basics of the CTM, refer to section 2.2.

In the application here, as in chapter 6, when there is an onramp and/or an offramp between links ℓ and $\ell + 1$, we determine the flow $q_{\ell,k}$ according to the junction model of Muralidharan et al. (2009). The ramp flows themselves are random variables.

Like in chapter 6, we perform freeway density estimation using simulated (noisy) density measurements from loop detectors and velocity measurements simulated as coming from vehicle-carried GNSS devices. As in equation 6.1, in the CTM the speed of traffic in link ℓ at time k is

$$v_{\ell,k}^{\text{CTM}} = L_l \cdot \frac{\rho_{\ell,k}}{q_{\ell,k}}. \tag{7.11}$$

where in this discussion we use the superscript "CTM" to distinguish that it is the link velocity predicted by the CTM model's fundamental diagram. A high vehicle density leads to congestion, and hence low speeds.

We deviate from chapter 6 in the specifics of the velocity measurement model. In addition to noisy density measurements from 41 loop detectors, we simulated GNSS speed measurements with a simulated penetration rate of 2% (i.e., each vehicle had a 2% chance of noisily reporting its speed). To generate the faulty third-party measurements, we gave each speed measurement a 30% probability of being faulty. We used two fault models: a faulty measurement had a 1/3 probability of reporting zero (i.e., a stopped car misreporting its location), and a 2/3 probability of drawing from a Gaussian distribution with mean 30 m/s and standard deviation 10 m/s. The non-fault model for velocity measurements, $g^j_{\theta_0}(\cdot)$, was Gaussian with a mean of the true link velocity and standard deviation of 20% of the mean (similar to Work et al. (2010)). We simulated a realization of our freeway model, with randomness introduced by the random onramp, offramp, and upstream boundary flows, to serve as the "true" state. Figure 7.1 shows the true state and velocity measurements used.

7.5.2 Derivation of the statistical tests

We present the derivation of the two statistical tests (Fisherian and Neyman-Pearsonian) for this problem. The tests derived here are the ones we use in item 3 in the algorithm presented in section 7.4.

7.5.2.1 Fisherian test

As mentioned, the non-fault model for a velocity measurement in link ℓ on timestep k here is

$$g_{\theta_0}(v_{\ell,k}|x_k) = \mathcal{N}\left(v^{\text{CTM}}_{\ell,k}, \left(0.2 \cdot v^{\text{CTM}}_{\ell,k}\right)^2\right) \tag{7.12}$$

where $v^{\text{CTM}}_{\ell,k}$ is calculated as in (7.11).

Given one particular velocity measurement $v_{\ell,k}$, we want to perform a test to conclude whether or not it came from this g_{θ_0}. The archetypical test statistic for Gaussian distributions with unknown means and known or dependent variances (as in (7.12)), is the z-statistic, which in this case is

$$z_{H_0}(v_{\ell,k}) = \frac{v_{\ell,k} - v^{\text{CTM}}_{\ell,k}}{\left|0.2 \cdot v^{\text{CTM}}_{\ell,k}\right|}.$$

We need to calculate the value of this test statistic under our null hypothesis H_0. We approximate this with (7.9), which means our Monte Carlo estimate of our test statistic is

$$\hat{z} = \sum_{p=1}^{P} \frac{v_{\ell,k} - v^{\text{CTM},p}_{\ell,k}}{\left|0.2 \cdot v^{\text{CTM},p}_{\ell,k}\right|}\, p(x^p_k|y_{0:k-1})$$

where $v_{\ell,k}^{\mathrm{CTM},p}$ means the value of $v_{\ell,k}^{\mathrm{CTM}}$ for particle p, and recall from section 2.1.3 that $p(x_k^p|y_{0:k-1})$ is the probability of particle p conditioned on all measurements up to time $k-1$.

We choose a two-sided significance test. We need to use (7.8) to compute the p-value. Again, it is known that the z-statistic's sampling distribution is a standard Gaussian distribution. So, our estimated p-value is

$$\widehat{\text{p-value}} = \Phi\left(-|\hat{z}|\right) + \left(1 - \Phi\left(|\hat{z}|\right)\right) = 2\Phi(-|\hat{z}|) \tag{7.13}$$

where $\Phi(\cdot)$ is the standard Gaussian cumulative distribution function.

Finally, the p-value estimated in (7.13) is compared to our preselected significance level α. If $\widehat{\text{p-value}} < \alpha$, we reject the measurement $v_{\ell,k}$.

7.5.2.2 Neyman-Pearsonian hypothesis test

This test's derivation is simpler than that of the Fisherian one presented in the previous section. The one complication is that, in our presentation in section 7.3, we assumed simple hypotheses. In this particular case, our fault hypothesis is a compound hypothesis (i.e., we have two proposed fault cases). Therefore, we need to generalize to the so-called "generalized likelihood ratio test," which permits non-simple hypotheses. In this situation, our hypotheses are

$$H_0 : = g(v_{\ell,k}|x_k) = \mathcal{N}\left(v_{\ell,k}^{\mathrm{CTM}}, \left(0.2 \cdot v_{\ell,k}^{\mathrm{CTM}}\right)^2\right) \tag{7.14a}$$

$$H_1 : g(v_{\ell,k}|x_k) \in \{\mathcal{N}\left(30, 100\right), \{0 \text{ with prob. } 1\}\}. \tag{7.14b}$$

We see that H_0 has the same likelihood as in equation (7.12), and H_1 consists of the two models from which we generated our faulty data.

For the generalized likelihood ratio test, we compute our likelihood ratio $\Delta(v_{\ell,k})$ by taking the maximum likelihood across any non-simple hypotheses. That means that our likelihood ratio here is

$$\Delta(v_{\ell,k}) = \frac{\mathcal{N}(v_{\ell,k}; v_{\ell,k}^{\mathrm{CTM}}, \left(0.2 \cdot v_{\ell,k}^{\mathrm{CTM}}\right)^2)}{\max\left(\mathcal{N}(v_{\ell,k}; 30, 100), 1_{v_{\ell,k}=0}\right)} \tag{7.15}$$

where by $\mathcal{N}(v_{\ell,k}; \mu, \sigma^2)$ we mean the likelihood value obtained by plugging in the value $v_{\ell,k}$ into the normal distribution $\mathcal{N}(\mu, \sigma^2)$.

Now, we can apply (7.5)-(7.6) to obtain our test. The final hypothesis test here is

$$\hat{\psi}(D) = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} 1_{\{\Delta>1\}}(v_{\ell,k}) \cdot p(x_k^p|y_{0:k-1}) < \alpha \\ 0 & \text{otherwise} \end{cases} \tag{7.16a}$$

for the selected level α, where

$$1_{\{\Delta>1\}}(v_{\ell,k}) = \begin{cases} 1 & \text{if } \Delta(v_{\ell,k}) > 1 \\ 0 & \text{otherwise} \end{cases} \tag{7.16b}$$

is our indicator function, and where $\Delta(v_{\ell,k})$ is from (7.15).

7.5.3 Results

We tested both the Fisherian and two instances of the Neyman-Pearsonian (each using a different fault hypothesis H_1) against this problem. The results are presented in Tables 7.1 through 7.3.

In the tables, we report both the performance of the statistical tests in rejecting faulty measurements, as well as the resulting estimation error obtained when using the non-rejected measurements. "Positives" refer to measurements for which we rejected H_0, i.e., sensors that our fault decision statistical test concluded were faulty. "True" and "False" refer to correct and incorrect decisions, respectively, of whether a measurement is faulty. "Labeling error" reports the overall percentage of incorrect decisions (False Negatives or False Positives). The estimation error is reported in terms of the mean absolute percentage error (MAPE), the average of $|\hat{\rho}_{\ell,k} - \rho_{\ell,k}|/\rho_{\ell,k}$ for all ℓ and k, with $\hat{\rho}_{\ell,k}$ the ℓth entry of $\hat{\rho}_k = \sum_{p=1}^{P} x_k^p \cdot p_\theta(x_k^p|y_k)$, i.e., the mean of the posterior particle filter PDF.

The two Neyman-Pearson estimation experiments are shown in Tables 7.1 and 7.2. As indicated in the table names, Table 7.1 shows results for a simulation where the likelihood ratio test had an incorrect faulty measurement likelihood, and Table 7.2 one with the correct faulty measurement likelihood. The faulty measurement likelihood of Table 7.1 was a Gaussian that only placed mass near zero, i.e., it was crafted to select the fake "stopped car" vehicles.

It indeed rejects the "stopped car" vehicles, but, as reflected in the relatively high labeling error rates, it does not reject the purely random measurements. On the other hand, the Neyman-Pearson fault detector with the correct H_1 model, unsurprisingly, performs better, rejecting many of the stopped-car and the purely random measurements.

The Fisherian results (Table 7.3) show that the estimation accuracy is quite sensitive to the selected α. There are much larger changes in labeling error and MAPE across the scale of α values selected than for either of the Neyman-Pearson results. This is not too surprising, as not having any alternative hypothesis H_1 to compare against, makes the p-value much more sensitive to small variations in the likelihood under H_0 than the likelihood ratio would be.

Of particular interest are the columns for $\alpha = 0.001$ and 0.01 in the Fisher results table (Table 7.3). For these values of α, we obtain results that are between the performance of the correct- and incorrect-fault-model Neyman-Pearson filters. This is an encouraging result, as it confirms our intuition that for a properly tuned α, not having a fault model can beat the performance of using an incorrect one.

As mentioned in the caption for the tables, a particle filter estimator that did not see any faulty data (i.e., the true measurement distribution was $g^j_{\theta_0}(\cdot)$) obtained a MAPE of 3.43%. Unsurprisingly, none of the fault-detecting estimators consistently managed to obtain this level of accuracy, although the Neyman-Pearson fault detector with the correct fault model did come within a standard deviation or two.

Table 7.1: Neyman-Pearson Fault Detection/Estimation (Incorrect H_1)

	$\alpha = 0.001$	$\alpha = 0.01$	$\alpha = 0.1$
True Positives	700 ±0	700 ±0	700.40±0.55
False Positives	58.40±9.50	62.80±5.72	76.60±7.02
True Negatives	4535.60±9.50	4531 ±5.72	1305.60±7.02
False Negatives	1306 ±0	1306 ±0	1305.60±0.55
Labeling Error (%)	20.67±0.14	20.74±0.09	20.94±0.10
Density MAPE (%)	3.80±0.12	3.86±0.05	3.93±0.11

Table 7.2: Neyman-Pearson Fault Detection/Estimation (Correct H_1)

	$\alpha = 0.001$	$\alpha = 0.01$	$\alpha = 0.1$
True Positives	1415.80±5.12	1433.80±1.79	1450.80±2.68
False Positives	94.80±8.07	106.20±16.80	118.20±17.61
True Negatives	4499.20±8.07	4487.80±16.80	4475.80±17.61
False Negatives	590.20±5.12	572.20±1.79	555.20±2.68
Labeling Error (%)	10.38±0.18	10.28±0.28	10.20±0.31
Density MAPE (%)	3.51±0.08	3.53±0.18	3.57±0.17

Table 7.3: Fisher Fault Detection/Estimation

	$\alpha = 0.001$	$\alpha = 0.01$	$\alpha = 0.1$
True Positives	1214.80±2.49	1294.60±4.62	1457 ±3.32
False Positives	39 ±2.74	76.40±12.10	349.40±29.52
True Negatives	4555 ±2.74	4517.60±12.10	4244.60±29.52
False Negatives	791.20±2.49	711.40±4.62	549 ±3.32
Labeling Error (%)	12.58±0.05	11.94±0.24	13.61±0.49
Density MAPE (%)	3.66±0.10	3.71±0.18	4.22±0.21

"Positives" refer to sensors for which we rejected H_0, i.e., sensors that our fault detection statistical test concluded were faulty. "True" and "False" refer to correct and incorrect decisions, respectively, of whether a sensor is faulty. In a simulation with no faulty velocity measurements, a lower-bound density MAPE of **3.43%** was achieved.

All values reported are the mean and standard deviation of five identical simulations with different random seeds.

MAPE = mean absolute percentage error.

7.6 Conclusion

This article presented a principled fault-detecting particle filter for real-time rejection of potentially faulty measurements. Our methods are based on the classical Fisherian and Neyman-Pearsonian statistical testing theories, and follows these theories to arrive at different testing methods for when the engineer has a reliable model of faults, and when she does not.

One item of interest is the subtle inversion of what is considered the "data" in our hypothesis tests. Notice that in this work, the "data" that we use to estimate our test statistic are actually the simulated particles, which come from the system model, and we perform our hypothesis test to accept or reject the observations. An interesting implication of this nuance is how this hypothesis-testing particle filter allows a highly-trusted model to overpower the data, whereas a standard particle filter will always accept every datapoint, even if it is a clear outlier. We believe that the proposed techniques are closely aligned with the contemporary effort to fuse model-based and data-driven estimation and control techniques in many information sciences: to bring priors obtained from the engineering discipline to the surge of "big data." These lessons are likely to be useful in many areas to bring GNSS and other "big data" to ITS and other built-environment applications.

Part III

Attentional Deep Neural Networks for Transportation Systems: Two Applications

Deep Learning at both the Macro and Micro Scales

Chapter 8

Neural-Attention-Based Deep Learning Architectures for Modeling Traffic Dynamics on Lane Graphs

8.1 Introduction

Well-designed deep neural networks (deep NNs) have exhibited a powerful ability to model complex nonlinear phenomena. While deep NNs may be most well known for their uses in image processing and statistical language modeling applications (LeCun et al., 2015), this modeling ability also makes them well-suited to predicting nonlinear physical phenomena like the dynamic behavior of road traffic (Lv et al., 2015).

In all deep NN applications, the particular neural architecture (i.e., the type of "layers" used) is of particular importance: it is desired to select an NN with a so-called "inductive bias" that admits learning the necessary element-to-element relationships in a generalizable form (Battaglia et al., 2018).

In this work, we present results on using NNs to model traffic dynamics at a lane level. We consider the road network as a directed graph, with the nodes being road lanes and various types of edges being different kinds of lane-to-lane relationships. We use an NN model to estimate macroscopic traffic characteristics for these lanes (e.g., the number of vehicles occupying it at a given time).

The motivation to consider macroscopic dynamics for lane graphs (rather than the traditional road dynamics graph of "links," with lanes lumped together (Treiber and Kesting, 2013)) is threefold. First, lanes are more of an "atomic unit" of the road network: while statistical models for individual lanes may freely be composed into one for the whole link, the reverse is not true.

Second, resolution of road features at a lane level is necessary for vehicle control and route planning applications. An NN architecture that can take in per-lane data and output per-lane network predictions would be useful in these settings.

Third, a lane-level NN model can address a shortcoming of traditional macroscopic traffic dynamics modeling. Macroscopic traffic models can describe the aggregate behavior of all vehicles on a road via a continuous partial differential equation (PDE) approximation (Treiber and Kesting, 2013; Ferrara et al., 2018). Zooming in to the lane level, however, requires modeling of more complex human behaviors (lane changes, for example, have proven difficult to model, both in predicting when drivers will make a maneuver and how it will affect the other vehicles on the road (Zheng, 2014)). This gap in our first-principles understanding is a natural place to attempt to statistically learn a model rather than derive one.

In this chapter, using a microscopic traffic simulator (Krajzewicz et al., 2012) as a testbed, we find that a lane-graph-based NN can achieve these goals. We present results showing that a deep NN can outperform a PDE-model-based method on an intersection queue estimation problem. The NN architecture leverages the lane graph, basing per-lane queue and occupancy estimates on upstream, downstream, and neighboring lanes' observations. We show that encoding these different lane-to-lane relationships in the lane graph differently (which some earlier NN layers for graph-structured data do not do) is critical for this problem. We also show that our NN architecture can permit the transfer of NN models between road networks, opening the door to cross-region data-driven intelligent transportation systems. All code used in development of these results is available online at `github.com/mawright/trafficgraphnn`.

The remainder of this chapter is organized as follows. Section 8.2 briefly reviews the idea of aggregate traffic models, NN operations for graph-structured data, and the specific application of deep NNs to road networks. All three problems have a large body of work; we review only a few closely-related items here. Section 8.3 presents the new NN layer we use in our lane graph models. Section 8.4 presents our developed software framework and the estimation problem we consider as a test case. Section 8.5 provides details for our simulation setup and NN methods. Section 8.6 presents in detail the results we outlined in the previous paragraph, and section 8.7 outlines next steps.

8.2 Background

8.2.1 Partial Differential Equation models of traffic: relevant points

As discussed in chapter 2, traffic flows at an aggregate level are often described by analogy to compressible fluid flows. Mathematically, this means a fluid-like partial differential equation (PDE). In this chapter, we make use of a particular method based on the PDE model of traffic due to Liu et al. (2009). The method of Liu et al. (2009) is a technique for estimating the number of vehicles queued at a red light, in situations where this value is not directly observable. More specifically, it is often the case that an intersection is instrumented only with inductive loop detectors that can measure when a vehicle is occupying a specific location, and if the queue at an intersection extends past the most-upstream detector, it is unobservable.

In Liu et al. (2009), it was proposed to apply the PDE approximation of traffic to estimate this unobserved queue. After the light turns green and traffic begins to move, Liu et al. (2009) propose to identify when the shockwaves predicted by the PDE model of traffic passes by these fixed detectors, then to apply a method-of-characteristics analysis to track the shock trajectories in space and time and finally back out the upstream PDE boundary location (i.e., the initial length of the queue). This calculation is performed on each lane individually to obtain a *per-lane, per-cycle* maximum queue estimate. There are obvious inaccuracies to applying a continuous PDE analysis at a scale where the continuum approximation is not valid, but nevertheless, the method represents an integration of domain knowledge to obtain an estimate where there was none before.

In this chapter, we make use of Liu et al. (2009)'s method both as a model-based baseline, and as an input to our NN models. For more details on our implementation, see Ehlers (2019).

8.2.2 Neural network operations for graph data

Deep NNs have had particular success in domains with regularly-structured data, such as images and text. In those domains, prior knowledge that the data structure encodes important information (e.g., that a pixel's meaning can be better understood by examining it in combination with nearby pixels than with far-away pixels) enable better statistical learning. The term *geometric deep learning* (Bronstein et al., 2017) has been used to describe efforts to generalize these lessons to structured data on more general topological domains, such as graphs.

Many early works on deep NNs for graph-structured data consider learning features in the graph's spectral domain (i.e., in the eigenbases of matrices associated with the graph) (Kipf and Welling, 2017; Defferrard et al., 2016). Another line of work ((Hamilton et al., 2017; Veličković et al., 2018), etc.) attempts to make inferences in the "graph domain," i.e., by relating nodes to their neighbors (c.f. the difference between how modern convolutional NNs' operations are carried out in the pixel/time domain, whereas in classical signal processing, convolutions are often performed in the frequency domain).

One item to note is that, for features learned in the spectral domain of a graph, those features are tied to the particular graph spectrum. The transferability of spectral graph NNs is unclear (Bronstein et al., 2017). The neighborhood-based approaches, on the other hand, do not have this constraint and can be freely transferred (but, of course, have the drawback that a node's "receptive field" is limited). In this chapter, we are explicitly interested in transferring between lane graphs, and so our proposed models fit in the neighborhood-based approach (we also present comparisons against a spectral-type approach (Kipf and Welling, 2017) in our detailed results).

More broadly, authors have recently argued that AI systems need to be allowed to learn to reason about related but distinct entities to truly mimic humans' learning capability (Battaglia et al., 2018). The existence of distinct entities with information tied to their relationships to one another is often rendered mathematically as graphs, and inferring meaning

about their relationships as classic message passing; designing NN architectures that can approximate this tractably is a topic of much discussion (Battaglia et al., 2018).

Our particular NN is based largely on a neural architecture termed *attention* (Bahdanau et al. (2015); Vaswani et al. (2017), etc.). This technique and our application of it are described in section 8.3.

8.2.3 Neural networks for traffic dynamics

We are of course not the first to use deep NNs for traffic flows. The difficulty of first-principles modeling of traffic flow, along with the availability of massive amounts of observational data, make it a model problem for machine learning methods (Lv et al., 2015). Here, we briefly discuss two recent works that make use of graph-type NN operations.

Li et al. (2018) and Cui et al. (2018) both propose graph NN layers that learn spatiotemporal relationships in traffic networks. Li et al. (2018) propose a "diffusion convolutional" operation with learned coefficients, used to model traffic movements as a random walk along a graph. Cui et al. (2018) propose a "traffic graph convolutional" operation that incorporates traffic-theoretic information by clipping adjacency matrices based on whether traffic can feasibly move from one location to another in a time interval, based on the road distance and a prescribed freeflow speed.

In both papers, this graph operation forms the internals of a recurrent neural network (RNN) layer that is iteratively applied to a traffic timeseries to learn an autoregressive model. Their methods are applied to freeway-network-scale traffic measurements in two major U.S. metro areas: Los Angeles in Li et al. (2018) and Seattle in Cui et al. (2018).

In comparison, while in the present chapter we do not attempt a joint learning of spatial and temporal relationships via a spatial RNN layer (see section 8.5.2 for our NN architecture), we operate at a finer scale (lane level) and explicitly consider the graph-to-graph transfer problem. We also consider the impact of having multiple edge types. This last point is of importance at a lane graph level, since the macroscopic meaning of lane-to-lane relationships are distinctly different for adjacent lanes (i.e, lane changes) and upstream/downstream lanes.

8.3 Multi-edge-type Attentional Neural Network Layers

Neural attention (Bahdanau et al., 2015) is a powerful and widely-applicable neural architecture. Neural attention is often described as letting the NN "attend" or "pay attention to" certain elements of the input data (Vaswani et al., 2017). This means that, in a multilayer NN, an early layer is able to send to later layers the most relevant information for downstream tasks.

Technically, we define an NN layer for graph data as accepting two items, 1) a graph (V, E) with node set V and edge set E and 2) $\{x_i \in \mathbb{R}^N : i \in V\}$, a data vector for every node. In this work, we restrict our focus to layers where the domain of the output data is

the same graph (V, E). That is, the output of the NN layer is $\{h_i \in \mathbb{R}^H : i \in V\}$, a per-node featurization of the input data, for some output feature dimension H.

In this chapter, we assume that our graph is a directed graph, and also allow the graph to be multidimensional, with each edge also having one of a finite number of edge dimensions $d \in \{1, \ldots, D\}$. We denote the collection of d-dimension edges as E^d. In our application, the edge dimensions correspond to different lane-to-lane topological relationships, e.g., upstream, downstream, adjacent, self, etc.

For computational purposes, define a fixed ordering for the nodes $\{1, \ldots, n\}, n = |V|$ and the corresponding per-edge-type adjacency matrices A_d, where the i, jth entry of A_d is 1 iff $(i, j) \in E^d$, and 0 otherwise. In this application, this means that one A_d will have the i, j entry equal to 1 iff lane i is downstream of lane j, another A_d will have its i, j entry to 1 iff lane i is adjacent to lane j, etc.

The basic function of a standard attentional layer is to learn two tasks in parallel: 1) a learned featurization of each node's data $f_i = f(x_i)$ and 2) an attention scoring $a : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ that quantifies the relative importance of the edge (i, j). Then, the attention scores are used to compute, for each node, a weighted average of the featurized data of all nodes (or some subset of them).

In particular, we adapt the "Graph Attention Layer" of Veličković et al. (2018) with a new generalization to multidimensional graphs. We perform our per-node embedding via a learned weight matrix $W \in \mathbb{R}^{F \times N}$, $f_i = W x_i$. That is, this matrix W transforms node i's raw data x_i into a featurized vector f_i. Then, we compute the attention scoring with a single (scalar output) fully-connected neural network layer, one for each edge type:

$$\begin{aligned} a_{i,j}^d &= \sigma\left(\left[\left(W_i^d\right)^T \ \left(W_j^d\right)^T\right]\left[(f_i)^T \ (f_j)^T\right]^T\right) \\ &= \sigma\left(\left(W_i^d\right)^T f_i + \left(W_j^d\right)^T f_j\right) \end{aligned} \tag{8.1}$$

where $W_i^d, W_j^d \in \mathbb{R}^F$ are learned weights and $\sigma(\cdot)$ denotes some elementwise NN nonlinearity (in this chapter, following Veličković et al. (2018), we use for $\sigma(\cdot)$ the LeakyReLU function with slope parameter 0.2).

The attention scores $a_{i,j}^d$ are then normalized via a softmax function,

$$\alpha_{i,j}^d = \frac{\exp(a_{i,j}^d)}{\sum_{\{k:(i,k)\in E^d\}} \exp(a_{i,k}^d)}. \tag{8.2}$$

The per-node, per-edge-type layer outputs are computed as a weighted average of f_j for nodes j to which i is d-related:

$$h_i^d = \left(\sum_{\{k:(i,k)\in E^d\}} \alpha_{i,k}^d f_k\right) + b^d \tag{8.3}$$

where b^d is a learned bias vector that is shared among all nodes i.

Finally, the per-edge-type weighted-average feature vectors are concatenated together,

$$h_i = \left[\left(h_i^1\right)^T \ \cdots \ \left(h_i^D\right)^T \right]^T \tag{8.4}$$

and this vector h_i becomes the layer output for node i.

Note that all learned parameters (W, W_i^d, W_j^d, b^d) are applied to all nodes' data in parallel.

We make use of *multi-head attention* (Vaswani et al., 2017), where the above calculations are all performed several times, in parallel. The idea is that different attention "heads" are able to learn different concepts. In a multi-head layer, every head's h_i (8.4) is concatenated together to become the final layer output. Subsequent layers then combine information across edge types and attention heads. Like Vaswani et al. (2017), we use fully-connected layers for this purpose.

8.4 Methods

8.4.1 Software framework

We wrote Python code to set up traffic simulations, extract and preprocess data, and train the NNs. We used the microscopic traffic simulator SUMO (Simulation of Urban MObility) (Krajzewicz et al., 2012). Our NN layers use the Keras (Chollet et al., 2015) "Layer" standard in the interest of making them generally applicable.

8.4.2 A model problem for spatiotemporal deep learning on lane graphs: traffic queue prediction

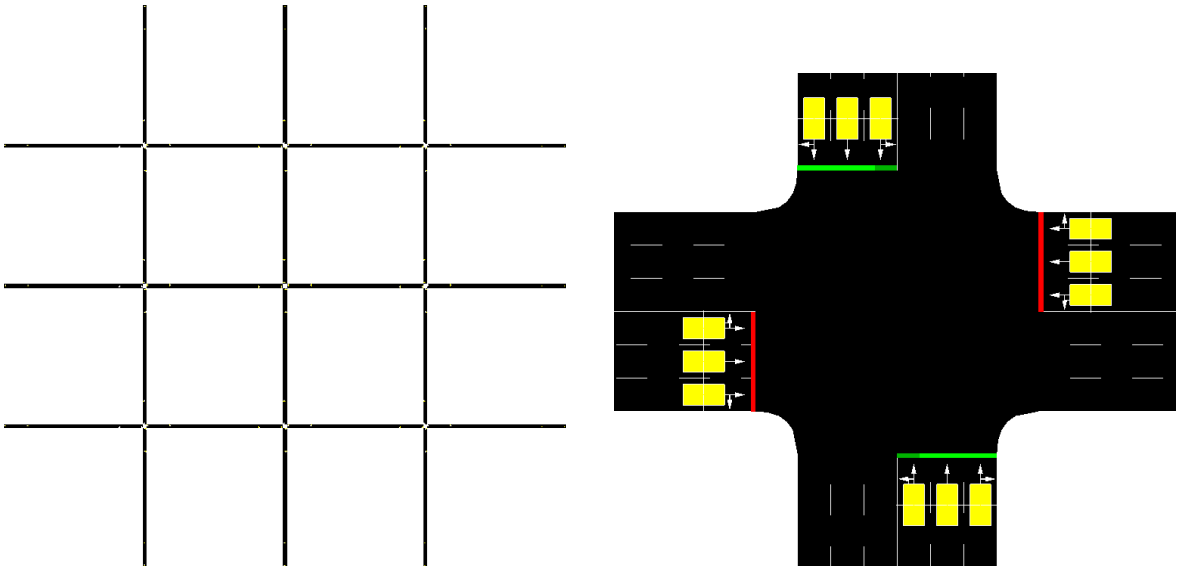
Our test case problem in this chapter is the estimation of the number of vehicles on a lane preceding a traffic signal, using readings from traditional fixed loop detectors. Accurate real-time vehicle counts at a traffic signal are necessary for effective signal control (Liu et al., 2009; Kurzhanskiy and Varaiya, 2015), but, as discussed in section 8.2.1 where we describe a model-based estimation method, these counts are in general unobservable from raw data alone.

We define two output quantities for our NN to predict: the maximum queue length at any point in a lane's red-green cycle, and the total count of vehicles on the lane at every timestep (the second quantity will be always no smaller than the first since it includes vehicles in transit who have not reached the signal queue yet).

8.5 Implementation Details

8.5.1 Simulation Details

We generated 100 SUMO simulations on the three-by-three grid network shown in Figure 8.1. For each simulation, we generated a sequence of random vehicle trips through the



(a) (b)

Figure 8.1: Training road network. (a): A 3x3 grid of three-lane roads, with attaching perimeter roads. Each road is 750m long. (b) Zoomed view of one intersection. The yellow boxes represent stopbar vehicle detectors.

road network using SUMO's `randomTrips` tool. Each vehicle entered the network on a randomly-selected entrance road (one of the roads whose upstream boundary is the edge of the road network) and was given a goal of a randomly-selected exit road (one of the roads whose downstream boundary is the edge of the network). A new vehicle entered the road in this manner every 0.4 s, for 3600 s. Each vehicle had a maximum speed given by multiplying SUMO's default maximum speed by a randomly-drawn multiplicative factor drawn from a Gaussian distribution with mean 1 and standard deviation 0.1, clipped at 0.2 and 2. We enabled SUMO's "rerouting" option so vehicles could reroute themselves around congestion during the simulation, and disabled its default "teleport" option. All other simulation parameters were left at their defaults.

On each 750-m lane, we placed two simulated loop detectors: one at the stopbar (visible in Fig 8.1(b)) and one 125 m upstream. Each detector recorded data every 1 s. These data are occupancy (a value in $[0, 1]$ reporting the portion of the 1-s period that a vehicle occupied the loop detector) and the speed of those vehicles (if any).

The input data for lane i at timestep t, $x_i^t \in \mathbb{R}^6$ was a vector with elements 1) the stopbar detector occupancy, 2) the stopbar detector speed, 3) the upstream detector occupancy, 4) the upstream detector speed, 5) a 0-1 indicator value that is 1 if the lane's traffic light is green and 0 if it is red, and 6) the PDE-model-based Liu et al. (2009) estimate of the queue.

We note that, when the traffic queue is not detected to have extended past the upstream detector, Liu et al. (2009) notes that the per-lane queue can be estimated via an input-output accounting. In practice, we found (Ehlers, 2019) that this was especially sensitive to lane

changes. Therefore, when a per-lane input-output difference resulted in a negative number of cars, we use a heuristic where we split this deficit evenly among the lanes that accounted for a surplus of cars (in reality, they had lane-changed out of the lane with a surplus and counted against the lane who saw it exit without entering).

The output vector $y_i^t \in \mathbb{R}^2$ has elements 1) the maximum queue length, in vehicles, of the lane during a red-green cycle, and 2) the count of the total number of vehicles on the lane during that timestep. These values were recorded via SUMO's "lane area" detectors and are given by the record fields `maxJamLengthInVehicles` and `nVehSeen`.

The queue length and the PDE-based queue length estimate are defined *per cycle* rather than *per timestep*. The PDE-based estimate (Liu et al., 2009) gives a per-lane estimate of the time of the maximum queue (based on the PDE characteristic curves), while the input-output relationship (used when the queue length does not extend past the upstream detector) can give an instantaneous per-lane estimate when the light changes from green to red. All other timesteps had this feature filled with the dummy value -1.

8.5.2 Our neural network architecture

In the NN literature, the problem setting where both the input and output domains are sequentially structured (i.e., timeseries data) is called the "sequence-to-sequence" problem (Sutskever et al., 2014; Neubig, 2017). The canonical NN architecture for this problem is a so-called "encoder-decoder" (Sutskever et al., 2014) approach.

One approach to modeling spatiotemporal data such as videos (Venugopalan et al., 2015) is to break the encoder and/or decoder into two parts. First, an encoder meant to learn spatial information (i.e., a convolutional NN stack or graph-type layers) operates on each timestep independently. Then, these per-timestep encodings are passed into an NN architecture meant to learn temporal correlations, such as an RNN layer stack. The NN model used in this chapter follows this paradigm.

We take in a set of per-lane input timeseries $\{x_i^t \in \mathbb{R}^N\}_{t=1}^T$ for lanes $i \in \{1, \ldots, n\}$. We pack each timestep's data into a data matrix for the entire graph, $X^t \in \mathbb{R}^{n \times N}$ We pass each data matrix into a graph encoder stack made up of several layers, each layer consisting of two sublayers: first, one of our multi-edge-type attention layers; second, a fully-connected layer. After each sublayer, we apply a layer normalization (Ba et al., 2016) operation with learned scaling and shift parameters. The idea of a layer with an attentional sublayer followed by a fully-connected sublayer is inspired by the Transformer architecture of Vaswani et al. (2017), though our fully-connected sublayer is simpler and we omit the residual connections. In this chapter, our graph encoder stack has two of these attention-fully connected layers, with ReLUs in between each sublayer.

The output of this graph encoding layer is a matrix of size $n \times F$, where F is the feature dimension of the last fully-connected layer. We have `batch_size` $\times T$ of these matrices, one for each timestep and element of our sample batch. The objective is that each node's F-long feature vector has aggregated information from other nodes' data.

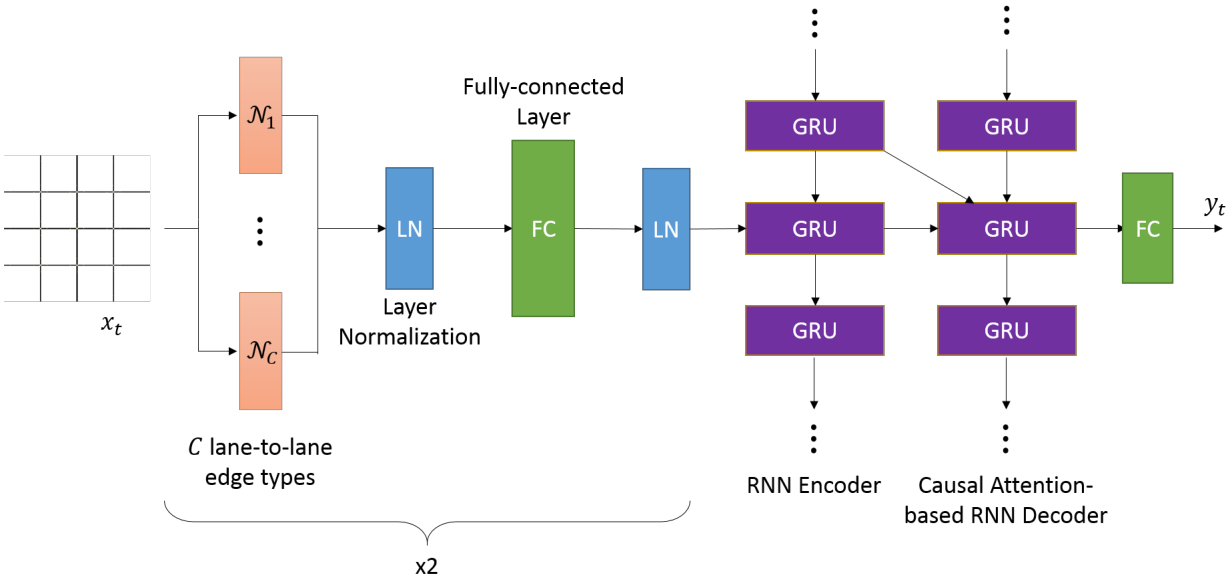


Figure 8.2: Schematic illustration of our NN architecture.

Our next step is to learn the temporal relationships. We pass each of the `batch_size` $\times n$ timeseries into an RNN independently. In this work, we use a two-layer RNN, using the gated recurrent unit (GRU) architecture. The first RNN layer is meant to encode the sequence forward in time. The second RNN layer uses an attentional decoding mechanism based on Bahdanau et al. (2015), with a modification that we do not let the RNN attend to timesteps in the future, using a masking technique similar to Vaswani et al. (2017); Veličković et al. (2018) and our graph encoder. We use ReLU nonlinearities at the output of each RNN layer as well.

The final layer in our NN is a fully-connected layer that maps the output of the second RNN layer to our output domain $\{y_i^t \in \mathbb{R}^M\}$. Since, in this case, our outputs (the queue length and the lane occupancy) are strictly nonnegative, we use a ReLU nonlinearity at the NN output as well, to restrict our NN's image to nonnegative numbers.

The entire NN architecture is depicted in Figure 8.2.

8.5.3 Training details

The 100 simulations were split into an 80-simulation training set, a 10-simulation validation set, and a 10-simulation test set. All learned parameters in the NNs were trained via minibatch stochastic gradient descent against the Huber loss, with gradients computed via Keras's automatic differentiation backend and gradient descent step-sizes computed with the Adam optimization algorithm (Kingma and Ba, 2015), with its parameters left at their Keras defaults. We annealed the Adam learning rate by multiplying it by a factor of 0.1 when the

Huber loss on the validation set did not decrease over 10 epochs. We used early stopping, ending training when the validation Huber loss did not decrease for 20 epochs.

The data for each simulation were chopped into periods of 30-second averages. For occupancy, speed, and the green indicator in x_i and vehicles on the lane in y_i, we took the mean over the period, and for the defined-per-cycle values of maximum queue length in y_i and PDE-based max queue estimate in x_i, we took the max. This 30-second averaging of detector data is prevalent in traffic engineering (Jia et al., 2001), despite recent awareness that it averages out important information (Coifman, 2015). This choice then mimics a lossy downsampling that is endemic to commonly-available data for practitioners. Another reason for doing this averaging is that we found that our NNs performed poorly on 1-second data. This is not too surprising; the 1-second data is very sparse, and it is known that RNNs have difficulty learning long-term dependencies (Trinh et al., 2018) despite many developments (including gated RNN architectures like the GRU) to tackle this problem.

For timesteps where the maximum queue length in y_i is undefined (see the last paragraph of section 8.5.1), we did not use the NN's output for that output feature and no gradient was taken.

8.6 Results and Discussions

We trained a variety of NNs with different combinations of encoded edge types in our layer from section 8.3. Table 8.1 gives quantitative results in terms of estimation mean absolute error (MAE) (i.e., the error values are in units of vehicles) on the 10-simulation held out test set.

In the table, the leftmost column describes the adjacency matrix structure included in our graph encoding layers described in section 8.3. The row for "$A = I$" indicates that no lane-to-lane adjacency information was encoded; instead, an identity matrix was passed in. This effectively means that the lane was only able to "attend" to itself, i.e., the graph encoding layer acted functionally identically to a fully-connected layer. Further entries in the first block of rows indicate that we add more adjacency matrices to encode more structure. "$+A_{\text{downstream}}$" means that we also include an adjacency matrix to denote downstream lane-to-lane connections, "$+A_{\text{upstream}}$" adds the reverse relation (i.e., the transpose of $A_{\text{downstream}}$), and $+A_{\text{neighbors}}$ means we added a matrix that indicates whether lanes were neighbors (i.e., they were on the same road). These additional adjacency matrices are cumulative: $+A_{\text{upstream}}$ also has the downstream adjacency dimension, and $+A_{\text{neighbors}}$ has all four.

The rows "$\{I, A_{\text{upstream}}\}$" and "$\{I, A_{\text{neighbors}}\}$" denote NNs with only the two noted adjacency matrices (as opposed to the cumulative adding of edge types in the previous rows).

All NNs described in Table 8.1 used four attention heads in the graph encoding sublayers. Information regarding parameter counts for each NN is given in the Appendix.

Next, we highlight several findings.

Table 8.1: PDE- and NN-based estimation results

Configuration	Mean Absolute Error (vehicles)	
	Cycle Queue Length	Lane Occupancy
$A = I$	1.04 ± 0.004	1.50 ± 0.003
$+A_{\text{downstream}}$	1.06 ± 0.01	1.47 ± 0.01
$+A_{\text{upstream}}$	1.06 ± 0.01	1.37 ± 0.01
$+A_{\text{neighbors}}$	0.96 ± 0.01	1.24 ± 0.01
$\{I, A_{\text{upstream}}\}$	1.07 ± 0.004	1.40 ± 0.01
$\{I, A_{\text{neighbors}}\}$	0.97 ± 0.01	1.40 ± 0.01
PDE-based estimate (Liu et al., 2009)	5.36	-

Values shown are mean abs. error ± one std. dev. over five random seeds.

8.6.1 Significantly Outperforming the PDE-Based Estimate

We found that our lane-graph NN models significantly outperform the method of Liu et al. (2009) in estimating the lane queues. The mean absolute error value of 5.36 vehicles is much higher than even our worst-performing NN. Figure 8.3 shows an example of the per-cycle lane queue estimates for one right-turn lane in the grid network.

Note that Liu et al. (2009)'s method was proposed to do lanewise queue estimation, but, as we mentioned earlier and discuss in more detail in Ehlers (2019), we found that it performs poorly in lanewise estimation in simulations with lane changing, like the ones analyzed in this work.

8.6.2 Benefits of Attending to Adjacent Lanes

We see in Table 8.1 that assimilating additional lanes' information via our multi-edge-type attentional encodings provides benefits to estimation. This is particularly notable in the occupancy estimation problem, where each additional edge type added gives a performance boost.

Interestingly, the cycle maximum queue length estimation performance seems to do worse when the downstream and upstream lanes' information are attended to. Under a two-sample t-test (unequal variances), the performance loss for both cases (adding the downstream lanes and adding both the downstream and upstream lanes) is statistically significant under most traditional levels ($p \approx 0.005$ and $p \approx 0.0005$, respectively) (the difference in estimation error between the attending downstream case and the attending downstream and upstream case is not statistically significant, $p \approx 0.41$). The fact that performance actually degrades instead of holding steady likely indicates overfitting.

Adding the neighboring lanes' information gave a clear performance boost in the max queue length problem, indicating that attending to those lanes is useful. Note also that while

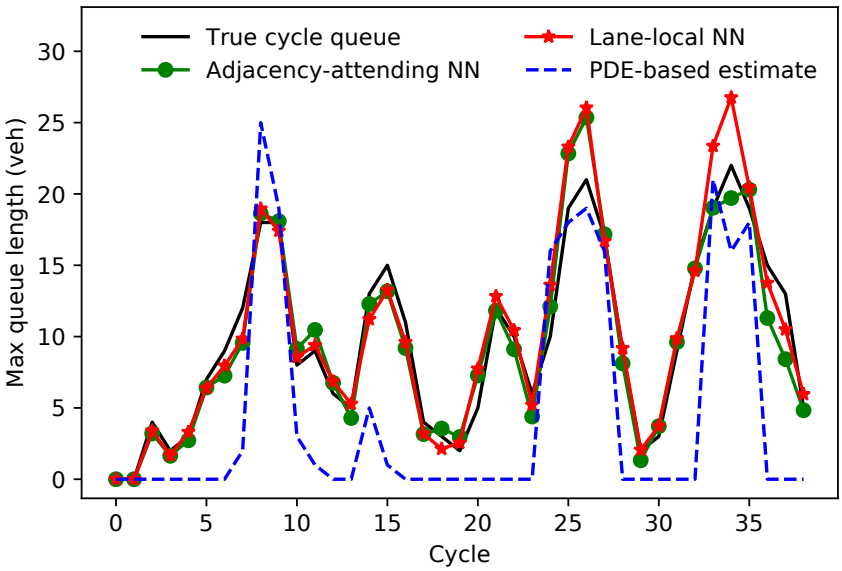


Figure 8.3: Example per-cycle queue estimates for a right-turn lane. The adjacency-attending NN is the one with all four edge-type encodings in Table 8.1 and the lane-local NN is the $A = I$ one. Note that the NN that can attend to adjacent lanes is able to predict the queue near cycle 35 more accurately than the lane-local one. When the queue does not back up enough to generate a shockwave to observe, the PDE-based method falls back to an input-output equation (Liu et al., 2009), which performs poorly in the presence of lane changing (Ehlers, 2019).

the configuration of only attending to neighboring lanes (the row $\{I, A_{\text{neighbors}}\}$) achieves about the same performance on queue estimation as the NN with all four adjacency matrices, it cannot perform as well on the lane occupancy problem. So, attending to *all* types of adjacent lanes is beneficial for this other, harder, problem.

8.6.3 Ablation Studies

We investigated the contribution of individual components of our NNs by selectively removing them. The results are shown in the first set of rows in Table 8.2 and discussed here.

8.6.3.1 The advantage of four-way edge type encoding is not simply due to an increased number of parameters

As discussed in section 8.3, in this work we augmented our graph encoding layers with extra adjacency matrices defining extra lane-to-lane relations by concatenating h_i^d's together (8.4). This means that the output dimension of the graph encoding layer increases linearly with D, and thus the fully-connected sublayer immediately downstream has a weight matrix whose

Table 8.2: Results for sections 8.6.3 and 8.6.4

Variation from Base Model	Mean Absolute Error (vehicles)	
	Cycle Queue Length	Lane Occupancy
(approx.) 1/4-size attn. dim.	0.98 ± 0.01	1.27 ± 0.01
Flattened adjacency matrix	1.22 ± 0.03	1.58 ± 0.03
One (4x-size) attention head	0.96 ± 0.01	1.24 ± 0.02
GCN, $\{I, A_{\text{up/downsteam}}\}$	1.81 ± 0.06	2.20 ± 0.05
GCN, $\{I, A_{\text{neighbors}}\}$	1.87 ± 0.04	2.54 ± 0.04
GCN, all adj. matrices A	1.49 ± 0.03	1.85 ± 0.03

"Base model" refers to the model with all four adjacency matrices ($+A_{\text{neighbors}}$ in Table 8.1). GCN = Graph Convolutional Network (Kipf and Welling, 2017). Values shown are mean abs. error ± one standard deviation over five random seeds.

number of rows also increases linearly with D. As can be seen in the Appendix, this can lead to a large increase in the total NN parameter count.

To verify that the increased performance of our extra edge-type encodings are not just due to increased parameter counts of downstream layers, we trained an NN with a (roughly) quarter-sized attention dimension F (it was not exactly 1/4 size, but was chosen to bring the parameter count close to the $A = I$ case, as shown in Table C.1). Its performance is shown in the first row of Table 8.2. We see there is a slight performance drop from the full-size base model, but we still perform substantially better than the other models with fewer adjacency dimensions. This suggests the high performance of the four-dimensional edge-type NN is indeed due to being able to attend to other lanes.

8.6.3.2 Flattening out edge type information is very costly

To study whether encoding different lane relationships in different ways (i.e., letting the model attend to upstream lanes in different ways than to neighboring lanes), we trained a model where we "flattened out" the adjacency dimensions. That is, here we used a single-dimension ($D = 1$) adjacency matrix where the i, j entry being 1 meant that lane i and j were related by any of the lane-to-lane relationships (self, downstream, upstream, neighboring).

The second row in Table 8.2 shows this model's performance. We see this model actually performed much worse, even worse in fact than the naive, non-adjacent-attending model ($A = I$ in Table 8.1). This suggests that attending to distinct relation types differently is critical, and attempting to treat them the same can lead to model confusion.

8.6.3.3 Is multi-head attention not needed when domain knowledge gives information of all edge-type relationships?

The third row in Table 8.2 shows the results when we use only one attention head. In this case, the attention head is 4x larger than the base model's four attention heads, meaning that the output dimension of this sublayer is the same. Quite interestingly, we saw no significant performance drop in our application, whereas Vaswani et al. (2017), when applying attention to a language translation task with no multi-edge-type encodings, saw a clear performance degradation with a single-headed configuration. We hypothesize this may be because our multi-edge-type encoding captures the benefits of learning multiple types of relationships that multi-head attention captures.

8.6.4 Comparing multi-edge-type attention to the Graph Convolutional Network layer

The "Graph Convolutional Network" (GCN) layer of Kipf and Welling (2017) is a popular spectral-type graph NN layer. The second group of rows in Table 8.2 presents results where replace our multi-edge-type attentional sublayers in our NN architecute with 256-hidden-unit GCN layers (using the Keras GCN implementation published by the authors, and using Kipf and Welling (2017)'s adjacency matrix "renormalization trick"), with subsets of adjacency matrices (as in section 8.6.2).

The GCN layer considers only one adjacency matrix and only undirected graphs. This means that to apply the GCN layer to our setting, we need to both flatten the adjacency matrices into one (see section 8.6.3.2) and drop all directionality information. This means, for instance, that we cannot encode downstream and upstream relations separately for the GCN. These topological assumptions not being valid for the nonlinear dynamics encountered on the road network may explain the GCN layer's poorer performance here.

8.6.5 Graph-to-Graph Transfer Learning: First Results

An important problem in geometric deep learning is the graph-to-graph transfer problem (Bronstein et al., 2017). In the classic spectral-based graph NN layers, the layer features are computed in the graph's spectral domain. This means that the applicability of these layers across graph topologies is undefined.

The local graph layers, on the other hand, since they rely only on local information, do not have this problem. Further, the particular learned attention scoring construction of attentional graph layers (8.1)-(8.3) means that different-sized d-neighborhoods can be used with the same layer.

Figure 8.4 shows a randomly-structured grid-like network with 750-m-long roads (the same length as the grid network), generated with SUMO's `NETGENERATE` function. We evaluated our NNs on 10 one-hour-long simulations on it.

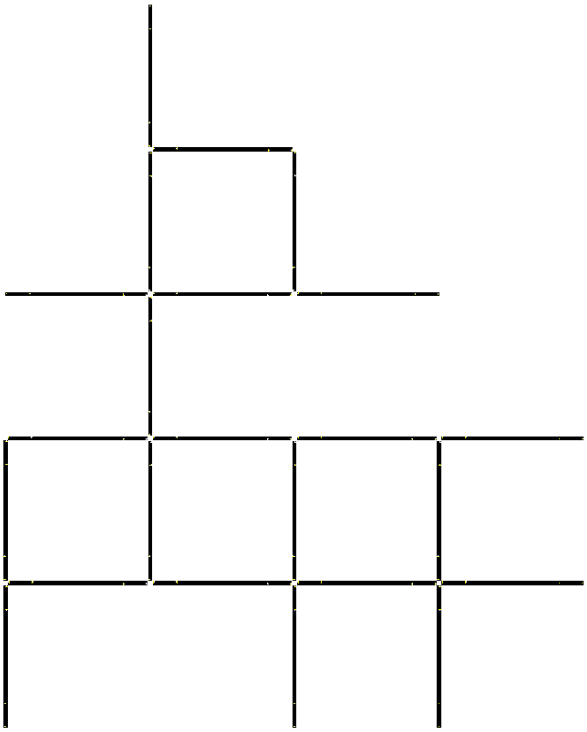


Figure 8.4: Random road network used in graph-to-graph transfer trial.

Table 8.3: Graph-to-graph transfer results for two exemplar NNs

Configuration	Mean Absolute Error (vehicles)	
	Cycle Queue Length	Lane Occupancy
$A = I$	5.77 ± 0.10	7.47 ± 0.14
All four adjacency matrices	6.32 ± 0.11	8.18 ± 0.14
PDE-based estimate (Liu et al., 2009)	15.04	-

Values shown are mean abs. error ± one std. dev. over five random seeds.

Figure 8.5 shows an example of how our four-edge-dimensional, four-headed NN predicts on the target network. Interestingly, the NN qualitatively follows the right general trend, but makes a few very large errors in understanding the queue near cycle 10 and near the end of the timeframe. Note the scale on the plot: a queue length of 60 is not typical on the grid network. Reexamining Figure 8.4, we see that the random network's irregularity leads to fewer alternate routes than the grid-based one, and therefore perhaps a greater likelihood of bottlenecking. This means that the queue length of nearly 60 vehicles is atypical for the grid-based one but may be not so out of the ordinary for the random one. Thus, the underestimation may be due to the fact that the NN had *never seen* that degree of bottlenecking in the grid network.

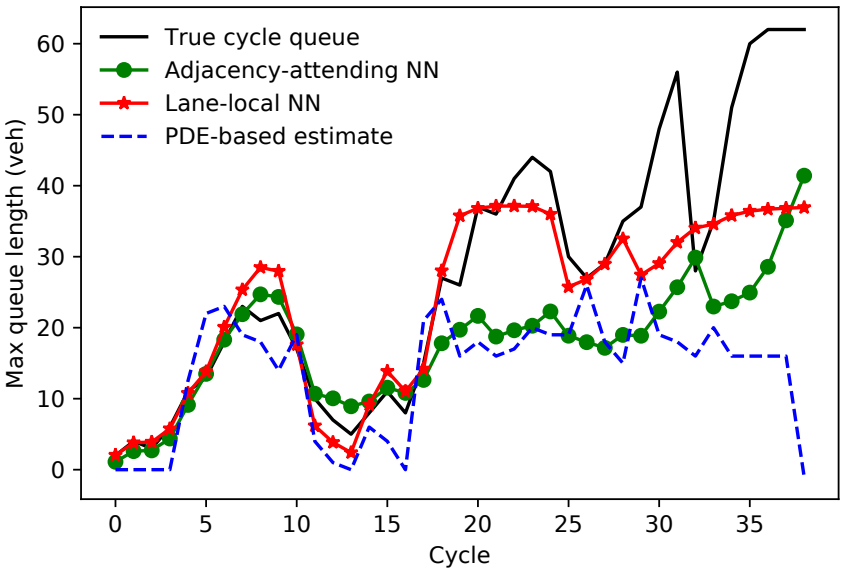


Figure 8.5: Example performance of the transferred NN from the grid network to the random network. The queue on a right-turn lane is shown.

Table 8.3 shows mean absolute errors for the four-dimensional NN, the NN with no adjoining lane information, as well as the PDE-based baseline. We see that both the NNs and the PDE-based estimate have much larger errors. The fact that the baseline error increases suggests that some of the lost accuracy for the NNs may be intrinsic to this network's dynamics presenting a more challenging problem in general. However, of interest is that the local-lane-information-only NN performs better on both prediction tasks. Since by construction the lane-local NN does not see what is happening on other lanes, we believe the fact that it now outperforms the adjacency-matrix-using network supports the hypothesis that the different road topology induced different inter-lane and inter-road dynamics as a source of increased error.

Put together, this interpretation justifies the idea of cross-network training to learn dynamics models that may only emerge on particular topologies. Extending our experiments to more networks, and more complex networks, in this manner is an item of ongoing work.

8.7 Conclusion and Future Work

We introduced the idea of applying attentional operations for deep learning models on lane graphs. Our results indicate that these particular architectures can be useful on the lane level, where first-principles models are difficult to derive.

We propose two immediate avenues for future work. First is the application of these NN architectures to more complex and varied traffic scenarios, such as more variable demands,

more variable vehicle behaviors, or further applications of the network transfer problem. Part of this work would be to ascertain whether the poor graph-to-graph transfer exhibited in section 8.6.5 can be improved by ensuring that the type of traffic dynamics encountered in the target road network were present in the training road network(s). The eventual goal of this avenue of work would be the transfer of these NN models from simulation to real-world lane graphs.

The other important avenue is further NN architecture engineering. We introduced a multi-edge-type attentional layer that can integrate information from different node-to-node relationships in different ways. Our studies on flattening out the edge types and the comparison with the GCN layer (sections 8.6.3.2 and 8.6.4, respectively) give evidence that this topological information is crucial to learn the complex dynamics on the lane graph. A shortcoming is that we do not (yet) include the time relationships in these edge types. Recent items in the NN literature like time convolutional networks (Bai et al., 2018), and the integration of those with attention (Yu et al., 2018) have challenged the assumption that RNNs are the optimal NN approach for timeseries data (learned downsampling convolutions in the time dimension would also be a more satisfying method of downsampling than our hard 30-second averaging). An open question is whether the time dimension can be treated as just another edge dimension, or whether it would be more effective to structure the architecture so that time means something distinct from spatial relations. A deep attention-based model for both space and time is particularly appealing for nonlinear dynamics problems, where the scalar attention scores (8.1) could be used to indicate the importance of different elements of the state space through entire dynamics trajectories according to the learned dynamics model.

Chapter 9

Attentional Policies for Cross-Context Multi-Agent Reinforcement Learning

9.1 Introduction

Deep reinforcement learning (RL) has been at the core of many breakthroughs in AI and controls domains in recent years. Examples include robotic locomotion (Lillicrap et al., 2016) and strategy games like Go at which computer programs had previously been noncompetitive (Silver et al., 2016, 2017).

The graduation of deep RL systems from closed environments to the real world will introduce new complexities. In the real world, a system will need to learn to interact not just with the environment, but with other (naturally or artificially) intelligent agents as well: they are *multi-agent systems* (Stone and Veloso, 1997; Hernandez-Leal et al., 2018).

Many new algorithms for deep RL have come from the statement of desired behaviors in policy learning, and then constructing new RL objective functions to encourage those desiridata. Such a pattern has been apparent in the multi-agent setting as well. Several recent papers on multi-agent-specific RL training algorithms propose value function estimators that can relate multiple agents' states and actions to observed scalar returns. For example, Sunehag et al. (2017) and Rashid et al. (2018) consider how to decompose a joint action-value function (Q function) into per-agent ones that, after learning, can guide each agent independently. Other works (Foerster et al. (2018); Lowe et al. (2017); Mao et al. (2019), etc.) consider a joint value function as playing the role of a critic in a multi-agent generalization of RL's actor-critic paradigm. The per-agent policy networks (i.e., the actors) are trained with *centralized* critic networks that aggregate information from all actors, then provide learning signals to move the actors to more cooperative behaviors.

In this chapter, we take a different approach. Taking a page from recent work in recognizing the utility of *relational inductive biases* (Battaglia et al., 2018) when crafting the interior architectures of a deep learning system, we redesign the policy networks such that agents can learn how to interact with other agents at the *policy* level. In particular, we apply neural

attention (Bahdanau et al., 2015; Vaswani et al., 2017) as a fundamental building block in our policy networks. We argue that this framework has appealing properties: among other benefits, it allows us to apply classic "single-agent" RL algorithms to the multi-agent problem with few modifications. The same architectures can be used for critic networks as well, which provides an appealing return to the classic, "single-agent" world, where actor and critic networks are often just architectural copies (c.f. the aforementioned methods where the critic networks are more complex than the policy networks).

Finally, the attentional construction lets us flexibly apply the same network to different contexts, both across agents and across different environmental situations from the perspective of each agent (i.e., we can both evaluate and train the same policy on multi-agent environments with different numbers of agents). We call the architecture "*cross-context*" to emphasize this flexibility.

The remainder of this chapter is organized as follows. In section 9.2, we briefly review the mathematical framework of RL in general and multi-agent RL. Section 9.3 reviews recent work in multi-agent RL and the use of "relational inductive biases" (of which attention is a particular instance) in learning. Section 9.4 reviews a benchmark multi-agent RL problem in coordinated autonomous vehicle control (Vinitsky et al., 2018) that we use as a framing problem. We then proceed to implementation: section 9.5 discusses our attentional policy networks for RL, section 9.6 discusses how these networks enable straightforward application of single-agent RL methods to multi-agent problems, and section 9.7 gives various implementation details. Section 9.8 presents our results, and finally, section 9.9 concludes with discussion on how the attentional policies enable flexible and decentralized multi-agent RL.

9.2 Definitions and Preliminaries

9.2.1 The general reinforcement learning setting

RL is typically presented in the mathematical framework of finite-horizon, discounted Markov decision processes (MDPs). These MDPs are defined by a tuple $(\mathcal{S}, \mathcal{A}, P, r, \rho_0, \gamma, T)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P is a stochastic transition function from $\mathcal{S} \times \mathcal{A}$ to $\mathcal{S}$ (i.e., it defines a conditional probability distribution $P(s_{t+1}|s_t, a_t)$ for $s_t, s_{t+1} \in \mathcal{S}, a_t \in \mathcal{A}$), $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, ρ_0 is a probability distribution on initial states s_0, $\gamma \in (0, 1]$ is a reward discounting factor, and T is the time horizon. The goal is to maximize the cumulative discounted reward $\sum_{t=0}^{T} \gamma^t r(s_t, a_t)$ where s_t and a_t are the state and action, respectively, at time t (Duan et al., 2016).

In the RL problem, the probability distributions and/or the reward function are unknown. The objective is to learn a *policy* π that defines a conditional probability distribution of actions given states, $a_t \sim \pi(a_t|s_t)$, such that the policy approximately maximizes the expectation of the discounted reward.

Typically we define our space of candidate policies as parameterized by some parameter vector θ (in modern deep RL, θ is the weights in a deep neural network). That is, the full

expression for the policy is $\pi(a_t|s_t;\theta)$ (also written with θ as a subscript, $\pi_\theta(a_t|s_t)$). The RL objective is then to find the optimal parameter vector θ^*, defined as

$$\theta^* = \arg\max_\theta E_\tau \left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right] \tag{9.1}$$

where $\tau = (s_0, a_0, s_1, a_1, \dots)$ is a shorthand for the entire trajectory. The policy $\pi_\theta(a_t|s_t)$ appears in (9.1) as part of the distribution over which the expectation is being taken.

Solving an RL problem requires devising a procedure for moving around in the space of θ to gather information about P and r (in the form of experienced samples) and using the information about the landscape of P and r to move towards θ^*. The abstract entity that draws actions a_t from $\pi_\theta(a_t|s_t)$, executes them, and observes the resulting next state $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$ and reward value $r(s_t, a_t)$ (for the particular s_t, a_t) is typically called the "agent."

9.2.2 Review of Proximal Policy Optimization (PPO)

9.2.2.1 The PPO Objective

Proximal Policy Optimization (PPO) algorithms (Schulman et al., 2017) are a class of popular and relatively simple algorithms for solving the RL problem (9.1). PPO algorithms are usually considered specific to the single-agent setting; later, we will outline how to apply them to our multi-agent setting.

A general PPO objective for a parameter vector θ at timestep t is to maximize a function of the form (Schulman et al., 2017)

$$L_t^{PPO}(\theta) = E_\pi \left[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S\left(\pi_\theta\right) - \beta D_{KL}(\pi_{\theta_{\text{old}}}||\pi_\theta)\right] \tag{9.2a}$$

with the "clipped" surrogate advantage objective function

$$L_t^{CLIP}(\theta) = \min\left(\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}} \cdot A(s_t, a_t),\ \text{clip}\left(\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}, 1-\epsilon, 1+\epsilon\right) \cdot A(s_t, a_t)\right) \tag{9.2b}$$

where $\pi_{\theta_{\text{old}}}$ is the policy distribution under some fixed parameter vector θ_{old} (taken to be the distribution at the beginning of a PPO iteration). The policy π_θ is still a conditional probability distribution for the random variable $a_t|s_t$, but we omit this argument in (9.2) to reduce notational clutter. Also, $\text{clip}(\cdot, 1-\epsilon, 1+\epsilon)$ is a function that projects its first argument to the interval $[1-\epsilon, 1+\epsilon]$, $L_t^{VF}(\theta)$ is the squared error of an estimate of the value function, i.e.,

$$L_t^{VF}(\theta) = \left(V(s_t) - \hat{V}_\theta(s_t)\right)^2 \tag{9.2c}$$

where $V(s_t)$, the value of state s_t, is the expected discounted reward of the trajectory beginning at state s_t,

$$V(s_t) = E_\tau \left[\sum_{t'=0}^{T-t} \gamma^{t'} r(s_{t'}, a_{t'})\right] \tag{9.2d}$$

and where $V_\theta(s_t)$ is a parametric estimate of this value as a function of s_t (with the parameters being, e.g., the parameters of a neural network that maps from $\mathcal{S}$ to $\mathbb{R}$). Later, in section 9.6.2, we will discuss a particular neural network architecture for this value function estimator.

In addition, $S(\pi_\theta)$ is the entropy of the distribution π_θ, $D_{KL}(\cdot||\cdot)$ the KL divergence, c_1 and c_2 are scaling constants, and β is a scalar whose value is updated adaptively during the training process. - Finally, $A(s_t, a_t)$ is the advantage of the state-action pair s_t, a_t, defined as the difference between the expected reward after taking action a_t, and $V(s_t)$,

$$A(s_t, a_t) = E_\tau \left[r(s_t, a_t) + \sum_{t'=1}^{T-t} \gamma^{t'} r(s_{t'}, a_{t'}) \right] - V(s_t). \tag{9.2e}$$

In other words, the advantage $A(s_t, a_t)$ quantifies how much better the action a_t is, in terms of total earned reward, than the average action at s_t.

9.2.2.2 PPO Implementation Details

The formulation of the PPO objective is to encourage the distribution $\pi_\theta(a_t|s_t)$ to move towards higher-advantage actions, but not move $\pi_\theta(a_t|s_t)$ *too far* from $\pi_{\theta_{\text{old}}}(a_t|s_t)$. Some implementations of PPO do not use the KL penalty ($\beta = 0$), some do not use the clipping of the objective ($\epsilon \to \infty$), and some use both to enforce this "distance" bound.

The execution of a PPO algorithm to train a policy π_θ iterates between an exploration step and an optimization step. First, the agent is allowed to interact with the environment for some pre-defined number of timesteps; this means that it is presented with a state s_t, selects an action a_t according to its policy π_θ, and observes the resulting new state s_{t+1} and reward r_t. Every tuple of information (s_t, a_t, s_{t+1}, r_t) is recorded in a buffer of samples $\mathcal{D}$.

After this interaction subroutine has been executed for the pre-defined number of timesteps, the execution step is complete and the optimization step begins. In the optimization step, a minibatch of samples $\{(s_t, a_t, s_{t+1}, r_t)\}, t = 1, \ldots, T_{\texttt{batch\_size}}$ are drawn randomly from $\mathcal{D}$. Then, an estimate of the cost function (9.2) is computed using those samples (in this discussion we use the subscript k rather than t to emphasize that the minibatch of samples need not be continuous in timestep t and can be drawn from several trajectories):

$$\hat{L}^{PPO}(\theta) = \frac{1}{K} \sum_{k=1}^{K} \left(\hat{L}_k^{CLIP}(\theta) - c_1 \hat{L}_k^{VF}(\theta) + c_2 S\left(\pi_\theta\right) - \beta D_{KL}(\pi_{\theta_{\text{old}}}||\pi_\theta) \right) \tag{9.3a}$$

where

$$\hat{L}_k^{CLIP}(\theta) = \min \left(\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}} \cdot \hat{A}(s_k, a_k),\ \text{clip} \left(\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}, 1-\epsilon, 1+\epsilon \right) \cdot \hat{A}(s_k, a_k) \right). \tag{9.3b}$$

The loss term $\hat{L}_k^{VF}(\theta)$ in (9.3a) is an empirical approximation to the value-function estimation loss (9.2c),

$$\hat{L}_k^{VF}(\theta) = \left(V_\theta\left(s_k\right) - R_k\right)^2 \tag{9.3c}$$

where

$$R_k = \sum_{t=k}^{T} \gamma^t r_t \tag{9.3d}$$

is the actually-observed discounted reward, beginning at timestep k, from a particular trajectory.

Also in (9.3a), $\hat{A}(s_k, a_k)$ is an estimate of the advantage of action a_k at state s_k, usually computed via the Generalized Advantage Estimator (GAE) (Schulman et al., 2016, 2017), which has the form

$$\begin{aligned} \hat{A}_t^{GAE} &= \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1} \\ \text{where } \delta_t &= r_t + \gamma V_\theta(s_{t+1}) - V_\theta(s_t) \end{aligned} \tag{9.3e}$$

where λ is a tuning parameter for the GAE, and γ is the discount rate for the Markov decision process. Note in (9.3e) that the formulation of the GAE calls for use of the *parametric estimator* of the value function, $V_\theta(s_t)$, rather than the empirical value R_t (Schulman et al., 2016). This estimator's parameters are adjusted via backpropagation from the $\hat{L}_k^{VF}(\theta)$ loss term (9.3c).

In (9.3d) and (9.3e), we note that the indices $t, t+1$, etc., mean that these correspond to the actual sequence of timesteps during exploration. Also, T means the time horizon of a trajectory.

The entire PPO optimization algorithm is summarized in Algorithm 9.1.

9.2.3 Multi-agent Reinforcement Learning

So far, we have just described the background to traditional, non-multi-agent RL. Multi-agent RL, as its name suggests, adds complications by having multiple agents. Let $\mathcal{I}_t$ denote the set of agents that are present in the environment at time t. We will say that each agent has its own observation space $\mathcal{O}^i$, which reflects a (partial) observation of the environment from that agent's perspective, and its own action space $\mathcal{A}^i$ and policy π^i which defines a per-agent conditional probability distribution on actions given the state, $a_t^i \sim \pi^i(a_t^i|s_t)$, with $a_t^i \in \mathcal{A}^i$.

In general, agent i need not only have information (sensor readings, etc.) *physically* local to itself. In a multi-agent setting where agents are meant to interact, it reasonable to assume that each agent will have information about the others. This information may be obtained by agent i from, for example, inter-agent communication or visual observation of other agents.

In the particular implementation presented in this work, the information that is transmitted by j, if any, is o_t^j, i.e., j's observation (we do not have a superscript i for the receiving agent here because we assume in our current formulation that all agents can potentiall receive the same observation vector o_t^j from sender j). A more general case where each agent i could potentially observe different information about a sending agent j, and the receiving agents' information are potentially of different dimensionalities, is possible, but in this chapter we will say that the local and received observations live in the same space to simplify later notation.

Algorithm 9.1: Proximal Policy Optimization (PPO) (Schulman et al., 2017)

Input: Initial policy parameters θ_0, environment $P(s_{t+1}|s_t, a_t)$, reward function $r(s_t, a_t)$, stopping criterion/criteria
Output: Final policy parameters θ
Initialize policy: $\theta \leftarrow \theta_0$;
while *Stopping criterion/critera not met* **do**
  Initialize buffer: $\mathcal{D} \leftarrow \emptyset$;
  for $t = 1, 2, \ldots, T_{\textit{execution\_steps}}$ **do**
    Observe state s_t from environment;
    Select action $a_t \sim \pi_\theta(a_t|s_t)$;
    Observe next state $s_{t+1} \sim P(s_t|s_t, a_t)$ and reward $r_t = r(s_t, a_t)$;
    Store data in buffer: $\mathcal{D} = \mathcal{D} \cup \{(s_t, a_t, s_{t+1}, r_t)\}$;
  end
  Take a snapshot of the policy parameters, $\theta_{\text{old}} \leftarrow \theta$;
  for $k = 1, 2, \ldots, K_{\textit{optimization\_steps}}$ **do**
    Sample a minibatch of samples $\{(s_t, a_t, s_{t+1}, r_t)\}_{t=1}^{T_{\text{batch\_size}}} \subseteq \mathcal{D}$;
    Compute $\hat{L}^{\text{PPO}}$ for the minibatch using (9.3);
    Compute gradients of θ with respect to $\hat{L}^{PPO}$, $\nabla \hat{L}^{PPO}(\theta)$;
    Take a gradient step: $\theta \leftarrow \theta + \eta \nabla \hat{L}^{PPO}(\theta)$ in the case of a scalar-valued step size η, or $\theta \leftarrow \theta + \eta \circ \nabla \hat{L}^{PPO}(\theta)$ for vector-valued step sizes such as the those from algorithms such as Adam (Kingma and Ba, 2015).;
  end
end

We also define a directed graph with edge set $\mathcal{E}_t$ that encodes the inter-agent relationships among the agents in $\mathcal{I}_t$. The edge $(i, j) \in \mathcal{E}_t$ if i gets information from j (i.e., it gets o_t^j). That is, if $(i, j) \notin \mathcal{E}_t$, then i does not receive o_t^j from j. In this directed graph, the presence of an edge $(i, j) \in \mathcal{E}_t$ need not necessarily imply the presence of the reversed edge $(j, i) \in \mathcal{E}_t$. Each edge also has a particular class c, $c \in \{1, \ldots, C\}$, to encode that agents relate to each other in meaningfully different ways.

9.3 Related Work

Multi-agent RL is said to be much more difficult than conventional, single-agent, RL. In addition to the typical obstacles in single-agent RL (like temporal credit assignment due to sparse rewards and navigating the exploration-exploitation tradeoff), multi-agent RL adds complications such as an intrinsically higher dimensionality, per-agent credit assignment, and (from the perspective of each individual agent) environmental nonstationarity during the learning process (i.e., if multiple interacting agents are all learning at the same time, then

one agent's knowledge about how others react to their actions quickly becomes outdated) (Hernandez-Leal et al., 2018).

9.3.1 Learning Multi-agent Cooperation

Several authors have proposed new RL training regimes to encourage the learning of cooperative behavior. In general, these approaches retain the idea of training individual policies per agent, but adjust the training goal to include context-specific multi-agent information.

Sunehag et al. (2017) and Rashid et al. (2018) consider the multi-agent Q-learning problem, where the joint (global) action-value function $Q(s_t, a_t)$ is well-defined, and study structural decompositions of it into per-agent action-value functions $Q^i(s_t^i, a_t^i)$ such that each agent's optimum a_t^i should be the same as the action it would have taken if the joint Q-function was maximized.

Foerster et al. (2018); Lowe et al. (2017); Iqbal and Sha (2019); Mao et al. (2019), among others, consider multi-agent RL where individual agents operate independently, but during actor-critic-style training, a centralized critic takes in information from all other agents. During policy execution (where the critic is not present), each agent operates independently, but has learned the behaviors favored by the centralized critic. Iqbal and Sha (2019) and Mao et al. (2019), in particular, propose centralized critics whose value function estimates make use of an attention module. Our proposed method is somewhat similar to the centralized critic methods, in that we also make use of a value function baseline that aggregates information (although our implementation is different from the aforementioned references), but we also permit the agents to explicitly attend to each other during execution. Of particular importance is that this means that the policies can adapt to situations of *varying numbers* of other agents.

Such a distinction is similar to the distinction between "self-attention," popularized by Vaswani et al. (2017) and the traditional attention of Bahdanau et al. (2015) (sometimes called "encoder-decoder" or just "decoder" attention). Our policy network structures can be thought of implementing self-attention,[1] whereas a centralized critic uses the classic encoder-decoder attention. To the best of our knowledge, this work represents the first proposal of attention mechanisms in the policy network itself.

9.3.2 Multi-agent Communication

In this work, we take the information that agent i has about agent j as given. The receiving/observing agent i then learns to process this information when generating its policy distribution. This is in contrast to a body of work (Foerster et al. (2016); Sukhbaatar et al. (2016); Lazaridou et al. (2017); Mordatch and Abbeel (2018), etc.) in which a communication

[1]The name is somewhat confusing since in our "self"-attention, agents are in fact attending to other agents. "Self-attention" is better thought of as being distinct from encoder-decoder attention in that in self-attention, all entities attend on each other, whereas in encoder-decoder attention, a fixed decoder attends on multiple encoded attendees, with the attentional decoding only happening in one direction.

policy is explicitly learned. In an application where i's information about j is *communicated* by j rather than *observed* by i, and communication has some associated bandwidth constraint or cost, it makes sense to add a communication policy in the learning objective.

As an example, Jiang and Lu (2018) proposed an attentional module to enable agents to decide *when* to communicate, and following Sukhbaatar et al. (2016) and Peng et al. (2017) a global LSTM coordinator is used to aggregate and disseminate this information back to the agents. They argue that enabling the learning of *selective* communication improves performance, by removing the need for receivers to filter out less-useful information. An extension of our framework where the attended-on information is emitted by a learned communication module as in Jiang and Lu (2018), but the processing is done in a decentralized agent-wise manner like in this work, is an interesting avenue for future work.

9.3.3 Relational Inductive Biases in Learning

Recent work in machine learning theory (Battaglia et al. (2018) present a broad review) has argued for the importance of so-called *Relational Inductive Biases* in effective learning. A relational inductive bias is an inductive bias (defined by Battaglia et al. (2018), citing Mitchell (1980), as a bias or prior that "allows a learning algorithm to prioritize one solution (or interpretation) over another, independent of the observed data") that encodes prior knowledge about the existence of discrete entities, and the *relations* among those entities. A convolutional neural network layer, for example, encodes a relational inductive bias that pixels are the entities of interest, and that *locality* of statistically correlated pixels is of prime importance (Battaglia et al., 2018).

Neural attention (Bahdanau et al., 2015; Luong et al., 2015; Vaswani et al., 2017) has been characterized as a form of relational inductive bias, where the attended-on elements (e.g., different words or time-slices) are the entities of interest, and the relations are quantified via the "attention weights," the output of a learned attention module (Battaglia et al., 2018).[1] Giving each agent a relational inductive prior to the other agents it needs to coordinate with, at the policy level, motivates our application.

9.4 A Cooperative Adaptive Cruise Control Problem for Multi-Agent Reinforcement Learning

9.4.1 Problem specification

In this work, we consider a benchmark multi-agent RL problem introduced by Vinitsky et al. (2018). This reference proposed several multi-agent reinforcement learning problems based on mixed-autonomy traffic (road traffic with mixtures of autonomous and human-driven vehicles). We will consider the "Merge" problem, shown in Figure 9.1.

[1]Below, in section 9.5, we will use an index i to index different attended-on elements, and α_{ij} in (9.5b) to

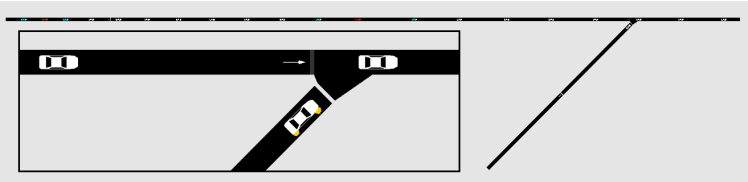


Figure 9.1: "Merge" benchmark road network with zoom-in showing simulated merging vehicles. Vehicles move to right.

In the problem, vehicles enter the freeway section in a Poisson process (and are removed when they reach the end). Most of the cars are considered "human-driven," with accelerations/decelerations governed by a behavioral model called the *Intelligent Driver Model* (IDM) (Treiber et al., 2000). The IDM has been widely used to simulate human drivers (Orosz et al., 2010). Under the IDM, drivers accelerate to reach a desired cruising speed and only slow down when they would otherwise collide with the vehicle in front of them. The freeway has an onramp, and vehicles on the freeway do not slow down preemptively to let the merging vehicles in, so when a merging vehicle enters, the vehicle(s) on the mainline behind them may need to rapidly brake. There are considerations for reaction time and kinematic constraints on the max deceleration. This means that sometimes the braking vehicle is not able to decelerate enough, causing a collision. If it does manage to avoid crashing, the braking cascades upstream (i.e., stop-and-go traffic), which hurts everyone, by reducing the road throughput and creating the danger of other collisions. The control problem is to take control of some subset of the vehicles (vehicles are randomly designated controllable or not as they enter) and prevent these dangerous and inefficient emergent dynamics.[2]

For each controlled vehicle $i \in \mathcal{I}_t$, its observation o_t^i is a five-dimensional vector of 1) its own speed, 2) the speed of the vehicle immediately in front of it, 3) the bumper-to-bumper distance of the immediately in front of it, 4) the speed of the vehicle immediately behind it, and 5) the bumper-to-bumper distance of the vehicle immediately behind it. The action for vehicle i, a_t^i, is the controlled vehicle's longitudinal acceleration (a scalar value). The problem's reward function encourages all vehicles to move quickly, while having the controlled vehicles maintain not-too-small inter-car distances. The reward is global (one reward for all agents), and takes the form (Vinitsky et al., 2018)

$$r_t = \frac{\max\left(\|v_{des}\|_2 - \|v_{des} - v_t\|_2, 0\right)}{\|v_{des}\|_2} - c_h \sum_{i \in \mathcal{I}_t} \max\left(\text{headway}_{max} - \text{headway}_{i,t}, 0\right) \tag{9.4}$$

where v is a vector with elements consisting of the current velocity of every vehicle currently in the network (both controlled and "human-driven"), v_{des} is a vector of the same size with

mean the attention weight of i attending on j.

[2]Recent work has shown how even one computer-controlled car in a population of human-driven ones can induce more efficient and safer driving patterns in the human drivers with whom they share the road (Stern et al. (2018); Sadigh et al. (2016), etc.).

every element equal to a pre-specified positive desired velocity (in Vinitsky et al. (2018)'s problem specification, v_{des} is the same for every vehicle i and every timestep), $\text{headway}_{i,t}$ is the headway of controlled vehicle i (i.e., the distance between i and the vehicle in front of it) at time t, headway_max is a desired maximum headway, and c_h is a scaling parameter. Examining the reward function (9.4), we can see that the first term leads to a higher reward when every vehicle has a velocity near the target velocity, and the second term leads to a lower reward when the controlled vehicles do not keep too small a headway between them and the vehicle they are following. The $\max(\cdot)$ in the first term means that no penalty is assigned for negative velocities (i.e., vehicles moving backward are assigned the same reward as zero-velocity vehicles), and the $\max(\cdot)$ in the second term means that there is no penalty for having $\text{headway}_{t,i} >> \text{headway}_{max}$.

In the instance that any two vehicles collide, the simulation immediately terminates. This means that in the event of a collision there is no reward from future timesteps. Since the objective of the RL problem is to maximize the discounted future reward (9.1), this means that collisions are heavily penalized. This termination occurs even if the colliding vehicles are both "human-driven," so an optimal policy will need to induce safe behavior in the non-controlled vehicles as well.

9.4.2 Problem background

This problem is a form of a Cooperative Adaptive Cruise Control (CACC) problem. The information each vehicle observes about its own speed, its preceding and following vehicles' speed, and its distance to those vehicles, is obtainable by onboard sensors (e.g., radar). This is enough information to perform adaptive cruise control (ACC), which is a term for computer-controlled cruise control with feedback based on onboard vehicle measurements. The technology of adding inter-vehicle communication so an ACC system can also use other cars' measurements, which is what our scenario captures, is called cooperative adaptive cruise control (CACC) and is an area of active research (Dey et al., 2016; Wang et al., 2018) (note also that Wang et al. (2018) explicitly mention freeway onramps, which our problem is based on, as a particularly important but difficult potential application for CACC). The literature of RL controllers for CACC is sparse compared to those that use traditional control methods. Finally, note also that most CACC methods in the literature or in industry focus on uninterrupted platoons of controlled and communicating vehicles, whereas the control problem considered here includes non-controlled vehicles between controlled ones (in addition, there is no explicit encoding in o_t^i as to whether the upstream/downstream vehicles are controlled or "human-driven").

9.4.3 Prior work on reinforcement learning for merge cooperative adaptive cruise control

In Vinitsky et al. (2018), the canonical solution uses a centralized single-agent approach rather than a multi-agent approach. There, a central controller receives all observations, stacks them

into one vector, and computes all actions. However, the number of controlled vehicles on the network will change as they enter and exit, so to use a traditional single-agent multi-layer perceptron (MLP) (i.e., successive fully-connected neural network layers) architecture, a fixed maximum number N of vehicles to control must be defined, and the network-wide observation vector is either truncated or zero-padded as needed. On the action end, if $|\mathcal{I}_t| < N$ (where recall we defined $\mathcal{I}_t$ as the set of all agents present at time t), extra actions are discarded, and when $|\mathcal{I}_t| > N$, some are left uncontrolled.

This style of centralized controller has some drawbacks beyond its potential lack of realism. Controlling at most N vehicles effectively throws away extra information when more than N vehicles are present. The padding and truncation likely also makes the learning problem harder because the RL agent is expected to learn by itself to not assign credit to the ignored actions (without knowledge that they have been ignored), making the credit assignment problem even more difficult. One solution is to train different policies for different numbers of agents and select between them as the situation changes, but training many policies would, among other issues, vastly increase the RL sample requirements. In contrast, our proposed method seeks to be cross-trainable by allowing valid backpropagation for any number of agents.

9.5 Attentional Architectures for RL

9.5.1 Self-attentional layer

We use a (self-) attention layer in our policy and value networks. The layer takes in a set of input data vectors $\{h^i_{in} \in \mathbb{R}^{1\times n}, i \in \mathcal{I}_t\}$ and outputs a set of featurized vectors $\{h^i_{out} \in \mathbb{R}^{1\times m}, i \in \mathcal{I}_t\}$. Note that in this chapter's work, we use a different kind of self-attentional layer than was used in chapter 8. Specifically, the attention calculation used in chapter 8 is the "Graph Attention Layer" of Veličković et al. (2018), which uses the so-called "additive attention" calculation of Bahdanau et al. (2015). On the other hand, the layer calculation used here is the "scaled-dot-product attention" calculation of Vaswani et al. (2017)[3] with a modified version of Shaw et al. (2018)'s "relative positional embeddings" to differentiate different types of edges, which in our notation is

$$h^i_{out} = \sum_{j\in\mathcal{I}_t:\ (i,j)\in\mathcal{E}_t} \alpha^{ij} \left(h^j_{in} W_V + a_V^{c(i,j)} \right) \tag{9.5a}$$

[3]Comparing the "additive" and the "scaled-dot-product" (i.e., multiplicative) attention mechanisms, Britz et al. (2017) found the additive mechanism to have a higher peak performance on one language-translation problem, but Vaswani et al. (2017) note that the multiplicative attention is much more computationally efficient, due to modern floating-point matrix-multiplication subroutines being more highly tuned than floating-point addition operations.

where $c(i,j)$ means the class of the edge (i,j), α^{ij} is the attention weight of i attending on j, given by

$$\alpha^{ij} = \frac{\exp(e^{ij})}{\sum_{k \in \mathcal{I}_t:\ (i,k) \in \mathcal{E}_t} \exp(e^{ij})} \qquad e^{ij} = \frac{h_{in}^i W_Q (h_{in}^j W_K + a_K^{c(i,j)})^T}{\sqrt{m}} \tag{9.5b}$$

and where $W_Q, W_K, W_V \in \mathbb{R}^{n \times m}$ and $a_V^c, a_K^c \in \mathbb{R}^{1 \times m}$ for $c \in \{1, \ldots, C\}$ are learned parameters.[4] The calculations in (9.5) are shown for a single set of data vectors $\{h_{in}^i, i \in \mathcal{I}_t\}$. In practice, multiple timesteps t are batched together for computational purposes; this can be done by concatenating the rows for each h_{in}^i to form matrices $\{H_{in}^i \in \mathbb{R}^{\texttt{batch\_size} \times n}, i \in \mathcal{I}_t\}$, and similarly for $a_V^{c(i,j)}$ and $a_K^{c(i,j)}$.

Equation (9.5a) and (9.5b) form a straightforward self-attention calculation in the spirit of Vaswani et al. (2017), with the adoption of Shaw et al. (2018)'s proposal to have unique bias vectors a_V and a_K for each of the C different ways that agents can relate to each other. In this chapter, like in chapter 8, we also make use of *multi-head* attention (Vaswani et al., 2017), where the calculations in (9.5) are performed multiple times in parallel, with independent W's and a's, and then each head's output vectors are concatenated.

This bias-vector encoding of edge types is different from the edge-type encoding used in chapter 8, where we trained separate sets of weight matrices $W_{\cdot}$ for each edge type c and concatenated the output vectors together similar to the multi-head encoding. One way of thinking about the differences of edge-type encoding techniques is that the edge-type-dependent bias method in this chapter may allow multiple edge-type relations to be learned within one attention head.

9.5.2 Our neural network architecture

In this work, our policy network is as follows. For a batch of timesteps $t = 1, 2, \ldots,$ `batch_size`, we stack all agent observations $o_t^i \in \mathbb{R}^n$ into a 3-D tensor $\mathbf{O} \in \mathbb{R}^{\texttt{batch\_size} \times \overline{|\mathcal{I}_t|} \times n}$, where $\overline{|\mathcal{I}_t|}$ means the maximum size of $\mathcal{I}_t$ of all timesteps in the batch (i.e., the maximum number of agents of all timesteps in the batch). We also use the directed graphs $\mathcal{E}_t$ for each timestep t in the batch; for computational purposes, we represent them as adjacency matrices and stack them into a 4-D tensor $\mathbf{A} \in \mathbb{R}^{\texttt{batch\_size} \times C \times \overline{|\mathcal{I}_t|} \times \overline{|\mathcal{I}_t|}}$, where C is the maximum number of edge classes. For those timesteps in the batch that have less than $\max_t |\mathcal{I}_t|$ agents present, their slices in the tensors are padded with zeroes as dummy values, up to a dimension of $\overline{|\mathcal{I}_t|}$. It can be seen that these dummy values do not contribute to the output of the attention layer for the non-padding observations o_t^i by noting that, since the adjacency matrices are padded with zeroes, there is no edge (i,j) for padded observation $o_t^j = \mathbf{0}$, since it will not be counted in the sum in (9.5a).

These tensors go into the attention layer described in section 9.5.1. The 3-D tensor $\mathbf{O} \in \mathbb{R}^{\texttt{batch\_size} \times \overline{|\mathcal{I}_t|} \times n}$ is sliced along the second dimension to obtain $\overline{|\mathcal{I}_t|}$ matrices $O^i \in \mathbb{R}^{\texttt{batch\_size} \times n}$. These matrices play the role of the input matrices H_{in}^i mentioned just after

[4] That is, they are part of the θ that is learned via (e.g.) Algorithm 9.1.

the statement of (9.5). We use four attention heads (i.e, four independent parallel copies of the attention layer) of $m = 16$ units each. After stacking these attention heads, that means that the output of the attention layer is a tensor of dimension $\texttt{batch\_size} \times \overline{|\mathcal{I}_t|} \times 64$ (64 being the concatenation of 4 attention heads of dimension 16).

The output of the previous layer is then fed into a fully-connected hidden layer with 64 units. This fully-connected layer is applied to each of the $\texttt{batch\_size} \cdot \overline{|\mathcal{I}_t|}$ slices along the trailing dimension independently, meaning that each of these 64-dimensional vectors have the same affine transformation applied to them.

Both the attentional and fully-connected sublayers are followed by a ReLu nonlinearity and a layer-normalization operation (Ba et al., 2016) (with learned scale and location parameters), in that order. This structure of an attentional sublayer followed by a shared-over-agents fully-connected sublayer is inspired by the Transformer architecture of Vaswani et al. (2017), though we use only one such layer and omit residual connections.

The output of the above layers then goes into the output layer, whose output parameterizes the stochastic policy. In this work, our stochastic policy is a per-agent Gaussian distribution with mean and log-variance computed by the same fully-connected layer for each agent. To reiterate, the same layers are used for all agents $i \in \mathcal{I}_t$, letting each agent i attend on all other agents independently.

9.6 Attentional Multi-Agent Proximal Policy Optimization

One key gain of using self-attentional architectures in the policy network is that classic "single-agent" RL training algorithms can be used with little modification (this is in contrast to, e.g., Sunehag et al. (2017); Foerster et al. (2018); Lowe et al. (2017); Iqbal and Sha (2019); Rashid et al. (2018); Mao et al. (2019), where restricting learning about inter-agent relationships to *outside* of the policy network requires multi-agent-specific modifications to RL). We reviewed traditional PPO in section 9.2.2. In this section, we discuss how the use of an attention-based architecture permits its application to our multi-agent problem.

9.6.1 The distribution for the attentional Proximal Policy Optimization objective

As we noted above, the traditional PPO algorithm assumes only a single agent. How do we apply the objective (9.3) when the domains of s_t and a_t vary over timesteps with the number of agents?

The key insight that allows us to apply PPO to the attentional multi-agent architecture is that every term is a function of statistics of the overall *policy* π_θ rather than any particular *agent*. The form of the joint distribution π_θ across agents varies over timesteps, but we

reduce the information *across agents* into per-timestep scalar statistics of π_θ (entropy and KL divergence in (9.3a), likelihood ratios in (9.3b)) before actually averaging over timesteps.

We formalize this result in the following elementary lemma:

Lemma 9.1. *For the attentional policy network, the conditional distribution of the actions given the state for timestep t is given by*

$$\pi_\theta(a_t|s_t) = \prod_{i \in \mathcal{I}_t} \pi_\theta(a_t^i | \{o_t^j : (i,j) \in \mathcal{E}_t\}). \tag{9.6}$$

Proof. We have defined that $\mathcal{A}^i$ is the action space for agent i, and defined the joint action space as the product space $\mathcal{A} = \prod_i \mathcal{A}^i$. Then, we can define the product measure on $\mathcal{A}$ as simply the product of the measures on the component action spaces,

$$\pi_\theta(a_t|s_t) = \prod_{i \in \mathcal{I}_t} \pi_\theta(a_t^i|s_t). \tag{9.7}$$

Then, we note that by the construction of the attentional network, we have that a_t^i is conditionally independent of $\{o_t^j : (i,j) \notin \mathcal{E}_t\}$ given $\{o_t^j : (i,j) \in \mathcal{E}_t\}$,

$$\pi_\theta(a_t^i|s_t) = \pi_\theta(a_t^i | \{o_t^j : (i,j) \in \mathcal{E}_t\}). \tag{9.8}$$

Combining (9.7) and (9.8), we immediately have the lemma. □

Remark 9.1. Note that the above construction only makes sense because θ and s_t are held fixed for all i. This means that the a_t^i are exchangeable in the de Finetti sense. In a non-attentional network where the relational inductive bias does not encode a permutation invariance (e.g., if an LSTM is used to sequentially encode the observations i attends to), this may not hold.

9.6.2 Attentional value function baseline

In this work, we estimate the value function $V(s_t)$ using a neural network with identical architecture to the policy network described in section 9.5.2, producing a tensor of dimension $\texttt{batch\_size} \times \overline{|I_t|} \times 64$. We then take a maximum over the second dimension to obtain a matrix of dimension $\texttt{batch\_size} \times 64$. Then, a scalar-output fully-connected layer is used to produce an estimate of the scalar value $V(s_t)$ for each timestep in the batch. This can be seen as a fully self-attentional critic rather than the encoder-decoder attentional critics used in, e.g., Iqbal and Sha (2019); Mao et al. (2019). This value function neural network is used to compute the value $V_\theta(s_t)$ in (9.3c) and (9.3e). Its parameters are trained simultaneously with the policy network's parameters, via backpropagation from the scalar loss $\hat{L}^{PPO}(\theta)$ in (9.3a).

9.7 Implementation Details

The "merge" baseline described above is implemented in the framework *Flow* (Wu et al., 2017a), which is a Python codebase built on the widely-used microscopic vehicle traffic simulator SUMO (Krajzewicz et al., 2012) that adapts SUMO to the widely-used RL problem standard "`env`" developed in OpenAI *Gym* (Brockman et al., 2016). We implemented our neural network architecture in the distributed computing framework *Ray* (Moritz et al., 2018), specifically its RLlib component (Liang et al., 2018). With only the minor modifications discussed above – using an attentional architecture as described in section 9.5 and modifying the form of the policy distribution π_θ as described in section 9.6 – we are able to use PPO true to the description in Algorithm 9.1. All PPO hyperparameters were left as the same as in Vinitsky et al. (2018), with the exception that we update our policy every 20 rollouts instead of every 50.

We also used Ray to produce a baseline solution similar to Vinitsky et al. (2018)'s fully-centralized single-agent approach, using MLP policy and value networks with the padding and truncation discussed in section 9.4.3. Our neural networks differ from Vinitsky et al. (2018)'s in that we use a shallower MLP network with different hidden layer sizes. For our single-agent reference, we use a two-hidden-layer networks with 64 units in each fully-connected layer and a tanh nonlinearity in between. This 64x64 architecture serves as a comparison to the attentional architecture that has the same number of hidden units.

Our code is available online at `github.com/mawright/attn_rl`.

9.8 Experimental Results

Vinitsky et al. (2018) proposed several different configurations of the "Merge" problem, varying in the penetration rate of autonomous vehicles and the maximum number of vehicles that are allowed to be controlled. At the low end, "Merge 0" requires the control of at most 5 vehicles, and on the high end, "Merge 2" requires the control of up to 17 vehicles. We report the results of several experiments in Figure 9.2 and discuss them in detail below.

Figure 9.2(a) shows learning curves for PPO on the "Merge 0" and "Merge 2" benchmarks, for both our attentional architecture and the reference MLP architecture with padding and truncation described in section 9.4. On both problems, we obtain superior performance to the MLP architecture.

9.8.1 Importance of relational inductive biases

Some of the superior performance of the decentralized controller comes from the power of the relational inductive biases encoded in the attention module. To study this, in Figure 9.2(b) we experiment on Merge 2 with varying numbers of relative position encodings C. In the base case, we use $C = 7$ different edge classes, and thus $C = 7$ different bias vector pairs (a_V, a_K) from (9.5). The edge class for the edge (i, j) is determined by the relative position from i

to j. One edge class is for the case $i = j$ (i.e., vehicle is attending to its own observation), one each for j being the controlled vehicle immediately in front of and immediately behind i, one each for j being two controlled vehicles in front of and behind i, one for all j's three or more controlled vehicles in front of i, and a final bias vector for all j's three or more controlled vehicles behind i. The base case uses $C = 7$, giving unique $c(i, j)$'s for the self-case of $i = j$ and the two subsequent controlled vehicles upstream and downstream; all other further-upstream vehicles share the same $c(i, j)$ relation, and all other further-downstream another $c(i, j)$. This configuration is "Max Relative Position = 3" in figure 9.2(b).

The line labeled "Max Relative Position = 1" uses only $C = 3$, where all downstream and all downstream vehicles to the attending agent are considered equivalently.

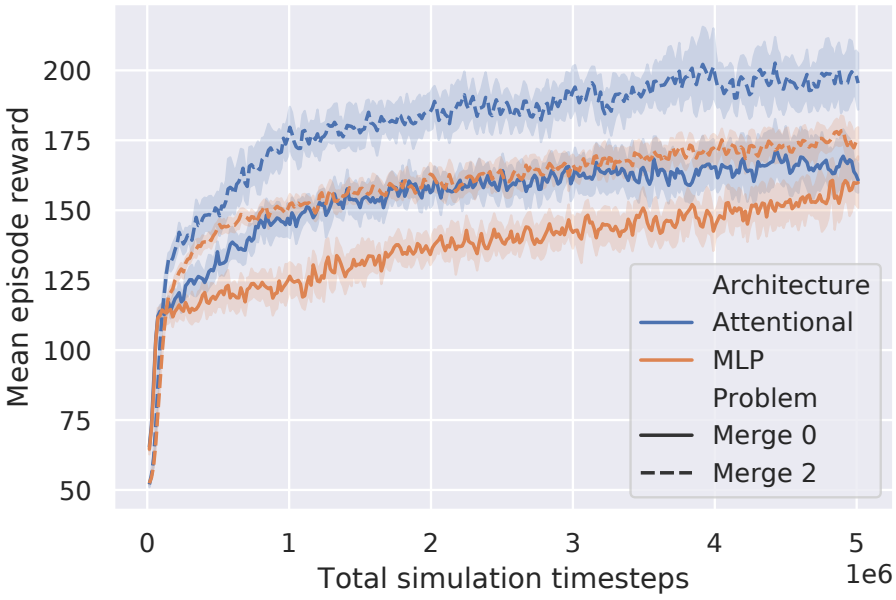
Finally, the line labeled "All (i,j) treated equivalently" means that $C = 1$, and each agent treats both itself and all other agents equivalently. We see that all configurations are able to eventually attain around the same maximum reward (the fact that we use multi-headed attention means that even for the all-agents-equivalent case, the policy can learn to use different heads to attend to different o_t^j's in different ways), but more informative relational inductive biases give increases in sample efficiency and less variance in learning.

9.8.2 Robustness to varying information availability

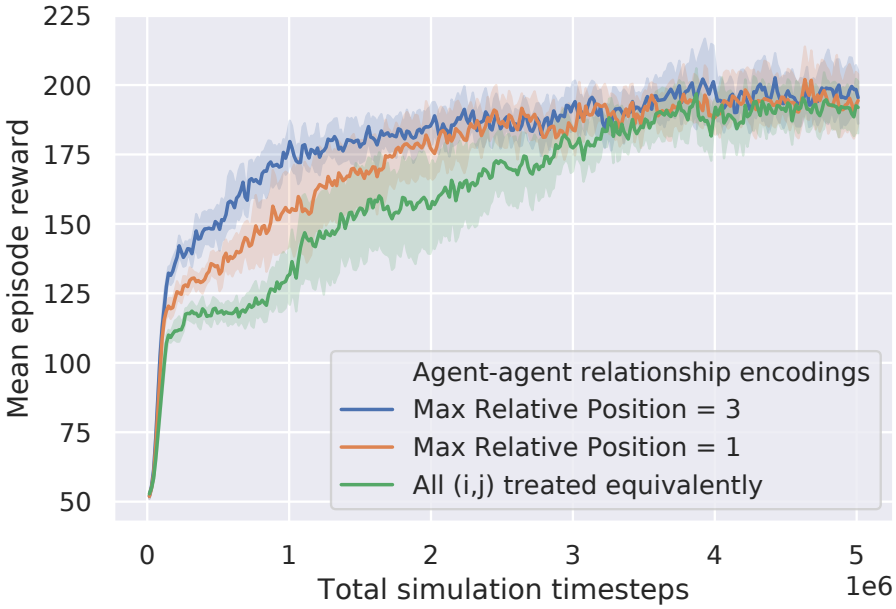
Finally, in figure 9.2(c) we test the attentional policies' robustness by introducing randomly lossy communication. To be more precise, we mean that on every timestep we randomly delete edges (i, j) for $i \neq j$ with varying probabilities. This means that each agent i will randomly not be able to attend to information from other agents. Mathematically, this means that in (9.5b), $e^{ij} \to -\infty$, which in turn means that $\alpha^{ij} = 0$ in (9.5a), and the output of the attentional layer for i will not have any contribution from o_t^j.

This test is done on Merge 2 with the "Max Relative Position = 1" configuration. Each line refers to trial with a different probability that each communication is dropped. The line denoted "$Rate = 0.2$," for example, means that every value α^{ij} (for $i \neq j$) in (9.5b) has a probability of 0.2 of being forced to zero through the mechanism just described.

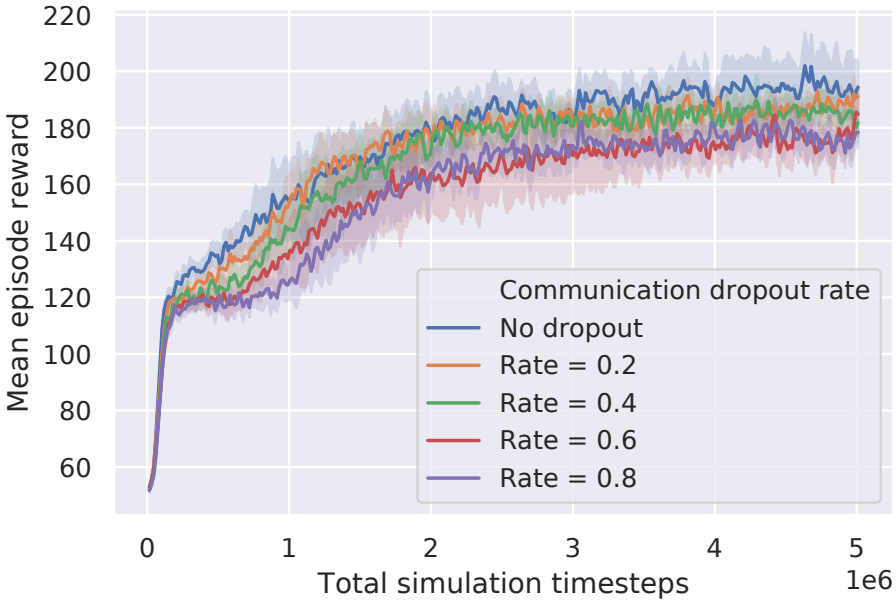
We find that while there may be somewhat of a performance decrease as the dropout rate increases, it is minor (the difference between the end-of-training mean reward for the 0.8-rate and both the 0.2-rate and no-dropout cases is statistically significant under a two-sample unequal variances t-test ($p \approx 0.01$ for both), but none of the other pairwise differences are). This suggests that even for highly-varying information environments, the architecture can generalize.


Mean episode reward
200
175
150
125
100
75
50
0
1
2
3
4
5
1e6
Total simulation timesteps
Architecture
Attentional
MLP
Problem
Merge 0
Merge 2

(a)


Mean episode reward
225
200
175
150
125
100
75
50
0
1
2
3
4
5
1e6
Total simulation timesteps
Agent-agent relationship encodings
Max Relative Position = 3
Max Relative Position = 1
All (i,j) treated equivalently

(b)



(c)

Figure 9.2: Figure continues from previous page. Learning curves for various experiments. All curves show the mean and 95% confidence interval of mean episode reward. Figure (a) compares the performance of the attentional policy to a standard MLP policy with dynamic padding and truncation to a fixed size ("Merge 0" is a version of the Merge problem that requires control of up to 5 vehicles, and "Merge 2" requires control of up to 17 vehicles). Figure (b) studies the performance for different levels of explicitly-encoded relational inductive bias. Figure (c) examines the robustness of the attentional policies to lossy communication by varying the degree to which communicated observations are randomly dropped. All curves are for sample sizes of 10 runs with different random seeds.

9.9 Conclusion: Attention's Real-World Applicability

It is worth noting a few details that make the attentional architecture appealing for multi-agent RL problems. Of key importance is that each agent's actions are computed fully in parallel. This means that each agent can actually compute its action locally, independently of the other agents, using only its knowledge of its and whatever other agents' states it has available. Computing all agents' actions in batch was only done for purposes of computational parallelism and ease of explanation.

Also of note is how using the attentional architecture allows for the straightforward application of a simple and relatively well-understood *single-agent* RL training algorithm (namely, PPO). The question of how each agent needs to reason about all other agents when determining its own action is explicitly moved to the policy network. The ability to deploy classic RL algorithms like PPO, as opposed to needing multi-agent-specific RL algorithms like QMIX (Rashid et al., 2018) is noteworthy.

Since all agents use the same policy, we may think about each agent's state and action, and its view of the states of the other agents, as an individual training example for the single policy. It seems that the only obstacle to a fully-decentralized training regime, where gradients can be computed locally, is the fact that to estimate the scalar reward, we need to centrally aggregate encoded information over agents in our value network by, e.g., our max-pooling. However, since all agents share the same policy, we should be able to assume that any agent with knowledge of the others' observations can make an estimate not only of its own action, but also the others'. This means that each agent can also produce a *local value function estimate*, as a function of the subset of the agents that it can observe. In other words, *every term* in the global (vanilla) policy gradient *and* a value function baseline can be locally estimated from each agent's perspective.

Future work should explore the extension of these points to move towards greater contextual transferability and decentralized learning in deep RL.

Chapter 10

Conclusions

10.1 Summary of contributions

The major contributions of the work presented in this dissertation are:

- In chapter 3, we presented the first (to the best of our knowledge) generally-applicable solution to the junction problem for the two-PDE class of macroscopic traffic models known as the "Generic Second Order Models" (Lebacque et al., 2007b). This tool and the analysis we present has promise in allowing transportation and system engineers to analyze and control the many new macro-scale emergent phenomena that will become apparent in any prospective (partial) deployment of autonomous vehicles as the characteristics of road traffic fundamentally change (Mehr and Horowitz, 2019; Mehr, 2019).

- Chapters 4 and 5 presented several new modeling constructions for a particular class of road networks. We discussed how equipping a freeway with a managed lane creates a situation of two parallel, interacting but qualitatively and quantitatively discrete traffic flows, and how this situation has been observed to lead to the emergence of previously-unseen phenomena (recall our discussion of emergence in complex systems in chapter 1, and discussions of these types of previously-unseen emergent phenomena in Daganzo and Cassidy (2008); Cassidy et al. (2010); Ponnu and Coifman (2017), as well as many others cited in the chapters). Our contribution was to propose models for several of these phenomena, then demonstrate how they can be incorporated into existing modeling paradigms.

- Chapters 6 and 7 present two contributions to the particle filter estimation paradigm suitable for complex systems where it is desired to use individual-agent-level measurements to estimate the state of a system defined on a higher level of the complex system hierarchy, with the motivating example of using vehicle-local speed measurements to estimate the macroscopic state of traffic on a road network. More specifically, chapter 6 presented an application of the Rao-Blackwellization technique for particle filters

(Doucet et al., 2000) that leveraged relationships in the macroscopic traffic flow theory and allowed us to assimilate these measurements in a principled manner. Then, chapter 7 presented our family of "hypothesis-testing particle filters," which we applied to reject outlier measurements in a principled manner, consistent with the classic theory of statistical testing. This method was also immediately applicable to assimilating local and small-scale measurements to estimate a higher-order system's state.

- Chapters 8 and 9 presented two new and exciting applications of neural attention towards designing deep neural networks that can apply structural domain knowledge of the transportation system, i.e. the domain knowledge of the kind of (self-) organization between system elements (in our cases, road lanes or vehicles) that lead to the complex system behaviors of interest. This idea is closely related to emerging trends in machine learning theory (Battaglia et al., 2018) on the need for proper inductive biases in learning.

10.2 Future work

We briefly discuss two avenues for immediate future work. First, there is a need to establish the hypothesis-testing particle filter of chapter 7 on a firmer (i.e., measure-theoretic level) theoretical foundation. This will include more formally defining the hypothesis tests proposed, as well as applying particle filter convergence results to prove estimator convergence.

Second, the attention-based neural networks have shown promise in confirming the usefulness of domain knowledge (e.g., that classifying different types of lane-to-lane relationships would improve statistical learning), but still in rather simple problem settings. There is great opportunity in applying these ideas to more complex systems or more realistic input data pipelines. For example, rather than chapter 9's approach that generated control inputs for vehicles, and assumed communication from the other vehicles, a more applicable situation is to attend to other vehicles (or other elements) that have been identified from some upstream computer vision system.

Bibliography

M. Abou-Zeid, J.-D. Schmöcker, P. F. Belgiawan, and S. Fujii. Mass effects and mobility decisions. *Transportation Letters*, 5(3):115–130, July 2013. ISSN 1942-7867. doi: 10.1179/1942786713Z.00000000011.

H. Adeli and X. Jiang. *Intelligent Infrastructure: Neural Networks, Wavelets, and Chaos Theory for Intelligent Transportation Systems and Smart Structures*. CRC Press, Oct. 2008. ISBN 978-0-429-18277-8. doi: 10.1201/9781482281767.

A. Allström, D. Gundlegård, and C. Rydergren. Evaluation of travel time estimation based on LWR-v and CTM-v: A case study in Stockholm. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pages 1644–1649, Sept. 2012. doi: 10.1109/ITSC.2012.6338823.

M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, Feb. 2002. ISSN 1053-587X. doi: 10.1109/78.978374.

A. Aw and M. Rascle. Resurrection of "Second Order" Models of Traffic Flow. *SIAM Journal on Applied Mathematics*, 60(3):916–938, 2000. ISSN 0036-1399. doi: 10.1137/S0036139997332099.

A. Aw, A. Klar, M. Rascle, and T. Materne. Derivation of Continuum Traffic Flow Models from Microscopic Follow-the-Leader Models. *SIAM Journal on Applied Mathematics*, 63 (1):259–278, Jan. 2002. ISSN 0036-1399. doi: 10.1137/S0036139900380955.

J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer Normalization. *arXiv:1607.06450 [cs, stat]*, July 2016. URL `http://arxiv.org/abs/1607.06450`.

D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *International Conference on Learning Representations*, 2015. URL `http://arxiv.org/abs/1409.0473`.

S. Bai, J. Z. Kolter, and V. Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv:1803.01271 [cs]*, Mar. 2018. URL `http://arxiv.org/abs/1803.01271`.

P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261 [cs, stat]*, June 2018. URL `http://arxiv.org/abs/1806.01261`.

J. Bhatti and T. E. Humphreys. Hostile Control of Ships via False GPS Signals: Demonstration and Detection. *Navigation*, 64(1):51–66, Mar. 2017. ISSN 00281522. doi: 10.1002/navi.183.

S. Blandin, A. Couque, A. Bayen, and D. Work. On sequential data assimilation for scalar macroscopic traffic flow models. *Physica D: Nonlinear Phenomena*, 241(17):1421–1440, Sept. 2012. ISSN 01672789. doi: 10.1016/j.physd.2012.05.005.

M. Bliemer. Dynamic Queuing and Spillback in Analytical Multiclass Dynamic Network Loading Model. *Transportation Research Record: Journal of the Transportation Research Board*, 2029:14–21, Dec. 2007. ISSN 0361-1981. doi: 10.3141/2029-02.

D. Britz, A. Goldie, M.-T. Luong, and Q. Le. Massive Exploration of Neural Machine Translation Architectures. *arXiv:1703.03906 [cs]*, Mar. 2017. URL `http://arxiv.org/abs/1703.03906`.

G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI Gym. *arXiv:1606.01540 [cs]*, June 2016. URL `http://arxiv.org/abs/1606.01540`.

M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, July 2017. ISSN 1053-5888. doi: 10.1109/MSP.2017.2693418.

C. Bucknell and J. C. Herrera. A trade-off analysis between penetration rate and sampling frequency of mobile sensors in traffic state estimation. *Transportation Research Part C: Emerging Technologies*, 46:132–150, Sept. 2014. ISSN 0968-090X. doi: 10.1016/j.trc.2014.05.007.

Caltrans Office of Project Development Procedures. How Caltrans Builds Projects, Aug. 2011. URL `http://www.dot.ca.gov/hq/tpp/index_files/How_Caltrans_Builds_Projects_HCBP_2011a-9-13-11.pdf`.

C. Carrion and D. Levinson. Value of travel time reliability: A review of current evidence. *Transportation Research Part A: Policy and Practice*, 46(4):720–741, May 2012. ISSN 0965-8564. doi: 10.1016/j.tra.2012.01.003.

G. Casella and C. P. Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996. doi: 10.1093/biomet/83.1.81.

C. G. Cassandras. Smart Cities as Cyber-Physical Social Systems. *Engineering*, 2(2):156–158, June 2016. ISSN 2095-8099. doi: 10.1016/J.ENG.2016.02.012.

M. J. Cassidy, K. Jang, and C. F. Daganzo. The smoothing effect of carpool lanes on freeway bottlenecks. *Transportation Research Part A: Policy and Practice*, 44(2):65–75, Feb. 2010. ISSN 09658564. doi: 10.1016/j.tra.2009.11.002.

M. J. Cassidy, K. Kim, W. Ni, and W. Gu. A problem of limited-access special lanes. Part I: Spatiotemporal studies of real freeway traffic. *Transportation Research Part A: Policy and Practice*, 80:307 – 319, 2015. ISSN 0965-8564. doi: http://dx.doi.org/10.1016/j.tra.2015.07.001.

M. Chang, J. Wiegmann, A. Smith, and C. Bilotto. A Review of HOV Lane Performance and Policy Options in the United States. Report FHWA-HOP-09-029, Federal Highway Administration, 2008.

C. Chen, J. Kwon, J. Rice, A. Skabardonis, and P. Varaiya. Detecting Errors and Imputing Missing Data for Single-Loop Surveillance Systems. *Transportation Research Record*, 1855 (1):160–167, Jan. 2003. ISSN 0361-1981. doi: 10.3141/1855-20.

F. Chollet et al. Keras, 2015. URL `https://keras.io`.

C. G. Claudel and A. M. Bayen. Convex Formulations of Data Assimilation Problems for a Class of Hamilton–Jacobi Equations. *SIAM Journal on Control and Optimization*, 49(2):383–402, Jan. 2011. ISSN 0363-0129, 1095-7138. doi: 10.1137/090778754.

B. Coifman. Empirical flow-density and speed-spacing relationships: Evidence of vehicle length dependency. *Transportation Research Part B: Methodological*, 78:54–65, Aug. 2015. ISSN 01912615. doi: 10.1016/j.trb.2015.04.006.

R. Corthout, G. Flötteröd, F. Viti, and C. M. Tampère. Non-unique flows in macroscopic first-order intersection models. *Transportation Research Part B: Methodological*, 46(3):343–359, Mar. 2012. ISSN 01912615. doi: 10.1016/j.trb.2011.10.011.

S. Cui, B. Seibold, R. Stern, and D. B. Work. Stabilizing traffic flow via a single autonomous vehicle: Possibilities and limitations. In *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pages 1336–1341. IEEE, 2017. doi: 10.1109/IVS.2017.7995897.

Z. Cui, K. Henrickson, R. Ke, and Y. Wang. Traffic Graph Convolutional Recurrent Neural Network: A Deep Learning Framework for Network-Scale Traffic Learning and Forecasting. *arXiv:1802.07007 [cs, stat]*, Feb. 2018. URL `http://arxiv.org/abs/1802.07007`.

C. F. Daganzo. The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269–287, Aug. 1994. doi: 10.1016/0191-2615(94)90002-7.

C. F. Daganzo. The cell transmission model, Part II: Network traffic. *Transportation Research Part B: Methodological*, 29(2):79–93, Apr. 1995. ISSN 01912615. doi: 10.1016/0191-2615(94)00022-R.

C. F. Daganzo. A behavioral theory of multi-lane traffic flow. Part I: Long homogeneous freeway sections. *Transportation Research Part B: Methodological*, 36(2):131–158, Feb. 2002. ISSN 01912615. doi: 10.1016/S0191-2615(00)00042-4.

C. F. Daganzo and M. J. Cassidy. Effects of high occupancy vehicle lanes on freeway congestion. *Transportation Research Part B: Methodological*, 42(10):861–872, Dec. 2008. ISSN 01912615. doi: 10.1016/j.trb.2008.03.002.

M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, volume 29, pages 3844–3852, 2016.

D. S. Dendrinos. Traffic-flow dynamics: A search for chaos. *Chaos, Solitons & Fractals*, 4(4):605–617, Apr. 1994. ISSN 0960-0779. doi: 10.1016/0960-0779(94)90069-8.

G. Dervisoglu, G. Gomes, J. Kwon, A. Muralidharan, P. Varaiya, and R. Horowitz. Automatic calibration of the fundamental diagram and empirical obsevations on capacity. *88th Annual Meeting of the Transportation Research Board, Washington, D.C., USA*, 2009.

K. C. Dey, L. Yan, X. Wang, Y. Wang, H. Shen, M. Chowdhury, L. Yu, C. Qiu, and V. Soundararaj. A Review of Communication, Driver Characteristics, and Controls Aspects of Cooperative Adaptive Cruise Control (CACC). *IEEE Transactions on Intelligent Transportation Systems*, 17(2):491–509, Feb. 2016. ISSN 1524-9050. doi: 10.1109/TITS.2015.2483063.

A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. In *The Oxford Handbook of Nonlinear Filtering*, pages 656–704. Oxford University Press, 2011. ISBN 978-0-19-953290-2.

A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-blackwellised Particle Filtering for Dynamic Bayesian Networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 176–183, Stanford, California, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-709-9.

Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking Deep Reinforcement Learning for Continuous Control. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 1329–1338, New York, NY, USA, 2016.

S. Ehlers. *Traffic Queue Length and Pressure Estimation for Road Networks with Geometric Deep Learning Algorithms.* Project Thesis, Leibniz Universität Hannover, Hannover, Germany, 2019. URL `https://arxiv.org/abs/1905.03889`.

D. J. Fagnant and K. Kockelman. Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice*, 77:167–181, July 2015. ISSN 0965-8564. doi: 10.1016/j.tra.2015.04.003.

S. Fan, Y. Sun, B. Piccoli, B. Seibold, and D. B. Work. A Collapsed Generalized Aw-Rascle-Zhang Model and Its Model Accuracy. *arXiv:1702.03624 [physics]*, Feb. 2017. URL `http://arxiv.org/abs/1702.03624`.

N. Farhi, H. Haj-Salem, M. Khoshyaran, J.-P. Lebacque, F. Salvarani, B. Schnetzler, and F. De Vuyst. The Logit lane assignment model: First results. In *Transportation Research Board 92nd Annual Meeting Compendium of Papers*, 2013.

A. Ferrara, S. Sacone, and S. Siri. *Freeway Traffic Modelling and Control*. Springer International Publishing, Cham, 2018. ISBN 978-3-319-75959-3 978-3-319-75961-6. doi: 10.1007/978-3-319-75961-6.

R. A. Fisher. *Statistical Methods for Research Workers*. Oliver & Boyd, 1925.

R. A. Fisher. *The Design of Experiments.* Oliver & Boyd, 1935a.

R. A. Fisher. The Logic of Inductive Inference. *Journal of the Royal Statistical Society*, 98 (1):39–82, 1935b. ISSN 0952-8385. doi: 10.2307/2342435.

K. Fitzpatrick, R. Avelar, and T. E. Lindheimer. Operating Speed on a Buffer-Separated Managed Lane. *Transportation Research Record: Journal of the Transportation Research Board*, 2616(1):1–9, Jan. 2017. ISSN 0361-1981, 2169-4052. doi: 10.3141/2616-01.

G. Flötteröd and J. Rohde. Operational macroscopic modeling of complex urban road intersections. *Transportation Research Part B: Methodological*, 45(6):903–922, July 2011. ISSN 01912615. doi: 10.1016/j.trb.2011.04.001.

J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 29, pages 2137–2145, 2016.

J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson. Counterfactual Multi-Agent Policy Gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 2974–2982, 2018. URL `http://arxiv.org/abs/1705.08926`.

M. Fransson and M. Sandin. *Framework for Calibration of a Traffic State Space Model*. Masters Thesis, Linkoping University, Norrkoping, Sweden, Oct. 2012.

C. Frazier and K. M. Kockelman. Chaos Theory and Transportation Systems: Instructive Example. *Transportation Research Record*, 1897(1):9–17, Jan. 2004. ISSN 0361-1981. doi: 10.3141/1897-02.

M. Garavello and B. Piccoli. Traffic Flow on a Road Network Using the Aw–Rascle Model. *Communications in Partial Differential Equations*, 31(2):243–275, Jan. 2006. ISSN 0360-5302, 1532-4133. doi: 10.1080/03605300500358053.

D. C. Gazis, R. Herman, and R. B. Potts. Car-Following Theory of Steady-State Traffic Flow. *Operations Research*, 7(4):499–505, 1959. doi: 10.1287/opre.7.4.499.

J. I. Ge and G. Orosz. Dynamics of connected vehicle systems with delayed acceleration feedback. *Transportation Research Part C: Emerging Technologies*, 46:46–64, 2014. ISSN 0968-090X. doi: 10.1016/j.trc.2014.04.014.

G. Gentile, L. Meschini, and N. Papola. Spillback congestion in dynamic traffic assignment: A macroscopic flow model with time-varying bottlenecks. *Transportation Research Part B: Methodological*, 41(10):1114–1138, 2007. doi: 10.1016/j.trb.2007.04.011.

P. Gipps. A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological*, 15(2):105–111, Apr. 1981. ISSN 01912615. doi: 10.1016/0191-2615(81)90037-0.

S. Godunov. A difference method for numerical calculation of discontinuous solutions of hydrodynamic equations. *Matematicheskii Sbornik*, 47:271–306, 1959.

N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings F Radar and Signal Processing*, 140(2):107, 1993. ISSN 0956375X. doi: 10.1049/ip-f-2.1993.0015.

B. D. Greenshields, W. Channing, and H. Miller. A study of highway capacity. *Proceedings of the 14th Annual Meeting of the Highway Research Board*, pages 448–477, 1935.

G. Grisetti, G. D. Tipaldi, C. Stachniss, W. Burgard, and D. Nardi. Fast and accurate SLAM with Rao–Blackwellized particle filters. *Robotics and Autonomous Systems*, 55(1):30–38, Jan. 2007. ISSN 0921-8890. doi: 10.1016/j.robot.2006.06.007.

Z. Guohui, Y. Shuming, and W. Yinhai. Simulation-Based Investigation on High-Occupancy Toll Lane Operations for Washington State Route 167. *Journal of Transportation Engineering*, 135(10):677–686, Oct. 2009. doi: 10.1061/(ASCE)0733-947X(2009)135:10(677).

F. L. Hall and L. M. Hall. Capacity and speed-flow analysis of the Queen Elizabeth Way in Ontario. *Transportation Research Record*, 1287:108–118, 1990. ISSN 0361-1981.

W. L. Hamilton, R. Ying, and J. Leskovec. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*, volume 30, pages 1024–1034, 2017.

B. Haut and G. Bastin. A second order model of road junctions in fluid models of traffic networks. *Networks and Heterogeneous Media*, 2(2):227–253, 2007. doi: 10.3934/nhm.2007.2.227.

D. Helbing. From microscopic to macroscopic traffic models. In J. Parisi, S. C. Müller, and W. Zimmermann, editors, *A Perspective Look at Nonlinear Media*, Lecture Notes in Physics, pages 122–139. Springer Berlin Heidelberg, 1998. ISBN 978-3-540-69681-0. doi: 10.1007/BFb0104959.

P. Hernandez-Leal, B. Kartal, and M. E. Taylor. Is multiagent deep reinforcement learning the answer or the question? A brief survey. *arXiv:1810.05587 [cs]*, Oct. 2018. URL `http://arxiv.org/abs/1810.05587`.

M. Herty and M. Rascle. Coupling Conditions for a Class of Second-Order Models for Traffic Flow. *SIAM Journal on Mathematical Analysis*, 38(2):595–616, Jan. 2006. ISSN 0036-1410, 1095-7154. doi: 10.1137/05062617X.

J. H. Holland. *Signals and Boundaries: Building Blocks for Complex Adaptive Systems*. MIT Press, Cambridge, Massachusetts, 2012. ISBN 978-0-262-30589-1.

J. H. Holland. *Complexity: A Very Short Introduction*. Oxford University Press, New York, NY, 1st edition, 2014. ISBN 978-0-19-966254-8.

S. P. Hoogendoorn and P. H. L. Bovy. Gas-Kinetic Model for Multilane Heterogeneous Traffic Flow. *Transportation Research Record*, 1678(1):150–159, Jan. 1999. ISSN 0361-1981. doi: 10.3141/1678-19.

S. P. Hoogendoorn and P. H. L. Bovy. Continuum modeling of multiclass traffic flow. *Transportation Research Part B: Methodological*, 34(2):123 – 146, Feb. 2000. ISSN 0191-2615. doi: http://dx.doi.org/10.1016/S0191-2615(99)00017-X.

S. P. Hoogendoorn and P. H. L. Bovy. State-of-the-art of vehicular traffic flow modelling. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 215(4):283–303, June 2001. ISSN 0959-6518. doi: 10.1177/095965180121500402.

R. Horowitz, A. A. Kurzhanskiy, A. Siddiqui, and M. A. Wright. Modeling and Control of HOT Lanes. Technical report, Partners for Advanced Transportation Technologies, University of California, Berkeley, Berkeley, CA, 2016. URL `https://escholarship.org/uc/item/9mx3903c`.

S. Iqbal and F. Sha. Actor-Attention-Critic for Multi-Agent Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2961–2970, Long Beach, CA, 2019. PMLR. URL `http://arxiv.org/abs/1810.02912`.

S. E. Jabari. Node modeling for congested urban road networks. *Transportation Research Part B: Methodological*, 91:229–249, Sept. 2016. ISSN 01912615. doi: 10.1016/j.trb.2016.06.001.

J. Jacobs. *The Death and Life of Great American Cities*. Random House, New York, NY, 1961. ISBN 978-0-679-74195-X.

K. Jang and M. J. Cassidy. Dual influences on vehicle speed in special-use lanes and critique of US regulation. *Transportation Research Part A: Policy and Practice*, 46(7):1108–1123, Aug. 2012. ISSN 09658564. doi: 10.1016/j.tra.2012.01.008.

K. Jang, S. Oum, and C.-Y. Chan. Traffic Characteristics of High-Occupancy Vehicle Facilities: Comparison of Contiguous and Buffer-Separated Lanes. *Transportation Research Record: Journal of the Transportation Research Board*, 2278:180–193, Dec. 2012. ISSN 0361-1981. doi: 10.3141/2278-20.

Z. Jia, C. Chen, B. Coifman, and P. Varaiya. The PeMS algorithms for accurate, real-time estimates of g-factors and speeds from single-loop detectors. In *2001 IEEE Intelligent Transportation Systems Proceedings*, pages 536–541. IEEE, 2001. ISBN 0-7803-7194-1. doi: 10.1109/ITSC.2001.948715.

J. Jiang and Z. Lu. Learning Attentional Communication for Multi-Agent Cooperation. In *Advances in Neural Information Processing Systems*, volume 31, pages 7254–7264, Montreal, Canada, 2018.

S. Jin, E. Cardellach, and F. Xie. *GNSS Remote Sensing: Theory, Methods and Applications*. Springer, Dordrecht, 2014. ISBN 978-94-007-7481-0.

W.-L. Jin. A kinematic wave theory of lane-changing traffic flow. *Transportation Research Part B: Methodological*, 44(8-9):1001–1021, Sept. 2010. ISSN 01912615. doi: 10.1016/j.trb.2009.12.014.

W.-L. Jin. A multi-commodity Lighthill–Whitham–Richards model of lane-changing traffic flow. *Transportation Research Part B: Methodological*, 57:361–377, Nov. 2013. ISSN 01912615. doi: 10.1016/j.trb.2013.06.002.

W.-L. Jin. A Riemann solver for a system of hyperbolic conservation laws at a general road junction. *Transportation Research Part B: Methodological*, 98:21–41, Apr. 2017. ISSN 01912615. doi: 10.1016/j.trb.2016.12.007.

R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, Mar. 1960. ISSN 0021-9223. doi: 10.1115/1.3662552.

R. Keener. *Theoretical Statistics: Topics for a Core Course*. Springer, New York, NY, 1st edition, 2010.

K. Kim and B.-J. Park. Safety features of freeway weaving segments with a buffer-separated high-occupancy-vehicle (HOV) lane. *International Journal of Injury Control and Safety Promotion*, 25(3):284–292, July 2018. ISSN 1745-7300, 1745-7319. doi: 10.1080/17457300.2018.1431943.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015. URL `http://arxiv.org/abs/1412.6980`.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL `https://arxiv.org/abs/1609.02907`.

M. Koshi, M. Iwasaki, and I. Ohkura. Some findings and an overview on vehicular flow characteristics. In *Proceedings of the 8th International Symposium on Transportation and Traffic Flow Theory*, volume 198, pages 403–426, 1983.

D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker. Recent Development and Applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, 5(3&4):128–138, Dec. 2012.

M. Kurant and P. Thiran. Layered Complex Networks. *Physical Review Letters*, 96(13): 138701, Apr. 2006. doi: 10.1103/PhysRevLett.96.138701.

A. A. Kurzhanskiy and P. Varaiya. Traffic management: An outlook. *Economics of Transportation*, 4(3):135–146, Sept. 2015. ISSN 22120122. doi: 10.1016/j.ecotra.2015.03.002.

T. R. Lakshmanan. The broader economic consequences of transport infrastructure investments. *Journal of Transport Geography*, 19(1):1–12, Jan. 2011. ISSN 0966-6923. doi: 10.1016/j.jtrangeo.2010.01.001.

A. Lazaridou, A. Peysakhovich, and M. Baroni. Multi-Agent Cooperation and the Emergence of (Natural) Language. In *International Conference on Learning Representations*, 2017. URL `http://arxiv.org/abs/1612.07182`.

J. P. Lebacque. The Godunov scheme and what it means for first order traffic flow models. In J. B. Lesort, editor, *Proceedings of the 13th International Symposium on Transportation and Traffic Theory*, pages 647–677. Pergamon, 1996.

J.-P. Lebacque and M. Khoshyaran. First-order macroscopic traffic flow models: Intersection modeling, network modeling. In *The 16th International Symposium on Transportation and Traffic Theory (ISTTT)*, pages 365–386, 2005.

J.-P. Lebacque, S. Mammar, and H. Haj-Salem. The Aw–Rascle and Zhang's model: Vacuum problems, existence and regularity of the solutions of the Riemann problem. *Transportation Research Part B: Methodological*, 41(7):710–721, Aug. 2007a. ISSN 01912615. doi: 10.1016/j.trb.2006.11.005.

J.-P. Lebacque, S. Mammar, and H. Haj-Salem. Generic second order traffic flow modelling. In *Proceedings of the 2007 Symposium on Transportation and Traffic Theory*, pages 755–776, 2007b.

L. Leclercq, J. Laval, and E. Chevallier. The Lagrangian coordinates and what it means for first order traffic flow models. In *Proceedings of the 17th International Symposium on Transportation and Traffic Theory*, pages 735–753. Elsevier, 2007.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature14539.

E. A. Lee and S. A. Seshia. *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*. MIT Press, Cambridge, Massachusetts, second edition edition, 2017. ISBN 978-0-262-53381-2.

A. Legrain, N. Eluru, and A. M. El-Geneidy. Am stressed, must travel: The relationship between mode choice and commuting stress. *Transportation Research Part F: Traffic Psychology and Behaviour*, 34:141–151, Oct. 2015. ISSN 1369-8478. doi: 10.1016/j.trf.2015.08.001.

E. L. Lehmann. The Fisher, Neyman-Pearson Theories of Testing Hypotheses: One Theory or Two? *Journal of the American Statistical Association*, 88(424):1242–1249, Dec. 1993. ISSN 01621459. doi: 10.2307/2291263.

Y. Li, R. Yu, C. Shahabi, and Y. Liu. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*, 2018. URL `http://arxiv.org/abs/1707.01926`.

E. Liang, R. Liaw, P. Moritz, R. Nishihara, R. Fox, K. Goldberg, J. E. Gonzalez, M. I. Jordan, and I. Stoica. RLlib: Abstractions for Distributed Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, pages 3059–3068, 2018. URL `http://arxiv.org/abs/1712.09381`.

M. J. Lighthill and G. B. Whitham. On kinematic waves I. Flood movement in long rivers. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 229(1178):281–316, May 1955a. doi: 10.1098/rspa.1955.0088.

M. J. Lighthill and G. B. Whitham. On kinematic waves II. A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 229(1178):317–345, May 1955b. ISSN 2053-9169. doi: 10.1098/rspa.1955.0089.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016. URL `http://arxiv.org/abs/1509.02971`.

H. X. Liu, X. Wu, W. Ma, and H. Hu. Real-time queue length estimation for congested signalized intersections. *Transportation Research C: Emerging Technologies*, 17(4):412–427, Aug. 2009. ISSN 0968090X. doi: 10.1016/j.trc.2009.02.003.

X. Liu, B. Schroeder, T. Thomson, Y. Wang, N. Rouphail, and Y. Yin. Analysis of Operational Interactions Between Freeway Managed Lanes and Parallel, General Purpose Lanes. *Transportation Research Record: Journal of the Transportation Research Board*, 2262:62–73, Dec. 2011. ISSN 0361-1981. doi: 10.3141/2262-07.

X. Liu, G. Zhang, Y. Lao, and Y. Wang. Modeling Traffic Flow Dynamics on Managed Lane Facility: Approach Based on Cell Transmission Model. *Transportation Research Record: Journal of the Transportation Research Board*, 2278:163–170, Dec. 2012. ISSN 0361-1981. doi: 10.3141/2278-18.

Y. Lou, Y. Yin, and J. A. Laval. Optimal dynamic pricing strategies for high-occupancy/toll lanes. *Transportation Research Part C: Emerging Technologies*, 19(1):64–74, Feb. 2011. ISSN 0968090X. doi: 10.1016/j.trc.2010.03.008.

E. Lovisari, C. Canudas de Wit, and A. Y. Kibangou. Data fusion algorithms for density reconstruction in road transportation networks. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 2804–2809. IEEE, 2015. doi: 10.1109/CDC.2015.7402641.

E. Lovisari, C. Canudas de Wit, and A. Y. Kibangou. Density/Flow reconstruction via heterogeneous sources and Optimal Sensor Placement in road networks. *Transportation Research Part C: Emerging Technologies*, 69:451–476, Aug. 2016. ISSN 0968090X. doi: 10.1016/j.trc.2016.06.019.

R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Advances in Neural Information Processing Systems*, volume 30, pages 6379–6390, June 2017. URL `http://arxiv.org/abs/1706.02275`.

M.-T. Luong, H. Pham, and C. D. Manning. Effective Approaches to Attention-based Neural Machine Translation. *arXiv:1508.04025 [cs]*, Aug. 2015. URL `http://arxiv.org/abs/1508.04025`.

Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang. Traffic Flow Prediction With Big Data: A Deep Learning Approach. *IEEE Transactions on Intelligent Transportation Systems*, 16(2): 865–873, Apr. 2015. ISSN 1524-9050. doi: 10.1109/TITS.2014.2345663.

H. Mao, Z. Zhang, Z. Xiao, and Z. Gong. Modelling the Dynamic Joint Policy of Teammates with Attention Multi-agent DDPG. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems*, 2019. URL `http://arxiv.org/abs/1811.07029`.

D. McFadden. Conditional Logit Analysis of Qualitative Choice Behavior. In P. Zarembka, editor, *Fronteirs in Econometrics*, pages 105–142. Academic Press, New York, 1973.

N. Mehr. *Smart Traffic Operation: From Human-Driven Cars to Mixed Vehicle Autonomy*. PhD Thesis, University of California, Berkeley, Berkeley, CA, 2019.

N. Mehr and R. Horowitz. How Will the Presence of Autonomous Vehicles Affect the Equilibrium State of Traffic Networks? *IEEE Transactions on Control of Network Systems*, pages 1–1, 2019. ISSN 2325-5870. doi: 10.1109/TCNS.2019.2918682.

M. Menendez and C. F. Daganzo. Effects of HOV lanes on freeway bottlenecks. *Transportation Research Part B: Methodological*, 41(8):809–822, Oct. 2007. ISSN 01912615. doi: 10.1016/j.trb.2007.03.001.

Metropolitan Transportation Commission. Bay Area Express Lanes, 2019. URL `http://bayareaexpresslanes.org`.

D. Michalaka, Y. Yin, and D. Hale. Simulating High-Occupancy Toll Lane Operations. *Transportation Research Record*, 2396(1):124–132, Jan. 2013. ISSN 0361-1981. doi: 10.3141/2396-14.

L. Mihaylova, R. Boel, and A. Hegyi. Freeway traffic estimation within particle filtering framework. *Automatica*, 43(2):290–300, Feb. 2007. ISSN 00051098. doi: 10.1016/j.automatica.2006.08.023.

J. H. Miller and S. E. Page. *Complex Adaptive Systems: An Introduction to Computational Models of Social Life*. Princeton Studies in Complexity. Princeton University Press, Princeton, N.J, 2007. ISBN 978-0-691-12702-6 978-0-691-13096-5.

M. Mitchell. *Complexity: A Guided Tour*. Oxford University Press, New York, NY, 2009. ISBN 978-0-19-512441-5.

T. M. Mitchell. The Need for Biases in Learning Generalizations. Technical Report CBM-TR-117, Rutgers University, Department of Computer Science, New Jersey, USA, 1980.

I. Mordatch and P. Abbeel. Emergence of Grounded Compositional Language in Multi-Agent Populations. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 1495–1502, 2018.

P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, and I. Stoica. Ray: A Distributed Framework for Emerging AI Applications. In *13th USENIX Symposium on Operating Systems Design and Implementation*, pages 561–577, Oct. 2018. URL `http://arxiv.org/abs/1712.05889`.

A. H. Munnell. Policy Watch: Infrastructure Investment and Economic Growth. *Journal of Economic Perspectives*, 6(4):189–198, Dec. 1992. ISSN 0895-3309. doi: 10.1257/jep.6.4.189.

L. Munoz, X. Sun, D. Sun, G. Gomes, and R. Horowitz. Methodological calibration of the Cell Transmission Model. In *Proceedings of the American Control Conference*, pages 798–803, Boston, MA, 2004.

A. Muralidharan and R. Horowitz. Imputation of Ramp Flow Data for Freeway Traffic Simulation. *Transportation Research Record*, 2099(1):58–64, Jan. 2009. ISSN 0361-1981. doi: 10.3141/2099-07.

A. Muralidharan, G. Dervisoglu, and R. Horowitz. Freeway traffic flow simulation using the Link Node Cell transmission model. In *Proc. of the 2009 American Control Conference*, pages 2916–2921, June 2009. doi: 10.1109/ACC.2009.5160597.

K. Nagel and S. Rasmussen. Traffic at the edge of chaos. In *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, volume 4, pages 222–235. MIT Press, 1994. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.13.433&rep=rep1&type=pdf`.

G. Neubig. Neural Machine Translation and Sequence-to-sequence Models: A Tutorial. *arXiv:1703.01619 [cs, stat]*, Mar. 2017. URL `http://arxiv.org/abs/1703.01619`.

G. F. Newell. A simplified theory of kinematic waves in highway traffic, part II: Queueing at freeway bottlenecks. *Transportation Research Part B: Methodological*, 27(4):289–303, Aug. 1993. ISSN 0191-2615. doi: 10.1016/0191-2615(93)90039-D.

J. Neyman and E. S. Pearson. On the use and interpretation of certain test criteria for purposes of statistical inference: Part I. *Biometrika*, 20A:175–240, 1928.

J. Neyman and E. S. Pearson. On the problem of the most efficient test of statistical hypotheses. *Philosophical Transactions of the Royal Society of London A*, 231:289–337, 1933a.

J. Neyman and E. S. Pearson. The testing of statistical hypotheses in relation to probabilites a priori. *Proceedings of the Cambridge Philosophical Society*, 29:492–510, 1933b.

D. Ngoduy. *Macroscopic Discontinuity Modeling for Multiclass Multilane Traffic Flow Operations*. PhD Thesis, Delft University of Technology, 2006.

D. Ngoduy and M. Maher. Calibration of second order traffic models using continuous cross entropy method. *Transportation Research Part C: Emerging Technologies*, 24:102–121, Oct. 2012. ISSN 0968090X. doi: 10.1016/j.trc.2012.02.007.

D. Ni and J. D. Leonard. A simplified kinematic wave model at a merge bottleneck. *Applied Mathematical Modelling*, 29(11):1054–1072, Nov. 2005. ISSN 0307904X. doi: 10.1016/j.apm.2005.02.008.

R. B. Northrop. *Introduction to Complexity and Complex Systems*. CRC Press, Boca Raton, 2011. ISBN 978-1-4398-3901-0.

J. Obenberger. Managed Lanes: Combining Access Control, Vehicle Eligibility, and Pricing Strategies Can Help Mitigate Congestion and Improve Mobility on the Nation's Busiest Roadways. *Public Roads*, 68(3):48–55, Nov/Dec 2004. URL `http://www.fhwa.dot.gov/publications/publicroads/04nov/08.cfm`. Publication Number FHWA-HRT-05-002.

G. Orosz, R. E. Wilson, and G. Stépán. Traffic jams: Dynamics and control. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368 (1928):4455–4479, Oct. 2010. ISSN 1364-503X, 1471-2962. doi: 10.1098/rsta.2010.0205.

G. A. Pagani and M. Aiello. The Power Grid as a complex network: A survey. *Physica A: Statistical Mechanics and its Applications*, 392(11):2688–2700, June 2013. ISSN 0378-4371. doi: 10.1016/j.physa.2013.01.023.

M. Papageorgiou, H. Hadj-Salem, and J. Blosseville. ALINEA: A local feedback control law for on-ramp metering. *Transportation Research Record*, 1320:58–64, 1991.

M. Papageorgiou, C. Kiakaki, V. Dinopoulou, A. Kotsialos, and Yibing Wang. Review of road traffic control strategies. *Proceedings of the IEEE*, 91(12):2043–2067, Dec. 2003. ISSN 0018-9219. doi: 10.1109/JPROC.2003.819610.

A. D. Patire, M. Wright, B. Prodhomme, and A. M. Bayen. How much GPS data do we need? *Transportation Research Part C: Emerging Technologies*, 58:325 – 342, 2015. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2015.02.011.

PeMS. California Performance Measurement System, 2019. URL `http://pems.dot.ca.gov`.

P. Peng, Y. Wen, Y. Yang, Q. Yuan, Z. Tang, H. Long, and J. Wang. Multiagent Bidirectionally-Coordinated Nets: Emergence of Human-level Coordination in Learning to Play StarCraft Combat Games. *arXiv:1703.10069 [cs]*, Mar. 2017. URL `http://arxiv.org/abs/1703.10069`.

B. Ponnu and B. Coifman. Speed-spacing dependency on relative speed from the adjacent lane: New insights for car following models. *Transportation Research Part B: Methodological*, 82: 74–90, Dec. 2015. ISSN 01912615. doi: 10.1016/j.trb.2015.09.012.

B. Ponnu and B. Coifman. When adjacent lane dependencies dominate the uncongested regime of the fundamental relationship. *Transportation Research Part B: Methodological*, 104:602–615, Oct. 2017. ISSN 01912615. doi: 10.1016/j.trb.2017.05.006.

A. Poole and A. Kotsialos. METANET Model Validation using a Genetic Algorithm. *IFAC Proceedings Volumes*, 45(24):7–12, Sept. 2012. ISSN 14746670. doi: 10.3182/20120912-3-BG-2031.00002.

A. Poole and A. Kotsialos. Second order macroscopic traffic flow model validation using automatic differentiation with resilient backpropagation and particle swarm optimisation algorithms. *Transportation Research Part C: Emerging Technologies*, 71:356–381, Oct. 2016. ISSN 0968090X. doi: 10.1016/j.trc.2016.07.008.

R. Rajkumar, I. Lee, L. Sha, and J. Stankovic. Cyber-physical systems: The next computing revolution. In *Design Automation Conference*, pages 731–736, June 2010. doi: 10.1145/1837274.1837461.

T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, and S. Whiteson. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4292–4301, Stockholm, Sweden, Mar. 2018. URL http://arxiv.org/abs/1803.11485.

S. J. Redding and M. A. Turner. Transportation Costs and the Spatial Organization of Economic Activity. In G. Duranton, J. V. Henderson, and W. C. Strange, editors, *Handbook of Regional and Urban Economics*, volume 5 of *Handbook of Regional and Urban Economics*, pages 1339–1398. Elsevier, Jan. 2015. doi: 10.1016/B978-0-444-59531-7.00020-X. URL http://www.sciencedirect.com/science/article/pii/B978044459531700020X.

J. Reilly, S. Martin, M. Payer, and A. M. Bayen. Creating complex congestion patterns via multi-objective optimal freeway traffic control with application to cyber-security. *Transportation Research Part B: Methodological*, 91:366–382, Sept. 2016. ISSN 01912615. doi: 10.1016/j.trb.2016.05.017.

P. F. Richards. Shock waves on the highway. *Operations Research*, 4(1):42–51, 1956. doi: 10.1287/opre.4.1.42.

M. Rosvall, A. Trusina, P. Minnhagen, and K. Sneppen. Networks and Cities: An Information Perspective. *Physical Review Letters*, 94(2):028701, Jan. 2005. doi: 10.1103/PhysRevLett.94.028701.

D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan. Planning for Autonomous Cars that Leverage Effects on Human Actions. In *Robotics: Science and Systems XII*. Robotics: Science and Systems Foundation, 2016. ISBN 978-0-9923747-2-3. doi: 10.15607/RSS.2016.XII.029.

J. W. Schneider. Null hypothesis significance tests. A mix-up of two different theories: The basis for widespread confusion and numerous misinterpretations. *Scientometrics*, 102(1):411–432, Jan. 2015. ISSN 0138-9130, 1588-2861. doi: 10.1007/s11192-014-1251-5.

J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In *International Conference on Learning Representations*, 2016. URL http://arxiv.org/abs/1506.02438.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms. *arXiv:1707.06347 [cs]*, July 2017. URL http://arxiv.org/abs/1707.06347.

T. Seo, A. M. Bayen, T. Kusakabe, and Y. Asakura. Traffic state estimation on highway: A comprehensive survey. *Annual Reviews in Control*, 43:128–151, 2017. ISSN 13675788. doi: 10.1016/j.arcontrol.2017.03.005.

P. Shaw, J. Uszkoreit, and A. Vaswani. Self-Attention with Relative Position Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*, volume 2, pages 464–468, New Orleans, LA, June 2018. doi: http://dx.doi.org/10.18653/v1/N18-2074.

J. Shen and X. Jin. Detailed traffic animation for urban road networks. *Graphical Models*, 74 (5):265–282, Sept. 2012. ISSN 15240703. doi: 10.1016/j.gmod.2012.04.002.

Y. Shiomi, T. Taniguchi, N. Uno, H. Shimamoto, and T. Nakamura. Multilane first-order traffic flow model with endogenous representation of lane-flow equilibrium. *Transportation Research Part C: Emerging Technologies*, 59:198–215, Oct. 2015. ISSN 0968090X. doi: 10.1016/j.trc.2015.07.002.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, Jan. 2016. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature16961.

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, Oct. 2017. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature24270.

H. A. Simon. *The Sciences of the Artificial*. MIT Press, Cambridge, Massachusetts, third edition, 1996.

K. A. Small and E. T. Verhoef. *The Economics of Urban Transportation*. Routledge, New York, NY, USA, 2nd edition, Oct. 2007. ISBN 978-0-203-64230-6. doi: 10.4324/9780203642306. URL `https://www.taylorfrancis.com/books/9780203642306`.

E.-S. Smits, M. C. Bliemer, A. J. Pel, and B. van Arem. A family of macroscopic node models. *Transportation Research Part B: Methodological*, 74:20–39, Apr. 2015. ISSN 01912615. doi: 10.1016/j.trb.2015.01.002.

R. E. Stern, S. Cui, M. L. Delle Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, R. Haulcy, H. Pohlmann, F. Wu, B. Piccoli, B. Seibold, J. Sprinkle, and D. B. Work. Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments. *Transportation Research Part C: Emerging Technologies*, 89:205–221, Apr. 2018. ISSN 0968-090X. doi: 10.1016/j.trc.2018.02.005.

P. Stone and M. Veloso. Multiagent Systems: A Survey from a Machine Learning Perspective:. Technical report, Defense Technical Information Center, Fort Belvoir, VA, Dec. 1997. URL `http://www.dtic.mil/docs/citations/ADA333248`.

Y. Sugiyama, M. Fukui, M. Kikuchi, K. Hasebe, A. Nakayama, K. Nishinari, S.-i. Tadaki, and S. Yukawa. Traffic jams without bottlenecks—experimental evidence for the physical mechanism of the formation of a jam. *New Journal of Physics*, 10(3):033001, Mar. 2008. ISSN 1367-2630. doi: 2008030504130900.

S. Sukhbaatar, A. Szlam, and R. Fergus. Learning Multiagent Communication with Backpropagation. In *Advances in Neural Information Processing Systems*, volume 29, pages 2244–2252, 2016.

A. Sumalee, R. Zhong, T. Pan, and W. Szeto. Stochastic cell transmission model (SCTM): A stochastic dynamic traffic model for traffic state surveillance and assignment. *Transportation Research Part B: Methodological*, 45(3):507–533, Mar. 2011. ISSN 01912615. doi: 10.1016/j.trb.2010.09.006.

X. Sun, L. Munoz, and R. Horowitz. Mixture Kalman filter based highway congestion mode and vehicle density estimator and its application. In *Proceedings of the 2004 American Control Conference*, volume 3, pages 2098–2103 vol.3, June 2004. doi: 10.23919/ACC.2004.1383770.

P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel. Value-Decomposition Networks For Cooperative Multi-Agent Learning. *arXiv:1706.05296 [cs]*, 2017. URL `http://arxiv.org/abs/1706.05296`.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, volume 27, pages 3104–3112, 2014.

System Metrics Group, Inc. Contra Costa County I-680 Corridor System Management Plan Final Report. Technical report, Caltrans District 4, 2015. URL `http://dot.ca.gov/hq/tpp/corridor-mobility/CSMPs/d4_CSMPs/D04_I680_CSMP_Final_Revised_Report_2015-05-29.pdf`.

C. M. Tampère, R. Corthout, D. Cattrysse, and L. H. Immers. A generic class of first order node models for dynamic macroscopic simulation of traffic flows. *Transportation Research Part B: Methodological*, 45(1):289–309, Jan. 2011. ISSN 01912615. doi: 10.1016/j.trb.2010.06.004.

T. Thomson, X. C. Liu, Y. Wang, B. J. Schroeder, and N. M. Rouphail. Operational Performance and Speed–Flow Relationships for Basic Managed Lane Segments. *Transportation Research Record: Journal of the Transportation Research Board*, 2286(1):94–104, Jan. 2012. ISSN 0361-1981, 2169-4052. doi: 10.3141/2286-11.

M. Treiber and A. Kesting. *Traffic Flow Dynamics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-32459-8 978-3-642-32460-4. doi: 10.1007/978-3-642-32460-4.

M. Treiber, A. Hennecke, and D. Helbing. Derivation, properties, and simulation of a gas-kinetic-based, nonlocal traffic model. *Physical Review E*, 59(1):239–253, Jan. 1999. doi: 10.1103/PhysRevE.59.239.

M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, Aug. 2000. ISSN 1063-651X, 1095-3787. doi: 10.1103/PhysRevE.62.1805.

J. Treiterer and J. Myers. The hysteresis phenomenon in traffic flow. In *Proceedings of the 6th International Symposium on Transportation and Traffic Theory*, volume 6, pages 13–38, Sydney, Australia, Aug. 1974.

T. H. Trinh, A. M. Dai, M.-T. Luong, and Q. V. Le. Learning Longer-term Dependencies in RNNs with Auxiliary Losses. In *Proceedings of the 35th International Conference on Machine Learning*, 2018. URL `http://arxiv.org/abs/1803.00144`.

J. van Lint, S. Hoogendoorn, and M. Schreuder. Fastlane: New Multiclass First-Order Traffic Flow Model. *Transportation Research Record: Journal of the Transportation Research Board*, 2088:177–187, Dec. 2008. ISSN 0361-1981. doi: 10.3141/2088-19.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017. URL `http://arxiv.org/abs/1706.03762`.

P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph Attention Networks. In *International Conference on Learning Representations*, 2018.

S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to Sequence – Video to Text. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4534–4542, Santiago, Chile, Dec. 2015. IEEE. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.515.

A. Vespignani. Complex networks: The fragility of interdependency. *Nature*, 464(7291): 984–985, Apr. 2010. ISSN 1476-4687. doi: 10.1038/464984a.

E. Vinitsky, A. Kreidieh, L. Le Flem, N. Kheterpal, K. Jang, C. Wu, R. Liaw, E. Liang, and A. Bayen. Benchmarks for reinforcement learning in mixed-autonomy traffic. In *Conference on Robot Learning*, pages 399–409, Zurich, Switzerland, 2018.

M. M. Waldrop. *Complexity: The Emerging Science at the Edge of Order and Chaos*. Simon & Schuster, New York, 1992. ISBN 978-0-671-76789-1.

J. L. Walker, E. Ehlers, I. Banerjee, and E. R. Dugundji. Correcting for endogeneity in behavioral choice models with social influence variables. *Transportation Research Part A: Policy and Practice*, 45(4):362–374, May 2011. ISSN 0965-8564. doi: 10.1016/j.tra.2011.01.003.

R. Wang, Y. Li, and D. B. Work. Comparing traffic state estimators for mixed human and automated traffic flows. *Transportation Research Part C: Emerging Technologies*, 78: 95–110, May 2017. ISSN 0968-090X. doi: 10.1016/j.trc.2017.02.011.

Y. Wang and M. Papageorgiou. Real-time freeway traffic state estimation based on extended Kalman filter: A general approach. *Transportation Research Part B: Methodological*, 39(2): 141–167, Feb. 2005. ISSN 01912615. doi: 10.1016/j.trb.2004.03.003.

Z. Wang, G. Wu, and M. J. Barth. A Review on Cooperative Adaptive Cruise Control (CACC) Systems: Architectures, Controls, and Applications. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2884–2891, Maui, HI, Nov. 2018. IEEE. ISBN 978-1-72810-321-1 978-1-72810-323-5. doi: 10.1109/ITSC.2018.8569947.

G. C. K. Wong and S. C. Wong. A multi-class traffic flow model–an extension of LWR model with heterogeneous drivers. *Transportation Research Part A: Policy and Practice*, 36(9): 827–841, 2002. doi: 10.1016/S0965-8564(01)00042-8.

D. B. Work, S. Blandin, O. P. Tossavainen, B. Piccoli, and A. M. Bayen. A Traffic Model for Velocity Data Assimilation. *Applied Mathematics Research eXpress*, 2010(1):1–35, Apr. 2010. ISSN 1687-1200, 1687-1197. doi: 10.1093/amrx/abq002.

M. Wright and R. Horowitz. Fusing Loop and GPS Probe Measurements to Estimate Freeway Density. *IEEE Transactions on Intelligent Transportation Systems*, 17(12):3577–3590, Dec. 2016. ISSN 1524-9050. doi: 10.1109/TITS.2016.2565438.

M. Wright, R. Horowitz, and A. A. Kurzhanskiy. A dynamic system characterization of road network node models. In *Proceedings of the 10th IFAC Symposium on Nonlinear Control Systems*, volume 49, pages 1054–1059, Monterey, CA, Aug. 2016. doi: 10.1016/j.ifacol.2016.10.307.

M. A. Wright and R. Horowitz. Generic second-order macroscopic traffic node model for general multi-input multi-output road junctions via a dynamic system approach. *arXiv:1707.09346 [cs, math]*, July 2017a. URL `http://arxiv.org/abs/1707.09346`.

M. A. Wright and R. Horowitz. Particle-filter-enabled real-time sensor fault detection without a model of faults. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 5757–5763, Dec. 2017b. doi: 10.1109/CDC.2017.8264529.

M. A. Wright and R. Horowitz. A Framework for Robust Assimilation of Potentially Malign Third-Party Data, and its Statistical Meaning. *IEEE Intelligent Transportation Systems Magazine*, 2018. URL `https://arxiv.org/abs/1809.01271`. Special Issue on GNSS for ITS, To appear.

M. A. Wright and R. Horowitz. Attentional Policies for Cross-Context Multi-Agent Reinforcement Learning. *arXiv:1905.13428 [cs, stat]*, May 2019a. URL `http://arxiv.org/abs/1905.13428`.

M. A. Wright and R. Horowitz. Macro-Scale Lane-Changing and Overtaking Behaviors at Road Junctions: Continuous-Time and Optimization-Problem Formulations. *In preparation*, 2019b.

M. A. Wright, G. Gomes, R. Horowitz, and A. A. Kurzhanskiy. On node models for high-dimensional road networks. *Transportation Research Part B: Methodological*, 105:212–234, Nov. 2017. ISSN 01912615. doi: 10.1016/j.trb.2017.09.001.

M. A. Wright, R. Horowitz, and A. A. Kurzhanskiy. A Dynamic-System-Based Approach to Modeling Driver Movements Across General-Purpose/Managed Lane Interfaces. In *Proceedings of the ASME 2018 Dynamic Systems and Controls Conference (DSCC)*, page V002T15A003. ASME, Sept. 2018. ISBN 978-0-7918-5190-6. doi: 10.1115/DSCC2018-9125.

M. A. Wright, S. F. G. Ehlers, and R. Horowitz. Neural-Attention-Based Deep Learning Architectures for Modeling Traffic Dynamics on Lane Graphs. In *2019 IEEE Conference on Intelligent Transportation Systems*, 2019a. URL `https://arxiv.org/abs/1904.08831`. To appear.

M. A. Wright, R. Horowitz, and A. A. Kurzhanskiy. Macroscopic Modeling, Calibration, and Simulation of Managed Lane-Freeway Networks, Part I: Topological and Phenomenological Modeling. *to be submitted to Transportation Research Part B: Methodological*, 2019b. URL `https://arxiv.org/abs/1609.09470`.

M. A. Wright, R. Horowitz, and A. A. Kurzhanskiy. Macroscopic Modeling, Calibration, and Simulation of Managed Lane-Freeway Networks, Part II: Network-scale Calibration Methods and Case Studies. *to be submitted to Transportation Research Part B: Methodological*, 2019c.

C. Wu, A. Kreidieh, K. Parvate, E. Vinitsky, and A. M. Bayen. Flow: Architecture and Benchmarking for Reinforcement Learning in Traffic Control. *arXiv:1710.05465 [cs]*, 2017a. URL `http://arxiv.org/1710.05465`.

C. Wu, A. Kreidieh, E. Vinitsky, and A. M. Bayen. Emergent Behaviors in Mixed-Autonomy Traffic. In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 398–407. PMLR, Nov. 2017b. URL `http://proceedings.mlr.press/v78/wu17a.html`.

B. M. Yerra and D. M. Levinson. The emergence of hierarchy in transportation networks. *The Annals of Regional Science*, 39(3):541–553, Sept. 2005. ISSN 1432-0592. doi: 10.1007/s00168-005-0230-4.

A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *International Conference on Learning Representations*, 2018. URL `https://arxiv.org/abs/1804.09541`.

Y. Yuan, J. W. C. van Lint, R. E. Wilson, F. van Wageningen-Kessels, and S. P. Hoogendoorn. Real-Time Lagrangian Traffic State Estimator for Freeways. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):59–70, Mar. 2012. ISSN 1524-9050, 1558-0016. doi: 10.1109/TITS.2011.2178837.

H. Zhang. A mathematical theory of traffic hysteresis. *Transportation Research Part B: Methodological*, 33(1):1–23, Feb. 1999. ISSN 01912615. doi: 10.1016/S0191-2615(98)00022-8.

H. Zhang. A non-equilibrium traffic model devoid of gas-like behavior. *Transportation Research Part B: Methodological*, 36(3):275–290, Mar. 2002. ISSN 01912615. doi: 10.1016/S0191-2615(00)00050-3.

Z. Zheng. Recent developments and research needs in modeling lane changing. *Transportation Research Part B: Methodological*, 60:16–32, Feb. 2014. ISSN 01912615. doi: 10.1016/j.trb.2013.11.009.

Appendix A

A Link Model

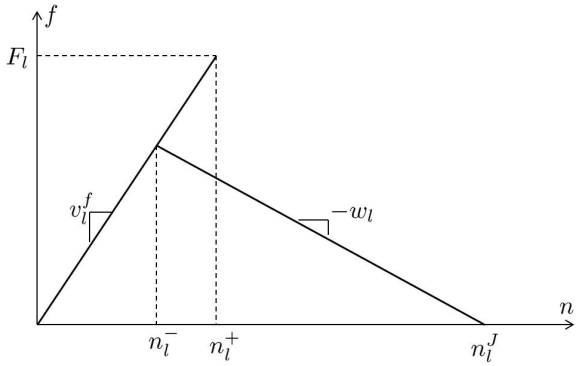


Figure A.1: The "backwards lambda" fundamental diagram.

For the majority of this work, we remain agnostic as to the particular functional relationship between density ρ_l, demand S_l (per commodity, ρ_l^c, S_l^c) and supply R_l, and flow f_l (also called the fundamental diagram) used in our first-order macroscopic model (4.1). Where a particular fundamental diagram is required, i.e. for the example implementation of the friction effect in (4.10), (4.11), (4.12), and (4.13), we use a fundamental diagram from Horowitz et al. (2016), shown in Figure A.1. This fundamental diagram captures the traffic hysteresis behavior with the "backwards lambda" shape often observed in detector data (Koshi et al., 1983):

$$S_l^c(t) = v_l^f(t)\rho_l^c(t)\min\left\{1, \frac{F_l(t)}{v_l^f(t)\sum_{c=1}^{C}\rho_l^c(t)}\right\}, \quad S_l(t) = \sum_{c=1}^{C} S_l^c(t), \tag{A.1}$$

$$R_l(t) = (1-\theta_l(t))\,F_l(t) + \theta_l(t)w_l(t)\left(\rho_l^J(t) - \sum_{c=1}^{C}\rho_l^c(t)\right), \tag{A.2}$$

where, for link l, F_l is the capacity, v_l^f is the free flow speed, w_l is the congestion wave speed, ρ_l^J is the jam density, and $\rho_l^- = \frac{w_l\rho_l^J}{v_l^f + w_l}$ and $\rho_l^+ = \frac{F_l}{v_l^f}$ are called the *low* and *high critical*

densities, respectively. As written here and used in this work, F_l, v_l^f, and w_l are in units per simulation timestep. The variable $\theta_l(t)$ is a congestion metastate of l, which encodes the hysteresis:

$$\theta_l(t) = \begin{cases} 0 & \rho_l(t) \leq \rho_l^-, \\ 1 & \rho_l(t) > \rho_l^+, \\ \theta_l(t-1) & \rho_l^- < \rho_l(t) \leq \rho_l^+, \end{cases} \tag{A.3}$$

where $\rho_l(t) = \sum_{c=1}^{C} \rho_l^c(t)$.

Examining (A.3) and (A.2), we see that when a link's density goes above ρ_l^+ (i.e., when it becomes congested), its ability to receive flow is reduced until the density falls below ρ_l^-.

An image of (A.1) and (A.2) overlaid on each other, giving a schematic image of the fundamental diagram, is shown in Figure A.1. Unless $\rho_l^- = \rho_l^+$, when it assumes triangular shape, the fundamental diagram is not a function of density alone (i.e., without $\theta_l(t)$): $\rho_l(t) \in \left(\rho_l^-, \rho_l^+\right]$ admits two possible flow values.

Appendix B

Dynamic Split Ratio Solver

Throughout this article, we have made reference to a dynamic-system-based method for solving for partially- or fully-undefined split ratios from Wright et al. (2017). This split ratio solver is designed to implicitly solve the logit-based split ratio problem

$$\beta_{ij}^c = \frac{\exp\left(\frac{\sum_{i=1}^{M}\sum_{c=1}^{C} S_{ij}^c}{R_j}\right)}{\sum_{j'=1}^{N}\exp\left(\frac{\sum_{i=1}^{M}\sum_{c=1}^{C} S_{ij'}^c}{R_{j'}}\right)}, \tag{B.1}$$

which cannot be solved explicitly, as the S_{ij}^c's are also functions of the β_{ij}^c's. The problem (B.1) is chosen to be a node-local problem that does not rely on information from the link model (beyond supplies and demands), and is thus independent of the choice of link model (Wright et al., 2017).

The solution algorithm is as follows, reproduced from Wright et al. (2017). More discussion is available in the reference.

- Define the set of commodity movements for which split ratios are known as $\mathcal{B} = \left\{\{i,j,c\} : \beta_{ij}^c \in [0,1]\right\}$, and the set of commodity movements for which split ratios are to be computed as $\overline{\mathcal{B}} = \left\{\{i,j,c\} : \beta_{ij}^c \text{ are unknown}\right\}$.
- For a given input link i and commodity c such that $S_i^c = 0$, assume that all split ratios are known: $\{i,j,c\} \in \mathcal{B}$.[1]
- Define the set of output links for which there exist unknown split ratios as $V = \left\{j : \exists\, \{i,j,c\} \in \overline{\mathcal{B}}\right\}$.
- Assuming that for a given input link i and commodity c, the split ratios must sum up to 1, define the unassigned portion of flow by $\overline{\beta}_i^c = 1 - \sum_{j:\{i,j,c\}\in\mathcal{B}} \beta_{ij}^c$.

[1] If split ratios were undefined in this case, they could be assigned arbitrarily.

- For a given input link i and commodity c such that there exists at least one commodity movement $\{i, j, c\} \in \overline{\mathcal{B}}$, assume $\overline{\beta}_i^c > 0$, otherwise the undefined split ratios can be trivially set to 0.

- For every output link $j \in V$, define the set of input links that have an unassigned demand portion directed toward this output link by $U_j = \left\{i : \exists \{i, j, c\} \in \overline{\mathcal{B}}\right\}$.

- For a given input link i and commodity c, define the set of output links for which split ratios for which are to be computed as $V_i^c = \{j : \exists i \in U_j\}$, and assume that if nonempty, this set contains at least two elements, otherwise a single split ratio can be trivially set equal to $\overline{\beta}_i^c$.

- Assume that input link priorities are non-negative, $p_i \geq 0$, $i = 1, \ldots, M$, and $\sum_{i=1}^{M} p_i = 1$.

- Define the set of input links with zero priority: $U_{zp} = \{i : p_i = 0\}$. To enable split ratio assignment for inputs with zero priorities, perform regularization:

$$\tilde{p}_i = p_i \left(1 - \frac{|U_{zp}|}{M}\right) + \frac{1}{M}\frac{|U_{zp}|}{M} = p_i \frac{M - |U_{zp}|}{M} + \frac{|U_{zp}|}{M^2}, \tag{B.2}$$

  where $|U_{zp}|$ denotes the number of elements in set U_{zp}. Expression (B.2) implies that the regularized input priority $\tilde{p}_i$ consists of two parts: (1) the original input priority p_i normalized to the portion of input links with positive priorities; and (2) uniform distribution among M input links, $\frac{1}{M}$, normalized to the portion of input links with zero priorities.

  Note that the regularized priorities $\tilde{p}_i > 0$, $i = 1, \ldots, M$, and $\sum_{i=1}^{M} \tilde{p}_i = 1$.

The algorithm for distributing $\overline{\beta}_i^c$ among the commodity movements in $\overline{\mathcal{B}}$ (that is, assigning values to the a priori unknown split ratios) aims at maintaining output links as uniform in their demand-supply ratios as possible. At each iteration k, two quantities are identified: $\mu^+(k)$, which is the largest *oriented* demand-supply ratio produced by the split ratios that have been assigned so far, and $\mu^-(k)$, which is the smallest oriented demand-supply ratio whose input link, denoted i^-, still has some unclaimed split ratio. Once these two quantities are found, the commodity c^- in i^- with the smallest unallocated demand has some of its demand directed to the j corresponding to $\mu^-(k)$ to bring $\mu^-(k)$ up to $\mu^+(k)$ (or, if this is not possible due to insufficient demand, all such demand is directed).

To summarize, in each iteration k, the algorithm attempts to bring the smallest oriented demand-supply ratio $\mu^+(k)$ up to the largest oriented demand-supply ratio $\mu^-(k)$. If it turns out that all such oriented demand-supply ratios become perfectly balanced, then the demand-supply ratios $(\sum_i \sum_c S_{ij}^c)/R_j$ are as well.

The algorithm is:

1. Initialize:

$$\begin{aligned}
\tilde{\beta}_{ij}^{c}(0) &:= \begin{cases} \beta_{ij}^{c}, & \text{if } \{i,j,c\} \in \mathcal{B}, \\ 0, & \text{otherwise;} \end{cases} \\
\overline{\beta}_{i}^{c}(0) &:= \overline{\beta}_{i}^{c}; \\
\tilde{U}_{j}(0) &= U_{j}; \\
\tilde{V}(0) &= V; \\
k &:= 0,
\end{aligned}$$

   Here $\tilde{U}_j(k)$ is the remaining set of input links with some unassigned demand, which may be directed to output link j; and $\tilde{V}(k)$ is the remaining set of output links, to which the still-unassigned demand may be directed.

2. If $\tilde{V}(k) = \emptyset$, stop. The sought-for split ratios are $\left\{\tilde{\beta}_{ij}^{c}(k)\right\}$, $i = 1, \ldots, M$, $j = 1, \ldots, N$, $c = 1, \ldots, C$.

3. Calculate the remaining unallocated demand:

$$\overline{S}_{i}^{c}(k) = \overline{\beta}_{i}^{c}(k) S_{i}^{c}, \quad i = 1, \ldots, M, \;\; c = 1, \ldots, C.$$

4. For all input-output link pairs, calculate oriented demand:

$$\tilde{S}_{ij}^{c}(k) = \tilde{\beta}_{ij}^{c}(k) S_{i}^{c}.$$

5. For all input-output link pairs, calculate oriented priorities:

$$\tilde{p}_{ij}(k) = \tilde{p}_{i} \frac{\sum_{c=1}^{C} \gamma_{ij}^{c} S_{i}^{c}}{\sum_{c=1}^{C} S_{i}^{c}} \tag{B.3}$$

   with

$$\gamma_{ij}^{c}(k) = \begin{cases} \beta_{ij}^{c}, & \text{if split ratio is defined a priori: } \{i,j,c\} \in \mathcal{B}, \\ \tilde{\beta}_{ij}^{c}(k) + \frac{\overline{\beta}_{i}^{c}(k)}{|V_{i}^{c}|}, & \text{otherwise,} \end{cases} \tag{B.4}$$

   where $|V_i^c|$ denotes the number of elements in the set V_i^c. Examining the expression (B.3)-(B.4), one can see that the split ratios $\tilde{\beta}_{ij}^{c}(k)$, which are not fully defined yet, are complemented with a fraction of $\overline{\beta}_{i}^{c}(k)$ inversely proportional to the number of output links among which the flow of commodity c from input link i can be distributed.

   Note that in this step we are using *regularized* priorities $\tilde{p}_i$ as opposed to the original p_i, $i = 1, \ldots, M$. This is done to ensure that inputs with $p_i = 0$ are not ignored in the split ratio assignment.

6. Find the largest oriented demand-supply ratio:

$$\mu^{+}(k) = \max_{j} \max_{i} \frac{\sum_{c=1}^{C} \tilde{S}_{ij}^{c}(k)}{\tilde{p}_{ij}(k) R_{j}} \sum_{i \in U_j} \tilde{p}_{ij}(k).$$

7. Define the set of all output links in $\tilde{V}(k)$, where the minimum of the oriented demand-supply ratio is achieved:

$$Y(k) = \arg \min_{j \in \tilde{V}(k)} \min_{i \in \tilde{U}_j(k)} \frac{\sum_{c=1}^{C} \tilde{S}_{ij}^c(k)}{\tilde{p}_{ij}(k) R_j} \sum_{i \in U_j} \tilde{p}_{ij}(k),$$

and from this set pick the output link j^- with the smallest output demand-supply ratio (when there are multiple minimizing output links, any of the minimizing output links may be chosen as j^-):

$$j^- = \arg \min_{j \in Y(k)} \frac{\sum_{i=1}^{M} \sum_{c=1}^{C} \tilde{S}_{ij}^c(k)}{R_j}.$$

8. Define the set of all input links, where the minimum of the oriented demand-supply ratio for the output link j^- is achieved:

$$W_{j^-}(k) = \arg \min_{i \in \tilde{U}_{j^-}(k)} \frac{\sum_{c=1}^{C} \tilde{S}_{ij^-}^c(k)}{\tilde{p}_{ij^-}(k) R_{j^-}} \sum_{i \in U_{j^-}} \tilde{p}_{ij^-}(k),$$

and from this set pick the input link i^- and commodity c^- with the smallest remaining unallocated demand:

$$\{i^-, c^-\} = \arg \min_{\substack{i \in W_{j^-}(k), \\ c : \overline{\beta}_{i^-}^c(k) > 0}} \overline{S}_i^c(k).$$

9. Define the smallest oriented demand-supply ratio:

$$\mu^-(k) = \frac{\sum_{c=1}^{C} \tilde{S}_{i^-j^-}^c(k)}{\tilde{p}_{i^-j^-}(k) R_{j^-}} \sum_{i \in U_{j^-}} \tilde{p}_{ij^-}(k).$$

- If $\mu^-(k) = \mu^+(k)$, the oriented demands created by the split ratios that have been assigned as of iteration k, $\tilde{\beta}_{ij}^c(k)$, are perfectly balanced among the output links, and to maintain this, all remaining unassigned split ratios should be distributed proportionally to the allocated supply:

$$\begin{aligned}
\tilde{\beta}_{ij}^c(k+1) = \; & \tilde{\beta}_{ij}^c(k) + \frac{R_j}{\sum_{j' \in V_i^c(k)} R_{j'}} \overline{\beta}_i^c(k), \\
& c : \overline{\beta}_i^c(k) > 0, \; i \in \tilde{U}_j(k), \; j \in \tilde{V}(k); \\
\overline{\beta}_i^c(k+1) = 0, \quad & c : \; \overline{\beta}_i^c(k) > 0, \;\; i \in \tilde{U}_j(k), \;\; j \in \tilde{V}(k); \\
\tilde{U}_j(k+1) = \emptyset, \quad & j \in \tilde{V}(k); \\
\tilde{V}(k+1) = \emptyset. &
\end{aligned} \tag{B.5}$$

If the algorithm ends up at this point, we have emptied $\tilde{V}(k+1)$ and are done.

- Else, assign:

$$\Delta\tilde{\beta}^{c^-}_{i^-j^-}(k) = \min\left\{\overline{\beta}^{c^-}_{i^-}(k), \left(\frac{\mu^+(k)\tilde{p}_{i^-j^-}(k)R_{j^-}}{\overline{S}^{c^-}_{i^-}(k)\sum_{i\in U_{j^-}}\tilde{p}_{ij^-}(k)} - \frac{\sum_{c=1}^{C}\tilde{S}^{c}_{i^-j^-}(k)}{\overline{S}^{c^-}_{i^-}(k)}\right)\right\} \tag{B.6}$$

$$\tilde{\beta}^{c^-}_{i^-j^-}(k+1) = \tilde{\beta}^{c^-}_{i^-j^-}(k) + \Delta\tilde{\beta}^{c^-}_{i^-j^-}(k); \tag{B.7}$$

$$\overline{\beta}^{c^-}_{i^-}(k+1) = \overline{\beta}^{c^-}_{i^-}(k) - \Delta\tilde{\beta}^{c^-}_{i^-j^-}(k); \tag{B.8}$$

$$\tilde{\beta}^{c}_{ij}(k+1) = \tilde{\beta}^{c}_{ij}(k) \text{ for } \{i,j,c\} \neq \{i^-, j^-, c^-\};$$

$$\overline{\beta}^{c}_{i}(k+1) = \overline{\beta}^{c}_{i}(k) \text{ for } \{i,c\} \neq \{i^-, c^-\};$$

$$\tilde{U}_j(k+1) = \tilde{U}_j(k) \setminus \left\{i:\ \overline{\beta}^{c}_{i}(k+1) = 0,\ c = 1,\ldots,C\right\}, \quad j \in \tilde{V}(k);$$

$$\tilde{V}(k+1) = \tilde{V}(k) \setminus \left\{j:\ \tilde{U}_j(k+1) = \emptyset\right\}.$$

In (B.6), we take the minimum of the remaining unassigned split ratio portion $\overline{\beta}^{c^-}_{i^-}(k)$ and the split ratio portion needed to equalize $\mu^-(k)$ and $\mu^+(k)$. To better understand the latter, the second term in $\min\{\cdot,\cdot\}$ can be rewritten as:

$$\frac{\mu^+(k)\tilde{p}_{i^-j^-}(k)R_{j^-}}{\overline{S}^{c^-}_{i^-}(k)\sum_{i\in U_{j^-}}\tilde{p}_{ij^-}(k)} - \frac{\sum_{c=1}^{C}\tilde{S}^{c}_{i^-j^-}(k)}{\overline{S}^{c^-}_{i^-}(k)} = \left(\frac{\mu^+(k)}{\mu^-(k)} - 1\right)\left(\sum_{c=1}^{C}\tilde{S}^{c}_{i^-j^-}(k)\right)\frac{1}{\overline{S}^{c^-}_{i^-}(k)}.$$

The right hand side of the last equality can be interpreted as: flow that must be assigned for input i^-, output j^- and commodity c^- to equalize $\mu^-(k)$ and $\mu^+(k)$ minus flow that is already assigned for $\{i^-, j^-, c^-\}$, divided by the remaining unassigned portion of demand of commodity c^- coming from input link i^-.

In (B.7) and (B.8), the assigned split ratio portion is incremented and the unassigned split ratio portion is decremented by the computed $\Delta\tilde{\beta}^{c^-}_{i^-j^-}(k)$.

10. Set $k := k + 1$ and return to step 2.

Appendix C

Parameter counts for chapter 8's neural networks

In our NNs, we set the output dimension (a.k.a. number of units) of all fully-connected layers, as well as the GRU dimensions, to 128 (except for the GCN NN, which had fully-connected layers of dimension 256). The base graph encoder layer dimension F was 96. Parameter counts for all our NN variants are shown in Table C.1.

Table C.1: Parameter counts for NN configurations

From:	NN Configuration	Param. count
Table 8.1	$A = I$	385,027
	$+A_{\text{downstream}}$, $\{I, A_{\text{upstream}}\}$, $\{I, A_{\text{neighbors}}\}$	486,403
	$+A_{\text{upstream}}$	587,779
	$+A_{\text{neighbors}}$	689,155
Table 8.2	approx. $1/4$-size attn. dim. ($F = 32$)	384,003
	Flattened adjacency matrix	385,027
	One (4x-size) attention head	689,155
	GCN (all versions)	481,027