BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Dissertation

ESTIMATION AND OPTIMIZATION METHODS FOR TRANSPORTATION NETWORKS

by

SALOMÓN WOLLENSTEIN-BETECH

B.S., Instituto Tecnológico y de Estudios Superiores de Monterrey, 2015

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2022

Approved by

First Reader

Christos G. Cassandras, Ph.D.
Distinguished Professor of Engineering
Professor and Division Head of Systems Engineering
Professor of Electrical and Computer Engineering

Second Reader

Ioannis Ch. Paschalidis, Ph.D.
Distinguished Professor of Engineering
Professor of Electrical and Computer Engineering
Professor of Biomedical Engineering
Professor of Systems Engineering
Professor of Computing & Data Sciences

Third Reader

David Castañón, Ph.D.
Professor of Electrical and Computer Engineering
Professor of Systems Engineering

Fourth Reader

Mauro Salazar, Dr. Sc. ETH Zürich
Assistant Professor of Mechanical Engineering
Eindhoven University of Technology

The first principle is that you must not fool yourself — and you are the easiest person to fool. Richard P. Feynman (1998)

To Diana

Acknowledgments

I would like to express my gratitude to my two advisors: Prof. Chistos G. Cassandras and Prof. Ioannis Ch. Paschalidis, for their direction, dedication, and interest towards my development as an independent researcher and the completion of this thesis. In particular, I am indebted to the trust they placed in me by supporting all my interests, even if these sometimes deviated from my main research area. Their example as dedicated researchers, support as mentors, and passion for learning are a source of inspiration to me and one that I wish to emulate during my professional life.

Besides my advisors, I would like to thank the rest of my dissertation committee. Prof. David Castañón, whose questions and comments made this dissertation better and who inspired me deeply through his classes. Prof. Mauro Salazar, who became my first outside collaborator and with whom I enjoyed hours of conversations on transportation, research, and engineering. Finally, Prof. Michael Caramanis who served as the Chair of my Ph.D. defense and for the fun and interesting hallways conversations we had during my time at BU.

I am grateful to the staff of the Division of Systems Engineering and the Center for Information and Systems Engineering. Especially, I thank Christina, Elizabeth, Maureen, and Ruth, for helping me navigate complicated paperwork, for spotlighting my research, and for organizing CISE seminars and fun events for the community.

In these past five years in Boston I have been lucky to find unforgettable friends who have made my Ph.D. not just an intellectual experience but also a personal growth journey. From BU, I thank Andrés, Andrew, Arian, Artin, Becky, Jimmy, Nan, Shirantha, Wei, and many more with whom I have shared unique memories. Similarly, I am grateful to many students from the MIT Economics department: Adam, Garima, Jacob, Jeremy, Karthik, Marc, Martina and many more, for their friendship and their welcoming environment. Moreover, I am thankful to my Mexican friends in Boston who brought a piece of home abroad. To name a few: Aarón, Alexa, Ari, Christopher, Daniel, Julián Yehuda, José Ramón, Marcia, Mijal, Teddy, and Victoria.

My residence during the last four years has been McCormick Hall, an MIT undergraduate dorm, where I served as a Graduate Resident Advisor (GRA). There, I have met extraordinary human beings who are passionate to learn and to push themselves to the limit. Living within this *learning community* has been a pivotal source of personal growth. Thanks to all the McCormick family for the fun study breaks, music jams, karaokes, and other fun events. Special mention to house team members: Al, Ashley, Brandon, Divya, Eleanor, Emma, Jackie, Marcy, Margaret, Rachel, Ryan, Samir, Scott, Soya and Tarfah. My deepest admiration goes to the heads of house, Raúl and Flavia, who are true *Mensch* (Yiddish for a person of integrity and honor). Two people who radiate sincere, energetic and compassionate care to everyone who crosses their path. I am lucky to have met them.

I am indebted with my family in-law who has shown me how far education and hard work can take us. Special thanks to Eduardo, who has provided his unconditional help and been instrumental in my success these past few years. And to Mónica and Gil for their constant encouragement and motivation.

My most sincere gratitude goes to my family in Mexico, in particular to my Mom and Dad, Mery and Moisés, whose love and encouragement has been pivotal to my development as a human being. To my siblings, Flor and Edy, who have shown me that success is accomplished by working towards something you truly believe in. My family constantly reminds me to not take things for granted, ask many questions and never fool myself. Without this, I would have never been able to complete this dissertation.

My Ph.D. adventure and thesis would have never been possible without the tenacious support of Diana Sverdlin-Lisker, my partner, wife, counselor, copy-editor, and best friend who I love more than words can describe. While not an original song composed by me —the thing she has asked for— I dedicate this thesis to her.

To all of you ...
¡Gracias ... Totales!

ESTIMATION AND OPTIMIZATION METHODS FOR TRANSPORTATION NETWORKS

SALOMÓN WOLLENSTEIN-BETECH

Boston University, College of Engineering, 2022

Major Professors: Christos G. Cassandras, Ph.D.
Distinguished Professor of Engineering
Professor and Division Head of Systems Engineering
Professor of Electrical and Computer Engineering

Ioannis Ch. Paschalidis, Ph.D.
Distinguished Professor of Engineering
Professor of Electrical and Computer Engineering
Professor of Biomedical Engineering
Professor of Systems Engineering
Professor of Computing & Data Sciences

ABSTRACT

While the traditional approach to ease traffic congestion has focused on building infrastructure, the recent emergence of Connected and Automated Vehicles (CAVs) and urban mobility services (e.g., Autonomous Mobility-on-Demand (AMoD) systems) has opened a new set of alternatives for reducing travel times. This thesis seeks to exploit these advances to improve the operation and efficiency of Intelligent Transportation Systems using a network optimization perspective. It proposes novel methods to evaluate the prospective benefits of adopting socially optimal routing schemes, intermodal mobility, and contraflow lane reversals in transportation networks.

This dissertation makes methodological and empirical contributions to the transportation domain. From a methodological standpoint, it devises a fast solver for the

Traffic Assignment Problem with Side Constraints which supports arbitrary linear constraints on the flows. Instead of using standard column-generation methods, it introduces affine approximations of the travel latency function to reformulate the problem as a quadratic (or linear) programming problem. This framework is applied to two problems related to urban planning and mobility policy: social routing with rebalancing in intermodal mobility systems and planning lane reversals in transportation networks. Moreover, it proposes a novel method to jointly estimate the Origin-Destination demand and travel latency functions of the Traffic Assignment Problem. Finally, it develops a model to jointly optimize the pricing, rebalancing and fleet sizing decisions of a Mobility-on-Demand service. Empirically, it validates all the methods by testing them with real transportation topologies and real traffic data from Eastern Massachusetts and New York City showing the achievable benefits obtained when compared to benchmarks.

Contents

List of Tables

List of Figures

List of Abbreviations

ADMM	....	Alternating Direction Method of Multipliers
AMoD	....	Autonomous Mobility-on-Demand
BPR	....	Bureau of Public Roads
CARS	....	Congestion-Aware Routing Scheme
CAV	....	Connected and Automated Vehicle
CD	....	Charge Depleting
CS	....	Charge Sustaining
CTPS	....	Central Transportation Planning Staff
EMA	....	Eastern Massachusetts Area
EV	....	Electric Vehicle
FW	....	Frank-Wolfe
GD	....	Gradient Descent
HWFET	....	Highway Fuel Economy Test
I-AMOD	....	Intermodal Autonomous Mobility-on-Demand
ILP	....	Integer Linear Program
IP	....	Integer Program
ITS	....	Intelligent Transportation Systems
KKT	....	Karush-Kuhn-Tucker
LA	....	Lane Assignment
LASO-TAP	....	Lane Assignment System-Optimal Traffic Assignment Problem
LP	....	Linear Program
MILP	....	Mixed Integer Linear Program
MIQP	....	Mixed Integer Quadratic Program
MoD	....	Mobility-on-Demand
MPO	....	Metropolitan Planning Organization
MSA	....	Method of Successive Averages
NLP	....	Non-Linear Program
NYC	....	New York City
OD	....	Origin Destination
PHEV	....	Plug-In Hybrid Electric Vehicle
PoA	....	Price of Anarchy
PSD	....	Positive Semidefinite

QCQP	....	Quadratically Constrained Quadratic Program
QP	....	Quadratic Program
RG	....	Relative Gap
RI	....	Relative Improvement
RKHS	....	Reproducing Kernel Hilbert Space
RL	....	Reinforcement Learning
SO	....	System-Optimal
SUE	....	Stochastic User Equilibrium
TAP	....	Traffic Assignment Problem
TAPAS	....	Traffic Assignment by Paired Alternative Segments
TSTT	....	Total System Travel Time
UC	....	User-Centric
UDDS	....	Urban Dynamometer Driving Schedule
U.S.	....	United States
V2I	....	Vehicle-to-Infrastructure communication
VI	....	Variational Inequality

Chapter 1

Introduction

High levels of urbanization and rapid technological developments have transformed human mobility. Only a century ago, people had to rely on carriages for the transportation of goods and only the wealthiest could afford traveling for long distances. Today, the world looks vastly different: more than 91% of households in the U.S. own a car (U.S. Census Bureau, 2019), and public transportation is available in almost every urban settlement in the world.

The rise in convenience of movement has also brought traffic congestion to our cities and thus the imperative need for improving efficiency, reducing greenhouse gas emissions, and reducing travel times. While many advances have been made in this direction, like the creation of new modes of urban mobility services (e.g., Mobility-on-Demand (MoD) systems like Uber or Lyft; and bike sharing systems); the rapid evolution of autonomous systems; and the emergence of Connected and Automated Vehicles (CAVs), there are still open questions related to the efficiency of *Intelligent Transportation Systems* (ITS). For example, how vehicle-to-infrastructure communication (V2I) in intersections or lane reversals can be used to reduce travel times while keeping the physical infrastructure fixed.

This thesis seeks to exploit many of the aftermentioned advances to improve the operation and efficiency of ITS using a network optimization perspective. It proposes novel methods to evaluate the prospective benefits of adopting socially optimal routing schemes, intermodal mobility, and contraflow lane reversals in transportation networks.

This work aims to inform transportation agencies and Mobility-on-Demand platforms on how to reduce congestion and improve the overall efficiency of the system.

This dissertation makes methodological and empirical contributions to the transportation domain. From a methodological standpoint, it devises a fast solver for the Traffic Assignment Problem with Side Constraints which supports arbitrary linear constraints on the flows. Instead of using standard column-generation methods, it introduces affine approximations of the travel latency function to reformulate the problem as a quadratic (or linear) programming problem. This framework is applied to two problems related to urban planning and mobility policy: social routing and rebalancing in intermodal mobility systems and planning lane reversals in transportation networks. Moreover, it proposes a novel method to *jointly* estimate the Origin Destination demand and travel latency functions of the Traffic Assignment Problem. To achieve that, it uses a *kernel-based* method that solves the joint estimation problem by using the ideas of *inverse optimization* for *variational inequalities* formulations. Finally, it develops a model to *jointly* optimize the pricing, rebalancing and fleet sizing decisions of a Mobility-on-Demand service. This method uses the passengers' destinations when setting prices, allowing the service to be more cost-efficient than when only considering origins (current state-of-the-art). Empirically, we test the described methods with real transportation topologies and real traffic data of Eastern Massachusetts and New York City to show the achievable benefits obtained when compared to benchmarks.

The workhorse model that is used throughout this thesis is the Traffic Assignment Problem (TAP) which quantifies the effect of a new transportation policy or infrastructure project on the overall travel time (or energy consumption). The TAP distributes traffic flows to specific links of the network. In other words, it assigns routes to commuters, which determines the travel times of each traveler.

Finally, this thesis hopes to inspire future researchers to work in transportation modeling. The reason for that is that many processes involve moving an entity from an origin to a destination. Examples include: water distribution networks, electric power networks, impulse signals traversing our neural networks, and packages delivered from factories to retailers. Thus, the analysis, understanding, and improvement of transportation networks is of interest for countless applications and some of the methods developed herein, focused on human mobility, may be suitable for other applications that have a network structure.

1.1 Transportation Networks

In the fields of optimization and control, *networks* are mathematical objects that help the practitioner represent a large and complicated system into a simple and compact framework. Networks are a key component for studying large-scale transportation models and they have been used in many applications in the transportation planning process. For example, to identify bottlenecks, analyze new modes of transportation, develop congestion pricing schemes, among others.

A network is formally represented by a set of *links* and a set of *nodes*. In transportation networks, it is common to use *links* to represent a form of traveling from one point to another (e.g., a road segment, a sidewalk, a bike lane). The *nodes* are the start and end points of the links and can be connected to more than one link.

An exciting aspect of the analysis of transportation systems is that traffic congestion and travelers commuting decisions are dependent on each other. Commuters use strategies or services (e.g., Google Maps, Waze) to avoid congested roads while congestion increases as more users are present in a link. These two interacting variables, travel choices and system congestion, are thus endogenous. Figure 1·1 depicts this dependency. Hence, the need for mathematical representations to address this endogeneity is fundamental for the understanding of the transportation networks.

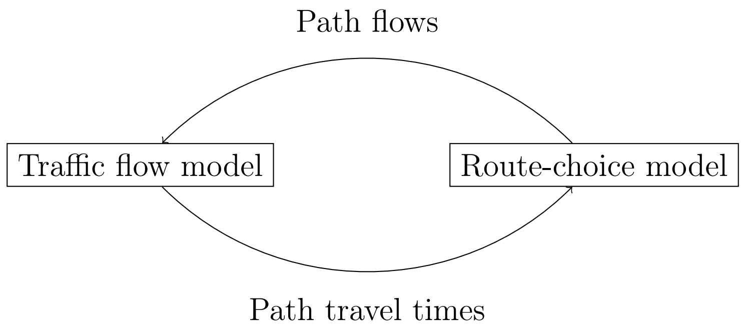


Figure 1·1: Interacting components in a Traffic Assignment. Adapted from Boyles et al. (2021)

1.2 Traffic Assignment

The *traffic assignment* is a procedure that aims to allocate the transportation demand of users' trips to specific transportation resources (links, modes of transportation). The input of the assignment is composed of three main components: (i) the transportation network defining the topology and modes of transportation; (ii) a complete description of the traveling patterns in the network, namely the *Origin-Destination* (OD) demand matrix; (iii) a *travel latency* function mapping traffic flows to travel times. The output of such model are the traffic flows in each link of the transportation network.

The purpose of determining a traffic assignment as part of the transportation planning process is to assess the effects of a new intervention or policy, as well as to capture the deficiencies of the transportation network. This is achieved since the assignment provides an estimate of future trips, traffic volumes, congestion, and average user travel times.

The *Traffic Assignment Problem* (TAP) is simply the solution of the traffic assignment given a particular transportation network and a set of *link latency functions*. These functions map the flow on each link to the corresponding travel time it takes to cross the link. We denote them by $t_{ij}(x_{ij})$ where ij indicate the link (i, j) connecting nodes i and j and x_{ij} is the flow present on link (i, j). Functions $t_{ij}(\cdot)$ are typically assumed to be increasing, non-negative and convex. This modeling assumption implies that as more vehicles are present on a link, travel times of crossing the link increase.

Many travel latency functions have been established in the literature and there is not a single perfect function (Branston, 1976), however, the most popular is the one introduced by Beckmann, McGuire, and Winsten (1955) and widely known as the *Bureau of Public Roads* (BPR) function. This function takes the form

$$t_{ij}(x_{ij}) = t_{ij}^0\Big(1 + \alpha\Big(\frac{x_{ij}}{m_{ij}}\Big)^{\beta}\Big), \tag{1.1}$$

where α and β are shape parameters, and t_{ij}^0 and m_{ij} are the free-flow travel time and the capacity of link (i, j), respectively. Figure 1·2 shows the travel latency function with $\alpha = 0.15$ and $\beta = 4$, parameters commonly used in the literature and by practitioners.

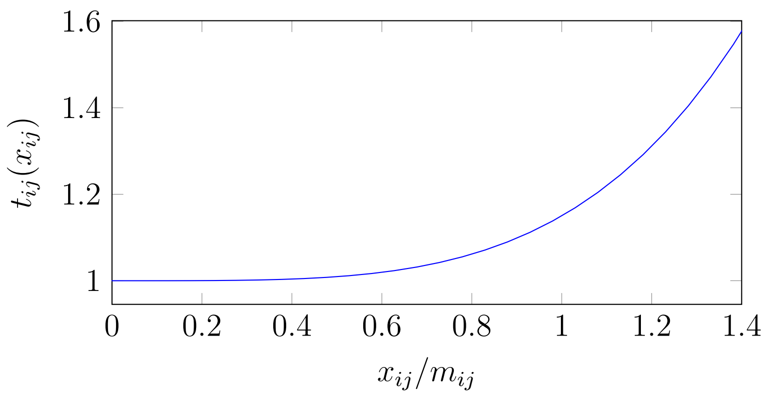


Figure 1·2: Typical travel latency function

By using these functions, the *static user equilibrium* traffic assignment is constructed by assuming that all paths between the same origin and destination have minimal and equal travel times. It is called an *equilibrium* point since if this were not true, users would switch from slower routes to faster ones, until travel times are equalized across travelers. It is known that this equilibrium state is not socially optimal, and that other configurations of traffic flow can reduce the overall travel time compared to the user equilibrium.

The main advantage of using this *static* traffic assignment and link latency functions like (1.1) is that the user equilibrium (and system optimum) states can be computed efficiently even for large-scale networks. Mostly for this reason, this modelling framework has been adopted by the transportation community for decades and remains a fundamental model for performing analysis of the network conditions.

This dissertation utilizes the static traffic assignment framework and identifies as a future research direction the development of the methods herein to the *dynamic* traffic assignment setting. For an extensive discussion on the advantages and drawbacks of each model, see Chapter 1 of Boyles et al. (2021).

1.3 Thesis Contributions and Overview

This section provides a brief motivation and the key contributions of each of the chapters in this thesis. Specific related work to each of the topics is covered in the body of each chapter. The problems are classified into two main sections defined by who is the stakeholder for the method. First, it focuses on *transportation planning agencies* and *policy-makers*, then, it moves to problems pertaining to *Autonomous Mobility-on-Demand* (AMoD) platforms.

1.3.1 Transportation Agencies

One of the main tasks of transportation agencies is providing transportation infrastructure which is safe, reliable, robust and resilient. To achieve that, it is fundamental to have models that can output accurate and useful predictions informing on the effect of incorporating a certain policy.

Three problems of interest to transportation agencies are developed in this thesis: (i) the estimation of the parameters of the TAP; (ii) the assessment of new routing policies for Connected and Automated Vehicles (CAVs); (iii) the optimization of existing infrastructure by better allocating the direction of lanes in the network.

Estimating Travel Patterns and User Behavior from Data

An important drawback that has been observed when using the TAP is that small perturbations in the inputs to the problem, namely the OD demands and travel time functions have a large impact on the equilibrium solution (Bertsimas, Gupta, & Paschalidis, 2015; Yang, Meng, & Bell, 2001). Therefore, the problem of accurately estimating these inputs is relevant to design interventions with reliable models.

Chapter 2 contributes to this topic as follows. First, unlike most of the available methods, this work tackles the estimation problem *jointly*, i.e., it calibrates the OD demands and travel latency function simultaneously by formulating a bilevel program. To solve the bilevel program, two methods are proposed and tested. Second, it extends the methods from single- to multi-class vehicle networks. This allows estimating the demand patterns and travel latency functions for different vehicle types such as self-driving vehicles, trucks, or bikes. Third, it performs two case studies using data from the transportation networks of Eastern Massachusetts Area (EMA) and New York City (NYC). The empirical results suggest that the joint methods converge to a solution that yields better estimates than solving the two estimation problems separately.

The Effect of System-Optimal Routing in Mixed Traffic

In a selfish attempt to beat traffic, every agent optimizes their own route by minimizing their travel time and leading to a *user-centric* solution known as the *Wardrop equilibrium* (Wardrop, 1952). Alternatively, if a social planner were to choose the routes for every commuter, the solution would lead to a *system (or social) optimal* allocation. The inefficiency of the user-centric approach boils down to the difference between a selfish versus a collaborative routing strategy.

A commonly used metric to measure this difference is the *Price of Anarchy* (PoA). The analysis of this phenomena is relevant for transportation planners when designing incentives schemes (e.g., congestion tolling) in the network. Moreover, the assessment of the magnitude of the PoA is important to understand the value that can be gained by collaboratively routing commuters.

Chapter 3 contributions are the following. First, using the estimated OD demand and travel latency functions, the PoA for different time slots in the Eastern Massachusetts network is estimated. It is shown that for the morning peak traffic the PoA can reach values of 1.10, suggesting that coordinating commuters routing decisions could reduce traffic congestion by 10%. Second, It proposes an algorithm which comes up with system-centric time-optimal and energy-optimal routes for collaborative CAVs in the presence of mixed traffic (having both CAVs and user-centric vehicles in the system). Using this method, the results show that even under small CAV penetration rates, CAVs and private vehicles benefit from the improvements. This work motivates and serves as a basis for future work on designing incentive schemes to steer user-centric behavior towards a system-optimal allocation.

Optimizing Lane Reversals

One possible way to reduce traffic congestion without building new roads is to increase the network's capacity by dynamically adjusting the direction of the lanes of the transportation infrastructure. The problem of identifying the best lanes in a network to reverse has been shown to be NP-hard given the dependence of the users' route selection on the lane direction decision. Solving this problem is relevant for transportation agencies as it helps identify which are the best links of the network to reverse. Moreover, it provides an estimate of the achievable benefits that this technology may provide.

Chapter 4 studies how to solve this problem efficiently by devising novel methodologies to solve the problem. First, it develops three tractable methods to solve the lane reversal problem while considering the routing decisions of commuters. The first method decouples the routing and lane assignment problems and solves them sequentially. The second approach uses a Frank-Wolfe algorithm which takes gradient steps for both the link's capacity and user routing. The last method convexifies the objective function to map the problem to a linear program. The second contribution lies on extending the convex approximation approach by incorporating any set of linear constraints, and provides examples for which this flexibility on the constraints is relevant for practical applications. Finally, case studies report overall reductions in travel times of 4.5% for the overall Eastern Massachusetts network with some roads improving up to 40% showing the achievable benefits of lane reversal for daily commuter traffic.

1.3.2 Autonomous Mobility-on-Demand Systems

In the last few years, the concept of a *sharing economy* has become ubiquitous across industries including transportation. A key tenet of a sharing economy is that consumer's cost for using a resource will shift from ownership to on-demand access based on need. In transportation, the sharing economy has had a large expansion in the so-called Mobility-on-Demand (MoD) platforms for which many (especially young) citizens have shifted to exclusively using MoD services and public transportation (Etehad & Nikolewski, 2016).

These MoD systems, like Uber or Lyft in the United States, use geolocalization and information services to improve the overall user experience. Although their connectivity has allowed them to offer a lower cost service by employing *ride-sharing* services (in which two or more passengers share a vehicle), there is an extensive area of opportunity to improve the overall operation of these systems by exploiting automation, connectivity, and coordination.

The second half of this dissertation studies optimal strategies to operate an MoD system that uses autonomous vehicles, creating Autonomous Mobility-on-Demand (AMoD) systems. Specifically, the goal is to tackle four operational decisions. (i) How many vehicles does the platform require to offer a desirable quality of service? (ii) How should the system set prices in order to maximize profits? (iii) How should vehicles be reallocated across the network in order to reduce customer wait times? and (iv) How to optimally route vehicles?

It is crucial to note that these decisions are inter-related; for example, if we increase the price for a specific origin-destination, we expect its demand to decrease, therefore, requiring less reallocation of vehicles to that region and presumably a smaller fleet. Therefore, we focus on answering these operational decisions *jointly*.

Optimizing Routing and Rebalancing of Autonomous Mobility-on-Demand systems

Chapter 5 studies how to improve the quality of service of an AMoD system by reducing the overall user travel time and by ensuring the availability of vehicles for their customers. To achieve this, the objective is to select the routing and rebalancing decisions *jointly* rather than separately (the current state-of-the-art).

The key methodological contribution is using the traffic assignment method and approximating the non-linear travel latency function with a piecewise affine function. This slight modification allows to render the non-linear program with rebalancing constraints (hard to solve) to a quadratic or a linear program, making it easier and faster to solve. We extend the two-line piecewise affine approximation presented in Salazar, Tsao, Aguiar, Schiffer, and Pavone (2019) to n segments and provide theoretical and empirical results showing that its solution is asymptotically optimal to the original problem. In addition to this contribution, we extend the routing and rebalancing of a fleet of AMoDs to consider reactive private flow[1] and intermodality when deciding the paths of the AMoD users. With this framework at hand, we analyze the trade-offs between the benefits of system-centric routing and the cost of rebalancing. Finally, we perform experiments using the transportation network of New York City.

Optimizing Pricing, Fleet Sizing, and Rebalancing of AMoD systems

Pricing policies play an important role in AMoD systems as they modulate the inflow of customers traveling between regions in the network. In contrast to the existing pricing methods in the literature, in Chapter 6 we use information on a customers' *destination* when designing the pricing policy. This allows the fleet manager to modulate demand such that the system is at an *equilibrium*[2] by solely adjusting prices. In addition to pricing, we incorporate the rebalancing policy optimization framework presented by Pavone, Smith, Frazzoli, and Rus (2012) and formulate a *joint static* optimization model.

Using these static models as a starting point, we propose *dynamic* policies that are more responsive to perturbations in the system (such as unexpected increases in demand). We build a simulation environment that takes into account the stochasticity of customer arrivals and travel times to test and compare the methodologies developed in the chapter. We use this tool to perform case studies that use traffic flow and taxi data from Eastern Massachusetts, New York City, and Chicago. Our results show that solving the problem jointly could increase profits between 1% and up to 50%, depending on the benchmark. Moreover, we observe that the proposed fleet size yields a 75% utilization rate of vehicles (i.e., the vehicle is idle 25% of the time) as compared

[1]in a network where AMoD and user-centric private vehicles interact, we provide routing decisions for the AMoD users that anticipate the behavior of the private vehicles.

[2]Infinite queues of customers or vehicles are not formed in the system.

to 5% for private vehicles.

Finally, Chapter 7 concludes and identifies future research directions.

1.4 Notational Conventions

In this thesis, all vectors are column vectors and are denoted by bold lowercase letters. Bold uppercase letters denote matrices. We write $\mathbf{x} = (x_1, \ldots, x_{\dim(\mathbf{x})})$ to denote the column vector $\mathbf{x}$, where $\dim(\mathbf{x})$ is its dimensionality. We use "prime" to denote the transpose of a matrix or vector. We let $\mathbf{0}$ and $\mathbf{I}$ be the vector of all zeroes and the identity matrix, respectively. Unless otherwise specified, $\|\cdot\|$ denotes the Euclidean norm, $|\mathcal{D}|$ denote the cardinality of a set $\mathcal{D}$, and $[\![\mathcal{D}]\!]$ the set $\{1, \ldots, |\mathcal{D}|\}$.

Chapter 2

Estimating Travel Patterns and User Behavior from Data

This chapter presents a kernel-based framework that jointly estimates the OD demand matrix and travel latency function in single and multi-class vehicle networks. To achieve that, we formulate a bilevel optimization problem and then we transform it to a Quadratic Constraint Quadratic Program (QCQP). To solve this QCQP, we propose a *trust-region feasible direction* algorithm that sequentially solves a quadratic program. In addition to the QCQP mehtod, we provide a more efficient alternating optimization method. Our results show that the QCQP method achieves better estimates when compared with the disjoint and sequential methods. We show the applicability of the method by performing case studies using data for the transportation networks of Eastern Massachusetts and New York City.

2.1 The Problem and Related Work

The inputs to the Traffic Assignment Problem (TAP) besides the network topology are: (*i*) the *Origin Destination* (OD) demand matrix; and (*ii*) a link *latency cost* or *travel time* function and are usually unknown. Hence, accurately estimating these quantities if of interest to traffic engineers and policy-makers.

Assuming access to traffic counts data, i.e., data reporting the number of vehicles in every link of the network, the estimation problem of the inputs of the TAP is

$$\min_{\boldsymbol{\beta},\mathbf{g}} \; F_1(\mathbf{x}(\boldsymbol{\beta},\mathbf{g}),\mathbf{x}^d),$$

where F_1 measures a non-negative "distance" between the TAP flows $\mathbf{x}(\mathbf{g},\boldsymbol{\beta})$ and the flow data $\mathbf{x}^d$; $\mathbf{g}$ the OD demand; $\boldsymbol{\beta}$ represents the parameters specifying a travel latency function and $\mathbf{x}(\mathbf{g},\boldsymbol{\beta})$ is the solution to the TAP.

2.1.1 Related Work

The problem of estimating $\mathbf{g}$ alone (for a fixed $\boldsymbol{\beta}$) has received extensive attention in the literature. In practice, urban planners estimate OD demand patterns through surveys. This task is expensive and time consuming, which makes it impractical to perform on a regular basis. In contrast, academics have estimated $\mathbf{g}$ assuming access to flow data and using a variety of methods.

An appropriate way to classify the literature on OD demand estimation is by considering the level of congestion in the network. For uncongested networks, where path enumeration connecting each OD pair is possible and flows do not impact the routing decisions of users, entropy maximization (Bell, 1983; Van Zuylen & Willumsen, 1980), multi-objective optimization (Brenninger-Göthe, Jörnsten, & Lundgren, 1989), generalized least squares (Cascetta, 1984; Hazelton, 2000), maximum likelihood estimation (Spiess, 1987), and Bayesian inference (Maher, 1983) have been employed. Cascetta and Nguyen (1988) provide a comprehensive comparison between these approaches. Alternatively, for congested networks, where there exists a circular dependence between the OD estimation problem and the traffic assignment problem, the problem has been posed as a bilevel problem. Fisk (1988) and Fisk (1989) where the first works to use this bilevel formulation and they tackled it by sequentially solving an entropy maximization and a user assignment problem. Subsequently, Spiess

(1990) proposed a gradient-descent heuristic algorithm that calculates derivatives by assuming that routing decisions are locally constant for small perturbations of $\mathbf{g}$ and solved the problem sequentially. Since then, many approaches have been proposed for the congested case, including coordinate descent (Florian & Chen, 1995), fixed point algorithms with generalized least squares (Cascetta & Postorino, 2001), and extensions to a stochastic user equilibrium setting (Maher, Zhang, & Van Vliet, 2001; Yang, Sasaki, Iida, & Asakura, 1992) that incorporates uncertainty in the routing decision process. In addition to these, Doblas and Benitez (2005) solves the problem by writing the augmented Largrangian and solving it using a Frank-Wolfe algorithm. A main aspect of all these approaches is that they assume full knowledge of $\boldsymbol{\beta}$.

Estimating *travel time functions* has received less attention in the literature. In general, this problem has been addressed by fitting data of a single link using regression (Mtoi & Moses, 2014; Skabardonis & Dowling, 1997), simulation (Lu, Meng, & Gomes, 2016), and more recently by incorporating stochasticity in the link's capacity (Neuhold & Fellendorf, 2014). In contrast to single-link methods, there has been some work trying to solve the *inverse* TAP which assumes knowledge of $\mathbf{x}$ and $\mathbf{g}$ and aims to recover $\boldsymbol{\beta}$. In this context, García-Ródenas and Verastegui-Rayo (2013) estimates a single parameter β and uses a column-generation alternating approach. Differently, Bertsimas et al. (2015) employs a kernel method to estimate the travel latency function allowing more flexibility in the specification function.

To the best of our knowledge, solving the joint (OD demand and travel time functions) estimation problem has only been studied in Russo and Vitetta (2011) where the authors considered the parametric BPR-type function and a heuristic algorithm to solve the joint problem. Different to Russo and Vitetta (2011), in this thesis the problem formulation allows more flexibility in the travel latency function by allowing it to be any monotonically increasing polynomial. Similar to this joint estimation problem, several studies have estimated the OD demand together with the route-choice *dispersion parameter* θ encountered in the stochastic user equilibrium (SUE). Liu and Fricker (1996) used a two-stage method and showed its convergence to a local minimum for uncongested networks. For the congested case, Yang et al. (2001) proposed a sequential quadratic procedure that requires estimating flow derivatives with respect to θ and $\mathbf{g}$. To compute the derivatives they used Tobin and Friesz (1988) and showed that their algorithm converges only under certain circumstances. Moreover, Lo and Chan (2003) and Wang et al. (2016) solved the same problem using an alternating method, and Meng, Lee, and Cheu (2004) tackled this problem using an augmented Lagrangian method.

2.2 Preliminaries

2.2.1 Single-Class Transportation Network Model

Consider a transportation network as a directed graph $G = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of nodes (road's intersections) and $\mathcal{A}$ is the set of links (roads). Let the node-link incidence matrix be denoted by $\mathbf{N} \in \{0, 1, -1\}^{|\mathcal{V}|\times|\mathcal{A}|}$ and $\mathbf{e}_a \in \{0, 1\}^{|\mathcal{A}|}$ be a zero vector with an entry equal to 1 corresponding to link a.

Let $\mathbf{w} = (w_s, w_t)$ denote an OD pair and $\mathcal{W} = \{\mathbf{w}_i : \mathbf{w}_i = (w_{si}, w_{ti}),\ i \in [\![\mathcal{W}]\!]\}$ be the set of all OD pairs. Furthermore, let $d^{\mathbf{w_i}} \geq 0$ be the demand flow that travels from origin w_{si} to destination w_{ti}. Denote by $\mathbf{d}^{\mathbf{w}} \in \mathbb{R}^{|\mathcal{V}|}$ the vector of all zeros except for the coordinates of nodes w_{si} and w_{ti} which take values $-d^{\mathbf{w}_i}$ and $d^{\mathbf{w}_i}$, respectively. We will also use vector $\mathbf{g} = (d^{\mathbf{w}_i}; i \in [\![\mathcal{W}]\!])$ to denote the demand flows for all OD pairs.

Let x_a be the total link flow of link $a \in \mathcal{A}$ and $\mathbf{x}$ the vector of these flows. Let $\mathcal{F}$ be the set of feasible flow vectors resulting from transporting the OD demand through the network, defined as

$$\mathcal{F} = \left\{ \mathbf{x} \in \mathbb{R}_+^{|\mathcal{A}|} : \mathbf{x} = \sum_{\mathbf{w}_i \in \mathcal{W}} \mathbf{x}^{\mathbf{w}_i},\ \mathbf{N}\mathbf{x}^{\mathbf{w}_i} = \mathbf{d}^{\mathbf{w}_i},\ \forall i \in [\![\mathcal{W}]\!] \right\}, \tag{2.1}$$

where $\mathbf{x}^{\mathbf{w}_i}$ is the flow vector attributed to OD pair $\mathbf{w}_i$.

For each OD pair $\mathbf{w}_i$ we define a set of possible routes $\mathcal{R}^{\mathbf{w}_i}$ where each route $r \in \mathcal{R}^{\mathbf{w}_i}$ is a sequence of links starting from the origin w_{si} and ending at the destination w_{ti}. We will write $a \in r$ if a route r contains link a. For each OD pair $\mathbf{w}_i \in \mathcal{W}$, we define the indicator function

$$\delta_r^{ai} = \begin{cases} 1, & \text{if } r \in \mathcal{R}^{\mathbf{w}_i} \text{ uses link } a, \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

Finally, let $t_a(\mathbf{x}) : \mathbb{R}_+^{|\mathcal{A}|} \mapsto \mathbb{R}_+$ be the *latency cost* (i.e., travel time) function for link a and write $\mathbf{t}(\cdot)$ for the vector of these link functions. Using the same structure used in Beckmann et al. (1955) we can characterize $t_a(x_a)$ as

$$t_a(x_a) = t_a^0 f(x_a/m_a),$$

where m_a is the flow capacity of link a, $f(\cdot)$ is a strictly increasing, positive, and continuously differentiable function, and t_a^0 is the free-flow travel time on link a. We set $f(0) = 1$, which ensures that if there is no constraint on flow capacity, the travel time t_a is equal to the free-flow travel time.

2.2.2 Multi-class Transportation Network Model

Let $\tilde{\mathcal{U}}$ be the set of *user* (or *vehicle*) classes and, without loss of generality, assume that all vehicle classes travel in the same transportation network. Let the *original* network be denoted with a directed graph $\tilde{\mathcal{G}} = \left(\tilde{\mathcal{V}}, \tilde{\mathcal{A}}, \tilde{\mathcal{W}}\right)$ where $\tilde{\mathcal{V}}$ is the set of nodes, $\tilde{\mathcal{A}}$ is the set of arcs, and $\tilde{\mathcal{W}}$ is the set of K OD pairs defined as $\tilde{\mathcal{W}} =$

$\big\{\mathbf{w}_k : \mathbf{w}_k := (w_{sk}, w_{tk}),\ k = 1, \ldots, K\big\}$.

In order to include multiple vehicle classes, we use the idea in Dafermos (1972) of making $|\tilde{\mathcal{U}}|$ copies of $\tilde{\mathcal{G}}$, each corresponding to a vehicle class. We use these graphs to create an *enlarged* network, denoted with $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \mathcal{W})$, where the sets of arcs, nodes, and OD pairs are constructed as the collection of all $|\tilde{\mathcal{U}}|$ graphs. Formally:

$$\mathcal{V} = \big\{v(i,u) : i \in [\![\tilde{\mathcal{V}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big\}, \tag{2.3a}$$

$$\mathcal{A} = \big\{a(i,u) : i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big\}, \tag{2.3b}$$

$$\mathcal{W} = \big\{\mathbf{w}(k,u) : \mathbf{w}(k,u) := (w_s(k,u), w_t(k,u)),\ k \in [\![\tilde{\mathcal{W}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big\}. \tag{2.3c}$$

Let the node-link incidence matrix of $\mathcal{G}$ be $\mathbf{N} \in \{0, 1, -1\}^{|\mathcal{V}| \times |\mathcal{A}|}$ and let $\mathbf{e}_{iu}$ denote the $|\mathcal{A}|$-dimensional vector where all entries are zero except the entry corresponding to link $a(i,u)$ which is set to one. To account for users' demand, for every OD $k \in [\![\tilde{\mathcal{W}}]\!]$ and class $u \in [\![\tilde{\mathcal{U}}]\!]$ let $d^{\mathbf{w}(k,u)}$ be the demand rate from node $w_s(k,u)$ to node $w_t(k,u)$. Moreover, let the vector $\mathbf{d}^{\mathbf{w}(k,u)} \in \mathbb{R}^{|\mathcal{V}|}$ be composed of all zeros except for the coordinates of nodes $w_s(k,u)$ and $w_t(k,u)$ which take values $-d^{\mathbf{w}(k,u)}$ and $d^{\mathbf{w}(k,u)}$, respectively. Let $\mathcal{F}$ be the set of feasible flow vectors defined as

$$\mathcal{F} = \Big\{\mathbf{x} \in \mathbb{R}_{\geq 0}^{|\tilde{\mathcal{A}}| \times |\tilde{\mathcal{U}}|} : \mathbf{x}_u = \sum_{k \in [\![\tilde{\mathcal{W}}]\!]} \mathbf{x}^{\mathbf{w}(k,u)},\ \mathbf{N}\mathbf{x}^{\mathbf{w}} = \mathbf{d}^{\mathbf{w}},\ \forall \mathbf{w} \in \mathcal{W}\Big\},$$

where $\mathbf{x} = \big(x_{iu};\, i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big)$, and x_{iu} denotes the flow of class u on link $a(i,u)$. Let $\mathbf{x}_u = \big(x_{iu};\, i \in [\![\tilde{\mathcal{A}}]\!]\big)$ be the flow vector for class u and $\mathbf{x}_i = \big(x_{iu};\, u \in [\![\tilde{\mathcal{U}}]\!]\big)$ the flow vector of all classes corresponding to the ith *physical* arc.

BPR-type Cost Functions

We reflect the total travel time of a link by coupling the flow from different classes of vehicles using a travel latency function. To do so, we characterize the set of travel latency functions as:

$$\mathbf{t}(\mathbf{x}) = \big(t_{iu}(\mathbf{x}_i);\ i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big), \tag{2.4}$$

where the cost on a *physical* link does not depend on the flows elsewhere, but only on the flows present in the link from all classes. To simplify the analysis, we select travel latency functions with the form of a *generalized* Bureau of Public Roads (BPR) form. This is, for each link $i \in [\![\tilde{\mathcal{A}}]\!]$ and user type $u \in [\![\tilde{\mathcal{U}}]\!]$ we have

$$t_{iu}(\mathbf{x}_i) = t^0_{iu} f\Big(\frac{\boldsymbol{\theta}'\mathbf{x}_i}{m_i}\Big) \tag{2.5}$$

where t^0_{iu} is the *free-flow travel time* for vehicle class u on link i; $f(\cdot)$ is a travel latency function which satisfies $f(0) = 1$, is strictly increasing and is continuously differentiable on $\mathbb{R}_+$; m_i is the *flow capacity* of link i, and $\boldsymbol{\theta} = \big(\theta_u;\ u \in [\![\tilde{\mathcal{U}}]\!]\big)$ is a weight vector such that $\theta_u \geq 1$, $\forall u \in [\![\tilde{\mathcal{U}}]\!]$. As a special case, the single-class network corresponds to $|\tilde{\mathcal{U}}| = 1$ and $\theta_1 = 1$. Finally, we make the following standard assumption in the context of the TAP for our analysis of the models.

Assumption A. *(i) $\mathcal{G}$ is strongly connected (there is at least one path connecting any origin with its destination). (ii) Demands $\mathbf{d}^{\mathbf{w}}$ are non-negative. (iii) Travel time functions are positive and continuous for every link.*

It has been shown that the single-class formulation is a special case of the multi-class model, and more interestingly, that we can treat the multi-class model as an enlarged single-class model (Patriksson, 1994). Thus, we only need to consider general multi-class models. However, due to coupling of flows in the latency function some of the properties of the single-class might differ to the multi-class case. These are further discussed in the following subsection and in Remark 1. For convenience, we vectorize the demand for vehicle class u with $\mathbf{g}_u = \big(d^{\mathbf{w}(k,u)};\ \in [\![\tilde{\mathcal{W}}]\!]\big)$ and denote with $\mathbf{g} = \big(\mathbf{g}_u; u \in [\![\tilde{\mathcal{U}}]\!]\big)$ the full demand vector.

2.3 Models

In this section we present the models employed to construct the joint problem. We review the variational inequality formulation of the TAP, as well as an *inverse* TAP model. We begin by defining the notion of a Wardrop Equilibrium.

Definition 1. *A feasible flow* $\mathbf{x}^* \in \mathcal{F}$ *is a* Wardrop Equilibrium *if for every OD pair* $\mathbf{w} \in \mathcal{W}$*, and any route* r_{iu} *connecting* $(w_s(i,u), w_t(i,u))$ *with positive flow* $h_{r_{iu}}$*, the cost of traveling along that route is no greater than the cost of traveling along any other route.*

For an OD pair $\mathbf{w}(k,u)$ let r_{ku} be a path connecting its origin to its destination and let $\mathcal{R}^{\mathbf{w}(k,u)}$ be the set of all paths. Furthermore, let $h_{r_{ku}}$ be the flow assigned to path r_{ku}. Then a Wardrop Equilibrium exists if

$$h_{r_{ku}} > 0 \implies c_{r_{ku}} = \pi_{\mathbf{w}(k,u)}, \quad \forall r_{ku} \in \mathcal{R}^{\mathbf{w}(k,u)} \tag{2.6a}$$

$$h_{r_{ku}} = 0 \implies c_{r_{ku}} \geq \pi_{\mathbf{w}(k,u)}, \quad \forall r_{ku} \in \mathcal{R}^{\mathbf{w}(k,u)} \tag{2.6b}$$

where $c_{r_{ku}}$ is the travel time in route r_{ku} and $\pi_{\mathbf{w}(k,u)}$ is the travel time on the fastest route of $\mathbf{w}(k,u)$.

2.3.1 Variational Inequality

One way of modeling and solving the TAP is by using a *Variational Inequality* (VI) formulation (Smith, 1979). In this context, let $\mathbf{h}$ be a feasible route flow vector. Utilizing Assumption A and the conditions described in (2.6), $\mathbf{h}^*$ is a Wardrop Equilibrium flow vector if and only if

$$\mathbf{c}(\mathbf{h}^*)'(\mathbf{h} - \mathbf{h}^*) \geq \mathbf{0}, \quad \forall \mathbf{h} \in \mathcal{H}, \tag{2.7}$$

where $\mathcal{H}$ is a convex feasible set for the route-based problem.

The intuition behind this result is that if there exists some $\hat{\mathbf{h}} \in \mathcal{H}$ such that

$\mathbf{c}(\mathbf{h}^*)'(\hat{\mathbf{h}} - \mathbf{h}^*) < \mathbf{0}$, then we have that $\mathbf{c}(\mathbf{h}^*)'\hat{\mathbf{h}} < \mathbf{c}(\mathbf{h}^*)'\mathbf{h}^*$ which implies that the total system travel time can be reduced for flows $\hat{\mathbf{h}}$ and travel times are $\mathbf{c}(\mathbf{h}^*)$. Hence, switching from $\mathbf{h}^*$ to $\hat{\mathbf{h}}$ must have reduced at least one vehicle's travel time even if the travel times $\mathbf{c}(\mathbf{h}^*)$ did not change.

Moreover, this result can be further studied in terms of flows (rather than routes). In short, if we assume that the route costs are additive for a set of links (i.e., the route cost is the summation of its link costs), then, we can rewrite these conditions in the link-based form

$$\mathbf{t}(\mathbf{x}^*)'(\mathbf{x} - \mathbf{x}^*) \geq \mathbf{0}, \quad \forall \mathbf{x} \in \mathcal{F}. \tag{2.8}$$

The existence and uniqueness of the solution to (2.8) for the general transportation networks is shown in Theorems 3.14 and 3.19 of Patriksson (1994) as long as Assumption A holds, the route costs are additive, and $\mathbf{t}$ is strictly monotone. These results are derived from the more general results of Variational inequality solutions presented in Harker and Pang (1990).

Unfortunately, as stated by Marcotte and Wynter (2004), it is not easy to verify that $t_{iu}(\mathbf{x}_i)$ is strictly monotone for general *multi-class* transportation networks. This happens because the $t_{iu}(\mathbf{x}_i)$ depends on more than one vehicle class. This, in turn, creates an asymmetric Jacobian matrix of $\mathbf{t}(\cdot)$ for which first-order methods may not be sufficient to find the solution to (2.8). We therefore cannot always guarantee obtaining *unique* link flows for each vehicle class. However, we still hope to accurately estimate the cost functions by using a weighted sum of link flows (cf. (2.5)) for different user types. Still, there are many standard and efficient algorithms to find a solution to (2.8) for both single- and multi-class networks such as the Method of Successive Averages (MSA) or the Frank-Wolfe algorithm (Frank & Wolfe, 1956).

2.3.2 User-Centric Inverse Model

The objective of the inverse model is to estimate the latency cost function $\mathbf{t}(\cdot)$ (specifically $f(\cdot)$ in (2.5)) using data. The key idea is to use observable equilibrium flow data and *known* OD demands to estimate $f(\cdot)$ by using inverse optimization.

The inverse formulation seeks to learn $f(\cdot)$ so that the flow observations are as close as possible to an equilibrium. Given that this formulation relies on measured data, we expect some measurement noise. Hence, the notion of an approximate solution to the variational inequality problem is needed. Therefore, for a given $\varepsilon > 0$, we define:

Definition 2 (ε-approximate VI)**.** *Given $\varepsilon > 0$; $\hat{\mathbf{x}} \in \mathcal{F}$ is called an ε-approximate solution to* (2.8) *if*

$$\mathbf{t}(\hat{\mathbf{x}})'(\mathbf{x} - \hat{\mathbf{x}}) \geq -\varepsilon, \quad \forall \mathbf{x} \in \mathcal{F}. \tag{2.9}$$

This ε-approximate model is studied in Bertsimas et al. (2015) where it is shown that $\hat{\mathbf{x}}$ is an optimal solution to (2.9) if: (i) $\mathcal{F}$ can be represented as the intersection of a small number of conic inequalities in standard form ($\mathcal{F} = \{\mathbf{x} \ : \ \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \in C\}$); (ii) $\mathcal{F}$ has an interior point (Slater condition); and (iii) there exists $\mathbf{y}$ such that

$$\mathbf{A}'\mathbf{y} \leq_C \mathbf{t}(\hat{\mathbf{x}}), \tag{2.10}$$

$$\mathbf{t}(\hat{\mathbf{x}})'\hat{\mathbf{x}} - \mathbf{b}'\mathbf{y} \leq \varepsilon. \tag{2.11}$$

In short, this result is achieved by leveraging Definition 2 and strong duality, and it is detailed in Theorem 2 of Bertsimas et al. (2015). Note that for our multi-class transportation network "$\mathbf{A}$" is constructed by $\mathbf{N}$ (see (2.4)) and has $|\mathcal{W}| \cdot |\tilde{\mathcal{A}}|$ rows. Hence $\mathbf{y}^{\mathbf{w}} \in \mathbb{R}^{|\mathcal{V}|}$ are the dual variables of $\mathbf{w}$ and these conditions for the multi-class transportation network are written as

$$\mathbf{e}'_{iu}\mathbf{N}'\mathbf{y}^{\mathbf{w}} \leq t_{iu}(\hat{\mathbf{x}}_i) \quad \forall i \in [\![\tilde{\mathcal{A}}]\!], \ \mathbf{w} \in \mathcal{W}, \ u \in [\![\tilde{\mathcal{U}}]\!], \tag{2.12}$$

$$\mathbf{t}(\hat{\mathbf{x}})'\hat{\mathbf{x}} - \sum_{\mathbf{w} \in \mathcal{W}} \mathbf{d}^{\mathbf{w}\prime}\mathbf{y}^{\mathbf{w}} \leq \varepsilon. \tag{2.13}$$

Figure 2·1: Diagram of $|\mathcal{K}|$ flow observations over NYC.

Assume now we are given $|\mathcal{K}|$ link flow observations. One can think of these as flow samples which are produced by the same $\mathbf{t}(\cdot)$ and different OD demand $\mathbf{g}$'s (see Figure 2·1 for intuition). For each $\kappa \in [\![\mathcal{K}]\!]$ we have $\mathbf{x}^{(\kappa)} = \left(x_{iu}^{(\kappa)};\, u \in [\![\tilde{\mathcal{U}}]\!]\right);\, i \in [\![\tilde{\mathcal{A}}]\!])$ and the inverse problem is defined as finding a function $\mathbf{t}(\cdot)$ such that $\mathbf{x}^{(k)}$ is an ε_κ-approximate solution of (2.8). Letting $\boldsymbol{\varepsilon} := (\varepsilon_\kappa; \kappa \in [\![\mathcal{K}]\!])$, we formulate the inverse VI problem as

$$\min_{\mathbf{t},\boldsymbol{\varepsilon}} \quad \|\boldsymbol{\varepsilon}\| \tag{2.14a}$$

$$\text{s.t.} \quad \mathbf{t}(\mathbf{x}^{(\kappa)})'(\mathbf{x} - \mathbf{x}^{(\kappa)}) \geq -\varepsilon_\kappa, \quad \forall \mathbf{x} \in \mathcal{F}^{(\kappa)}, \kappa \in [\![\mathcal{K}]\!], \tag{2.14b}$$

$$\varepsilon_\kappa > 0, \quad \forall \kappa \in [\![\mathcal{K}]\!], \tag{2.14c}$$

where the optimization is over $\boldsymbol{\varepsilon}$ and the selection of function $\mathbf{t}(\cdot)$, and where $\mathcal{F}^{(\kappa)}$ depends on κ based on the different OD demands.

Note however that (2.14) is not solvable yet as we have not specified $\mathbf{t}(\cdot)$. Also, observe that the set of constraints restricts the travel time function to be within ε_k units of the Wardrop equilibrium flows for each sample. In this sense, if we solve the problem using a large $[\![\mathcal{K}]\!]$ corresponding to multiple observed networks, we will find a more "stable" travel latency function as we expect the variance of the estimated parameters to decrease.

To solve (2.14), we require to give more structure to $\mathbf{t}(\cdot)$, or more specifically to $f(\cdot)$, and to express (2.14b) with their optimality conditions. Aiming to recover a function with good data reconciling and generalization properties, we apply an approach which expresses the function $f(\cdot)$ in a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ as in Bertsimas et al. (2015). Then, we transform (2.14) using the RHKS, (2.12), and (2.13), which lead to the following optimization problem:

$$\begin{aligned}
\min_{f,\boldsymbol{\varepsilon},\mathbf{y}} \quad & \|\boldsymbol{\varepsilon}\| + \gamma \|f\|^2_{\mathcal{H}} && \text{(2.15a)}\\
\text{s.t.} \quad & \mathbf{e}'_{iu}\mathbf{N}'\mathbf{y}^{\mathbf{w}} \leq t^0_{iu} f\left(\frac{\boldsymbol{\theta}'\mathbf{x}^{(\kappa)}_i}{m_i}\right), \forall i \in [\![\tilde{\mathcal{A}}]\!],\ u \in [\![\tilde{\mathcal{U}}]\!],\ \mathbf{w} \in \mathcal{W}^{(\kappa)},\ \kappa \in [\![\mathcal{K}]\!], && \text{(2.15b)}\\
& \sum_{i=1}^{|\tilde{\mathcal{A}}|}\left(\sum_{u=1}^{|\tilde{\mathcal{U}}|} t^0_{iu} x^{(\kappa)}_{iu}\ f\left(\frac{\boldsymbol{\theta}'\mathbf{x}^{(\kappa)}_i}{m_i}\right)\right) - \sum_{\mathbf{w}\in\mathcal{W}_\kappa} (\mathbf{d}^{\mathbf{w}})'\mathbf{y}^{\mathbf{w}} \leq \varepsilon_\kappa, \quad \forall \kappa \in [\![\mathcal{K}]\!], && \text{(2.15c)}\\
& f\left(\frac{\boldsymbol{\theta}'\mathbf{x}^{(\kappa)}_i}{m_i}\right) < f\left(\frac{\boldsymbol{\theta}'\mathbf{x}^{(\kappa)}_{\tilde{i}}}{m_{\tilde{i}}}\right), \forall i,\ \tilde{i} \in [\![\tilde{\mathcal{A}}]\!] \text{ s.t.} \frac{\boldsymbol{\theta}'\mathbf{x}^{(\kappa)}_i}{m_i} < \frac{\boldsymbol{\theta}'\mathbf{x}^{(\kappa)}_{\tilde{i}}}{m_{\tilde{i}}};\ \forall \kappa \in [\![\mathcal{K}]\!], && \text{(2.15d)}\\
& f(0) = 1, && \text{(2.15e)}\\
& \boldsymbol{\epsilon} \geq \mathbf{0}, \quad f \in \mathcal{H},
\end{aligned}$$

where $\mathcal{W}^{(\kappa)}$ is the OD demand matrix of the κ-th network and where we have replaced constraints (2.14b) with (2.15b)-(2.15e). Moreover, note that in this case, when considering $|\mathcal{K}|$ flow observations, then $\mathbf{y} \in \mathbb{R}^{|\mathcal{W}|\times|\mathcal{K}|}$ via $\mathcal{W}^{(\kappa)}$.

In the objective, γ is a regularization parameter and $\|f\|^2_{\mathcal{H}}$ denotes the squared norm of $f(\cdot)$ in $\mathcal{H}$ (note that to solve (2.15), $f(\cdot)$ still need to be specified). The first constraint (2.15b) corresponds to dual feasibility, (2.15c) states that the solution to the κ-th network is within ε_κ distance, in other words, it is the suboptimality constraint (primal-dual gap). (2.15d) enforces $f(\cdot)$ to be strictly increasing, and (2.15e) is for normalization purposes (see (2.5)). Note that a larger γ implies giving more weight to estimating a "better" $f(\cdot)$ that generalizes better out-of-sample, rather than fitting the data better. In contrast, a smaller γ would recover a "tighter" $f(\cdot)$ in terms of data reconciliation but might not provide good generalization.

One question that may arise is: How does $\|\boldsymbol{\varepsilon}\|$ behaves as we increase the number of data samples? To answer this question, we refer to Theorems 6 and 7 of (Bertsimas et al., 2015, Sec. 6) where the convergence of (2.15) is discussed in detail.

So far, solving (2.15) is ambiguous since it optimizes over undefined functions $f(\cdot)$. To make the estimation problem solvable, we specify $\mathcal{H}$ (and thus the class of $f(\cdot)$) by choosing its reproducing kernel as a polynomial $\phi(x, y) = (c + xy)^n$ for some choice of $c \in \mathbb{R}_{\geq 0}$ and $n \in \mathbb{N}$ (Evgeniou, Pontil, & Poggio, 2000). In other words, we aim to fit $f(\cdot)$ with a polynomial of degree n. We believe this assumption a good choice since it matches our intuition on how congestion affects the latency cost of links (cf. (2.5)). The polynomial kernel function is

$$\phi(x, y) = (c + xy)^n = \sum_{i=0}^{n} \binom{n}{i} c^{n-1} x^i y^i.$$

Using the representer theorem for kernel functions (Evgeniou et al., 2000), we modify the polynomial objective function to a quadratic function parameterized by $\boldsymbol{\beta} = \{\beta_j : j = [\![n]\!]\}$ where n is the degree of the polynomial. This renders to the following tractable Quadratic Programming (QP) problem (see (3.2), (3.2), and (3.6) in Evgeniou et al. (2000) for details)

$$\begin{aligned} \min_{\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\varepsilon}} \quad & \|\boldsymbol{\varepsilon}\| + \gamma \sum_{j=0}^{n} \frac{\beta_j^2}{\binom{n}{j} c^{n-j}} \\ \text{s.t.} \quad & (2.15\text{b}) - (2.15\text{d}), \quad \beta_0 = 1. \end{aligned} \tag{2.16}$$

The output of (2.16) contains $\boldsymbol{\beta}^*$, and therefore the $f(\cdot)$ estimator is

$$\hat{f}(x) := \sum_{i=0}^{n} \beta_i^* x^i = 1 + \sum_{i=1}^{n} \beta_i^* x^i,$$

where we set $\beta_0 = 1$ to have $f(0) = 1$. Note that the well-recognized vanilla BPR function (i.e., $f(x) = 1 + 0.15x^4$) is a special case of the proposed polynomial $\hat{f}(x)$ when $\boldsymbol{\beta} = [1, 0, 0, 0, 0.15]$.

Remark 1. *In the above QP formulation, we have assumed that the parameter vector* $\boldsymbol{\theta}$ *and the set of user classes* $\tilde{\mathcal{U}}$ *are the same for all* $|\mathcal{K}|$ *networks. Since* (2.5) *is a weighed sum of link flows of different classes of vehicles. Hence, as noted previously, we aim to accurately recover the travel latency function from such weighted sum of link flows. In Sec. 2.5 we will illustrate this by conducting a numerical experiment.*

To facilitate the analysis, let us write (2.16) using the following compact notation:

$$\min_{\boldsymbol{\beta},\mathbf{y},\boldsymbol{\varepsilon}} \quad \boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} \tag{2.17a}$$

$$\text{s.t.} \quad \mathbf{A}(\mathbf{g})\mathbf{y} + \mathbf{B}\boldsymbol{\beta} + \mathbf{C}\boldsymbol{\varepsilon} + \mathbf{h} \leq \mathbf{0}, \tag{2.17b}$$

where $\mathbf{H}$ is a positive definite matrix and $\mathbf{A}(\mathbf{g})$ depends on the demand vector $\mathbf{g}$ via constraint (2.15c). This formulation was proposed for single-class networks by Bertsimas et al. (2015) and employed by J. Zhang et al. (2018). We have extended this model for the multi-class case and in the next section we present our main contribution, the joint estimation of $f(\cdot)$ and $\mathbf{g}$ for the multi-class TAP.

2.4 The Joint Problem

2.4.1 Bilevel Formulation

Unlike most previous work in this field, we would like to recover the *travel time* function $f(\cdot)$ (i.e., $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_n)$) and the OD demand vector $\mathbf{g}$ jointly. To achieve this, we define the joint problem as a bilevel formulation and we propose an algorithm to find its solution.

To ease notation, for any $\boldsymbol{\beta}$ and $\mathbf{g}$ we let $\mathbf{x}(\boldsymbol{\beta}, \mathbf{g}) = (\mathbf{x}_i(\boldsymbol{\beta}, \mathbf{g});\ \forall i \in [\![\tilde{\mathcal{A}}]\!])$ be the optimal solution to the TAP (i.e., (2.8)). Assume that we observe $|\mathcal{K}|$ different equilibrium link flow vectors $\mathbf{x}^{(\kappa)}$, and let $\mathbf{x}^d := (\mathbf{x}^{(\kappa)}, \forall \kappa \in [\![\mathcal{K}]\!])$ be the vectorized version of all the observed data. Using these definitions we write the bilevel optimization problem as

$$\min_{\boldsymbol{\beta},\mathbf{g}} \quad \hat{F}(\boldsymbol{\beta}, \mathbf{g}) := \sum_{\kappa=1}^{|\mathcal{K}|} \sum_{u=1}^{|\tilde{\mathcal{U}}|} \sum_{i=1}^{|\tilde{\mathcal{A}}|} ([\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})]_{iu} - x_{iu}^{(\kappa)})^2 \tag{2.18a}$$

$$\text{s.t.} \quad \mathbf{x}(\boldsymbol{\beta}, \mathbf{g}) = \text{TAP}(\boldsymbol{\beta}, \mathbf{g}), \quad \forall u \in [\![\tilde{U}]\!] \tag{2.18b}$$

$$(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\varepsilon}) = \arg\min_{\boldsymbol{\beta},\mathbf{y},\boldsymbol{\varepsilon}} \big\{ \boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta}, \tag{2.18c}$$

$$\text{s.t.} \ \mathbf{A}(\mathbf{g})\mathbf{y} + \mathbf{B}(\mathbf{x}^d)\boldsymbol{\beta} + \mathbf{C}\boldsymbol{\varepsilon} + \mathbf{h} \leq \mathbf{0} \big\}.$$

Notice that this problem is a bilevel problem and the lower-level problem/constraint (2.18c) could be avoided completely. The reasoning behind including (2.18c) as a constraint to the joint estimation problem is to enforce a stronger, and optimal, relationship between $\mathbf{g}$ and $\boldsymbol{\beta}$ by leveraging the data and the inverse method presented in the previous section. This aims to guide the optimization to better outcomes.

Remark 2. *Typically, most related work aiming to solve this problem include a term in* (2.18a) *which penalize the deviation of the estimated demand* $\mathbf{g}$ *from a* known *demand vector* $\mathbf{g}^0$*. To include this penalty one requires* prior knowledge *(or a good estimate) of the true demand through* $\mathbf{g}^0$*. We believe this term has been historically included due to the fact that the optimization problem is non-convex in terms of* $\mathbf{x}$*. Hence, this prior knowledge aims to steer the solution towards a local optimum close to the initial demand estimate* $\mathbf{g}^0$*. In contrast, we avoid imposing such connection with an initial demand estimate. Still, our methodology allows, if desired, to include such term in the objective function.*

2.4.2 Optimality conditions of (2.18c)

To write (2.18) as a computationally solvable joint optimization problem, we replace the lower-level problem (2.18c) by its optimality conditions and write (2.18) as a single-level problem. Note that both (2.18) and (2.17) have convex quadratic objectives as the only non-zero entries in their objective are in the diagonal of the $\mathbf{Q}$ matrix in their canonical QP standard form. We begin our analysis by writing the Lagrangian function:

$$\mathcal{L}(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\varepsilon}; \boldsymbol{\nu}) = \boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} + \boldsymbol{\nu}'(\mathbf{A}(\mathbf{g})\mathbf{y} + \mathbf{B}(\mathbf{x}^d)\boldsymbol{\beta} + \mathbf{C}\boldsymbol{\varepsilon} + \mathbf{h}), \tag{2.19}$$

where $\boldsymbol{\nu}$ denotes the dual variables corresponding to constraints (2.17b). Furthermore, the first-order conditions of (2.17) are

$$\partial\mathcal{L}/\partial\boldsymbol{\varepsilon} = 2\mathbf{I}\boldsymbol{\varepsilon} + \mathbf{C}'\boldsymbol{\nu} = \mathbf{0} \Rightarrow \boldsymbol{\varepsilon} = -(1/2)\mathbf{I}^{-1}\mathbf{C}'\boldsymbol{\nu}, \tag{2.20a}$$

$$\partial\mathcal{L}/\partial\boldsymbol{\beta} = 2\mathbf{H}\boldsymbol{\beta} + \mathbf{B}(\mathbf{x}^d)'\boldsymbol{\nu} = \mathbf{0} \Rightarrow \boldsymbol{\beta} = -(1/2)\mathbf{H}^{-1}\mathbf{B}(\mathbf{x}^d)'\boldsymbol{\nu}, \tag{2.20b}$$

$$\partial\mathcal{L}/\partial\mathbf{y} = \mathbf{A}(\mathbf{g})'\boldsymbol{\nu} = \mathbf{0}, \tag{2.20c}$$

which, by substituting $\boldsymbol{\varepsilon}$ and $\boldsymbol{\beta}$ in (2.19) using (2.20), yields to the dual function

$$D(\boldsymbol{\nu}) = -(1/4)\boldsymbol{\nu}'(\mathbf{CIC}' + \mathbf{B}(\mathbf{x}^d)\mathbf{H}^{-1}\mathbf{B}(\mathbf{x}^d))\boldsymbol{\nu} + \mathbf{h}'\boldsymbol{\nu}.$$

It follows that for each primal-dual pair $(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\varepsilon};\ \boldsymbol{\nu})$ in (2.18c), the necessary and sufficient conditions are

$$\mathbf{A}(\mathbf{g})\mathbf{y} + \mathbf{B}(\mathbf{x}^d)\boldsymbol{\beta} + \mathbf{C}\boldsymbol{\varepsilon} + \mathbf{h} \leq \mathbf{0}, \tag{2.21a}$$

$$\mathbf{A}(\mathbf{g})'\boldsymbol{\nu} = \mathbf{0}, \tag{2.21b}$$

$$\boldsymbol{\nu} \geq \mathbf{0}, \tag{2.21c}$$

$$\boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} = -\frac{1}{4}(\boldsymbol{\nu}'\mathbf{CIC}' + \boldsymbol{\nu}'\mathbf{B}(\mathbf{x}^d)\mathbf{H}^{-1}\mathbf{B}(\mathbf{x}^d)' + \mathbf{h}')\boldsymbol{\nu}, \tag{2.21d}$$

where (2.21a)-(2.21c) correspond to primal and dual feasibility and (2.21d) equates the primal and dual objectives to ensure strong duality.

2.4.3 Relaxation and Trust-Region Feasible-Direction Method

At this point we are ready to convert the bilevel problem to a single-level joint problem. We do this by replacing the lower-level problem (2.18c) by its KKT conditions (2.21). In other words, we minimize (2.18a) subject to (2.21). However, note that (2.21d), corresponding to strong duality, is a non-convex quadratic equality constraint. To address this issue, we relax this constraint by requiring that the gap is upper bounded by some $\xi \in \mathbb{R}_{\geq 0}$. We penalize this gap in the objective using some λ. Then, the new joint formulation becomes

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{g},\mathbf{y},\boldsymbol{\nu},\boldsymbol{\varepsilon},\xi} \quad & F(\boldsymbol{\beta},\mathbf{g},\xi) := \hat{F}(\boldsymbol{\beta},\mathbf{g}) + \lambda\xi && \text{(2.22a)}\\
\text{s.t.} \quad & (2.21\text{a}), (2.21\text{b}),\\
& \boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} + \frac{1}{4}\boldsymbol{\nu}'(\mathbf{CIC}' + \mathbf{B}(\mathbf{x}^d)\mathbf{H}^{-1}\mathbf{B}(\mathbf{x}^d)')\boldsymbol{\nu} - \mathbf{h}'\boldsymbol{\nu} \leq \xi, && \text{(2.22b)}\\
& \boldsymbol{\nu}, \boldsymbol{g}, \boldsymbol{\beta}, \xi \geq \mathbf{0}, && \text{(2.22c)}
\end{aligned}$$

where $\boldsymbol{\beta}$ and $\mathbf{g}$ are the quantities of interest; $\mathbf{y}$ and $\boldsymbol{\nu}$ are the dual variables defined in (2.15) and (2.19), respectively; and $\boldsymbol{\varepsilon}$ and ξ are the primal-dual gap relaxations.

Unfortunately, both the objective and the constraints, through $\mathbf{A}$, are nonlinear functions of $\mathbf{g}$, $\mathbf{y}$ and $\boldsymbol{\nu}$. Therefore, to solve (2.22), we propose an iterative *trust-region feasible direction* method. To do so, let $\mathbf{z} = (\boldsymbol{\beta}, \mathbf{g}, \xi)$ and denote with j the iteration count. With this sequential approach, we evaluate the gradient of $F(\cdot)$ at the previous iteration and seek the steepest feasible direction of descent by solving the following optimization problem:

$$\min_{\mathbf{z}^j,\mathbf{y},\boldsymbol{\nu},\boldsymbol{\varepsilon}} \quad \nabla F(\mathbf{z}^{j-1})'(\mathbf{z}^{j-1} - \mathbf{z}^j) \qquad \text{(2.23a)}$$

$$\begin{aligned} \text{s.t.} \quad & (2.21\text{a}), (2.21\text{b}), (2.22\text{b}) \\ & \mathbf{g}^{j-1} - c_{1j}\mathbf{e} \ \leq \ \mathbf{g}^j \leq \ \mathbf{g}^{j-1} + c_{2j}\mathbf{e}, && (2.23\text{b}) \\ & \boldsymbol{\beta}^{j-1} - d_{1j}\mathbf{e} \ \leq \ \boldsymbol{\beta}^j \leq \ \boldsymbol{\beta}^{j-1} + d_{2j}\mathbf{e}, && (2.23\text{c}) \\ & \boldsymbol{\nu}, \mathbf{z}^j \geq \mathbf{0}, && (2.23\text{d}) \end{aligned}$$

where $\mathbf{e}$ denote the vector of all ones and c_{1j}, c_{2j}, d_{1j}, d_{2j} are used as step-size parameters. The matrix $\mathbf{A}$ in the constraint (2.21a), and (2.21b) is a function of $\mathbf{g}$ and we approximate it by using $\mathbf{g}^{j-1}$. Then, the gradient in (2.23a) is expressed by

$$\begin{aligned} \nabla F(\mathbf{z}^j)' = \Bigg[& \sum_{u=1}^{|\tilde{\mathcal{U}}|} \sum_{i=1}^{|\tilde{\mathcal{A}}|} 2(x_{iu}(\mathbf{z}^j) - x_{iu}^*) \frac{\partial x_{iu}(\boldsymbol{\beta}^j, \mathbf{g}^j)}{\partial \beta_l}, \ l = [\![n]\!]; \\ & \sum_{i=1}^{|\tilde{\mathcal{A}}|} 2(x_{iu}(\mathbf{z}^j) - x_{iu}^*) \frac{\partial x_{iu}(\boldsymbol{\beta}^j, \mathbf{g}^j)}{\partial g_{ku}}, \ k = [\![\tilde{\mathcal{W}}]\!], u = [\![\tilde{\mathcal{U}}]\!]; \lambda \Bigg]. \end{aligned} \tag{2.24}$$

As a result, problem (2.23) has a linear objective (provided that we can evaluate the partial derivatives) and constraints that are linear and convex quadratic, making it a tractable problem. In fact, this problem can be written as a quadratic program by replacing ξ^j in the objective (2.23a) by the left-hand side of constraint (2.22b). Given these "constant" approximations of the constraints at the prior iterate, the role of c_{1j}, c_{2j}, d_{1j}, d_{2j} is to ensure that the optimization takes place in a relatively small "trust" region for $(\boldsymbol{\beta}^j, \mathbf{g}^j)$ that is not too far from the prior iterate $(\boldsymbol{\beta}^{j-1}, \mathbf{g}^{j-1})$. Note, however, that to solve (2.23), we still need to estimate the partial derivatives of the flows with respect to $\mathbf{g}$ and $\boldsymbol{\beta}$.

2.4.4 Derivatives

It has been observed that it is hard to derive analytical expressions for the partial derivatives of $\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})$ with respect to $\boldsymbol{\beta}$ and $\mathbf{g}$. To overcome this, we use classical approximation techniques. We refer the interested reader to a comprehensive discussion carried out by Patriksson (2004).

Directional flow derivatives with respect to the OD demand

We derive an approximation to the gradient of $\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})$ with respect to $\mathbf{g}$. To that end, fix $i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]$ and add the flows over the OD pairs demands

$$x_{iu}(\boldsymbol{\beta}, \mathbf{g}) = \sum_{k \in [\![\tilde{\mathcal{W}}]\!]} \sum_{r \in \mathcal{R}_u^k} \delta_r^{a(i,u)} p_u^{kr} g_{ku} = \sum_{k \in [\![\tilde{\mathcal{W}}]\!]} g_{ku} \sum_{r \in \mathcal{R}_u^k} \delta_r^{a(i,u)} p_u^{kr}, \tag{2.25}$$

where $\mathcal{R}_u^k$ denotes the set of feasible routes for vehicle class u in OD pair k; p_u^{wr} stands for the percentage of class u vehicles using route $r \in \mathcal{R}_u^k$, and

$$\delta_r^{a(i,u)} := \begin{cases} 1, \text{ if route } r \text{ uses link } a(i,u); \\ 0, \text{ otherwise.} \end{cases} \tag{2.26}$$

As pointed out by Noriega and Florian (2007) and Spiess (1990), for all $k \in [\![\tilde{\mathcal{W}}]\!]$ and all $u \in [\![\tilde{\mathcal{U}}]\!]$, and assuming that the route probabilities are locally constant, equation (2.25) implies

$$\frac{\partial x_{iu}(\boldsymbol{\beta}, \mathbf{g})}{\partial g_{ku}} = \begin{cases} \sum_{r \in \mathcal{R}_u^k} \delta_r^{a(i,u)} p_u^{kr}; \\ 0, \text{ otherwise.} \end{cases} \tag{2.27}$$

If we only consider the shortest route $r_{ku}(\boldsymbol{\beta}, \mathbf{g})$ based on the travel latency cost, we have that

$$\frac{\partial x_{iu}(\boldsymbol{\beta}, \mathbf{g})}{\partial g_{ku}} \approx \delta_{r_{ku}(\boldsymbol{\beta}, \mathbf{g})}^{a(i,u)} = \begin{cases} 1, \text{ if } a(i,u) \in r_{ku}(\boldsymbol{\beta}, \mathbf{g}); \\ 0, \text{ otherwise,} \end{cases} \tag{2.28}$$

where $a(i,u) \in r_{ku}(\boldsymbol{\beta}, \mathbf{g})$ indicates that the shortest route $r_{ku}(\boldsymbol{\beta}, \mathbf{g})$ uses link $a(i,u)$. By (2.28) we obtain an approximation to the Jacobian matrix

$$\left[\frac{\partial x_{iu}(\boldsymbol{\beta}, \mathbf{g})}{\partial g_{ku}};\, i \in [\![\tilde{\mathcal{A}}]\!],\, u \in [\![\tilde{\mathcal{U}}]\!],\, k \in [\![\tilde{\mathcal{W}}]\!]\right]. \tag{2.29}$$

The reasons to consider only the shortest routes for the purpose of calculating these gradients are: (i) GPS routing services are widely-used by vehicle drivers so they tend to always select the fastest routes, (ii) considering only the fastest routes significantly simplifies the calculation of the route-choice probabilities, and (iii) extensive numerical experiments and algorithms show that such an approximation of the gradients performs well, e.g., TAPAS, or MSA.

Directional flow derivatives with respect to the coefficients of the travel latency function

To the best of our knowledge there are two main techniques to calculate directional derivatives of the link flows with respect to a perturbation ρ on the cost coefficients $\boldsymbol{\beta}$. Tobin and Friesz (1988) and Patriksson (2004) proposed a sensitivity analysis method with respect to routes that require solving a large linear system that it is hard to solve for large-scale networks. To overcome this issue, Josefsson and Patriksson (2007) developed a QP formulation to calculate such derivatives. However, both the LP and QP have a similar complexity as solving the TAP. Therefore, although we are able to use any of these methods to calculate $\partial x_{iu}(\boldsymbol{\beta}, \mathbf{g})/\partial \beta_l$, we prefer to use a finite-difference approximation. This is because, (i) the complexity of solving the TAP is similar to that of the QP proposed in Josefsson and Patriksson (2007); (ii) there are fast algorithms such as MSA, Frank-Wolfe, TAPAS to solve the TAP efficiently; and (iii) the TAP allows to include all routes connecting any OD pair of any class rather than defining a route set. Using $\text{TAP}_{iu}(\cdot)$ to denote the solution of the TAP

```
Algorithm 1 Joint cost-demand estimation.
Input: $\mathcal{G}$; $\lambda$; $\rho$; $(\mathbf{g}^0, \boldsymbol{\beta}^0)$: initial OD demand and travel latency function; step-size
    rule; $(\eta, T)$ : precision parameters.
Initialize: Set $j = 0$, calculate $F(\boldsymbol{\beta}^0, \mathbf{g}^0)$, if: $F(\boldsymbol{\beta}^0, \mathbf{g}^0) = 0$ then: terminate and
    output $(\boldsymbol{\beta}^0, \mathbf{g}^0)$ else:
1: Compute $\mathbf{x}_u^j = \text{TAP}_u(\mathbf{g}^j, \boldsymbol{\beta}^j)$ for every $u \in [\![\tilde{\mathcal{U}}]\!]$.
2: Obtain derivatives $\partial \mathbf{x}(\boldsymbol{\beta}^j, \mathbf{g}^j)/\partial \boldsymbol{\beta}^j$ and $\partial \mathbf{x}(\boldsymbol{\beta}^j, \mathbf{g}^j)/\partial \mathbf{g}^j$ using (2.29) and (2.30),
   respectively.
3: Approximate the descent direction $\nabla F(\boldsymbol{\beta}^j, \mathbf{g}^j)$ using (2.24).
4: Choose a trust region/step size.
5: Solve for a steepest descent direction using (2.23), and obtain $\boldsymbol{\beta}^{j+1}$ and $\mathbf{g}^{j+1}$.
6: (Termination criterion)
     • if $\frac{\|F(\boldsymbol{\beta}^j, \mathbf{g}^j) - F(\boldsymbol{\beta}^{j+1}, \mathbf{g}^{j+1})\|_2}{F(\boldsymbol{\beta}^0, \mathbf{g}^0)} \geq \eta$ and $j < T$ then let $j = j + 1$ and go to
       Step 1,
     • else Terminate and output: $(\boldsymbol{\beta}^{j+1}, \mathbf{g}^{j+1})$.
```

for link i and class u, for some small enough ρ we compute

$$\frac{\partial x_{iu}(\boldsymbol{\beta}^j, \mathbf{g}^j)}{\partial \beta_l} \approx \frac{\text{TAP}_{iu}(\boldsymbol{\beta}^j + \rho \mathbf{e}_l, \mathbf{g}^j) - \text{TAP}_{iu}(\boldsymbol{\beta}^j, \mathbf{g}^j)}{\rho},$$

where $\mathbf{e}_l$ is the lth unit vector.

Using these two approximation to the partial derivatives we have developed a complete method for solving the joint problem. We summarize our approach in Algorithm 1.

Our algorithm proceeds as a *trust-region feasible direction method* that seeks to solve the joint cost-demand estimation problem (defined in (2.18)). The key idea of the algorithm is to jointly select, at every iteration j, the vectors $\boldsymbol{\beta}^j$ and $\mathbf{g}^j$ that maximize the reduction in the objective function of (2.18) by solving (2.23). To achieve this, we evaluate the gradient of the function at the previous iteration of $\boldsymbol{\beta}^j$ and $\mathbf{g}^j$ and restrict our new iterates to be within a trust region. By recursively performing this procedure, we expect (and observe empirically) the algorithm to converge to a local minimum.

Note that Alg. 1 is a method that solves a sequence of quadratic problems. This approach has some similarities with Yang et al. (2001) where, in general, convergence cannot be guaranteed. This is due to the dependence between $\boldsymbol{\beta}$ and $\mathbf{g}$. In addition, a convergence analysis would need to account for the fact that $\mathbf{A}(\mathbf{g}^j)$ is approximated by $\mathbf{A}(\mathbf{g}^{j-1})$, which is not straightforward. A potential approach to address this issue is by selecting a small step-size and by using the minimization rule. To that end, we need to perform a line search at every iteration (this is standard in alternating methods, see Lundgren and Peterson (2008) and García-Ródenas and Marín (2009)). However computing $\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})$ is computationally demanding and line search would be too expensive.

Moreover, we point out that solving (2.23) recursively is computationally intensive as it involves solving a large optimization problem (e.g., the dimensions of $\boldsymbol{\nu}$ and $\mathbf{y}$ are $\mathcal{O}(|\mathcal{K}| \times |\mathcal{A}|)$ and $\mathcal{O}(|\mathcal{K}| \times (|\mathcal{W}| \times |\mathcal{V}|))$, respectively) with a linear objective and quadratic constraints. We leave the development of techniques to reduce the computational burden, such as the one described by Bertsimas et al. (2015), to future work.

Both the computational burden and the hardness of proving convergence motivate us to propose an *alternating method* described in Alg 2. This algorithm proceeds by adjusting $\mathbf{g}$ using gradient descent, and then estimating the *restricted* travel latency function using (2.15) with constraint (2.23c). The algorithm possesses two advantages over Alg. 1. First, it decomposes the problem making it more computationally tractable. Second, we can guarantee its convergence. This follows by observing that $F(\boldsymbol{\beta}, \mathbf{g})$ is lower-bounded by 0 and by observing that at every iteration the objective is either reduced or stays at its current value. To show this, we argue that if $c_1 < 0 < c_2$ then Step 3 certifies $F(\boldsymbol{\beta}^j, \mathbf{g}^{j+1}) \leq F(\boldsymbol{\beta}^j, \mathbf{g}^j)$ since we can always select $\alpha = 0$. Moreover, by imposing Step 6 we ensure a similar argument for $\boldsymbol{\beta}$. Thus, by using the monotone

Algorithm 2 Alternating cost-demand estimation.

```
Input: $\mathcal{G}$; $\lambda$; $\rho$; $(\mathbf{g}^0, \boldsymbol{\beta}^0)$: initial OD demand and travel latency function; step-size
    rule; $(\eta, T)$ : precision parameters.
Initialize: Set $j = 0$, calculate $F(\boldsymbol{\beta}^0, \mathbf{g}^0)$, if: $F(\boldsymbol{\beta}^0, \mathbf{g}^0) = 0$ then: terminate and
    output $(\boldsymbol{\beta}^0, \mathbf{g}^0)$ else:
 1: Compute $\mathbf{x}_u^j = \text{TAP}_u(\mathbf{g}^j, \boldsymbol{\beta}^j)$ for every $u \in [\![\tilde{\mathcal{U}}]\!]$.
 2: Obtain derivative $\partial \mathbf{x}(\boldsymbol{\beta}^j, \mathbf{g}^j)/\partial \boldsymbol{\beta}^j$ using (2.29).
 3: Find best step-size by doing line-search:
        $\alpha = \arg\min_{c_1 \leq \alpha \leq c2} F(\boldsymbol{\beta}^j, \mathbf{g} - \alpha \nabla_{\mathbf{g}} F(\boldsymbol{\beta}^j, \mathbf{g}^j))$
 4: Take a gradient-step: $\mathbf{g}^{j+1} = \mathbf{g}^j - \alpha \nabla_{\mathbf{g}} F(\boldsymbol{\beta}^j, \mathbf{g}^{j+1})$
 5: Obtain $\boldsymbol{\beta}^{j+1}$ by solving (2.15) for a fixed $\mathbf{g}^{j+1}$ and constraint (2.23c).
 6: If $F(\mathbf{g}^{j+1}, \boldsymbol{\beta}^{j+1}) > F(\mathbf{g}^{j+1}, \boldsymbol{\beta}^j)$ let $\boldsymbol{\beta}^{j+1} = \boldsymbol{\beta}^j$.
 7: (Termination criterion): Same as in Alg. 1
```

convergence theorem Bertsekas (2016) we guarantee its convergence. Its main drawback is that the steepest directions of both $\mathbf{g}$ and $\boldsymbol{\beta}$ are done independently rather than jointly as in Alg 1 and it may not converge to a local minimum of the joint problem.

2.5 Model Validation and Case Studies

We report numerical experiments conducted on a benchmark network and subnetworks from the Eastern Massachusetts Area (EMA) and New York City (NYC). Note that for the subnetworks we will estimate the OD demand based on the observed flow and that *through traffic* (traffic originated at and destined for points outside the subnetwork) will be estimated as belonging to an OD pair. For example, for Figure 2·2b, demand initiated north of the top node (e.g., New Hampshire) and going south (e.g., Rhode Island) would be counted as part of the OD demand from node 1 to 8.

We begin by validating our methods on single-class and multi-class variants of the well-known Braess Network. Then, we perform additional validation experiments using real networks: first on the EMA network focusing on an interstate highway subnetwork and then on an urban network in NYC.

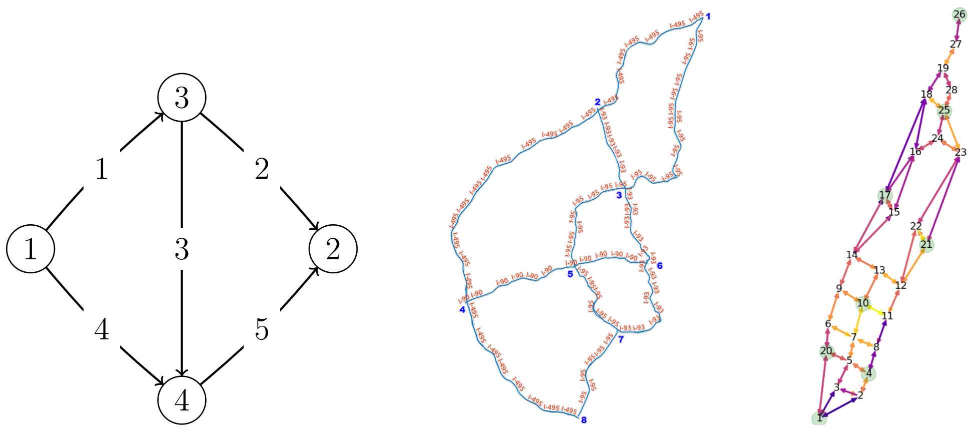
2.5.1 Model Validation

To perform these experiments, we generate "ground truth" data by selecting specific OD demands and cost functions $(\boldsymbol{\beta}^*, \mathbf{g}^*)$. Then, we solve the TAP problem using these "true" inputs to obtain $\mathbf{x}^*$, the "true" flow (note that, normally, we would obtain $\mathbf{x}^*$ from data). To test the performance, we take the generated flows, $\mathbf{x}^*$ as input to Alg. 1 and compare the resulting $(\boldsymbol{\beta}^j, \mathbf{g}^j)$ to the ground truth. In addition to reporting the performance of the methods developed herein, we compare them with simpler algorithms. In particular: (i) the approach of just estimating OD demands by assuming a fixed travel latency cost function ($f(\cdot) = \text{BPR}$), and (ii) a gradient descent method (GD) using the estimated derivatives in (2.29), (2.30).

Single-Class Braess Network

As our first experiment, we use the Braess network (Fig. 2·2a). We generate ground truth data by considering a single OD pair which transports $4,000$ vehicles per hour from node 1 to 2 and with a true travel latency function $f(x) = 1+0.45x^4$. The resulting "true" flows when solving the TAP for this network are: $(2095, 1904, 2095, 0, 1904, 1)$ for links $(1, 2, 3, 4, 5)$ respectively. Then, we initiate Alg. 1 with $\mathbf{g}_0 = \mathbf{0}$ and $\boldsymbol{\beta}_0 = (1, 0, 0, 0, 0.15, 0)$. We set $c = 30$, $\lambda = 0.1$, $\rho = 0.1$, $n = 5$, and adaptive step-sizes $c_{1j} = c_{2j} = \frac{200}{\sqrt{j}}$, $d_{1j} = d_{2j} = \frac{0.02}{j^{3/4}}$.

In the left plot of Fig. 2·3 we observe the objective function of the joint problem (2.22a) converging to zero when solving the problem jointly. In contrast, when only adjusting OD Demands with a fixed $f(\cdot)$, the method converges to a higher value. In addition, the right hand side plot of Fig. 2·3 shows the norm between the true demand and the estimated one. We see that this distance converges to a lower value when solving the joint problem in contrast to all other methods. Highlighting the benefits of solving the joint formulation over other approaches. It is imperative to



(a) Braess' **(b)** EMA **(c)** NYC

Figure 2·2: Network topologies.

point out that the algorithm is sensitive to the selected parameters. In particular, the selection of step sizes is relevant as it may cause the algorithm to diverge. We believe this happens because the matrix $\mathbf{A}$ is evaluated at the previous iterations and therefore, the selection of $(c_{1j}, c_{2j}, d_{1j}, d_{2j})$ has a direct impact on both the closeness to the previous iteration and the algorithm's convergence rate.

Multi-Class Braess Network

We introduce two types of vehicles: *cars* and *commercial trucks* ($|\tilde{\mathcal{U}}| = 2$). For each vehicle class we generate ground truth demands $\mathbf{g}^*_{\text{car}} = [4000, 0]$ and , $\mathbf{g}^*_{\text{truck}} = [400, 0]$ for OD pairs $(1, 2)$ and $(2, 1)$, respectively. Recall that now we are interested on estimating the OD demand for each vehicle type, i.e., $\mathbf{g}_{\text{car}}$ and $\mathbf{g}_{\text{truck}}$, and the travel latency function $f(\cdot)$. We select $T = 500$, $\lambda = 0.1$, $\rho = 0.1$, $n = 5$, $c_{1j} = c_{2j} = \frac{400}{j}$, $d_{1j} = d_{2j} = 0.05/j$ and run Alg. 1 with parameters as in the single-class case (i.e., $f(x) = 1 + 0.15x^4$, $\mathbf{g}^0 = \mathbf{0}$). To consider the relatively lower speed and larger physical dimensions of trucks compared to cars, we assume the flow weight vector to be $\boldsymbol{\theta} = (1, 2)$ for cars and trucks, respectively. We assume $t^0_{i,\text{car}} = t^0_i$ and $t^0_{i,\text{truck}} = 1.1t^0_i$, where t^0_i is the *free-flow travel time* for the *physical* link indexed by i. The details

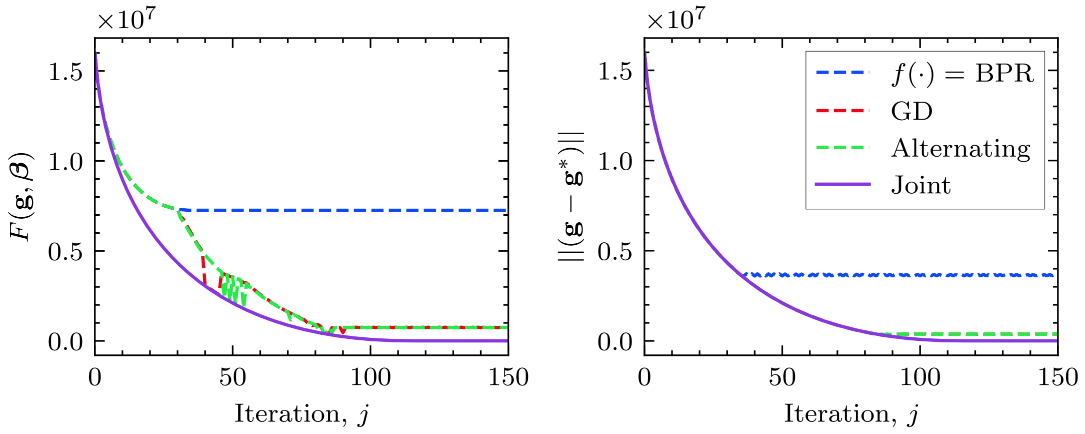


Figure 2·3: Model validation on Braess Network. $f(\cdot)$ solves the problem using a static BPR function; *GD* uses a gradient decent; *Alternating* solves the two problems sequentially; *Joint* solves the problem using Alg. 1

regarding the road characteristics used for the free flow speeds and capacities, as well as the algorithms employed in this chapter are available in our online public repository[1]. Figures 2·4c and 2·4d show how Alg. 1 recovers the true OD demands for both vehicle classes. In Fig. 2·4b we see how $\hat{f}(\mathbf{x})$ (our estimate of $f(x)$) is getting closer to the true travel latency function. Finally, in Fig. 2·4a we show how the objective function converges to a value close to zero. In this experiment we observe that we are able to recover the truth demand and travel latency function regardless the limitation of the multi-class model as pointed in Sec 2.3.1.

Eastern Massachusetts Area Seeking for more realistic networks, we perform a validation experiment using the road network of EMA. We select this network since it helps to emulate the conditions of a interstate highway road network. The graph (Fig. 2·2b) consists of 8 nodes, 24 links and 56 OD pairs.

We run Algs. 1 and 2 with $T = 60$, $\lambda = 0.1$, $\rho = 0.1$, $n = 5$, $c_{1j} = c_{2j} = \frac{250}{j}$, $d_{1j} = d_{2j} = \frac{0.02}{\sqrt{j}}$ and we initialize it with $\boldsymbol{\beta}^0 = (1, 0, 0, 0, 0, 15, 0)$ and $\mathbf{g}^0 = \mathbf{0}$. In

[1]https://github.com/salomonw/tnet

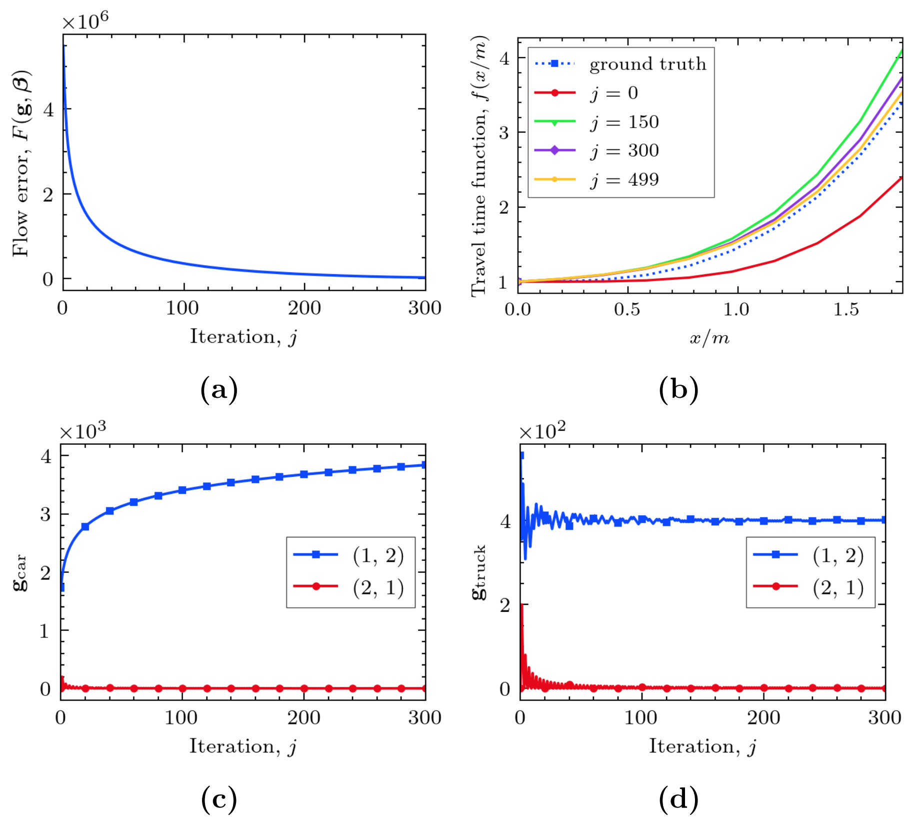


Figure 2·4: Validation Results for the Multi-Class Braess Network. (a) Objective Function, (b) $f(\cdot)$ evolution, (c) Car demand estimator, (d) Truck demand estimator.

the left plot of Fig. 2·5, we observe that jointly adjusting the cost function while estimating demands is beneficial for approximating the flows. We believe this is the case since Alg. 1 takes the joint steepest descent direction compared with taking these steps separately as in the Alternating method. In the right hand side plot of Fig. 2·5, we see the progress of $\|(\mathbf{g}^j - \mathbf{g}^*)\|$ (which is not minimized explicitly). We observe the joint method reaches a lower level than all of the other approaches, especially against the BPR approach. However, after iteration 9, this quantity begins to increase. We believe this happens due to the fact that the optimization process advances by adjusting both $\hat{f}$ and $\mathbf{g}$ and might deviate the estimated demand while tuning the travel latency function.

New York City To test our method in congested urban areas, we created a NYC validation network (Fig. 2·2c) consisting of 28 nodes, 90 edges and 8 Zones (green dots). To build this network we used two data sources: OpenStreetMaps, from which we retrieve the network topology and road characteristics, and *Uber Movement Speed Data set*, which we use for assigning speed data to each link in the network. With this in hand, we estimated travel times, speeds, densities and flows. See Appendix B for details.

We run the algorithms with $T = 30$, $\lambda = 0.1$, $\rho = 0.1$, $n = 5$, $c_{1j} = c_{2j} = \frac{100}{\sqrt{j}}$, $d_{1j} = d_{2j} = \frac{0.02}{j^{(3/4)}}$, $\boldsymbol{\beta}^0 = (1, 0, 0, 0, 0, 15, 0)$, and $g_i^0 = 1000$ for all $i \in [\![\mathcal{W}]\!]$. Consistent with the other examples we observe in Fig. 2·6 similar results as in the Braess and the EMA networks where using the joint methodology is beneficial.

2.5.2 Case Studies

Eastern Massachusetts Area

We use speed data from April, 20th, 2012 from 8 to 9 a.m. provided by the Central Transportation Planning Staff (CTPS) of the Boston Metropolitan Planning Organization (MPO). We converted this speed to flow data using the procedure described in Appendix C. With this, we run the algorithms using $T = 60$, $\lambda = 1e-3$, $\rho = 0.1$, $n = 5$, $c_{1j} = c_{2j} = \frac{100}{j}$, $d_{1j} = d_{2j} = \frac{0.02}{\sqrt{j}}$, $\boldsymbol{\beta}^0 = (1, 0, 0, 0, 0, 15, 0)$, and $\mathbf{g}^0 = \mathbf{0}$. And report the estimated OD matrix in Table 2.1.

Table 2.1: Estimated OD (veh/h) demands for the EMA network.

O/D	1	2	3	4	5	6	7	8	sum
1	0.0	1.1	0.0	515.1	298.1	803.4	36.3	310.0	1964.1
2	0.0	0.0	0.0	590.5	205.3	645.3	130.5	325.8	1897.3
3	0.0	1.1	0.0	598.6	173.6	662.4	130.3	257.0	1822.9
4	0.0	1.1	0.0	0.0	349.6	720.4	20.2	331.7	1423.1
5	0.0	1.1	0.0	563.6	0.0	645.1	87.4	320.9	1618.1
6	0.0	1.1	0.0	483.5	184.3	0.0	1.7	212.8	883.3
7	0.0	1.1	0.0	619.2	283.4	807.9	0.0	197.9	1909.5
8	1706.3	0.0	0.0	1355.2	65.8	898.0	1.7	0.0	**4027.0**
sum	1706.3	6.6	0.0	**4725.6**	1560.1	**5182.5**	408.1	1956.1	15545.3

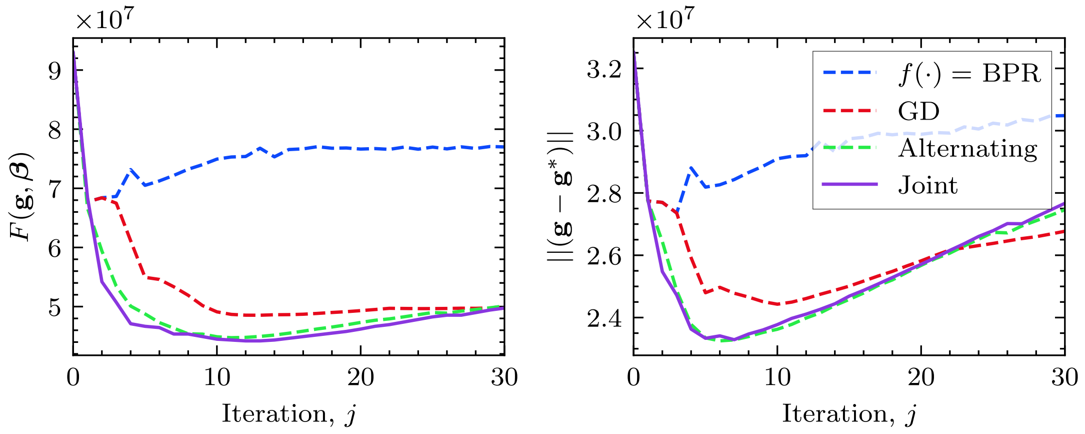


Figure 2·5: Validation Results for the EMA Network. Left and right plots show the progress over the iterations of the objective function and the OD demand deviation from a specified "truth" demand $\mathbf{g}^*$, respectively.

From field observations we understand that the traffic patterns for EMA are similar to most urban areas. In the morning, commuters travel to work towards the city center and on the afternoon, they travel to residential areas. Table 2.1 matches this intuition by estimating higher demand for OD pairs with destination in the city center. In particular for nodes 4 (Worcester) and 6 (Downtown Boston). Likewise, the origin for which most flow is generated is 8 (Taunton), corresponding to the southern most node which accounts for all the southern flow that commutes towards the Boston Metropolitan Area. In addition, our results also suggests that travelers are more sensitive to lower flows than the ones proposed by the BPR function (see that the estimated Joint $f(\cdot)$ is above the BPR in the left plot of Fig. 2·7). This is relevant for designing and planning transportation networks.

New York City

We perform a similar case study using the NYC subnetwork. We run our algorithms for traffic data on Feb 13th, 2017 at 9 am. The right hand plot in Fig 2·8 show our results of the joint approach yielding to lower flow error than the other estimation

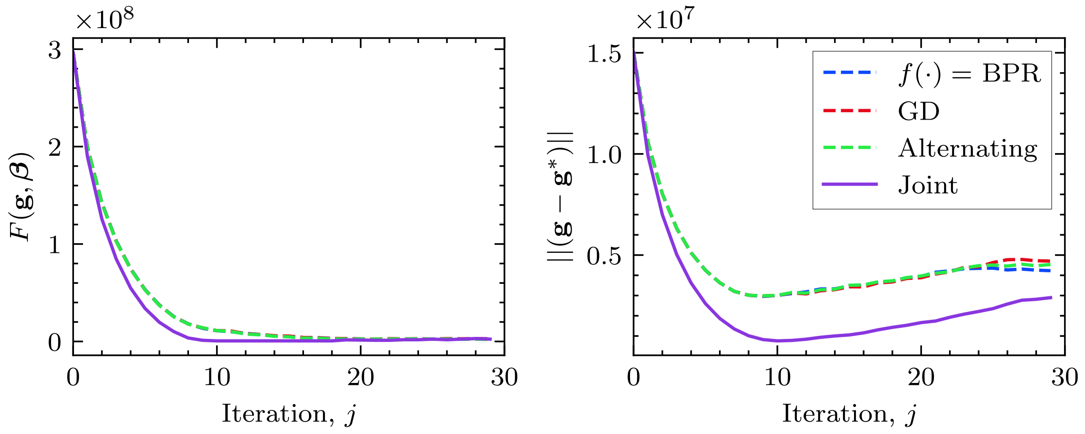


Figure 2·6: Validation Results for the NYC Network. Left and right plots show the progress over the iterations of the objective function and the OD demand deviation from a specified "truth" demand $\mathbf{g}^*$, respectively

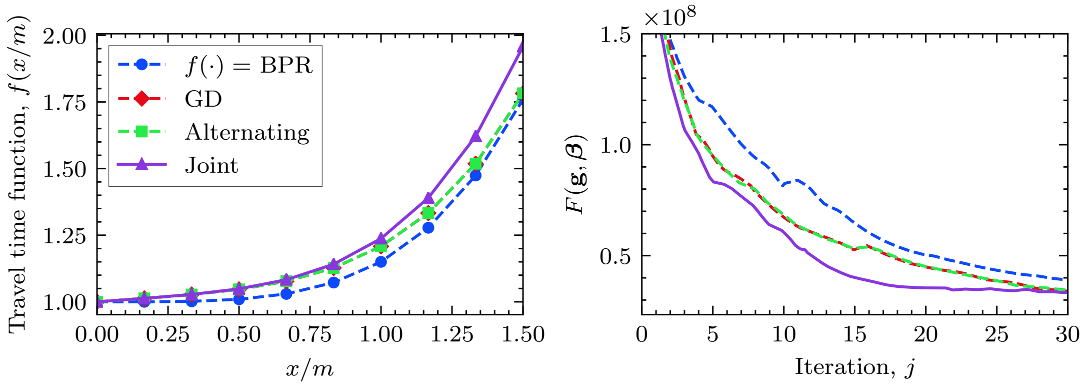


Figure 2·7: Case Study results for EMA

methods. Similar to the EMA case, these results confirm our understanding of the morning traffic pattern in NYC. Table 2.2 shows that node 10 is a favorite origin and destination. This node represents Midtown, one of the busiest neighborhoods in Manhattan due to high density of offices and businesses. Moreover, we see that nodes 1 and 20 are desired destinations. These nodes represent Wall Street and Lower Manhattan areas, respectively. This follows our intuition as the south of Manhattan (also known as the Financial District) it is densely populated with offices.

In summary, the numerical results reported in this Section suggest that Alg. 1

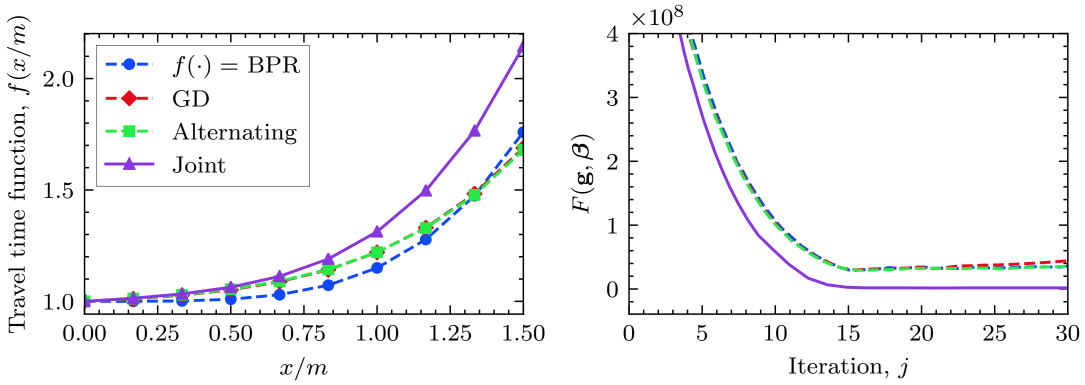


Figure 2·8: Case Study results for the NYC

works well in terms of reducing the objective function value of (2.18) while improving the estimation accuracy for the cost function and demands.

Table 2.2: Estimated OD demands (veh/h) for the NYC network.

O/D	1	4	20	10	17	21	25	26	sum
1	0.0	20.6	243.2	210.1	0.0	0.0	0.0	0.0	473.9
4	0.0	0.0	312.8	199.3	0.0	0.0	0.0	0.0	512.2
20	0.0	156.2	0.0	0.0	0.0	239.4	162.8	0.0	558.4
10	713.8	0.0	0.0	0.0	270.1	0.0	0.0	19.7	**1003.6**
17	0.0	86.4	0.0	0.0	0.0	185.5	140.5	0.0	412.4
21	0.0	86.4	0.0	0.0	0.0	0.0	140.5	0.0	226.9
25	0.0	0.0	0.0	52.2	0.0	0.0	0.0	41.1	93.3
26	0.0	0.0	0.0	52.2	0.0	0.0	0.0	0.0	52.2
sum	**713.8**	349.6	**556.0**	**513.9**	270.1	424.8	443.8	60.8	3332.8

2.6 Limitations

We acknowledge some limitations of our method that we believe can serve as a basis for future work, in particular we identify:

1. We cannot guarantee the adjusted OD demands to be close to the ground truth demand (since we are estimating $\mathbf{g}$ by closing the gap between $\mathbf{x}$ and $\mathbf{x}^*$) and the problem is non-convex.

2. In practice, the output of our algorithm would heavily depend on the initial demand data (since the joint problem is non-convex), as well as, on the accuracy

of the flow observations. Hence, good initial estimates are important for the success of recovering the true parameters.

3. The selection of $(c_{1j}, c_{2j}, d_{1j}, d_{2j})$ has a direct impact on both the closeness to the previous iterate and the algorithm's convergence rate which trades-off speed and accuracy. To overcome the parameter selection issue, we suggest that practitioners use cross-validation techniques or armijo-type rules when possible.

4. At each iteration we are solving a QP which can be solved in polynomial time. However, the requirement for memory increases exponentially with the size of the network and the number of OD pairs. Therefore, decomposition techniques for the QP should be explored to increase the computational efficiency of the method.

2.7 Summary and Future Work

In this chapter, we addressed the problem of estimating OD demands and cost functions jointly in a multi-class transportation network. We tackle this problem by rewriting the bilevel optimization problem (2.18) using the lower-level optimality conditions (2.22). To solve the resulting model, we propose an iterative approach (2.23) by relaxing some constraints and allowing a penalized gap to exist between the primal and dual objectives. To solve the joint problem we proposed a *trust-region feasible direction* and an *alternating* method described in Algs. 1 and 2, respectively.

Our results show that we can always reduce the objective function value of (2.18) to some extent, sometimes significantly, thanks to the the construction of the algorithms. However, the precision of the output highly depends on the inputs due to the non-convexity nature of the bilevel problem. We show empirically that there is value in jointly solving this problem as it reaches better estimates of the flows $\mathbf{x}$, cost function $f(\cdot)$, and OD demands $\mathbf{g}$. Nevertheless, we believe the GD and alternating methods are good alternatives when dealing with large networks. We hope this work nudges transportation experts to include cost function adjustments when estimating OD demands from data.

We identify potential future directions: First, to perform sensitivity analysis of link congestion metrics with respect to key quantities, such as link capacity and free-flow speed through the estimation of $f(\cdot)$ to enable the transportation agencies to prioritize congestion-reducing interventions (Houshmand, Wollenstein-Betech, & Cassandras, 2019; Wollenstein-Betech, Salazar, et al., 2021). Second, to integrate our algorithms to a dynamic OD demand estimation problem setting e.g., (Pitombeira-Neto, Loureiro, & Carvalho, 2020). The potential outcome would be to provide a more trustworthy method to predict the "Estimated Time of Arrival" more accurately. Third, it is possible to improve the running time of our algorithm by (i) utilizing previous iterations as starting feasible solutions for subsequent iterates; (ii) exploring better stopping criteria; (iii) implementing coordinate descent schemes and accelerated methods; (iv) employing more efficient data structures, and (v) aggregating flows at the link- or bush-level.

Chapter 3

The Effect of System-Optimal Routing in Mixed Traffic

In this chapter we study the problem of routing Connected and Automated Vehicles (CAV) in the presence of mixed traffic (coexistence of private vehicles and CAVs). We consider all centralized routed CAVs to belong to the same fleet seeking to minimize their overall traveling costs (travel time or energy consumption) while private vehicles (non-CAVs) choose their routes by selfishly minimizing their traveling times. Figure 3·1 shows a schematic representation of the problem. We propose an algorithm that deals with the interaction between CAVs and non-CAV and investigate the penetration rate effect of CAVs in the network. In addition to the methodological contribution, we assess the results using real traffic data from Eastern Massachusetts. Our data-driven results suggest that for a network of only CAVs, collaborative routing can improve the traveling times to up to 10%. Moreover, for lower penetration rates, we observe that not only the CAVs cost is reduced, but also the non-CAVs cost. Motivating the question on how policy-makers should incetivize the adoption of CAVs.

3.1 The Problem and Related Work

The *Price of Anarchy* (PoA) for traffic networks was first introduced by Koutsoupias and Papadimitriou (1999) and it is defined as the ratio between the social cost of the least efficient Nash equilibrium over the minimum achievable cost. Even if this term was introduced in 1999, this inefficiency have been studied, through examples,

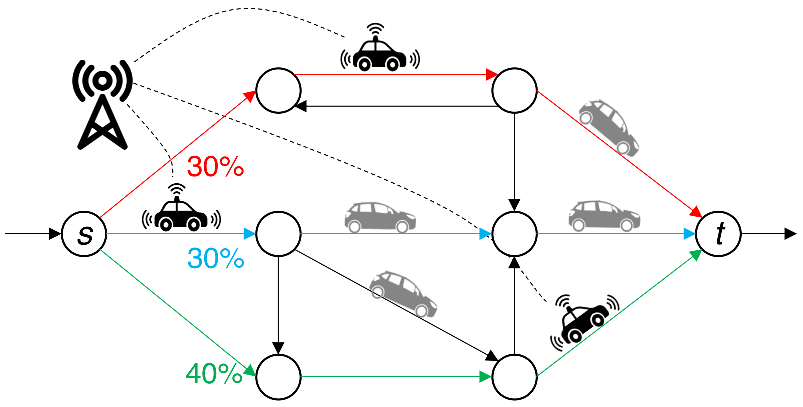


Figure 3·1: The centralized controller is assigning routes to CAVs traveling from s to t. Red, blue, and green numbers show the percentage of vehicles routed via that route.

since Pigou (1920) and later by Braess (1968). Later, Roughgarden and Tardos (2002) formally proved that the PoA in routing games with affine latency functions may never exceed 4/3 and for travel latency functions that are polynomials of degree n, the worst case grows order $\Theta(n/log(n))$. In addition, a recent work by Colini-Baldeschi, Cominetti, Mertikopoulos, and Scarsini (2020) shows that under heavy traffic conditions, the user-centric routing scheme becomes as efficient as the socially optimum one, i.e. PoA $= 1$.

Much of the work in this field focuses on showing the bounds of the PoA under different scenarios. However, few works have focused on estimating the PoA using real traffic data and on calculating the inefficiencies in mixed traffic, i.e., when only a percentage of the vehicles in the network are taking socially-optimal routes and the rest act selfishly.

For data-driven PoA estimation, Youn, Gastner, and Jeong (2008) reported that the PoA ranged between 1.2 and 1.3 (20%-30% inefficiencies) for the Boston/Cambridge, New York City, and London networks when the travel latency function is a polynomial of degree 10. In addition to this result, Grange, Melo-Riquelme, Burgos, González, and Raveau (2017) analyzed the transportation networks of Santiago, Chicago, and

Anaheim estimating inefficiencies of around 6% when using the classical BPR function.

In turn, analyzing socially-optimum routes in a mixed traffic setting is harder to perform due to the dependency of the decisions of the user-centric vehicles on the CAVs decisions, and vice versa. The study of the existence and uniqueness of Nash equilibria points for different sets of assumptions is studied in Harker (1988) and Yang, Zhang, and Meng (2007). In turn, other lines of research have defined a *Stackelberg game* to model the case where the CAVs objective is to minimize the total system travel time (instead of only the total travel time of their fleet). For the Stackelberg game case, the *leader* agent (in this case the CAVs) selects a routing policy anticipating the behavior of the *follower* (the user-centric vehicles). In the context of transportation networks, Korilis, Lazar, and Orda (1997) derived sufficient conditions to solve the *leader* (CAVs) problem when the network has *parallel links*. More recently, Lazar, Coogan, and Pedarsani (2017) and Mehr and Horowitz (2020) described that the PoA can be even larger by having CAVs choosing selfish routes. They argue that since CAVs will reduce the headways between vehicles using the cruise control, the congestion in certain links may be higher and therefore the system would experience higher travel times. To make such analysis, they require to use the assumptions of parallel links similarly as in the Stackelberg game setup in Korilis et al. (1997) Unlike these works, in this chapter we aim to find a Nash equilibria solution using the diagonalization approach explored in both Harker (1988) and Yang et al. (2007) to remove the assumption of parallel links and apply it to any network topology.

In this chapter, we use real traffic data and we leverage the methodology presented in chapter 2 to estimate both the travel latency functions and the OD demand matrices to come up with calculations for the PoA. Moroever, we use the diagnoalization scheme to observe the differences between the two classes of vehicles (CAVs and non-CAVS) and analyze the differences in terms of time and energy consumption

3.2 Models

We begin this section using a classical example introduced by Pigou (1920) to motivate the inefficiencies of selfish routing. Consider the network shown in Figure 3·2 and assume that every hour 10 vehicles are traveling from node s to t. If users take the top road, their travel time is equal to 10 regardless the number of vehicles in the road (think of it as a very wide highway). Alternatively, if users use the bottom road, they will experience a travel time proportional to the number of vehicles in the road (we could think of this as a shorter, but narrower bridge). Then, by having full information of the conditions, the worst travel time a commuter can experience by using the bottom link is 10. Hence, users the bottom road looking for the promise that someone will deviate and they have a shorter travel time. In this case, the Total System Travel Time (TSTT) is equal to 100 since 10 travelers are taking a route of 10 units.

In contrast, if a social planner could assign the routing decisions, the best allocation would be to send 5 passengers through the top road and 5 via the bottom road. In this case, the TSTT would be 75. The PoA which is defined as the ratio between these two strategies is then PoA $= \frac{\text{User-Centric TSTT}}{\text{System-Optimal TSTT}} = \frac{100}{75} = 4/3$.

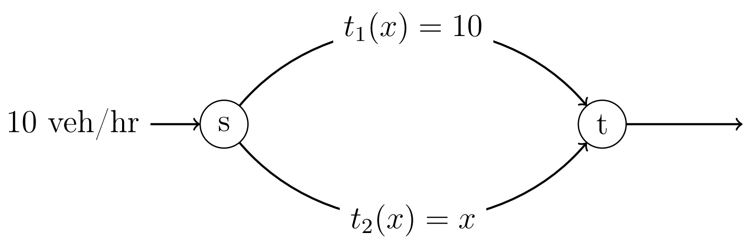


Figure 3·2: Pigou's example

We now introduce methodologies to obtain the time- and energy-based user-centric and system-optimal solutions for general networks and polynomial travel time functions.

3.2.1 Preliminaries

We model any traffic network as a directed graph $G = (\mathcal{V}, \mathcal{A}, \mathcal{W})$ where $\mathcal{V}$ is the set of nodes, $\mathcal{A}$ is the set of links, and $\mathcal{W} = \{\mathbf{w}_i : \mathbf{w}_i = (w_{si}, w_{ti}), i = 1, \ldots, K\}$ is the set of K Origin Destination (OD) pairs. We assume that all OD pairs start and end at one of the network's nodes. Let the node-link incidence matrix of G be $\mathbf{N} \in \{0, 1, -1\}^{|\mathcal{V}| \times |\mathcal{A}|}$, and the link-route incident matrix be denoted by $\mathbf{A}$. Let us define $d^w \geq 0$ as the demand rate for any OD pair $w \in \mathcal{W}$. Moreover, the route choice probability matrix is defined as $\mathbf{P} = [p_{ir}]$, where p_{ir} is the probability of taking route r while traveling through OD pair i. Let $\mathbf{g} = (g_i; i \in [\![\mathcal{W}]\!])$ be the OD demand vector.

Let us define the power-set of routes $\mathbf{R} = \{\mathcal{R}_i, i \in [\![\mathcal{W}]\!]\}$, where $\mathcal{R}_i$ is the set of allowable routes (more than one) for each OD pair i. Finally, the link-route incidence matrix is denoted by $\mathbf{A} = \{\alpha^i_{a,r}, i \in [\![\mathcal{W}]\!], r \in \mathcal{R}_i, a \in \mathcal{A}\}$ in which:

$$\alpha^i_{a,r} = \begin{cases} 1; & \text{if route } r \in \mathcal{R}_i \text{ uses link a} \\ 0; & \text{otherwise.} \end{cases}$$

Additionally, let A_i be the sub-matrix of $\mathbf{A}$ including the columns of $\mathbf{A}$ where $r \in \mathcal{R}_i$. The total flow is denoted by $\mathbf{x} = \{x_a; a \in \mathcal{A}\}$ where x_a is the flow on each link $a \in \mathcal{A}$.

3.2.2 System-Optimal Routing

We begin by assuming an all-CAV network for which a centralized controller can select the users routes. The system-centric problem consists of minimizing the total traveling time of CAVs in the network by coordinating the selection of routes.

The system-centric time-optimal problem is formulated as follows:

$$\min_{\mathbf{P}} \quad \sum_{a \in \mathcal{A}} t_a(x_a) x_a \tag{3.1a}$$

$$\text{s.t.} \quad \mathbf{x} = \mathbf{A}\mathbf{P}^T \mathbf{g}, \tag{3.1b}$$

$$\sum_{r \in \mathcal{R}_i} p_{ir} = 1, \quad \forall i \in [\![\mathcal{W}]\!], \tag{3.1c}$$

$$p_{ir} \in [0, 1], \quad \forall i \in [\![\mathcal{W}]\!],\ \forall r \in \mathcal{R}_i, \tag{3.1d}$$

where $t_a(x_a)$ is the traveling time of link a assumed to be

$$t_a(x_a) = t_a^0 \sum_{i=1}^{n} \beta_i (\frac{x_a}{m_a})^{(i-1)}, \tag{3.2}$$

where t_a^0 and m_a are the free flow travel time and flow capacity of a, respectively; $\boldsymbol{\beta} = (\beta_i, i = 1, 2, ..., n)$ is a vector of coefficients; (3.1b) is the flow assignment from OD routes to links; and (3.1c) enforces that sum of all the fraction of vehicles traveling through an OD pair is 1.

The decision variable is the routing-probability matrix $\mathbf{P} = [p_{ir}]$ that for each OD pair $i \in [\![\mathcal{W}]\!]$ assigns fractions of vehicles to allowable existing routes $r \in \mathcal{R}_i$ between any given OD pair. The inputs to the problem are the link-route incident matrix ($\mathbf{A}$), and the OD demand vector $\mathbf{g}$.

Note that instead of solving for individual links to follow for each vehicle, here we are assigning CAVs directly to routes between each OD pair. This transformation helps us reduce the decision space to select routes between OD pairs rather than finding link-based decisions. However, it also requires to defined a finite set of allowable routes per OD pair.

3.2.3 System-Optimal Routing in the Presence of Mixed Traffic

We address the system-centric time-optimal routing of CAVs in the presence of mixed traffic (CAV and non-CAV). In this case, only a portion of vehicles are CAVs and can be controlled through a centralized controller. As a result, instead of finding a routing scheme that minimizes total costs for all vehicles in the system, we focus on minimizing travel time for the CAV portion of traffic. We consider all CAVs belong to the same fleet (e.g. AMoD) and the fleet management company is trying to minimize total traveling time of the fleet.

To solve this problem we make four assumptions: (i) The non-CAV traffic flow is the solution to the user-centric routing approach; (ii) there exists a centralized controller which can route the CAV portion of traffic; (iii) We allow only to select between m for each OD pair in the network; (iv) travel time functions $t(\cdot)$ are monotonically increasing and continuously differentiable.

We define $\gamma \in [0,1]$ to be the fraction of demand that consists of CAVs. Relatedly, we define $\mathbf{P}_c = [p^c_{ir}]$ to be the route choice probability matrix for the CAVs and $\mathbf{g}_c = (g^c_i; i \in [\![\mathcal{W}]\!])$ the CAVs OD demand vector. Let us define $\mathbf{x}^c = \{x^c_a; a \in \mathcal{A}\}$ and $\mathbf{x}^{nc} = \{x^{nc}_a; a \in \mathcal{A}\}$ as the network flow of CAVs and non-CAVs, respectively. The optimization problem is then:

$$\min_{\mathbf{P}_c} \quad \sum_{a\in\mathcal{A}} t_a(x_a)x^c_a \tag{3.3a}$$

$$\text{s.t.} \quad \mathbf{x} = \mathbf{x}^c + \mathbf{x}^{nc}, \tag{3.3b}$$

$$\mathbf{x}^c = \mathbf{A}\mathbf{P}_c^T\mathbf{g}^c, \tag{3.3c}$$

$$\sum_{r\in\mathcal{R}_i} p^c_{ir} = 1, \quad \forall i \in [\![\mathcal{W}]\!], \tag{3.3d}$$

$$p^c_{ir} \in [0,1], \quad \forall i \in [\![\mathcal{W}]\!],\ \forall r \in \mathcal{R}_i. \tag{3.3e}$$

Constraint (3.3b) states that the total flow in the network is the summation of CAV's flow ($\mathbf{x}^c$) and non-CAV's flow ($\mathbf{x}^{nc}$). Notice that we are minimizing the travel time for the CAV share of traffic but evaluating the travel time function with the total flow on the link. As a result in (3.3a), the traveling time of each link which is a function of both CAVs flow and non-CAVs flow ($t_a(x_a)$), is multiplied by the flow of CAVs only (x^c_a). The inputs to the optimization problem are the link-route incident matrix $\mathbf{A}$, OD demand vector $\mathbf{g}$, and non-CAV flow $\mathbf{x}^{nc}$.

By solving Problem 3.3, we find optimal flows of CAVs over each OD pair (route-probability matrix $\mathbf{P}_c$). In other words, when a CAV enters the network at an origin O given its destination D, the algorithm gives it the desired socially optimal route to follow in terms of a sequence of links.

It is worth mentioning that the main difference between the system-centric problem and the user-centric problem relies on the objective functions. As stated in Section 2.4 of Patriksson (2004), the system-centric problem can be reformulated as a user-centric problem by slightly changing the travel cost function. Therefore, the results on the existence and uniqueness of the solution for the user-centric problem extend to the system-centric case. As a requirement for such a result we need positive and strictly increasing travel time functions on $\mathbf{P}$ which is achieved by having increasing polynomial functions.

3.2.4 System-Optimal Eco-Routing in the Presence of Mixed Traffic

We solve the eco-routing problem for a fleet of CAVs in the presence of mixed traffic. Eco-routing refers to the procedure of finding the most energy efficient route. This problem shares similar properties to (3.3) with the main difference that we minimize energy instead of time. As a result, we need an energy model to calculate energy consumption.

To accomplish this, we leverage two energy models that are functions of average speed which itself is a function of traffic flow. These models are suitable for Plug-in Hybrid Electric Vehicles (PHEVs) and for conventional vehicles. For a more detailed description of these energy models, please see Appendix E.

Eco-routing Problem Formulation for PHEVs

Following Houshmand and Cassandras (2018), the main feature of the eco-routing model is that it categorizes the consumption into three (although more can be specified) different modes based on the average speed: heavy traffic, medium traffic, and low traffic links. We then assign a *drive cycle or pattern* to each class based on the average speed (Table 3.1).

Table 3.1: Drive cycle assignment of each link

Traffic Mode	Average speed on the link (mph)	Drive Cycle
Heavy Traffic	[0,20]	NYC
Medium Traffic	[20,40]	UDDS
Low Traffic	≥ 40	HWFET

The eco-routing method seeks to find both the optimal routes, and optimal switching strategies between charge depleting (CD) and charge sustaining (CS) modes on each link. To model this, we consider two sets of decision variables: the CAV route-probability matrix, $\mathbf{P}_c = \{p^c_{ir}, i \in [\![\mathcal{W}]\!], r \in \mathcal{R}\}$ and the CD/CS switching strategy on each link, $\mathbf{Y} = \{y^i_{a,r}, i \in [\![\mathcal{W}]\!], r \in \mathcal{R}, a \in \mathcal{A}\}$. Here, $y^i_{a,r}$ represents the fraction of the link's length (l_a) over which we use CD mode. Thus, if we only use the CD mode over link $a \in \mathcal{A}$, then $y^i_{a,r}$ would equal to one[1].

Then, for a given vectorized OD demand of CAVs, $\mathbf{g}_c = (g^c_i; i \in [\![\mathcal{W}]\!])$, we can formulate the problem as follows:

$$\min_{\mathbf{P}_c,\mathbf{Y}} \quad \sum_{i\in[\![\mathcal{W}]\!]} \sum_{r\in\mathcal{R}_i} \sum_{a\in\mathcal{A}} \Big(c_{gas} \frac{l_a}{\mu^a_{cs}(v_a(x_a))}(1-y^i_{a,r}) + c_{ele}\frac{l_a}{\mu^a_{CD}(v_a(x_a))} y^i_{a,r}\Big) x^c_a \tag{3.4a}$$

$$\text{s.t.} \quad \sum_{a\in\mathcal{A}} \frac{\alpha^i_{a,r} y^i_{a,r} l_a}{\mu^a_{cd}} \leq E^{r,i}_0, \quad \forall r \in \mathcal{R}_i,\ \forall i \in [\![\mathcal{W}]\!], \tag{3.4b}$$

$$\mathbf{x} = \mathbf{x}^c + \mathbf{x}^{nc}, \tag{3.4c}$$

$$\mathbf{x}^c = \mathbf{A}\mathbf{P}^T_c \mathbf{g}^c, \tag{3.4d}$$

$$\sum_{r\in\mathcal{R}_i} p^c_{ir} = 1, \quad \forall i \in [\![\mathcal{W}]\!], \tag{3.4e}$$

[1]If one would like to solve this problem for Electric Vehicles (EVs), we only set $\mathbf{Y} = 1$. For Hybrid EVs and for conventional vehicles we set $\mathbf{Y} = 0$.

$$p_{ir}^c \in [0,1], \quad \forall i \in [\![\mathcal{W}]\!], \forall r \in \mathcal{R}_i \tag{3.4f}$$

$$y_{a,r}^i \in [0,1], \quad \forall a \in \mathcal{A},\ \forall i \in [\![\mathcal{W}]\!],\ \forall r \in \mathcal{R}_i, \tag{3.4g}$$

where c_{gas} and c_{ele} are the cost of gas, and electricity, respectively; $\mathbf{P}_c = [p_{ir}^c]$ is the route choice probability matrix for the CAV portion of traffic; and l_a is the length of link a. Constraint (3.4b) states that the total electrical energy used should not be more than the available energy in the battery pack at the start of that path ($E_0^{r,i}$). $\mu_{CS}(\cdot)$ and $\mu_{CD}(\cdot)$ are functions of velocity (Table E.1, and 3.1), and velocity is a function of flow: $v_a(x_a) = l_a/t_a(x_a)$. We assume that non-CAV flow ($\mathbf{x}_{nc}$) is known.

This model finds the eco-route for PHEVs. It is relevant noting that the energy model used in (3.4) is a piece-wise constant function of velocity. Hence, the problem is now non-convex and may have differentiability issues. To overcome this issue, we use a sigmoid function to smooth out the energy function and make it differentiable. However, the cost function still suffers from non-convexity and therefore, the optimizer may converge to a sub-optimal solution.

Eco-routing Problem Formulation for Conventional Vehicles

For the conventional vehicle case, we adopt the energy model proposed by Boriboonsomsin et al. (2012) (Appendix E). which models energy as a polynomial function of the link's average speed.

The difference between this formulation and (3.3) lies in that $t_a(x_a)$ is replaced with $e_a(x_a)$ (the average fuel consumption per mile). Employing this, the eco-routing problem for CAVs conventional vehicles is:

$$\min_{\mathbf{P}_c} \quad \sum_{a \in \mathcal{A}} c_{gas} l_a e_a(v_a(x_a)) x_a^c \tag{3.5a}$$

$$\text{s.t.} \quad \mathbf{x} = \mathbf{x}^c + \mathbf{x}^{nc}, \tag{3.5b}$$

$$\mathbf{x}^c = \mathbf{A}\mathbf{P}_c^T\mathbf{g}^c, \tag{3.5c}$$

$$\sum_{r\in\mathcal{R}_i} p_{ir}^c = 1 \quad \forall i \in [\![\mathcal{W}]\!], \tag{3.5d}$$

$$p_{ir}^c \in [0,1], \quad \forall i \in [\![\mathcal{W}]\!], \forall r \in \mathcal{R}_i \tag{3.5e}$$

where the average energy consumption in a link is modeled by

$$ln(e_a) = \sum_{i=0}^{4} \theta_i (v_a)^i + \theta_5 R_a,$$

and $\boldsymbol{\theta} = (\theta_i, i = 0, 1, ..., 5)$ is the energy cost coefficient (Table E.2).

3.2.5 Non-CAV Flow Modeling

One of the assumptions is the non-CAV flow is an input to our models. To do so, we model non-CAVs flow considering their reaction to CAVs decisions. To achieve this task, we assume non-CAVs act selfishly by minimizing their travel time and could be modeled using the classical *Traffic Assignment Problem* (TAP).

Since, this modeling framework has been already defined in Chapter 2 Section 2.3, we only write the TAP in terms of non-CAV flows and take into account the presence of the CAV flow in the network.

$$\min_{\mathbf{x}^{nc}} \quad \sum_{a\in\mathcal{A}} \int_{x_a^c}^{x_a^c + x_a^{nc}} t_a(s)ds \tag{3.6a}$$

$$\text{s.t} \quad \mathbf{x}^{nc} = \sum_{\mathbf{w}\in\mathcal{W}} \mathbf{x}^{nc,w}, \tag{3.6b}$$

$$\mathbf{N}\mathbf{x}^{nc,w} = \mathbf{d}^{nc,w}, \quad \forall \mathbf{w} \in \mathcal{W} \tag{3.6c}$$

$$\mathbf{x}^{nc,w} \geq \mathbf{0},$$

where $\mathbf{N}$ is the node-link incidence matrix; and $\mathbf{d}^{nc,w}$ and $\mathbf{x}^{nc,w}$ are the vector denoting the non-CAV demand and flow of OD pair w, respectively. This problem can be solved using many methodologies, a classical one is the Method of Successive Averages described in Appendix F.

3.2.6 Mixed Traffic Modeling

The basis of this methodology is that whenever CAVs change their routing decisions, non-CAVs adjust theirs and vice versa. To obtain the non-CAV flow for a given CAV penetration rate γ, we first consider only non-CAVs in the network and the OD demand of non-CAVs is given by:

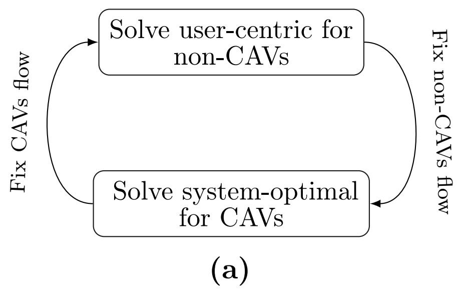
$$\mathbf{g}^{nc} = (1-\gamma)\mathbf{g} \tag{3.7}$$

Even though we choose a uniformly demand distribution for non-CAVs across OD pairs, without loss of generality, we can use any other given demand for both CAVs and non-CAVs.

Considering non-CAV demand $\mathbf{g}^{nc}$, we solve the user-centric routing problem which minimizes their travel time. In this respect, we use the *Method of Successive Averages* (MSA). After finding $\mathbf{x}^{nc}$, we solve the time optimal (3.3) or energy optimal (3.5) routing problem for CAVs demand:

$$\mathbf{g}^{c} = \gamma\mathbf{g} \tag{3.8}$$

Since non-CAVs were unaware of CAVs in the system while solving the TAP, we re-solve the problem considering CAV flow on each link. Hence, we re-iterate by considering CAVs solution $\mathbf{x}^{c}$. Furthermore, the TAP is solved again for non-CAVs. Re-iteration of this process continuous until convergence (Figure 3·3).


Solve user-centric for non-CAVs
Solve system-optimal for CAVs
Fix CAVs flow
Fix non-CAVs flow
(a)

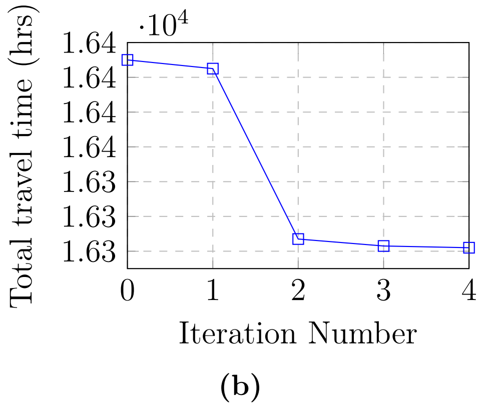


Figure 3·3: (a) Procedure for solving the system-centric routing problem ; (b) Convergence plot for iterating through TAP and social problem

3.3 Numerical Results

We perform three case studies. First, by leveraging speed data and the estimates of the OD demand we estimate the PoA for the Eastern Massachusetts network (Figure 3·10b). Second, we study the effect of the CAV penetration rate on the total time savings and energy savings on the Braess network (Figure 3·5). Third, we apply the framework to the Eastern Massachusetts network. For finding the energy optimal routes we assume the road grade is zero ($R_a = 0$ and we assume the cost of gas to be 2.75 \$/gal. We solve the NLP problems using IPOPT solver (Wächter & Biegler, 2006) in Julia (Bezanson, Edelman, Karpinski, & Shah, 2017).

For the eco-routing case, we only show the results for conventional vehicles using the energy model discussed in Appendix E. This is because the other energy model (CD/CS) introduces non-convexity to the NLP problem and it is challenging to solve the problem using commercially available optimizers. As mentioned earlier, eco-routing results are sensitive to the energy model employed. Given a more accurate, convex, smooth and differentiable model may obtain different results. Hence, the eco-routing results shown in this chapter should only be considered as preliminary results which show the potentials of saving energy using centralized routing of CAVs.

3.3.1 Price of Anarchy in Eastern Massachusetts

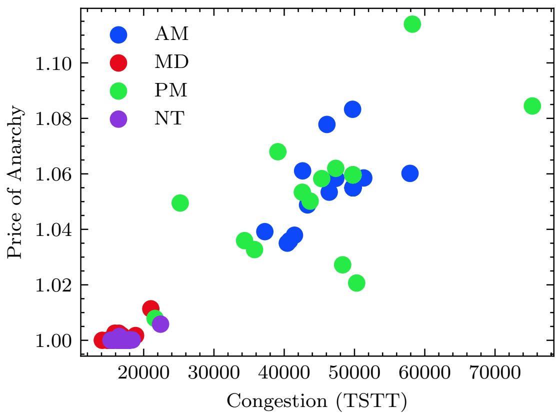
We tested the models for 100% and 0% CAVs penetration rates using real traffic speed data for the Eastern Massachusetts road network. The data was captured by a leading private company on traffic data called INRIX and the dataset was provided to us by the Central Transportation Planning Staff (CTPS) of the Boston Metropolitan Planning Organization (MPO). In Appendixes B.0.1 and C we describe dataset and the preprocessing steps used to generate this analysis.

For our experiments, we use data of April 2012 and 2015 and we separate this into four different time periods: the morning peak (AM) from 7:00-9:00 hrs, midday (MD) from 11:00-13:00 hrs, afternoon peak (PM) from 17:00-19:00 hrs and night (NT) from 21:00-23:00 hrs[2]. For each of these periods, we estimated the OD using the Generalized Least Square method Hazelton (2000) and described in Appendix C.0.5. Once the OD demand is estimated, and together with the BPR function (3.2) with $\boldsymbol{\beta} = (1, 0, 0, 0, 0.15)$, we solve the time-optimal user-centric (3.6) and system-optimal (3.1) problems. For every solution of these two methods, we compute the PoA.

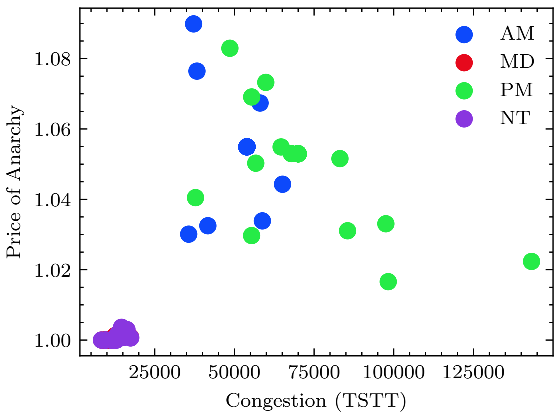
Figure 3·4 shows the PoA for the different days on April as a function of the level of congestion of the network approximated by the Total System Travel Time (TSTT). The results report that although on average April 2015 was more congested than April 2012 (see the x-axis), the PoA was not necessarily higher. We believe this is due to the fact that if congestion increases too much, the PoA decreases to one (Colini-Baldeschi et al., 2020). In fact, from both plots we can infer that for this network the PoA is maximized when the TSTT is around 60,000. Finally, we see that the PoA is higher for the morning and afternoon peak periods, suggesting that socially routing decisions are particularly useful at these times. These results promises that implementing system-optimal routing techniques in transportation systems could

[2]For more details of the data, we have created an interactive web-based visualization interface available in: https://salomonw.github.io/congestionmaps/DynamicPage/AM/

improve overall travel times between 0% and 10%.



(a) 2012



(b) 2015

Figure 3·4: Price of Anarchy as a function of the level of congestion of the network for different days in April 2012 and 2015

3.3.2 Braess' Network Example

To demonstrate how system-optimal routing of CAVs can affect both CAVs and non-CAVs, we perfrom a toy experiment with the Braess' network. Note that in this case instead of using the BPR function, we use the travel time functions shown on each link of Braess' network in Figure 3·5. We consider a demand of 4000 veh/hr traveling from node 1 to 4, the lengths of links 1,2,3 and 5 equal to 30.5 miles and the length of link 4 equal to zero. Time-optimal results are shown in Figure 3·6, in which we compare traveling time of CAVs with non-CAVs under different penetration rates. The energy cost for traveling through the optimal routes are also shown in Figure 3·6b. In addition, we report the traveling time of CAVs and non-CAVs under different penetration rates using the case of 0% CAVs as a baseline and reported the time savings in Figure 3·6c.

As shown in Figure 3·6, introducing CAVs into the system not only improves the time saving of CAVs, but also helps non-CAVs to save time. This is because smart routing decisions of CAVs reduce the traffic intensity in the highly congested roads, which consequently help non-CAVs to travel faster. As we inject CAVs to the system, we see that travel time (as well as energy cost) per vehicle of CAVs starts decreasing compared with the uncontrolled traffic. Moreover, traveling time of commuting through the fastest route decreases as we inject more CAVs to the system.

It is interesting to see that when a small percentage of CAVs are in the system, there is no improvement for anyone. This happens because CAVs are optimizing for themselves and since their fraction is not sufficient to change the network conditions, their decision is to act selfishly. However, as we increase the penetration rate, CAVs create a more balanced flow distribution in the network in which both CAVs and non-CAVs can benefit from. In the Braess network example, it can be seen that if all the cars in the system are replaced with CAVs, we can save 18.9% in terms of travel time. This value is often referred to as the Price of Anarchy (PoA).

In addition to the time-optimal case, we also solve the eco-routing problem. Figure 3·7 shows the same trend as time optimal results. In other words, centralized eco-routing of CAVs can benefit both CAVs and non-CAVs. The maximum energy savings happens at the 100% CAV penetration rate and is equal to 19.1%.

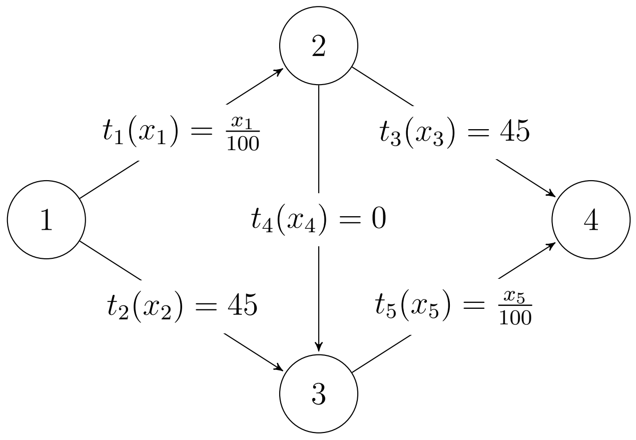
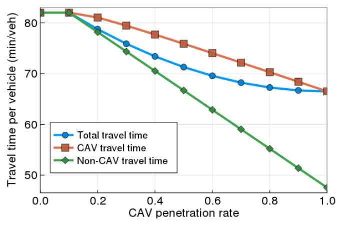
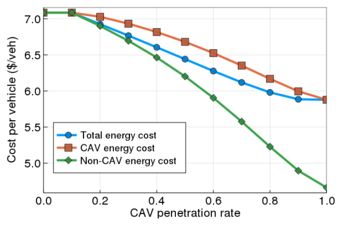


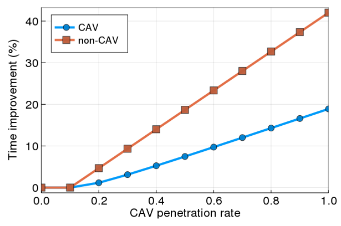
Figure 3·5: Simple 5 link directed graph (Braess's network)



(a) Traveling time comparison results

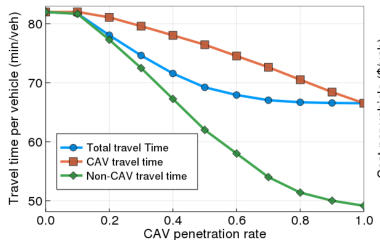


(b) Energy cost comparison results

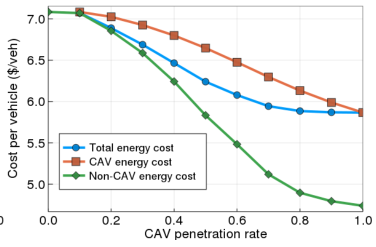


(c) Travel Time improvement results

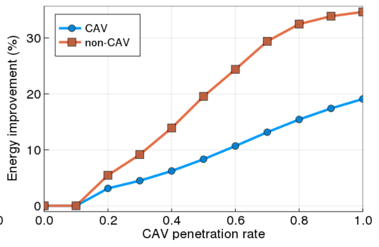
Figure 3·6: Braess network time optimal routing results under different penetration rates.



(a) Traveling time comparison results



(b) Energy cost comparison results



(c) Energy cost improvement results

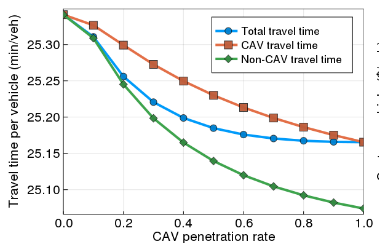
Figure 3·7: Braess network eco-routing (energy optimal) routing results under different penetration rates.

3.3.3 EMA Interstate Highways Network

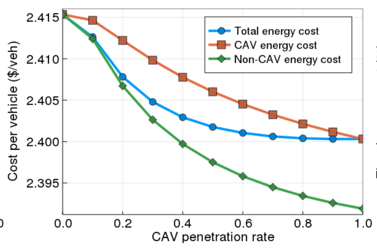
To obtain more realistic results, we perform a data-driven case study using the actual traffic data from the Eastern Massachusetts (EMA) road network. This data was collected by INRIX and was provided to us by the Boston Region MPO. The subnetwork including the interstate highways of EMA (Figure 3·10b) is chosen for the case study.

For this network, we use the OD demand which has been calculated using an inverse optimization framework in J. Zhang et al. (2018). This OD demand consists of 56 OD pairs and we allow up to 3 routes between each origin and destination (top 3 shortest routes in distance).

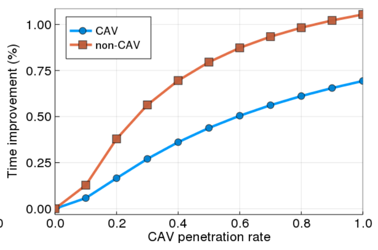
Time optimal and energy optimal results are shown in Figures 3·8 and 3·7, respec-



(a) Traveling time comparison results

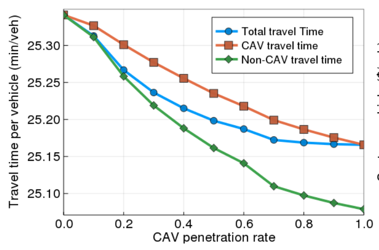


(b) Energy cost comparison results

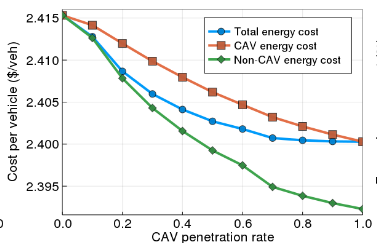


(c) Travel Time improvement results

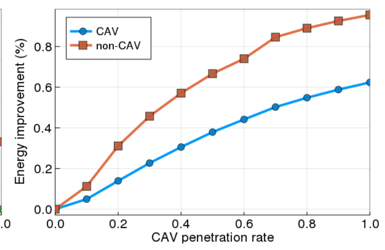
Figure 3·8: EMA network time optimal routing results under different penetration rates.



(a) Traveling time comparison results



(b) Energy cost comparison results

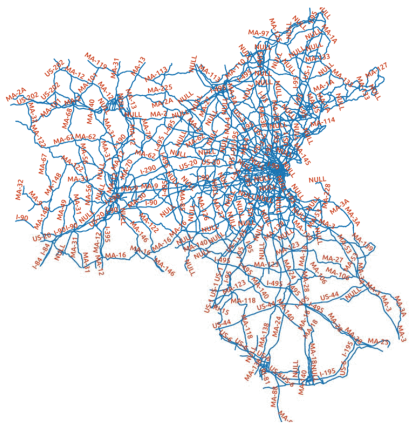


(c) Energy cost improvement results

Figure 3·9: EMA network eco-routing (energy optimal) routing results under different penetration rates.

tively. The results follow the same behavior as the results of Braess example. We again see that as the CAV penetration rate increases, both CAVs and non-CAVs benefit from optimal routing decisions of non-CAVs. Notice that in Figure 3·8 the differences of the improvement are very small. This is because for the specific scenario that we analyzed, the PoA was very small, i.e., ratio between 100% and 0% penetration rate. However, we obtained a similar shape for other values in which the PoA is higher. For example, for cases where the PoA is 1.08 as shown in Figure 3·4.

Figure 3·11a reports the time improvement of different OD pairs with their corresponding OD demand for 100% and 50% CAV penetration rates. It is interesting to see in Figure 3·11a that two OD pairs with relatively high demand are being affected by -0.8% and -1.3% for 50% and 100% γ's respectively. However, we see a


(a)

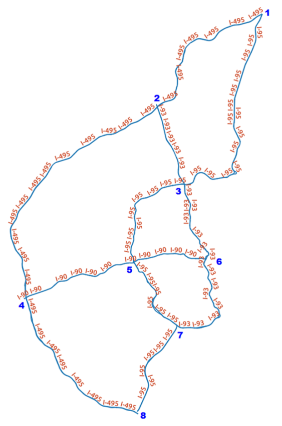
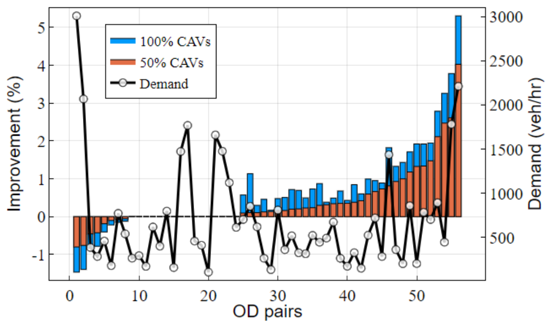
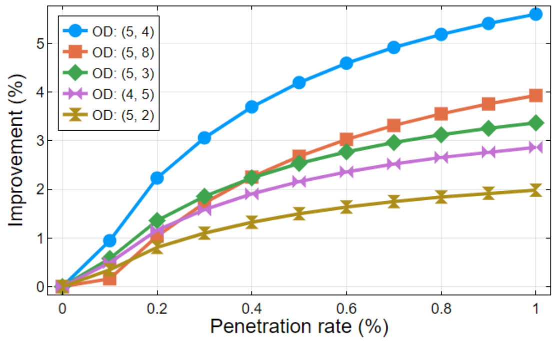

(b)

Figure 3·10: (a) All available road segments in the road map of Eastern Massachusetts J. Zhang et al. (2018) ; (b) Interstate highway sub-network of eastern Massachusetts

5% benefit on most of the OD pairs. In this manner we are able to identify which OD pairs are getting worse and which ones are improving. This gives the opportunity to better understand the dynamics of the network and develop pricing and incentive mechanisms.



(a)



(b)

Figure 3·11: (a) Time saving improvement of different OD pairs for 50% and 100% CAV time optimal routing (EMA network) ; (b) Travel time improvement of different OD pairs as a function of CAV penetration rate (EMA network)

3.4 Summary and Future Work

In this chapter, we proposed system-centric optimal routing algorithms for a fleet of CAVs in the presence of mixed traffic. We consider two objectives for routing: (1) minimizing travel time (2) minimizing energy consumption cost. Moreover, in order to model the routing behavior of regular vehicles, we consider they make selfish decisions by minimizing their own travel time. Then, by iteratively solving the TAP, and finding optimal routes for CAVs we estimate non-CAV flow in the network. Historical traffic data and a theoretical example were used to validate the models. The results suggest that optimal routing of CAVs can not only benefit CAVs, but the smart routing decision of CAVs helps ease traffic congestion in the network which helps regular vehicles as well. Additionally, we empirically showed that even a small CAV penetration rate has significant impact on the overall traveling cost of the network.

As of future work, we see three main directions. First, to implement these algorithms in real-time and model it using a microsimulator. This will help understand how this static planning framework compares with a more granular microscopic model. Second, to increase the number of fleets in the network. So far, we the method only considers two, but it may be worth exploring the results for more fleets. Finally, exploring pricing and incentives methodologies to steer the behavior of selfish users is of high relevance for the implementation of such routing policies.

Chapter 4

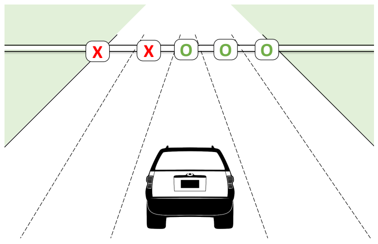
Optimizing Lane Reversals

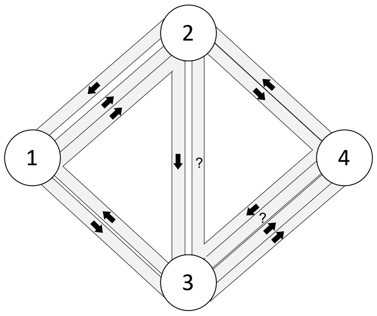
This chapter studies how to reduce the overall travel time of commuters in a transportation network by reversing the direction of some lanes in the network. This problem has been shown to be NP-hard given the dependence of the users' route selection on the lane direction decision. Herein, we propose and compare three efficient methodologies to solve the routing and lane reversal problem jointly.

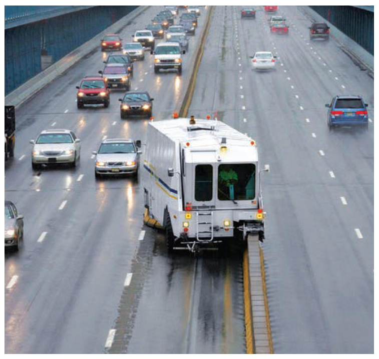
First, we introduce an alternating method that decouples the routing and lane assignment problems. Second, we propose a Frank-Wolfe method that jointly takes gradient steps to adjust both the lane assignment and routing decisions. Third, we propose a convex approximation method that uses a threshold-based approach to convexify the joint routing and lane reversal problem. Using the convex approximation, we extend the main formulation to limit a maximum number of reversals, as well as to incorporate multiple origin-destination (OD) patterns and we present extensive experiments using the Eastern Massachusetts transportation network.

4.1 The Problem and Related Work

Unfortunately, lane reversals have been applied in practice in very limited instances. In particular, they have been carried out for emergency evacuations (e.g., Hurricane Florence in South Carolina (Fausset, 2018)), or after the completion of an event (e.g., NASCAR races in New Hampshire (Arcand, 2019)). For both of these cases, the reversals must be well communicated to drivers and carefully planned before being







(a) **(b)** **(c)**

Figure 4·1: (a) Illustration of a typical lane reversal signal. (b) Diagram of the Braess' network with specific number of lanes and directions. (c) Barrier transfers (or road zippers).

implemented. Hence, communicating the status of the infrastructure to drivers is a major challenge when applying lane reversals.

Current lane reversals systems built to reduce traffic congestion indicate the direction of the reversible lanes with overhead signals or road zippers, see Figures 4·1a and 4·1c, respectively. Therefore, the infrastructure manager has to provide very structured (not flexible) schedules to avoid driver confusion. In addition to communication, the attractiveness of lane reversals has been also curtailed by the poor human ability to respond to a coordinated change. This has led to traffic gridlocks and to an increase in traffic accidents (Martínez, 2021).

Propitiously, the rise of Connected and Automated Vehicles (CAVs) can address such limitations by their ability to communicate with the infrastructure. Therefore, CAVs will enable the possibility to implement lane reversals more aggressively, either by doing it for more roads, or by dynamically changing the direction of a single road more often. From the perspectives of the traffic engineer, assessing the benefits of lane reversals and understanding which roads to revert is quite important.

4.1.1 Related Work

Most of the literature concerned with contraflow lane reversals has concentrated on two main applications: (i) reversing lanes for evacuation routing during emergencies and (ii) alleviating traffic congestion.

For evacuation planning, the problem has been studied using simulation and network flow models. Simulation methods by Jha, Moore, and Pashaie (2004) and Theodoulou and Wolshon (2004) showed that evacuation route capacity can be improved by 53% and 73%, respectively. Different than simulation, network flow models by Cova and Johnson (2003) formulate the problem as a mixed-integer program and use a generic solver to find a feasible solution. Their numerical results suggests evacuation time improvements in Salt Lake City of 30% to 40%. Similarly, Zhao, Feng, Li, and Bernard (2016) and Xie and Turnquist (2011) uses Tabu search to solve the problem and Kim, Shekhar, and Min (2008) showed that the network flow lane reversal problem is NP-complete and suggested a greedy heuristic algorithm to solve it.

The research concerned with reducing traffic congestion has focused on reversing lanes for a single bottleneck road (e.g., tunnels and bridges) or for the full network. For single roads, rule-based (Ampountolas, dos Santos, & Carlson, 2019) and fuzzy controllers (Xue & Dong, 2000; Zhou et al., 1993) have been studied and they typically rely on the fundamental diagram of traffic flow (Lighthill & Whitham, 1955). Moreover, with the advent of CAVs, a lane-free approach has been recently proposed where a fluidic boundary is dynamically adapted depending on the number and type of vehicles present in a road (Malekzadeh, Papamichail, Papageorgiou, & Bogenberger, 2021). For networked problems, its canonical mathematical representation is an Integer Linear Program (ILP) which is generally NP-hard. To tackle it, Chu, Lam, and Li (2019) uses a distributed alternating direction method of multipliers (ADMM) to decompose the problem into smaller instances which are still integer programs. Their numerical results show that travel times for a small subnetwork of New York City could be improved by 61%. In addition, Hausknecht, Au, Stone, Fajardo, and Waller (2011) and Meng and Khoo (2008) solve the networked reversal problem using genetic algorithms and report increases in efficiencies on the order of 72% for the city of Austin. Note that both models do not have guarantees on finding an optimal solution. Finally, Levin and Boyles (2016) used a microscopic cell-transmission model for which they solved the lane reversal problem using a heuristic method based on congestion estimates showing a 21.8% reduction in total system travel time.

The rest of the chapter is structured as follows. In Section 4.2 we present the model preliminaries and the problem formulation. In Section 4.3 we introduce the three main methodologies employed to solve the lane assignment problem. Section 4.4 discusses extensions of our model. In particular, constraining the model to a maximum number of reversals and employing the model for multiple origin-destination demand patterns. Section 4.5 reports numerical results of the different methods over a case study using the Eastern Massachusetts Area (EMA) and a smaller test network, and Section 4.6 concludes.

4.2 Model and Problem Formulation

4.2.1 Preliminaries

We represent the transportation network using a directed graph denoted by $\mathcal{G}$ and composed by the set of *nodes* $\mathcal{V}$ and the set of *links* $\mathcal{A}$. We assume $\mathcal{G}$ to be *strongly-connected* such that every node is reachable from any other node in the network, and let the node-arc incidence matrix of $\mathcal{G}$ be $\mathbf{N} \in \{0, 1, -1\}^{|\mathcal{V}|\times|\mathcal{A}|}$. For every link (i, j) in $\mathcal{A}$, we denote its number of assigned lanes to be z_{ij}; its *capacity per lane* be c_{ij}; and its total capacity be $m_{ij} = z_{ij}c_{ij}$ expressed in vehicles per hour. To establish the relationship between *opposite direction* links, i.e., $(i,j),(j,i) \in \mathcal{A}$, we let n_{ij} be the maximum number of lanes in a *road* (i.e., n_{ij} is the maximum number of lanes a link can have if all lanes in the road are facing the same direction). Hence, $n_{ij} = n_{ji}$.

To model the user trips, let K be the number of Origin-Destination (OD) pairs and $\mathcal{W} = \{\mathbf{w}_k = (s_k, t_k) : k = 1, \dots, K\}$ the set of all OD pairs. For every $\mathbf{w}_k$, let the demand rate of trips (veh/hr) from its source node $s_k \in \mathcal{V}$ to the target node $t_k \in \mathcal{V}$ be $d_{\mathbf{w}_k} \geq 0$.

In addition to user demand, we use $x_{ij}^{\mathbf{w}_k}$ to track the flow in every link $(i,j) \in \mathcal{A}$ associated with OD pair $\mathbf{w}_k$. Moreover, let x_{ij} represent the (aggregated) total link flow in (i,j), i.e.,

$$x_{ij} = \sum_{k=1}^{K} x_{ij}^{\mathbf{w}_k}, \qquad \forall (i,j) \in \mathcal{A}, \tag{4.1}$$

and let $\mathbf{x} = (x_{ij};\ (i,j) \in \mathcal{A})$ be the vector of all link flows.

To quantify the travel times in every link, let $t_{ij}(x_{ij}, z_{ij}) : \{\mathbb{R}_{\geq 0}, \mathbb{N}_{\geq 0}\} \mapsto \mathbb{R}_{\geq 0}$ be the *latency cost* (or travel time) function of link (i,j) which depends on the link's flow and on the number of lanes assigned to the link. Using the same structure as in Beckmann et al. (1955), we characterize $t_{ij}(x_{ij}, z_{ij})$ as:

$$t_{ij}(x_{ij}, z_{ij}) = t_{ij}^0 f\Big(\frac{x_{ij}}{c_{ij} z_{ij}}\Big), \tag{4.2}$$

where $f(\cdot)$ is a strictly increasing, positive, and continuously differentiable function, and t_{ij}^0 is the free-flow travel time on arc (i,j). We set $f(0) = 1$, which ensures that if there is no constraint on the arc's capacity, the travel time t_{ij} equals the free-flow travel time t_{ij}^0. A widely used function by transportation engineers is the *Bureau of Public Roads (BPR)* function (Traffic Assignment Manual, 1964)

$$f(x_{ij}/m_{ij}) = 1 + \alpha (x_{ij}/m_{ij})^{\beta}. \tag{4.3}$$

A common choice is $\alpha = 0.15$ and $\beta = 4$. An extensive discussion on how to estimate these functions using flow data is in Wollenstein-Betech, Sun, Zhang, and Paschalidis (2019) and in chapter 2 of this dissertation. Finally, we let the vector of travel latency functions be $\mathbf{t}(\mathbf{x}, \mathbf{z}) = (t_{ij}(x_{ij}, z_{ij});\ \forall (i,j) \in \mathcal{V})$.

4.2.2 LASO-TAP Problem

Using the definitions presented above, we formulate the *Lane-Assignment System-Optimal Traffic Assignment Problem* (LASO-TAP) as follows:

$$\min_{\mathbf{x},\mathbf{z}} \quad \mathbf{t}(\mathbf{x}, \mathbf{z})'\mathbf{x} \tag{4.4a}$$

$$\text{s.t.} \quad \sum_{i:(i,j)\in\mathcal{A}} x_{ij}^{\mathbf{w}_k} - \sum_{\ell:(j,\ell)\in\mathcal{A}} x_{j\ell}^{\mathbf{w}_k} = \begin{cases} -d_{\mathbf{w}_k}, & \text{if } j = s_k, \\ 0, & \text{if } j \neq s_k, t_k, \\ d_{\mathbf{w}_k}, & \text{if } j = t_k, \end{cases}$$
$$\forall k = 1, \dots, K,\ \forall j \in \mathcal{V}, \tag{4.4b}$$

$$z_{ij} + z_{ji} \leq n_{ij}, \quad \forall (i,j) \in \mathcal{A}, \tag{4.4c}$$

$$\mathbf{x}^{\mathbf{w}_k} \in \mathbb{R}_+^{|\mathcal{A}|}, \quad \mathbf{z} \in \mathbb{N}_+^{|\mathcal{A}|}, \tag{4.4d}$$

where in the objective (4.4a) we are minimizing the overall travel time; constraint (4.4b) ensures that $\mathbf{x}$ is a feasible vector which complies with demand satisfaction and conservation of flow; constraint (4.4c) ensures that the number of assigned lanes does not exceed the maximum number of available lanes; and (4.4d) restricts $\mathbf{z}$ to be a vector of integer variables.

Origin-based formulation

The total number of variables introduced in the LASO-TAP formulation (4.4) is $|\mathcal{A}|(|\mathcal{W}| + 1)$, which is typically dominated by the number of OD pairs $|\mathcal{W}|$. In practice, this number can be very large, sometimes up to $|\mathcal{V}|^2$. Hence, solving (4.4) for real networks require large memory capabilities. To mitigate this problem, we aggregate flows by origin, similar to Rossi, Iglesias, Alizadeh, and Pavone (2020). This reduces the number of variables to $|\mathcal{A}|(1+|\mathcal{V}|)$, which significantly improves the computation time.

Let the set of origin (sources) be $\mathcal{O} = \{s_k : \ d_{\mathbf{w}_k} > 0, \ k = 1, \ldots, K\}$ and let the flow vector with $o \in \mathcal{O}$ as it source to all possible destinations be $\mathbf{x}^o$. The total user flow on a link is then $\mathbf{x} = \sum_{o \in \mathcal{O}} \mathbf{x}^o$ and we define the set of user origin-link variables to be $\mathbf{x}^{\mathcal{O}} = \{\mathbf{x}^o : \ o \in \mathcal{O}\}$. For every origin $o \in \mathcal{O}$ and every node $j \in \mathcal{V}$, let ϕ_{oj} be the node *imbalance* describing its excess demand or supply. This is:

$$\phi_{oj} = \begin{cases} \sum\limits_{t_k : \mathbf{w}_k \in \mathcal{W}} -d_{\mathbf{w}_k}, & \text{if } j = o, \\ 0, & \text{if } (o,j) \notin \mathcal{W}, \\ d_{\mathbf{w}_k}, & \text{if } j = t_k \text{ and } s_k = o. \end{cases} \tag{4.5a}$$

With these definitions, we can formulate the *origin-based* LASO-TAP as follows:

$$\min_{\mathbf{x}^{\mathcal{O}}, \mathbf{z}} \quad \mathbf{t}(\mathbf{x}, \mathbf{z})'\mathbf{x} \tag{4.6a}$$

$$\text{s.t.} \quad \sum_{i:(i,j) \in \mathcal{A}} x_{ij}^o - \sum_{\ell:(j,\ell) \in \mathcal{A}} x_{j\ell}^o = \phi_{oj} \quad \forall j \in \mathcal{V}, \ \forall o \in \mathcal{O}, \tag{4.6b}$$

$$z_{ij} + z_{ji} \leq n_{ij}, \quad \forall (i,j) \in \mathcal{A}, \tag{4.6c}$$

$$\mathbf{x}^{\mathbf{w}_k} \in \mathbb{R}_+^{|\mathcal{A}|}, \quad \mathbf{z} \in \mathbb{N}_+^{|\mathcal{A}|}. \tag{4.6d}$$

Despite the reduction from (4.4) to (4.6), the LASO-TAP remains hard to solve for several reasons. First, the interaction between the decision variables in (4.4a) and (4.6a) makes the objective non-convex. This comes from the fact that when multiplying (4.3) by x_{ij} we get the term $\gamma_{ij} x_{ij}^{\beta+1}/z_{ij}^{\beta}$ where $\gamma_{ij} = \alpha t_{ij}^0 / c_{ij}^{\beta}$. Second, we are optimizing over a set of integer variables $\mathbf{z}$ which increases the computational complexity of the problem. Still, (4.6) reduces the dimensionality of (4.4) and makes the problem more manageable to solve. In the following sections we present modifications to this problem formulation to enable efficient solution methods.

It is important to note that the problem as stated in (4.4) and (4.6) is a lane *assignment* problem rather than a lane *reversal* problem. This distinction is negligible when we are capable to reverse *all* lanes in the network. However, in reality the transportation infrastructure is not very flexible and might not be able to handle too many lane reversals. Hence, we are interested in considering a *sparse* lane assignment problem in which we limit the number of lane reversals. We develop this extension in Section 4.4.

Remark 3. *Note that the LASO-TAP as stated in* (4.4) *and* (4.6) *is seeking a System-Optimal (SO) solution to the Traffic Assignment Problem (TAP). We can expect a SO behavior when vehicles collaborate with each other when deciding their routes which is for CAVs in chapter 3. However, by slightly modifying the objective function, the same algorithms that solve a SO TAP are capable of solving the User-Centric (UC) TAP (Patriksson, 1994). In this chapter we will focus on the SO case, but our framework can handle both SO and UC TAPs.*

4.3 Methods

We have discussed how the LASO-TAP problem is hard to solve for several reasons. First, the objective is non-convex. Second, the optimization is made over a set of integer variables $\mathbf{z}$. Thus, let us consider three main strategies to find effective solutions to the problem. To describe these strategies, we first define a condition that will help us identify if a current solution is at an *equilibrium point* or not.

Intuitively, we say that Problem (4.6) is at an equilibrium point if the objective does not improve when we perform a single reversal in the network. We now state this formally.

Definition 3 (Equilibrium)**.** *An equilibrium point of problems* (4.4) *and* (4.6) *is found* *if for all* $(i,j) \in \mathcal{A}$

$$\begin{aligned} \psi_{ij} &\geq 0 \quad \forall z_{ij} = 0, \ldots, n_{ij} - 1, \ \textit{and} \\ \psi_{ij} &\leq 0 \quad \forall z_{ij} = n_{ij}, \end{aligned}$$

where ψ_{ij} *is defined as*

$$\begin{aligned} \psi_{ij} = &\big(x^+_{ij} t_{ij}(x^+_{ij}, z_{ij}+1) + x^+_{ji} t_{ji}(x^+_{ji}, z_{ji}-1)\big) - \big(x_{ij} t_{ij}(x_{ij}, z_{ij}) + x_{ji} t_{ji}(x_{ji}, z_{ji})\big) \\ &+ \sum_{(k,l)\in\mathcal{A}/\{(i,j),(j,i)\}} \Big(\big(x^+_{kl} t_{kl}(x^+_{kl}, z_{kl}) + x^+_{lk} t_{lk}(x^+_{lk}, z_{lk})\big) - \big(x_{kl} t_{kl}(x_{kl}, z_{kl}) + x_{lk} t_{lk}(x_{lk}, z_{lk})\big)\Big), \end{aligned}$$

and x^+_{ij} *indicates the modified flows when we reverse a lane in* (i,j)*.*

To check whether the equilibrium condition holds or not, we have to run a TAP to get x^+_{ij}. Therefore, to evaluate the vector $\boldsymbol{\psi} = (\psi_{ij}; (i,j) \in \mathcal{A})$ we require solving $|\mathcal{A}|/2$ TAPs. Checking this condition in a sequential algorithm is computationally expensive, but could be used regularly as a measure of closeness to an equilibrium point. Note that many equilibrium points may be encountered and that assessing the global optimality of such a point may be a difficult, and often an intractable task.

In the following subsections we present three families of algorithms that could be used to efficiently solve the LASO-TAP problem. First, we introduce an alternating method consisting of decoupling the lane and traffic assignment problems. Then, we provide a *feasible direction* or *Frank-Wolfe* algorithm. This framework reduces the complexity by solving a fluidic version of the problem. Finally, we present a novel convex approximation to the problem that can be solved to optimality.

4.3.1 Alternating method

We consider the method of sequentially and iteratively solving two disjoint and easier problems. The idea consists of first solving the traffic assignment problem (TAP) using any of the standard methods (e.g., Method of Successive Averages, Frank-Wolfe, among others) and then solving for the best *lane assignment* (LA) for those specific flows. With the new lane allocation, we re-solve the TAP and LA problems and repeat this procedure until convergence.

The LA problem of allocating lanes for a fixed flow vector $\mathbf{x}$ is defined as

$$\min_{\mathbf{z}} \quad J^{\mathbf{z}}(\mathbf{z}) := \sum_{(i,j)\in\mathcal{A}} x_{ij} t_{ij}(x_{ij}, z_{ij}) \tag{4.7a}$$

$$\text{s.t.} \quad z_{ij} + z_{ji} \leq n_{ij}, \quad \forall (i,j) \in \mathcal{A}, \tag{4.7b}$$

$$\mathbf{z} \in \mathbb{N}_{+}^{|\mathcal{A}|}, \tag{4.7c}$$

where the objective (4.7a) only depends on $\mathbf{z}$.

Problem (4.7) is an integer programming problem with a convex objective and linear constraints. The convexity comes from the fact that each element $J_{ij}^{\mathbf{z}}$ is convex in z_{ij} when we use a BPR function of the form of (4.3). This is because its second derivative is nonnegative for all $z_{ij} \geq 0$, and by the fact that the summation of convex functions is convex. Moreover, $J_{ij}^{\mathbf{z}}$ is monotonically decreasing as it decreases when we increase the capacity of arc (i,j).

For a deeper discussion on how to efficiently solve (4.7) we refer to Wollenstein-Betech, Paschalidis, and Cassandras (2021). However, one can observe that (4.7) is separable since both the objective and constraints are separable for each pair of opposite direction links (i,j) and (j,i). This enables the possibility to distributively solving this problem and obtaining the global optimal solution. Solving the lane assignment problem is thus on order of $\mathcal{O}(\|\mathbf{n}\|_\infty)$. This complexity is only true when we allow to reverse any lane in the network. However, if we are interested in including other coupling constraints, for example constraining a maximum number of reversals, then we would have to solve the full convex integer programming problem.

The family of algorithms presented herein consists of sequentially solving TAP and LA until convergence. We refer to a *family* of algorithms since we can restrict the

LA problem to have a maximum of reversals at every iteration. For example if we restrict the LA to only select the best reversal at every iteration, we would end up with a greedy algorithm.

This *greedy* approach will switch the best lane, then will optimize flows, and then will look for the next best reversal, and so forth. This approach is natural when urban planners reverse a lane, i.e., wait to see how traffic responds, re-assess the network conditions, and select the next most congested link.

4.3.2 Frank-Wolfe method

A different approach to solve the LASO-TAP problem is to relax the integer variables to continuous ones. This relaxation is often employed when dealing with integer programming problems and has the benefit of generating a lower bound. The relaxed problem leads to a non-convex continuous programming problem for which standard methods can be used to find a local stationary point.

In our context, we first derive an equilibrium condition for the relaxed case which is analogous to the one described in Definition 3 and we quantify the impact on the objective when we reverse a small fraction of a lane while assuming that flows $\mathbf{x}$ remain unchanged. Formally, we perturb the capacity by an infinitesimal amount δ for a fixed $\mathbf{x}$. Then, its impact on the objective can be estimated using

$$\begin{aligned}\frac{\partial J^{\mathbf{z}}}{\partial z_{ij}} = \lim_{\delta \longrightarrow 0} \Big(& \big(x_{ij}t_{ij}(x_{ij}, z_{ij} + \delta) + x_{ji}t_{ji}(x_{ji}, z_{ji} - \delta)\big) \\ & - \big(x_{ij}t_{ij}(x_{ij}, z_{ij}) + x_{ji}t_{ji}(x_{ji}, z_{ji})\big)\Big)/\delta. \end{aligned} \tag{4.8}$$

An equilibrium point for the capacity variables $\mathbf{z}$ is that for all (i,j) in $\mathcal{A}$ we have that

$$\partial J^{\mathbf{z}}/\partial z_{ij} = 0, \quad \text{for } z_{ij} \in (0, n_{ij}),$$

$$\partial J^{\mathbf{z}}/\partial z_{ij} \geq 0, \quad \text{for } z_{ij} = 0,$$

$$\partial J^{\mathbf{z}} / \partial z_{ij} \leq 0, \quad \text{for } z_{ij} = n_{ij}.$$

That is, there is no benefit to reversing δ units of capacity for any arc (i, j) for a current flow solution.

Since the relaxed problem is not convex due to the interaction between the flow and capacity variables in $f(x_{ij}/m_{ij})$, we cannot ensure that the stable point will be a *global* optimal but rather a *local* optimal solution. We argue that this local solution is better than the current *nominal* lane allocation since at every step we aim to improve the overall travel time by modifying $\mathbf{z}$.

To solve the joint $(\mathbf{x}, \mathbf{z})$ problem we consider an *enlarged* Frank-Wolfe algorithm which is similar to the one used for solving the TAP. At every iteration we find the best routing decisions based on the current status of the infrastructure and then we immediately take a step of adjusting the capacity for the current flow solution.

Algorithm 3 provides a detailed description of this methodology. When the algorithm terminates, we simply project the fluidic (continuous) variables to its nearest integer. Note that, similarly to the Frank-Wolfe methodology for solving TAP, this method is memory-efficient (Patriksson, 1994).

4.3.3 Convex approximation

The goal of this subsection is to develop an approximation to the problem such that efficient algorithms can be employed to find a *global* optimal solution instead of a *local* solution.

The key idea is to avoid the interaction in the objective of x_{ij} with the capacity m_{ij} such that we construct a convex objective. To do so, we fix each link's capacity m_{ij} using a nominal (current) number of lanes n_{ij}^0 (its vectorized version $\mathbf{n}^0$). Then, we relate the new lane assignment with the flow in a threshold-based fashion.

Algorithm 3 LASO Frank-Wolfe

```
1: Initialization. Set counter $l := 0$. Perform all-or-nothing assignment with $\mathbf{t}^l = \mathbf{t}(\mathbf{0}, \mathbf{z}^0)$. This yields $\mathbf{x}^l$.
2: Update SO travel time. $\mathbf{t}^{l+1} \leftarrow \mathbf{t}(\mathbf{x}^l, \mathbf{z}^l) + \mathbf{x}^l \nabla_{\mathbf{x}^l} \mathbf{t}(\mathbf{x}^l, \mathbf{z}^l)$.
3: Capacity direction finding. Obtain $\nabla J^{\mathbf{z}}(\mathbf{z}^l)$ using (4.8).
4: Capacity step size selection. Use $\alpha_1^l = 1/l$.
5: Move capacity. $\mathbf{z}^{l+1} \leftarrow \mathbf{z}^l - \alpha_1^l \nabla J^{\mathbf{z}}(\mathbf{z}^l)$.
6: Flow direction finding. Perform All-or-nothing assignment with $\mathbf{t}^{l+1}$ and $\mathbf{z}^{l+1}$ and get auxiliary flows $\mathbf{y}^{l+1}$.
7: Flow step size selection. Use line search and select $\alpha_2$ by solving
   $$\min_{0 \le \alpha_2 \le 1} \sum_{(i,j) \in \mathcal{A}} \int_0^{x_{ij}^l + \alpha_2 (y_{ij}^l - x_{ij}^l)} t_{ij}^{l+1}(\omega, z_{ij}^{l+1}) d\omega$$
   .
8: Move flow. $\mathbf{x}^{l+1} \leftarrow \mathbf{x}^l + \alpha_2 (\mathbf{y}^l - \mathbf{x}^l)$.
9: Relative gap. Calculate the Relative Gap (RG) using
   $$\text{RG} = \frac{\left(\mathbf{x}^{l+1}\right)' \left(\mathbf{t}(\mathbf{x}^{l+1}, \mathbf{z}^{l+1}) + \nabla_{\mathbf{x}^{l+1}} \mathbf{t}(\mathbf{x}^{l+1}, \mathbf{z}^{l+1})\right)}{\sum_{k=1}^{K} d_k h_w} - 1$$
   where $h_k$ is the SO shortest travel time (i.e., $\mathbf{t}(\mathbf{x}^l, \mathbf{z}^l) + \nabla_{\mathbf{x}^l} \mathbf{t}(\mathbf{x}^l, \mathbf{z}^l)$) from $w_{sk}$ to $w_{tk}$.
10: Stopping criterion. If RG $< \xi_1$ and $\|\nabla J^{\mathbf{z}}(\mathbf{z})\|_2 \le \xi_2$ or $l > L$ then continue to Step 11. Otherwise, let $l = l + 1$ and go to Step 2.
11: Project $\mathbf{z}^{l+1}$ to closest integer: $\mathbf{z}^{\text{int}} = \Pi(\mathbf{z}^{l+1})$ and output $(\mathbf{x}^{l+1}, \mathbf{z}^{\text{int}})$.
```

Specifically, we propose the following objective:

$$\min_{\mathbf{x}, \mathbf{z}} \quad \mathbf{x}' \mathbf{t}(\mathbf{x}, \mathbf{n}^0) + \lambda \| \max\{\mathbf{0}, \mathbf{x} - \Theta(\mathbf{z})\} \|_2. \tag{4.9}$$

where λ is a regularizer that trades off routing efficiency versus not exceeding a link's capacity. Note that this objective seeks a joint solution that avoids exceeding the capacity threshold $\Theta(z_{ij})$ that depends on $\mathbf{z}$.

A natural value would be $\Theta(z_{ij}) = c_{ij} z_{ij}$, which points to the capacity of the link $(i, j) \in \mathcal{A}$ when z_{ij} lanes are assigned to it. However, we consider the more general case in which any other linear $\Theta(z_{ij})$ function can be imposed, e.g., a fraction of the link's capacity that the practitioner would like to avoid exceeding.

Using this objective, the approximated LASO-TAP is formulated as follows:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{z},\mathbf{s}} \quad & \mathbf{x}'\mathbf{t}(\mathbf{x},\mathbf{n}^0) + \lambda\|\mathbf{s}\|_2 && \text{(4.10a)}\\
\text{s.t.} \quad & \sum_{i:(i,j)\in\mathcal{A}} x_{ij} - \sum_{\ell:(j,\ell)\in\mathcal{A}} x_{j\ell} = \phi_j \quad \forall j \in \mathcal{N}, && \text{(4.10b)}\\
& z_{ij} + z_{ji} \leq n_{ij}, \quad \forall (i,j) \in \mathcal{A}, && \text{(4.10c)}\\
& \mathbf{s} \geq \mathbf{x} - \Theta(\mathbf{z}), && \text{(4.10d)}\\
& \boldsymbol{\varepsilon} \geq 0, \quad \mathbf{s} \geq 0, && \text{(4.10e)}\\
& z_{ij} = \{0, 1, \ldots, n_{ij}\} \quad \forall (i,j) \in \mathcal{A} && \text{(4.10f)}
\end{aligned}$$

which results in a convex mixed integer program with linear constraints.

Piecewise-affine approximation

Following a similar approach as in Wollenstein-Betech, Salazar, et al. (2021), we consider approximating the travel latency function, i.e., $t_{ij}(x_{ij}, n)$, with a piecewise affine function as shown in Figure 4·2.

For every piecewise segment in the range $\theta_{ij}^{(l)} \leq x_{ij} \leq \theta_{ij}^{(l+1)}$ we introduce a slack variable $\varepsilon_{ij}^{(l)}$. Hence, $t_{ij}(x_{ij}, n_{ij}^0)$ is approximated by

$$\hat{t}_{ij}(\boldsymbol{\varepsilon}_{ij}, n_{ij}^0) := t_{ij}^0 \left(1 + \sum_{l=1}^{n} \frac{a_l}{n_{ij}^0} \varepsilon_{ij}^{(l)}\right),$$

where $a_1 < a_2 < \cdots < a_n$ are the slopes of the n segments in the piecewise-affine approximation. Using this function we obtain a quadratic objective

$$\hat{J}_{ij}(\boldsymbol{\varepsilon}_{ij}) := \mathbf{1}'\boldsymbol{\varepsilon}_{ij}\hat{t}_{ij}(\boldsymbol{\varepsilon}_{ij}, n_{ij}^0) \approx x_{ij} t_{ij}(x_{ij}, n_{ij}^0),$$

where $\mathbf{1}'\boldsymbol{\varepsilon}_{ij} = x_{ij}$ (recall that $\mathbf{1}$ is a vector of all ones).

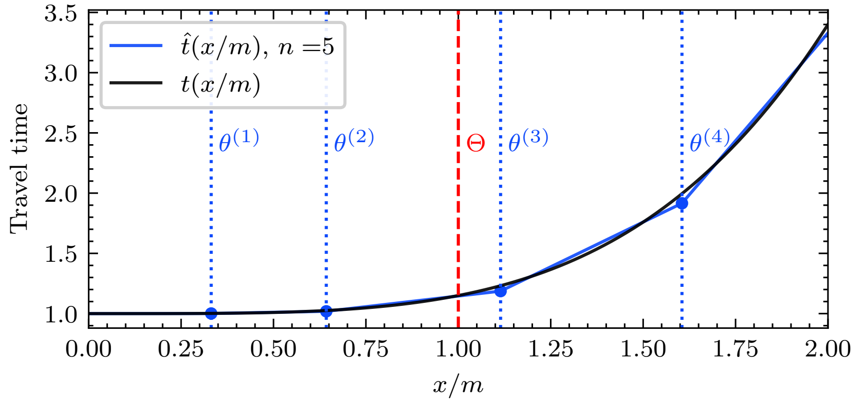


Figure 4·2: Piecewise affine approximation of the travel time function

In Wollenstein-Betech, Salazar, et al. (2021) and Section 5.3.1 of this thesis, we show that this piecewise linearization for the classical TAP results in a convex quadratic program (QP) that could be further approximated by a linear program (LP). Using this piecewise affine approximation, we formulate the LASO-TAP problem as a mixed integer quadratic programming problem (MIQP):

$$\min_{\boldsymbol{\varepsilon},\mathbf{z},\mathbf{s}} \quad \mathbf{1}'\hat{\mathbf{J}}(\mathcal{E}) + \lambda\|\mathbf{s}\|_2 \tag{4.11a}$$

$$\text{s.t.} \quad \sum_{i:(i,j)\in\mathcal{A}} \mathbf{1}'\boldsymbol{\varepsilon}_{ij} - \sum_{k:(j,k)\in\mathcal{A}} \mathbf{1}'\boldsymbol{\varepsilon}_{jk} = \phi_j, \quad \forall j \in \mathcal{N}, \tag{4.11b}$$

$$0 \le \epsilon_{ij}^{(l)} \le \theta_{ij}^{(l+1)} - \theta_{ij}^{(l)}, \quad l = 1,\dots,L_{ij}-1, \tag{4.11c}$$

$$z_{ij} + z_{ji} \le n_{ij}, \quad \forall (i,j) \in \mathcal{A}, \tag{4.11d}$$

$$\mathbf{s} \ge \mathbf{x} - \Theta(\mathbf{z}), \tag{4.11e}$$

$$\mathbf{s} \ge 0, \tag{4.11f}$$

$$z_{ij} = \{0, 1, \dots, n_{ij}\} \quad \forall (i,j) \in \mathcal{A}, \tag{4.11g}$$

where $\hat{\mathbf{J}}(\mathcal{E}) = (\hat{J}_{ij}(\boldsymbol{\varepsilon}_{ij});\ (i,j) \in \mathcal{A})$. Note that, we can also formulate this problem as a mixed integer linear program (MILP) by approximating the routing part of the objective as in Wollenstein-Betech, Salazar, et al. (2021) and by penalizing the slack variables $\mathbf{s}$ with an ℓ_1 norm instead of an ℓ_2 norm.

Although the problem we are dealing with is NP-hard due to the integrality of $\mathbf{z}$, exact methods such as branch and bound have been found to perform well in practice. Otherwise, we can relax $\mathbf{z}$ by letting it be a nonnegative continuous variable. When applying this relaxation, we can give two different interpretations. First, we can think of the non-integer part of the solution as a percentage of the time in which the lane is reversed. For example, if the solution indicates that an optimal lane configuration is equal to $z_{12} = 4/3$ and $z_{21} = 5/3$ for a 3 hours period, we could assign 1 lane to link $(1, 2)$ for the two hours and 2 lanes assigned for one hour. Similarly, link $(2, 1)$ will have 1 lane assigned during one hour and a second lane assigned for 2 hours. The second approach consists of projecting the continuous solution to the closest integer (as in the proposed Frank-Wolfe method).

We conclude by observing that this convex approximation approach gives us four different methodologies coming from the *convex approximation* framework: the integer-based NP-hard methods MIQP and MILP; and the polynomial-time methods QP and LP with their appropriate projections.

4.3.4 Discussion

A few comments are relevant to consider. First, one should note that contraflow lane reversals could violate our assumption regarding the strong connectivity of $\mathcal{G}$. This may happen since we could disconnect certain nodes in the network while performing a lane reversal. One way to avoid this issue is to restrict the solution to maintain at least one lane in each direction of the road. While this approach maintains connectivity, it may sacrifice some efficiency by limiting the optimization procedure. An alternative way to handle this issue is by assuming a tiny amount of demand for every pair of nodes in the network. In this way, we ensure that there is a path connecting any two nodes. Note however, that this trick may increase the dimensionality of the problem by increasing the number of variables. This is because now every node will become an origin in (4.6).

Second, the convexified version requires tuning λ. Although there is no specific way to set λ, we observe that it is trading off routing with not exceeding a link's capacity. These two processes are somehow aligned because as flows get closer to a link's capacity, they also become less attractive for routing purposes due to higher travel times (see Figure 4·2).

Third, we would like to argue the flexibility that the convexified (the MIQP, MILP, QP and LP) approaches provide in comparison with the alternating and Frank-Wolfe methods. Their main advantage is that they allow the inclusion of additional linear constraints to our problem. This offers the possibility to extend the framework to more realistic and practical examples. Another advantage of the convex approaches, and more specifically the LP approach, is the ability to provide with very low effort sensitivity analysis of the solution with respect to the demand vector. Moreover, the LP method can accommodate *robust* (or uncertainty-aware) instances of the problem in which the OD demand lies inside an uncertainty set and we aim to choose the best lane configuration of the worst selection of OD demands (Ben-Tal, El Ghaoui, & Nemirovski, 2009). This can be achieved by leveraging Bertsimas and Sim (2003). Finally, commercial software for solving LPs and QPs is widely available and has been improved over the years such that it is quite efficient to compute a solution to the problem using this strategy.

4.4 Extensions

In the previous section we introduced three main methodologies to solve the lane assignment problem while considering the routing decisions of commuters. Now, we exploit the convex approximation methodology to extend the framework to be suitable for more practical implementations, especially to limit the number of reversals.

Imposing a maximum number of reversals becomes relevant for several reasons. First, when considering the case of human-driven vehicles, we would like to have fewer reversible roads in the network to avoid confusion. Second, if the transportation agency is planning to invest on infrastructure in these roads, e.g., surveillance cameras, barrier transfer machines (see Fig. 4·1c), then the agency will limit the number of reversible lanes/links in the network due to budget constraints.

The extensions presented herein are suitable for the MIQP and MILP frameworks but are not suitable for LP and QP. This is because we rely on inequalities that are well-defined for the integer variables but not for the continuous ones.

4.4.1 Maximum number of lane reversals

So far, we have selected the lane configuration such that it complies with the constraint $z_{ij} + z_{ji} = n_{ij}$. Note that this does not consider the *current* infrastructure status and it assigns lanes to links regardless of implementation costs. However, transportation infrastructure is, in general, not flexible to perform many reversals during a day. Therefore, we would like to limit the number of allowable lane reversals.

To achieve that, we seek to ensure that our solution does not deviate too much from the *nominal* (or *current*) configuration. This can be modeled by $|n^0_{ij} - z_{ij}| \leq \xi$ where n^0_{ij} is the nominal capacity of the link (i, j) and ξ is the maximum number of lane reversals allowed. Then, using the convex formulation (4.11) we can simply add, for each link $(i, j) \in \mathcal{A}$, a slack variable $r_{ij} \geq 0$ and the linear constraints $r_{ij} \geq n^0_{ij} - z_{ij}$; $r_{ij} \geq -(n^0_{ij} - z_{ij})$; and $\sum_{(i,j)\in\mathcal{A}} r_{ij} \leq \xi$.

4.4.2 Maximum number of link reversals

In a similar way to the previous subsection, suppose now that the planner would like to limit the number of allowable *link* (instead of lane) reversals. This is relevant because when investing in a lane reversal infrastructure for one road, the cost of reversing one versus multiple lanes may not be significant. However, building the initial lane reversal infrastructure for a particular road is.

To model this, we introduce for every arc $(i,j) \in \mathcal{A}$ the variable

$$q_{ij} = \min\{1,\ |n^0_{ij} - z_{ij}|\}, \tag{4.12}$$

$$= \max\{-1, -r_{ij}\}. \tag{4.13}$$

where r_{ij} is defined as in Section 4.4.1. To introduce this to our MIQP or MILP, we let ξ be the number of allowable link reversals and we add the slack variables q_{ij} with constraints $q_{ij} \leq 0$; $q_{ij} \leq r_{ij}$; $q_{ij} \geq -1$; and $\sum_{(i,j)\in\mathcal{A}} q_{ij} \leq \xi$.

4.4.3 Multi-period optimization

We have considered the problem of finding the best lane configuration for a single OD demand matrix. However, in practice we would like to decide where to invest based on *multiple* traffic patterns. For example, by considering the morning and the afternoon peak traffic.

To achieve this, suppose that we have an OD demand matrix for every time interval $t \in \mathcal{T}$. For example, $\mathcal{T} = \{\text{AM}, \text{MD}, \text{PM}, \text{NT}\}$ corresponding to morning, midday, afternoon and night traffic patterns, respectively. Then, the problem becomes

$$\min_{\{\mathcal{E}_t, \mathbf{c}_t, \mathbf{s}_t \ \mid\ t\in\mathcal{T}\}} \quad \sum_{t\in\mathcal{T}} \mathbf{1}'\hat{\mathbf{J}}_t(\mathcal{E}_t) + \lambda\|\mathbf{s}_t\|_2, \tag{4.14a}$$

$$\text{s.t.} \quad \sum_{i:(i,j)\in\mathcal{A}} x_{ijt} - \sum_{\ell:(j,\ell)\in\mathcal{A}} x_{j\ell t} = \phi_{jt},\ \forall j \in \mathcal{N},\ \forall t \in \mathcal{T}, \tag{4.14b}$$

$$z_{ijt} + z_{jit} \leq n_{ij}, \quad \forall (i,j) \in \mathcal{A},\ \forall t \in \mathcal{T}, \tag{4.14c}$$

$$\mathbf{s}_t \geq \mathbf{x}_t - \Theta(\mathbf{z}_t), \quad \forall t \in \mathcal{T}, \tag{4.14d}$$

$$\mathbf{r}_t \geq \mathbf{n}_t^0 - \mathbf{z}_t, \quad \forall t \in \mathcal{T}, \tag{4.14e}$$

$$\mathbf{r}_t \geq -(\mathbf{n}_t^0 - \mathbf{z}_t), \quad \forall t \in \mathcal{T}, \tag{4.14f}$$

$$\mathbf{q} \geq -\mathbf{1}, \tag{4.14g}$$

$$\mathbf{q} \leq -\sum_{t \in \mathcal{T}} \mathbf{r}_t, \tag{4.14h}$$

$$\mathbf{1}'\mathbf{q} \leq \xi, \tag{4.14i}$$

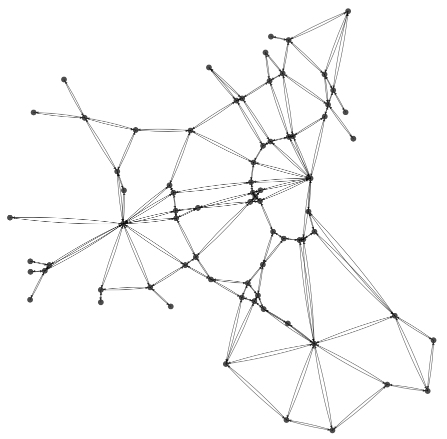
$$\boldsymbol{\varepsilon}_t, \mathbf{s}_t \geq 0,\ \mathbf{z}_t \in \mathbb{N}_+, \quad \forall t \in \mathcal{T}. \tag{4.14j}$$

Problem 4.14 is an augmented version of (4.11) with the additional coupling constraint (4.14h) which limits the number of links to invest over all the OD demand setting in $\mathcal{T}$. Notice that constraint (4.14h) is the only coupling constraint over $\mathcal{T}$. Therefore, this formulation is suitable for using decomposition techniques, such as Dantzig-Wolfe (Dantzig & Wolfe, 1960).

4.5 Numerical Results and Case Studies

To validate and compare the methods described above, we consider several numerical examples over two different transportation networks shown in Figure 4·3. The *test* network is a small example that is useful to test our methods. Additionally, we perform a case study using the Eastern Massachusetts interstate highways (EMA) subnetwork (Figure 4·3a).

The EMA road network is relevant in the context of lane reversals as it captures the dynamics of suburban/urban mobility where we expect lane reversal strategies to be beneficial. This is because arterial roads typically have a large number of lanes and, at the same time, they experience high traffic congestion. The values of the network topology (e.g., capacities, free flow speeds), as well as the code used to perform the experiments is publicly available in our online repository.[1]



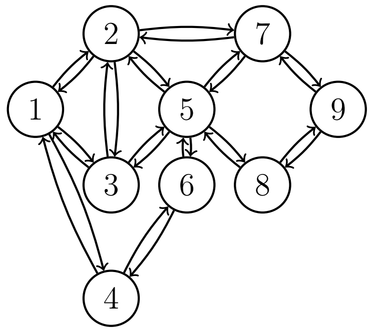


Figure 4·3: (a) EMA transportation network composed of 74 nodes, 258 arcs, 581 lanes, and 1113 OD pairs; (b) Test network consists of 9 nodes, 26 arcs, 61 lanes, and 5 OD pairs.

We present our numerical results in the same order as our methods were presented in Section 4.3. First, we analyze the convergence of different alternating methods and the Frank-Wolfe algorithm. Then, we report numerical results for different selections of the λ parameter for the convex programming approach and we compare all approaches. Finally, we provide an example of one of our extensions which limits the number of link reversals and we experiment with the symmetry of the OD demand.

4.5.1 Alternating and Greedy

Using the alternating method described in Section 4.3.1 we solve the LASO-TAP. In particular we use three different methodologies. First, we introduce the *greedy* (or *One*) approach where the idea is to first solve the TAP. Then, find the best possible reversal and switch it. Once we have changed the lane, we re-solve the TAP. We do

[1]https://github.com/salomonw/contraflow-lane-reversal

this iteratively until we converge to a value. The second approach, which we called *Five*, implements the same idea, but instead of changing only the best possible lane, it changes the top 5 lane reversals at every iteration. Similarly, our last alternating approach referred to as *Full* follows the same procedure but at each step reverses all the possible lanes that improve the traveling times. Figure 4·4 indicates the convergence of these three approaches.

We observe two main patterns in these results. First, the convergence of the algorithm is reached within a few (two to three) iterations. This is consistent with many bi-level formulations involving the traffic assignment problem. Second, and more interestingly, the greedy approach performs poorly compared to the other methods. Our justification for this behavior is that when flows (or routes) are re-optimized, they are trying to use the links of the network infrastructure efficiently. Hence, we find a good allocation of flows to lanes such that the whole system is closer to a stationary point.

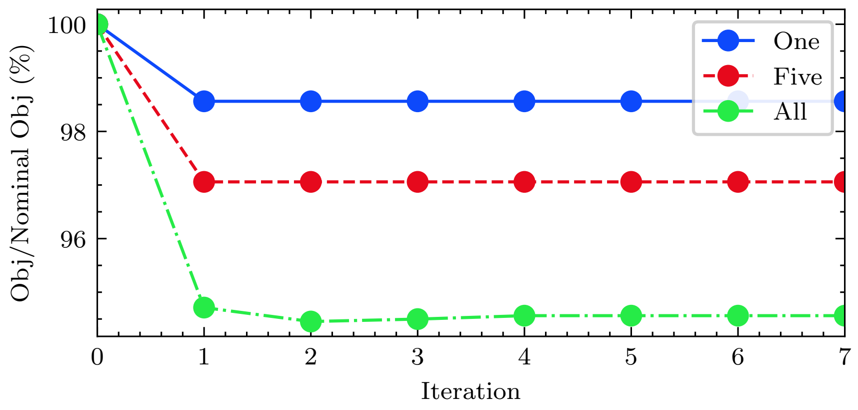


Figure 4·4: Convergence of the alternating method for different numbers of reversals per iteration. *"One"* corresponds to reversing only one lane at every iteration while *"All"* intends to reverse all the beneficial lanes at every step.

4.5.2 Convergence of Alg. 3

With the aim of validating our Frank-Wolfe methodology described in Section 4.3.2, we observe its trajectory over the iterates using the EMA transportation network.

Figure 4·5 shows (i) the value of the objective function $J(\mathbf{x}, \mathbf{z})$; (ii) the ℓ_2 norm of the gradient with respect to the lane reversals, i.e., $\|\nabla J^{\mathbf{z}}(\mathbf{z})\|_2$. When this value equals zero, we know that an infinitesimal change in reversing a lane will not be beneficial to the system; (iii) the relative gap (RG) pointing to the closeness to a solution that follows a Wardrop equilibrium. Hence, these results verify the effectiveness and convergence of Alg. 3 in the sense that it is minimizing the overall travel times by adjusting the routing and the links capacity at every iteration.

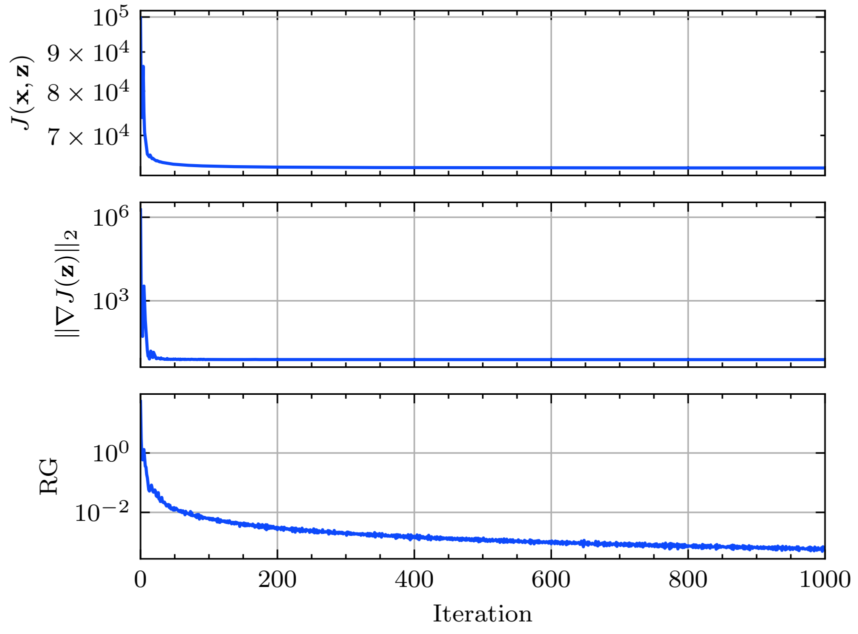


Figure 4·5: Convergence of the Frank-Wolfe algorithm proposed in Section 4.3.2

4.5.3 Dependence on λ

As we discussed in Subsection 4.3.4, one of the main drawbacks of our convex formulation is the inclusion of the parameter λ. This parameter serves as a trade-off between the routing and the lane assignment decisions. Hence, we would like to observe the performance of our solution for different values of λ. To perform the experiment, we use the EMA network and we solve the problem for $\lambda \in (0, 1 \times 10^5)$.

Figure 4·6 reports the objective for different values of λ and for different algorithms. The numerical results show that the model is robust for different values of λ, i.e., the performance does not change substantially for very different λ's. This is a positive result, since it suggest that calibrating λ does not play a very important role in the performance of this method.

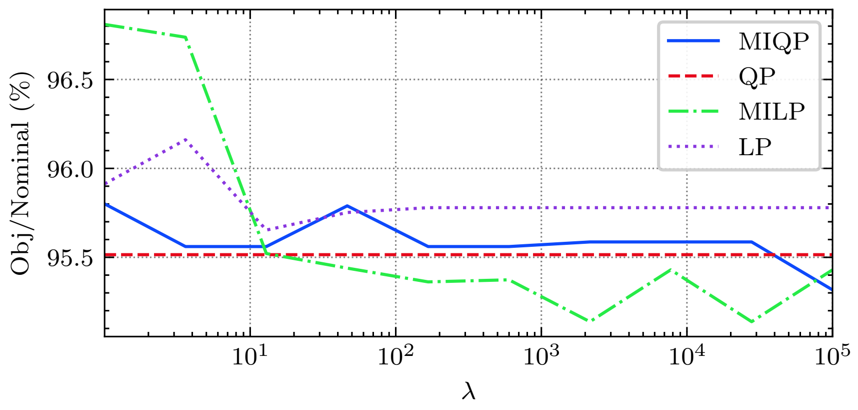


Figure 4·6: Objective function for different values of λ and for different convex methods over the EMA network.

4.5.4 Comparison between methods

Now, we compare all the methods proposed. We solve the routing problem, i.e., the TAP, with the current lane configuration and call it the *nominal* solution. Then, we solve the LASO-TAP problem for the test and the EMA networks using all the methodologies described earlier, i.e., Frank-Wolfe (FW), the three variations of the alternating method, and the different convex integer and continuous programs.

In Table 4.1 we report the computational time of each method, as well as their performance compared to the nominal lane allocation. Specifically, we take $1 - (\mathrm{OBJ}_{\mathrm{method}}/\mathrm{OBJ}_{\mathrm{nominal}})$ to obtain a *relative improvement* (RI) metric with respect to the nominal allocation.

We observe that the convex approximation approaches perform very well in terms of obtaining efficient solutions to the problem. In particular LP, QP and MILP compute the solution relatively fast compared with the FW or the alternating methods and achieve good solutions while providing the flexibility to add linear constraints.

Table 4.1: Results indicating the relative improvement in the overall travel time between a method and the nominal capacity.

Method	Test		EMA	
	RI (%)	Time (s)	RI (%)	Time (s)
Nominal	0.0	0.01	0.0	0.42
FW	12.7	11.5	0.7	956.2
Alternating (1)	10.3	0.62	1.4	12.2
Alternating (5)	14.6	0.60	2.9	12.4
Alternating (full)	14.7	0.59	5.4	12.4
MIQP	17.5	0.02	4.7	101.37
QP	17.4	0.02	4.5	0.71
MILP	17.5	0.02	4.6	5.67
LP	17.3	0.01	4.2	0.36

4.5.5 Maximum number of lane reversals

This experiment computes the Pareto optimal frontier using the extension of our convex approximation that limits the number of link reversals for which we compute the performance for a different value of the maximum number of allowable link reversals. Figure 4·7 shows this relationship for the EMA network.

As expected, we observe that the first lanes are the ones that contribute the most to the improvement of the overall travel times. This methodological advantage and corresponding results are of particular interest to urban planners that might need to prioritize the most critical roads and may have a budget that depends on the overall benefit of reducing travel times for the system. In addition, our numerical results imply that it is not necessary to invest in too many lane reversals to achieve a large fraction of achievable improvement. For example, by investing between 10 and 15 links in the network (out of 129 possible links) the solution has already reached most of the benefit.

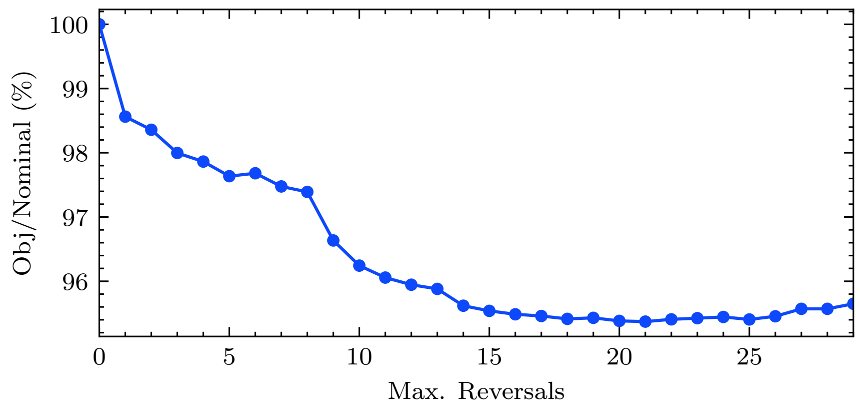


Figure 4·7: Performance on the overall travel times as the number of links allowed to perform reversals increases.

4.5.6 Effect of OD demand symmetry

A characteristic of lane reversal strategies is that larger benefits occur when the demand is *not symmetric*. We use an example to introduce the concept of *symmetry* in this context. Consider the demand patterns of a large metropolitan area on a weekday morning. In most cases, we expect a large fraction of the demand to be traveling towards the city center. Hence, we expect high traffic heading towards the city center and low traffic traveling to suburban areas. We refer to this case as *asymmetric* demand. In contrast, the traffic flows between two major cities could serve as an example of a *symmetric* OD demand.

We generated an experiment using the Test network (Figure 4·3b) to exemplify the effect of symmetry in the performance of lane reversals. To do so, let $\rho \in (0, 1)$ be a parameter describing the fraction of the OD demand traveling from west to east and $(1-\rho)$ the fraction of vehicles going from east to west. More explicitly, we defined the OD matrix to be $d_{(1,9)} = 15,000\rho$, $d_{(1,9)} = 15,000(1-\rho)$ and $d_{(s,t)} = 0$ for all $(s,t) \in \mathcal{W} \setminus \{(1,9),(9,1)\}$.

Figure 4·8 shows the *relative improvement* of the lane reversal solution, using MIQP and QP, with respect to the nominal capacity solution. We observe that for the cases in which demand is not symmetric, the improvements in travel time can be as high as to 65%. In contrast, when the demand is symmetric, there are fewer benefits (considering uniform capacities across the network). Notice that negative values can be observed (as in the QP case) since we are solving an approximation method (both by the convex approach and by projecting the continuous QP solution) that could deviate from the optimal value in certain cases. These results provide us a tractable example to understand the potential of lane reversals depending on the symmetry of the demand and suggest the instances for which these interventions provide large benefits. Some real-life examples in which asymmetric demands can be found include: morning and afternoon peaks, massive events, and holiday travel.

4.6 Summary and Future Work

The problem of identifying the best lanes to reverse in a congested network is challenging because it requires to solve a mixed integer non-convex programming problem. The literature dealing with this problem has been focused on using heuristic algorithms for solving it.

In this chapter we propose three strategies to reduce the complexity of the problem. Our first method uses the principle of *decomposition*, or *divide and conquer*, by separating the joint routing and assignment problem into separate ones. Our second methodology uses the idea of *relaxation* by converting the integer variables to con-

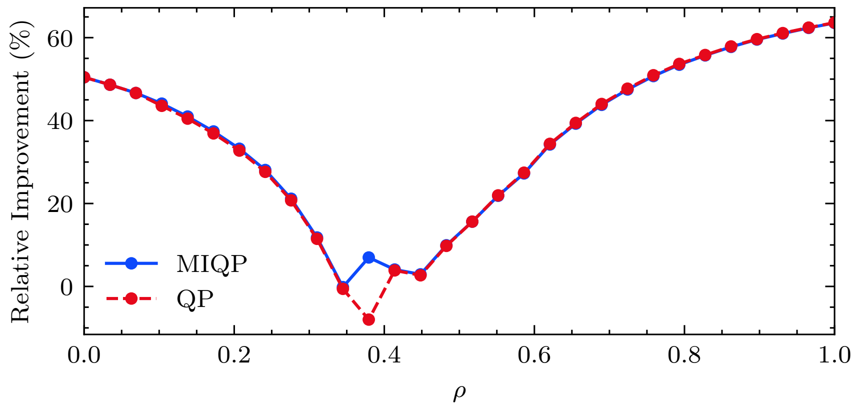


Figure 4·8: The relative improvement for different OD demand distributions in the Test network. The x-axis ρ indicate the fraction of demand traveling from west to east. Similarly, $(1-\rho)$ captures the fraction of vehicles traveling from east to west. The extremes ($\rho = 0$ and $\rho = 1$) are the most asymmetric traveling patterns

tinuous ones. Lastly, our third approach *convexifyies* the objective by modifying the objective function. This last method is interesting since it allows including additional constraints to the problem, e.g., a maximum number of reversals.

We provide numerical results for all our methods showing their performance over a test network and a case study using the transportation network of Eastern Massachusetts. Interestingly our results show that the greedy approach, the one that reverses the most relevant lane at every iteration, can result in sub-optimal solutions. In contrast, the convexification approach is efficient in both the quality of the solution and the computational burden.

We foresee the development of sensitivity analysis and robust optimization methods as an interesting future research direction. This next step is interesting specifically for the convexification method, as it permits to use known methods for linear programming. This will contribute to the existing literature regarding robust network optimization.

Chapter 5

Optimizing Routing and Rebalancing of Autonomous Mobility-on-Demand systems

This chapter studies congestion-aware route-planning policies for intermodal Autonomous Mobility-on-Demand (AMoD) systems, whereby a fleet of autonomous vehicles provides on-demand mobility jointly with public transit under mixed traffic conditions (consisting of AMoD and private vehicles). First, we devise a network flow model to jointly optimize the AMoD routing and rebalancing strategies in a congestion-aware fashion by accounting for the endogenous impact of AMoD flows on travel time. Second, we capture the effect of exogenous traffic stemming from private vehicles adapting to the AMoD flows in a user-centric fashion by leveraging a sequential approach. Since our results are in terms of link flows, we then provide algorithms to retrieve the explicit recommended routes to users. Finally, we showcase our framework with two case-studies considering the transportation sub-networks in Eastern Massachusetts and New York City, respectively. Our results suggest that for high levels of demand, pure AMoD travel can be detrimental due to the additional traffic stemming from its rebalancing flows. However, blending AMoD with public transit, walking and micromobility options can significantly improve the overall system performance.

5.1 The Problem and Related Work

We study the *routing* and *load-balancing* processes of a fleet of vehicles belonging to an AMoD service when they interact with self-interested vehicles in the network. In contrast to commonly used platforms today (e.g., Uber, Lyft, DiDi), our objective is to take these two decisions *jointly* rather than separately.

In the chapter, we devise methodologies that optimize the operations of AMoD systems with the goal of reducing traffic congestion. To achieve this, we develop a coordinated intermodal routing procedure that seeks to minimize the overall commuters travel time while ensuring that all travelers are being served by the same platform.

5.1.1 Related Work

Current drivers in MoD platforms, such as Uber, Lyft or DiDi, choose their paths by using routing apps (e.g., Waze and Google Maps). These apps recommend routes using traditional exact shortest path algorithms such as Dijkstra's (Dijkstra, 1959), Bellman-Ford (Bellman, 1958), and incremental graph (Ramalingam & Reps, 1996) or by heuristics such as A-star (Hart, Nilsson, & Raphael, 1968), tabu search (Hertz, Laporte, & Mittaz, 2000) and genetic algorithms (Ahn & Ramakrishna, 2002).

This *User-Centric* (UC) approach to routing, in which every driver minimizes their own travel time, is suboptimal compared to *System-Optimal* (SO) routing schemes achievable when vehicles are coordinated by a central controller as explained in Chapter 3 of this thesis.

In a mixed traffic setting, the interaction between a fleet of CAVs using SO routing coupled with reactive UC private vehicles is investigated theoretically by Lazar et al. (2017), where a reduction in headways is considered thanks to the adaptive cruise control technology included in CAVs. However, this analysis requires a network configuration of parallel links and is not suitable for general transportation networks.

To overcome this, Harker (1988); Houshmand et al. (2019); Yang et al. (2007) propose an iterative approach to find a solution between these two classes of vehicles known as *diagonalization scheme.*

Aside from routing, *rebalancing* is tackled in practice by providing drivers with a real-time heat-map of the users' demand such that the driver is incentivized to relocate to an area that will maximize its profits.

Rebalancing has been studied by researchers using *proactive* (or *planning*) strategies that redistribute the fleet across regions in order to meet a forecasted demand[1]. Using this perspective, Pavone et al. (2012) shows that rebalancing is necessary to avoid building unbounded customer queues and to stabilize the system. Moreover, they propose a rebalancing policy that minimizes the empty vehicle travel time under static (steady-state) conditions using a fluidic model. Furthermore, R. Zhang and Pavone (2016) proposes a queueing-theoretical approach to account for customers leaving the system when their waiting times are long. This method repeatedly solves a Linear Program (LP) that balances the fleet availability across the regions. Similarly, Spieser, Samaranayake, Gruel, and Frazzoli (2016) proposes a method that minimizes the number of customer dropouts instead of the empty driven miles to focus on service quality. Different from these queueing models, simulation-based methods are also employed (Hörl, Ruch, Becker, Frazzoli, & Axhausen, 2018; Levin, Kockelman, Boyles, & Li, 2017; Swaszek & Cassandras, 2019a).

More recently, schemes that consider the effects of rebalancing in routing and congestion have been analyzed. Threshold approximations of the travel time function have been used to study congestion effects in Rossi, Zhang, Hindy, and Pavone (2018), sometimes capturing the interaction with public transit (Salazar, Lanzetti, Rossi,

[1]Note that this process finds good coverage of vehicles over regions of the system and it is not focused on *matching* or *assigning* vehicles to customers. This vehicle-passenger *assignment* is explored in Alonso-Mora, Samaranayake, Wallar, Frazzoli, and Rus (2017); Bei and Zhang (2018); R. Chen and Cassandras (2022)

Schiffer, & Pavone, 2020; Salazar, Rossi, Schiffer, Onder, & Pavone, 2018), or with the power-grid (Belakaria, Ammous, Smith, Sorour, & Abdel-Rahim, 2019; Rossi et al., 2020). These threshold schemes work as binary decisions allowing for the use of a road (or not), depending on whether the flow has exceeded the threshold or not, but do not capture different travel times for different flow levels on each link.

To account for flow-based routing schemes most work leverages the classical Bureau of Public Roads (BPR) congestion model together with network optimization methods. In particular, Solovey, Salazar, and Pavone (2019) provides a Frank-Wolfe algorithm, where dummy nodes are added to the transportation network to account for rebalancing flows and where the BPR function is evaluated when designing routes. However, this approach cannot include other modes of transportation such as walking, micromobility options, or public transit. Against this backdrop, a piecewise-affine approximation of the travel time function is introduced in Salazar et al. (2019) which converts the joint problem to a quadratic program. In this work, we extend this approximation in order to account for more accurate, fast and implementable models.

The remainder of the chapter is organized as follows: In Section 5.2 we present the models used and the problem formulation. In Section 5.3 we develop the piecewise-affine approximation formulation along with the main analytical results. In Sections 5.4 and 5.5, we provide a framework for the mixed traffic problem and route-recovering strategies, respectively. Finally, we present experiments using the Eastern Massachusetts and New York City transportation networks in Section 5.6 and we conclude in Section 5.7.

5.2 Problem Formulation

Consider an AMoD system which provides a mobility service through multiple modes of transportation (e.g., autonomous taxi-rides, walking and mass transit). To model the system, let $\mathcal{G}$ be a network composed by the road layer and L additional layers, each representing a different transportation mode (Fig. 5·1). We denote by $\mathcal{G}_{\mathrm{R}} = (\mathcal{V}_{\mathrm{R}}, \mathcal{A}_{\mathrm{R}})$ the road layer and by $\mathcal{G}_{\mathrm{l}} = (\mathcal{V}_{\mathrm{l}}, \mathcal{A}_{\mathrm{l}})$, for $l = 1, \ldots, L$, the other layers where $(\mathcal{V}_{\mathrm{R}}, \mathcal{A}_{\mathrm{R}})$ and $(\mathcal{V}_{\mathrm{l_i}}, \mathcal{A}_{\mathrm{l_i}})$ are the sets of vertices and arcs for each layer. Then, the supergraph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ is composed of all layers and a set of *switching* arcs, denoted by $\mathcal{A}_{\mathrm{S}}$, that connect the network layers to allow AMoD users to switch modes (see dotted lines in Fig. 5·1). Formally $\mathcal{G}$ is composed of the set of vertices $\mathcal{V} = \mathcal{V}_{\mathrm{R}} \cup \mathcal{V}_1 \cup \ldots \cup \mathcal{V}_L$ and arcs $\mathcal{A} = \mathcal{A}_{\mathrm{R}} \cup \mathcal{A}_1 \cup \ldots \cup \mathcal{A}_L \cup \mathcal{A}_{\mathrm{S}}$.

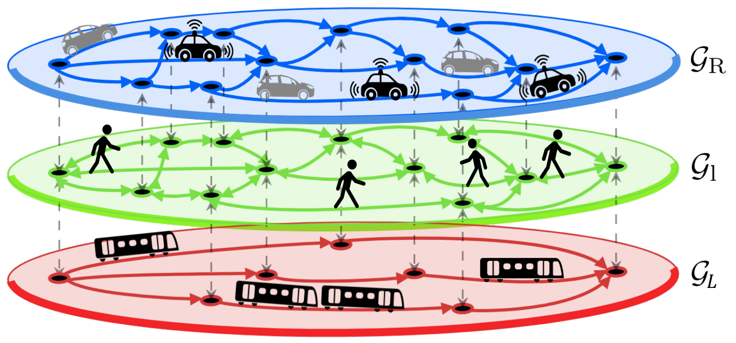


Figure 5·1: Intermodal AMoD network (supergraph) consisting of three layers: the road network (blue with black AMoD cars and grey private vehicles, respectively), walking pathways (green) and subway lines (red); the dashed arrows represent switching arcs.

In order to model the demanded trips, let $\mathbf{w} = (w_s, w_t)$ denote an Origin-Destination (OD) pair and $d_{\mathbf{w}} \geq 0$ the demand rate at which customers request service per unit time from origin w_s to destination w_t. Let W be the total number of OD pairs and $\mathcal{W} = \{\mathbf{w}_k : \mathbf{w}_k = (w_{sk}, w_{tk}), k = 1, \ldots, W\}$ the set of OD pairs. Let a vectorized version of the demand be $\mathbf{g} = (d_{\mathbf{w}_k}; k = 1, \ldots, W)$.

To keep track of the AMoD user flow on a link, we let $x_{ij}^{\mathbf{w}}$ denote the AMoD flow induced by OD pair $\mathbf{w}$ on link $(i, j) \in \mathcal{A}$. Given that the AMoD system needs to rebalance its vehicles to ensure service, we let x_{ij}^{r} be the *rebalancing flow* on link $(i, j) \in \mathcal{A}_{\mathrm{R}}$. Finally, to consider the interaction between the AMoD provider and the other (private) vehicles, we let x_{ij}^p be the self-interested *private vehicle* flow on $(i,j) \in \mathcal{A}_{\text{R}}$. We use the term "private" as we assume that self-interested users must arrive at their destination with their vehicle and do not have the option of switching transportation mode since they have a parking constraint. To simplify notation, we let the AMoD user flow on any link be

$$x_{ij}^u = \sum_{\mathbf{w} \in \mathcal{W}} x_{ij}^{\mathbf{w}}, \qquad \forall (i,j) \in \mathcal{A}, \tag{5.1}$$

and the total flow on a link be

$$x_{ij} = x_{ij}^u + x_{ij}^r + x_{ij}^p, \qquad \forall (i,j) \in \mathcal{A}. \tag{5.2}$$

Note that neither rebalancing flow $\mathbf{x}^r$ nor private vehicle flow $\mathbf{x}^p$ exists on layers $l = 1, \ldots, L$ nor on the switching links in Fig. 5·1. Hence, for those links we set $x_{ij}^r = x_{ij}^p = 0$ for all $(i,j) \in \mathcal{A} \setminus \mathcal{A}_{\text{R}}$.

We specify the time it takes to cross link (i,j) as $t_{ij}(x) : \mathbb{R}_+^{|\mathcal{A}|} \mapsto \mathbb{R}_+$. Using the same structure as in Beckmann et al. (1955), we characterize t_{ij} as a *travel time* function that maps the flow x_{ij} on a link to a travel time as follows:

$$t_{ij}(x_{ij}) = t_{ij}^0 f(x_{ij}/m_{ij}), \tag{5.3}$$

where m_{ij} is the link capacity, t_{ij}^0 is the free-flow travel time on link (i,j), and $f(\cdot)$ is a strictly increasing, positive, and continuously differentiable function. To ensure that the travel time is equal to the free-flow travel time when there is no flow on the link, we consider functions with $f(0) = 1$.

Throughout this chapter, we use the Bureau of Public Roads (BPR) function to decide the routes of AMoD users and private vehicles given the network flow levels. However, our analysis allows for any strictly increasing travel time function. For the rest L layers (excluding the road layer) we consider a constant travel time (independent of the flow) on every link.

5.2.1 System-Optimal Routing and Rebalancing of AMoD Systems

Recall that our goal is to find the system-optimal congestion-aware routes and rebalancing policies of an AMoD provider. Let $d^u_{\mathbf{w}}$ be customer rate requests to the AMoD system for passengers traveling from w_s to w_t, and $\mathbb{1}_{i=j}$ be the indicator function equal to 1 when $i = j$ and 0 otherwise. The problem we aim to solve is then expressed by

$$\min_{\mathbf{x}^{\mathcal{W}},\mathbf{x}^r} \quad J(\mathbf{x}) := \sum_{(i,j)\in\mathcal{A}} t_{ij}(x_{ij})x^u_{ij} + \sum_{(i,j)\in\mathcal{A}_{\mathrm{R}}} c_{ij}x^r_{ij} \tag{5.4a}$$

$$\text{s.t.} \quad \sum_{i:(i,j)\in\mathcal{A}} x^{\mathbf{w}}_{ij} + \mathbb{1}_{j=w_s}d^u_{\mathbf{w}} = \sum_{k:(j,k)\in\mathcal{A}} x^{\mathbf{w}}_{jk} + \mathbb{1}_{j=w_t}d^u_{\mathbf{w}}, \quad \forall \mathbf{w}\in\mathcal{W}, j\in\mathcal{V}, \tag{5.4b}$$

$$\sum_{i:(i,j)\in\mathcal{A}_{\mathrm{R}}} \left(x^r_{ij} + x^u_{ij}\right) = \sum_{k:(j,k)\in\mathcal{A}_{\mathrm{R}}} \left(x^r_{jk} + x^u_{jk}\right), \forall j\in\mathcal{V}_{\mathrm{R}}, \tag{5.4c}$$

$$\mathbf{x}^{\mathcal{W}}, \mathbf{x}^r \geq \mathbf{0}, \tag{5.4d}$$

where we use bold notation $\mathbf{x}$ to represent a vector containing all the elements of x_{ij}.

The dimensions of the decision vectors $\mathbf{x}^r$ and $\mathbf{x}^{\mathcal{W}}$ are given by $\mathbf{x}^r \in \mathbb{R}^{|\mathcal{A}_{\mathrm{R}}|}$, and $\mathbf{x}^{\mathcal{W}} = \{\mathbf{x}^{\mathbf{w}} \in \mathbb{R}^{|\mathcal{A}|} \mid \mathbf{w}\in\mathcal{W}\}$. Constraints (5.4b) take care of flow conservation and demand compliance as in a multi-commodity transportation setting. Constraints (5.4c) ensure the rebalancing of the AMoD fleet (only on the road network). The last sets of constraints (5.4d) restrict the flows to non-negative values.

The objective $J(\mathbf{x})$ is composed of two terms. The first term considers the total travel time of AMoD users. This evaluates the travel time function $t_{ij}(x_{ij})$ with respect to the total flow given by (5.2) which includes variables corresponding to private vehicle flow x^p_{ij} (assumed to be fixed), and the rebalancing flow x^r_{ij}. When taking the product $t_{ij}(x_{ij})x^u_{ij}$, we obtain a non-convex function which makes the problem hard to solve. To address this issue, we use a piecewise-affine approximation of $t_{ij}(x_{ij})$ which is further developed in Section 5.3.

The second term acts as a linear regularizer whose purpose is to penalize rebalancing flows. This will ensure that a cost for rebalancing of the fleet are taken into account in the optimization problem. We use $c_{ij} = \lambda t^0_{ij}$ where λ is a constant. Therefore, we use a small λ to guide the rebalancing flow through good paths, without dominating the AMoD user routing decisions. Note that, if normalization is needed to ensure a good regularization parameter, we can always bound each component on (5.4a) using the link capacities and a large enough value for $t(\cdot)$.

5.2.2 Private Vehicles Flow Modeling

Aiming to understand the interaction between a SO AMoD fleet and private vehicles, we assume some user-choice model behind private vehicle decisions. To do so, we use the *User-Centric* (UC) routing as in the Traffic Assignment Problem (TAP). Given OD demands, this model finds the flows in the network which achieve a Wardrop equilibrium (Wardrop, 1952). This is equivalent to each private user deciding to take the route that minimizes their own travel time.

We impose that private vehicles can travel exclusively through the road network $\mathcal{G}_\mathrm{R}$ as opposed to traveling in the full network $\mathcal{G}$. Let $x^{p,\mathbf{w}}_{ij}$ be the flow on link (i,j) induced by private vehicle demand $d^p_\mathbf{w}$ of OD pair $\mathbf{w}$. Then the UC problem of private vehicles is

$$\min_{\mathbf{x}^p} \quad \sum_{(i,j)\in\mathcal{A}_\mathrm{R}} \int\limits_{x^u_{ij}+x^r_{ij}}^{x_{ij}} t_{ij}(s)ds \tag{5.5a}$$

$$\text{s.t} \quad \sum_{i:(i,j)\in\mathcal{A}_\mathrm{R}} x^{p,\mathbf{w}}_{ij} + d^p_\mathbf{w}\mathbb{1}_{j=w_s} = \sum_{k:(j,k)\in\mathcal{A}_\mathrm{R}} x^{p,\mathbf{w}}_{jk} + d^p_\mathbf{w}\mathbb{1}_{j=w_t}, \quad \forall \mathbf{w}\in\mathcal{W}, j\in\mathcal{V}_\mathrm{R}, \tag{5.5b}$$

$$\mathbf{x}^{p,\mathbf{w}} \geq \mathbf{0}. \tag{5.5c}$$

Notice that this version of the UC TAP is slightly different to the classic one (Patriksson, 1994) since it considers the AMoD flow in its objective (see limits of the integral on

(5.5a)).

To solve this problem we assume that the AMoD flow is fixed and private vehicles plan their routes considering AMoD flows as exogenous. By assuming this, we can use the *Frank-Wolfe* algorithm (Frank & Wolfe, 1956) to solve (5.5). Let us use the shorthand notation of $\texttt{TAP}(\mathbf{g}, \mathbf{x}^e)$ to indicate the solution of (5.5) with $\mathbf{x}^e$ equal to any generic exogenous flow. Hence $\mathbf{x}^p \in \texttt{TAP}(\mathbf{g}^p, \mathbf{x}^u + \mathbf{x}^r)$.

5.2.3 AMoD in Mixed Traffic

Critically, AMoD flows react to the decisions made by private vehicles and these, in turn, react to AMoDs' flows. Hence, whenever private vehicles make their routing decisions, the AMoD fleet adjusts theirs, and vice versa. This creates a nested optimization problem between these two classes of vehicles.

To give a formal definition of this game-theoretical problem we use the following bilevel optimization formulation

$$\min_{\{\mathbf{x^w}\}_{\mathbf{w}\in\mathcal{W}}, \mathbf{x}^r, \mathbf{x}^p} \; J(\mathbf{x}) \tag{5.6a}$$

$$\begin{aligned} \text{s.t.} \quad & (5.4\text{b}) - (5.4\text{d}), \\ & \mathbf{x}^p \in \texttt{TAP}(\mathbf{g}^p, \mathbf{x}^u + \mathbf{x}^r), \end{aligned} \tag{5.6b}$$

which has the same structure as (5.4) with the additional constraint (5.6b). The latter constraint refers to the TAP (the lower-level problem), which depends on the solution of the full problem (upper-level). Note that the upper-level problem is minimizing over the AMoD users, rebalancing, and privately-owned vehicle flows.

This phenomenon has been studied in Harker (1988) and Yang et al. (2007) where the existence and uniqueness of Nash equilibria points is discussed. To compute the solution to the problem, we leverage the iterative approach (diagonalization scheme) presented in Chapter 3 of this thesis to compute the private vehicles' and AMoD flows.

5.3 AMoD Routing and Rebalancing Problem

The problem of routing and rebalancing as stated in (5.4) is non-convex for typical travel time functions such as the BPR. This happens due to the term $t(x_{ij})x^r_{ij}$ in the objective function. To address this issue, we approximate the travel time function with a piecewise-affine function. The results presented here are key contributions of this Chapter and of the overall dissertation.

5.3.1 Piecewise-affine Approximation

Let the function approximating $t(x)$ be of the form:

$$\hat{t}_{ij}(x) = \begin{cases} t^0_{ij}\left(1 + a_1 \dfrac{(x - \theta^{(1)}_{ij})}{m_{ij}}\right), & \text{if } \theta^{(1)}_{ij} \le x \le \theta^{(2)}_{ij} \\ \qquad \vdots & \\ t^0_{ij}\left(1 + \sum_{l=1}^{n} \left(\dfrac{a_l(\theta^{(l)}_{ij} - \theta^{(l-1)}_{ij})}{m_{ij}}\right) + \dfrac{a_n(x - \theta^{(n)}_{ij})}{m_{ij}}\right), & \text{if } \theta^{(n)}_{ij} \le x, \end{cases}$$

where a_l is the slope of segment $l = 1, \dots, n$ of $\hat{t}$ with $a_1 \le \dots \le a_n < \infty$, and $\theta^{(l)}_{ij}$ is a threshold dividing segments on the travel time function for link (i, j). In our case, we let $\theta^{(l)}_{ij} = \theta^{(l)} m_{ij}$ where $\theta^{(l)}$ is the normalized threshold in the travel time and capacity-normalized function depicted in Fig. 5·2.

To model this piecewise-affine function in the optimization problem, we introduce the following set of slack variables

$$\varepsilon^{(n)}_{ij} = \max\{0,\ x_{ij} - \theta^{(n)}_{ij}\}, \tag{5.8a}$$

$$\vdots$$

$$\varepsilon^{(k)}_{ij} = \max\{0,\ x_{ij} - \theta^{(k)}_{ij} - \sum_{l=k+1}^{n} \varepsilon^{(l)}_{ij}\}, \tag{5.8b}$$

$$\vdots$$

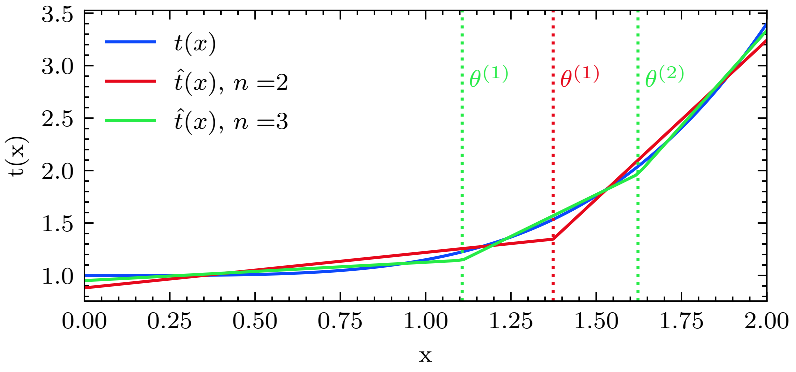


Figure 5·2: Travel time function approximation.

$$\varepsilon_{ij}^{(0)} = \max\{0,\ x_{ij} - \sum_{l=1}^{n} \varepsilon_{ij}^{(l)}\}, \tag{5.8c}$$

where each $\varepsilon_{ij}^{(k)}$ denotes the extra flow exceeding threshold $\theta_{ij}^{(k)}$ and up to $\theta_{ij}^{(k+1)} - \theta_{ij}^{(k)}$, thus, $\varepsilon_{ij}^{(k)} \in [0, \theta_{ij}^{(k+1)} - \theta_{ij}^{(k)}]$. We include these variables in the problem by adding the linear constraints $\varepsilon_{ij}^{(k)} \geq 0$ and $\varepsilon_{ij}^{(k)} \geq \theta_{ij}^{(k)} - \sum_{l=k+1}^{n} \varepsilon_{ij}^{(l)}$, provided that the objective is a function of $\varepsilon_{ij}^{(k)}$.

Using these definitions we can generate a tractable cost function. We focus our attention on an element-wise analysis of the first term (non-convex part) of objective function (5.4a) using $\hat{t}$ instead of t for which we obtain the objective function

$$\hat{J}(x_{ij}, \boldsymbol{\varepsilon}_{ij}) := t_{ij}^{0}\Big(1 + \sum_{l=1}^{n}(a_l \varepsilon_{ij}^{(l)}/m_{ij})\big)\big(\sum_{k=1}^{n}\varepsilon_{ij}^{(k)} - x^p\Big), \tag{5.9}$$

derived as follows:

$$\hat{t}_{ij}(x_{ij})x_{ij}^{u} = t_{ij}^{0}\Big(1 + \sum_{l=1}^{n}(a_l \varepsilon_{ij}^{(l)}/m_{ij})\Big)x_{ij}^{u}, \tag{5.10a}$$

$$= t_{ij}^{0}\Big(1 + \sum_{l=1}^{n}(a_l \varepsilon_{ij}^{(l)}/m_{ij})\Big)\Big(\sum_{k=1}^{n}\varepsilon_{ij}^{(k)} - x^r - x^p\Big), \tag{5.10b}$$

$$\leq t_{ij}^{0}\Big(1 + \sum_{l=1}^{n}(a_l \varepsilon_{ij}^{(l)}/m_{ij})\Big)\Big(\sum_{k=1}^{n}\varepsilon_{ij}^{(k)} - x^p\Big). \tag{5.10c}$$

In (5.10b) we express x_{ij}^u by using a combination of (5.2) and (5.8). In the last step (5.10c), we add the term $\sum_{l=1}^n t_{ij}^0 a_l \varepsilon_{ij} x_{ij}^r / m_{ij}$ to $\hat{J}_{ij}$.

By adding this term, we consider a relaxation of the original problem (i.e., an upper bound of $\hat{J}_{ij}$). This modification, which is detailed later (see Remark 4), allows our proposed objective to be a convex quadratic function. Hence, we define the AMoD problem to be

$$\begin{aligned}
\min_{\mathbf{x}^{\mathcal{W}},\mathbf{x}^r,\boldsymbol{\varepsilon}} \quad & \sum_{(i,j)\in\mathcal{A}} \hat{J}(x_{ij},\boldsymbol{\varepsilon}_{ij}) + \sum_{(i,j)\in\mathcal{A}_{\mathrm{R}}} c_{ij} x_{ij}^r, && (5.11a)\\
\text{s.t.} \quad & (5.4b) - (5.4d) \\
& \varepsilon_{ij}^{(k)} \geq \theta_{ij}^{(k)} - x_{ij}, \quad \forall (i,j)\in\mathcal{A}, \quad k=1,\ldots,n, && (5.11b)\\
& \varepsilon_{ij}^{(k)} \geq 0, \qquad\qquad \forall (i,j)\in\mathcal{A}, \quad k=1,\ldots,n, && (5.11c)
\end{aligned}$$

where $\boldsymbol{\varepsilon} = \{\varepsilon_{ij}^{(j)} \mid (i,j)\in\mathcal{A}, \quad k=1,\ldots,n\}$. Now, we present the main analytical result of this work.

Theorem 1. *Problem* (5.11) *is a linearly constrained convex Quadratic Program (QP) with linear equality constraints.*

Proof. We prove this by construction. We show that the $\mathbf{Q}$ matrix in the QP standard form (i.e., $\min_{\mathbf{x}} \mathbf{x}'\mathbf{Q}\mathbf{x}$, s.t. $\mathbf{A}\mathbf{x} \leq \mathbf{b}$) can be modified to be positive semidefinite (PSD). Note that in (5.11a) the only quadratic term is of the form $\varepsilon_{ij}^{(l)}\varepsilon_{ij}^{(k)}$ and its matrix representation (i.e., $\boldsymbol{\varepsilon}'\mathbf{Q}\boldsymbol{\varepsilon}$) does not guarantee that $\mathbf{Q}$ is PSD. However, we observe that since we are minimizing, when $x_{ij} \leq \theta_{ij}^{(k)}$ then $\varepsilon_{ij}^{(k)} = 0$ and when $x_{ij} \geq \theta_{ij}^{(k+1)}$ then $\varepsilon_{ij}^{(k)} = (\theta_{ij}^{(k+1)} - \theta_{ij}^{(k)})$. Therefore,

$$\varepsilon_{ij}^{(l)}\varepsilon_{ij}^{(k)} = \begin{cases} (\theta_{ij}^{(l+1)} - \theta_{ij}^{(l)})\varepsilon_{ij}^{(k)}, & \text{if } l < k, \\ \varepsilon_{ij}^{(l)}\varepsilon_{ij}^{(l)}, & \text{if } l = k, \\ \varepsilon_{ij}^{(l)}(\theta_{ij}^{(k+1)} - \theta_{ij}^{(k)}), & \text{if } l > k, \end{cases}$$

where the first case comes from the fact that in order for $\varepsilon_{ij}^{(k)}$ to be greater than zero, the flow x_{ij} must have exceeded $\theta_{ij}^{(l+1)}$ for $l < k$. Therefore, $\varepsilon_{ij}^{(l)}$ is at its maximum value of $(\theta_{ij}^{(l+1)} - \theta_{ij}^{(l)})$. The same analogy applies to the third case. Hence, the link-wise objective function of the QP without the rebalancing term is rewritten as

$$\hat{J}_{ij}^{\mathrm{QP}}(x_{ij}^u, \boldsymbol{\varepsilon}_{ij}) = t_{ij}^0 \Bigg(x_{ij}^u + \sum_{l=1}^{n} \frac{a_l}{m_{ij}} \Big(\sum_{k=1}^{l-1} (\theta_{ij}^{(k+1)} - \theta_{ij}^{(k)}) \varepsilon_{ij}^{(l)} + (\varepsilon_{ij}^{(l)})^2 + \sum_{k=l+1}^{n} (\theta_{ij}^{(l+1)} - \theta_{ij}^{(l)}) \varepsilon_{ij}^{(k)} \Big) \Bigg).$$

Using this new formulation, we note that the $\mathbf{Q}$ matrix is the identity matrix which is PSD and therefore J_{ij}^{QP} is convex quadratic using (Bertsekas, 2016, Prop. 3.1.1). □

A relevant trade-off worth noting is on the number of piecewise affine segments used to approximate the travel latency function. Even though a larger n will provide better approximations of $t(\cdot)$, and hence a better solution to the problem, this implies adding $|\mathcal{A}|$ additional variables and linear constraints to the formulation for every additional segment included in the estimation.

Remark 4. *We observe that the effect of adding* $\sum_{l=1}^{n} t_{ij}^0 a_l \varepsilon_{ij} x_{ij}^r / m_{ij}$ *to* (5.10a) *implies taking into account congestion-aware rebalancing routing. However, this congestion-aware routing of the rebalancing vehicles has a lower impact in* J_{ij}^{QP} *than the AMoD users. This is because* $a_0 = 0$ *for* $\mathbf{x}^r$ *(i.e., the first term in* (5.10c) *does not include* $\mathbf{x}^r$*). Hence, the interpretation of this addition is that the rebalancing flows evaluate the travel latency function with the same structure as the AMoD flows but with* $t_{ij}^0 = 0$.

5.3.2 Linear Relaxation

Seeking a simpler formulation and faster computation performance of (5.11), we notice that it is possible to relax the QP to a Linear Program (LP) by modifying the only quadratic term in (5.11a), i.e., $(\varepsilon_{ij}^{(l)})^2$. We approximate this using $\varepsilon_{ij}^{(l)} \theta_{ij}^{(l+1)}$ and observe that when $x_{ij} \leq \theta_{ij}^{(l)}$ or $x_{ij} \geq \theta_{ij}^{(l+1)}$ we recover exactly $(\varepsilon_{ij}^{(l)})^2$. However, a gap exists when $x_{ij} \in (\theta_{ij}^{(l)}, \theta_{ij}^{(l+1)})$ which can be diminished by adding more linear segments to $\hat{t}(\cdot)$ and consequently decreasing the range of $(\theta_{ij}^{(l)}, \theta_{ij}^{(l+1)})$.

Lemma 5.3.1. *Assuming the distance between the break points of the linear segments is uniform, i.e.,* $\theta_{ij}^{(l+1)} - \theta_{ij}^{(l)} = \theta^{(n)}/n$*, for* $l = 1, \ldots, n-1$*, then the objective function of* *the LP formulation approximates the QP objective function by an error upper-bounded by* $a_n(\theta^{(n)})^2/4n^2)\sum_{(i,j)\in\mathcal{A}} t^0_{ij} m_{ij}$.

Proof. See Appendix A.0.1. $\square$

Theorem 2. *Let the total flow and the capacity of every link be upper-bounded and assume* $a_0 \leq a_1 \leq, \dots, \leq a_n < \infty$. *Then, as* $n \to \infty$, *the solution of the LP problem recovers the solution of the QP.*

Proof. See Appendix A.0.1 $\square$

Interestingly, these two reformulations, QP and LP, together with Theorems 1 and 2 show that an LP can be solved instead of the original convex program described in (5.4). This LP approximates the solution of the QP which, in turn, approximates the solution of the original problem. These two are asymptotically optimal in the number of segments used to describe the nonlinear function $t(\cdot)$ in the objective.

5.3.3 Origin-based Formulation (Flow-bundling)

So far, we have formulated the problem such that for every OD pair $\mathbf{w} \in \mathcal{W}$ we introduce $|\mathcal{A}|$ decision variables. The total number of variables in our QP (or LP) is then $(n + 1 + |\mathcal{W}|)|\mathcal{A}|$, which is typically dominated by the number of OD pairs $|\mathcal{W}|$. In practice, this number can be very large, sometimes up to $|\mathcal{V}|^2$. Hence, solving the problem using the previous formulations may require large memory capabilities.

To mitigate this issue, we leverage similar ideas to Rossi et al. (2020) which aggregate flows by origin with the objective to reduce the number of variables and constraints of the QP and LP without losing information. This flow aggregation by origin allows to reduce the number of variables to be in the order of $(n + 1 + |\mathcal{V}|)|\mathcal{A}|$, which makes the problem significantly faster to solve.

Let us denote the set of origin (sources) $\mathcal{S} = \{w_s : d^u_{(w_s,w_t)} > 0, \ \forall (w_s, w_t) \in \mathcal{W}\}$ and the flow on the network with $s \in \mathcal{S}$ as it source by $\mathbf{x}^s$; the total user flow on a link is then $\mathbf{x}^u = \sum_{s\in\mathcal{S}} \mathbf{x}^s$ and the set of user origin-link variables be $\mathbf{x}^{\mathcal{S}} = \{\mathbf{x}^s \mid s \in \mathcal{S}\}$.

For every origin s, let $\psi^s(j)$ be the node *imbalance* describing the excess demand or supply at each node. This is

$$\psi^s(j) = \begin{cases} \sum_{t:(s,t)\in\mathcal{W}} -d^u_{(s,t)}, & \text{if } j = s, \\ 0, & \text{if } j \neq s, t, \\ d^u_{(s,t)}, & \text{if } j = t. \end{cases}$$

Using this definition in hand, we establish the origin-based problem as follows

$$\begin{aligned} \min_{\mathbf{x}^{\mathcal{S}}\geq\mathbf{0},\mathbf{x}^r\geq\mathbf{0}} \quad & \sum_{(i,j)\in\mathcal{A}} \hat{J}_{ij}(x_{ij}, \boldsymbol{\varepsilon}_{ij}) + \sum_{(i,j)\in\mathcal{A}_{\mathrm{R}}} c_{ij}x^r_{ij} && \text{(5.14a)} \\ \text{s.t.} \quad & \sum_{i:(i,j)\in\mathcal{A}} x^s_{ij} - \sum_{k:(j,k)\in\mathcal{A}} x^s_{jk} = \psi^s(j), \quad \forall j \in \mathcal{N},\ \forall s \in \mathcal{S}, && \text{(5.14b)} \\ & \sum_{i:(i,j)\in\mathcal{A}_{\mathrm{R}}} \big(x^r_{ij} + x^u_{ij}\big) = \sum_{k:(j,k)\in\mathcal{A}_{\mathrm{R}}} \big(x^r_{jk} + x^u_{jk}\big), \forall j \in \mathcal{V}_{\mathrm{R}}, && \text{(5.14c)} \\ & \text{(5.11b)},\ \text{(5.11c)}, \end{aligned}$$

where $x_{ij} = x^u_{ij} + x^r_{ij} + x^p_{ij} = \sum_{s\in\mathcal{S}} x^s_{ij} + x^r_{ij} + x^p_{ij}$.

We proceed to show that the resulting flows of the solution of the origin-based problem (5.14) are the same as the OD-based problem (5.11). To accomplish this, we use the result below.

Lemma 5.3.2. *Let $\mathbf{x}^{\mathcal{S}*}$ be the solution to the origin-based problem* (5.14) *and $\mathbf{x}^{s*}$ the flows associated with origin s. Then the subset of arcs $\mathcal{A}^{s*} = \{(i,j) : x^s_{ij} > 0,\ \forall (i,j) \in \mathcal{A})\}$ with positive flow from origin s has no direct cycles.*

Proof. See Appendix A.0.1 □

Lemma 5.3.3. *The link-flow solution of the origin-based problem* (5.14) *is equivalent to the solution of the OD-based problem* (5.11)*. i,e,. for any origin s , we have $\sum_{t:(s,t)\in\mathcal{W}} x^{\mathbf{w}*} = \mathbf{x}^{s*}$.*

Proof. See Appendix A.0.1 □

Therefore, using the result of Lemma 5.3.3, we can restate the network model in terms of the origin-based flows which reduces the size of the model, memory requirements, and solution time.

5.3.4 Disjoint Strategy

We have discussed methods to solve the SO routing and rebalancing problem jointly. Yet another approach is to tackle these two problems separately. Explicitly, we initiate the method by setting $\mathbf{x}^r = 0$ and then we repeatedly solve the routing problem followed by the rebalancing problem. Mathematically, this is to first solve

$$\min_{\mathbf{x}^{\mathcal{W}} \geq 0} \quad \sum_{(i,j)\in\mathcal{A}} t_{ij}(x_{ij})x_{ij}^u \tag{5.15}$$

$$\text{s.t.} \quad \sum_{i:(i,j)\in\mathcal{A}_{\mathrm{R}}} \left(x_{ij}^r + x_{ij}^u\right) = \sum_{k:(j,k)\in\mathcal{A}_{\mathrm{R}}} \left(x_{jk}^r + x_{jk}^u\right), \forall j \in \mathcal{V}_{\mathrm{R}}, \tag{5.16}$$

followed by using the optimal $\mathbf{x}^{u*}$ to solve

$$\min_{\mathbf{x}^r \geq 0} \quad \mathbf{c}'\mathbf{x}^r \tag{5.17}$$

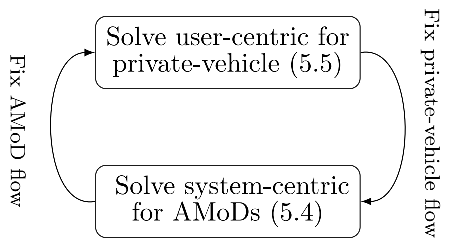
$$\text{s.t.} \quad \sum_{i:(i,j)\in\mathcal{A}_{\mathrm{R}}} \left(x_{ij}^r + x_{ij}^u\right) = \sum_{k:(j,k)\in\mathcal{A}_{\mathrm{R}}} \left(x_{jk}^r + x_{jk}^u\right), \forall j \in \mathcal{V}_{\mathrm{R}}, . \tag{5.18}$$

It is relevant to highlight that this strategy is interesting given its fast computation. Problem (5.16) is a constrained nonlinear program (NLP) which can be solved using any of the typical algorithms for the TAP, for example Frank-Wolfe or TAPAS; and problem (5.18) is a LP with $|\mathcal{V}|$ variables.

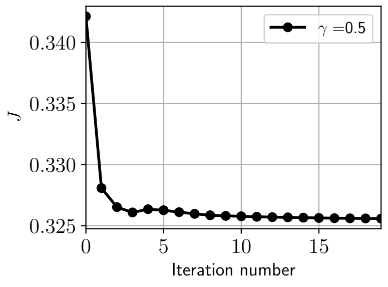
5.4 AMoD in Mixed Traffic

We have not yet discussed how to address the nested problem (5.6) which considers the interaction between the fleet of AMoDs vehicles and self-interested private vehicles. We employ the framework in Chapter 3 which applies a sequential approach (diagonalization scheme Harker (1988); Houshmand et al. (2019); Yang et al. (2007)) to find an equilibrium between the AMoD and private flows.

Rather than addressing the bilevel problem (5.6), we solve (5.5) for the private vehicles and (5.4) for the AMoD fleet (using any of the methods in the previous section) and iterate until convergence. Namely, for a private vehicle demand $\mathbf{g}^u$ we solve $\mathbf{x}^p = \texttt{TAP}(\mathbf{g}^p, \mathbf{0})$. Thereafter, we solve (5.4) for AMoD demand $\mathbf{g}^u$ with fixed input $\mathbf{x}^p$ (the output of the earlier solved TAP). Since private vehicles were not aware of the AMoD flow in the system while finding their routes, we re-solve the TAP by considering a fixed AMoD flow equal to $\mathbf{x}^u + \mathbf{x}^r$, i.e., we solve $\mathbf{x}^p = \texttt{TAP}(\mathbf{g}^p, \mathbf{x}^u + \mathbf{x}^r)$. Further, we iterate this process until it converges. An example is shown in Fig. 5·3b.


Solve user-centric for private-vehicle (5.5)
Solve system-centric for AMoDs (5.4)
Fix AMoD flow
Fix private-vehicle flow

(a)



(b)

Figure 5·3: (a): A sketch of the procedure for solving the bilevel problem (5.6). (b): An example of the total cost converging for an AMoD penetration rate of 0.5 on the NYC sub-network.

Remark 5. *Notice that when employing this iterative method, some of the parameters can be updated. In particular, if one uses the disjoint strategy to solve the routing and rebalancing problem, one could update the* $\mathbf{c}$ *vector at each subsequent iteration by the calculated travel times* $\mathbf{t}(\mathbf{x})$ *at the current iteration. When doing this, one obtains a more precise cost function by weighting vector* $\mathbf{c}$ *with the updated travel times.*

5.5 Route Recovery Strategies

All methods discussed thus far solve the routing and rebalancing problem by choosing $\mathbf{x}^u$ and $\mathbf{x}^r$ that minimize a performance metric. Even if this flow solution allows us to assess the network deficiencies and to plan for infrastructure improvements, flows do not give explicit routes to a given vehicle. Therefore, we need to extract the routes to implement the desired flow-based solution we derive. An advantage of the proposed models in (5.11) and (5.14) in contrast to classical link-based TAP is the fact that they allow for tracing and recovering the routes (or paths).

5.5.1 AMoD User Flow

OD-pair model

Let the optimal solution of the routing and rebalancing problem be $(\mathbf{x}^{\mathcal{W}*}, \mathbf{x}^{r*})$ and denote with $\mathcal{R}_\mathbf{w}$ a set of routes for OD pair $\mathbf{w}$. For each $\mathbf{w}$, we let $\boldsymbol{\pi} \in [0,1]^{|\mathcal{R}_\mathbf{w}|}$ be a vector with elements denoting the fraction of vehicle flow routed through route $i \in \mathcal{R}_\mathbf{w}$. We denote with $\mathbf{A}$ the route-link incidence matrix of $\mathcal{R}_\mathbf{w}$. With these definitions, we provide a column-generation approach in which we find the routes of an OD-pair by sequentially solving the linear program

$$\min_{\boldsymbol{\pi}\in[0,1]} \quad \|\mathbf{A}\boldsymbol{\pi} d_\mathbf{w} - \mathbf{x}^{\mathbf{w}*}\| \tag{5.19a}$$

$$\text{s.t.} \quad \boldsymbol{\pi}'\mathbf{1} = 1, \tag{5.19b}$$

where the product $\mathbf{A}\boldsymbol{\pi} d_\mathbf{w}$ is equal to the estimated link flow induced by routing $d_\mathbf{w}\pi_i$ flow through each route. The constraint ensures that the vector $\boldsymbol{\pi}$ is a probability distribution.

To address the problem of selecting which routes to include in $\mathcal{R}_\mathbf{w}$ (column selection) we use the greedy approach of adding the next shortest route to $\mathcal{R}_\mathbf{w}$ and re-solving problem (5.19). To terminate the algorithm, we employ a user-defined parameter ξ

(as shown in Alg. 4).

It is worth pointing out that this procedure can run in parallel for each OD pair. For uncongested networks, we expect it to converge fast. This is because when there is little congestion, the majority of vehicles will be routed through the shortest paths, which are the first ones to be added to the set $\mathcal{R}_{\mathbf{w}}$. Finally, note that this formulation is only available if we have information on $\mathbf{x}^{\mathbf{w}*}$ for all $\mathbf{w} \in \mathcal{W}$.

Algorithm 4 Route-recovery for a specific OD pair

```
1: procedure RouteRecovery($\mathbf{A}$, $\mathbf{d_w}$, $\mathbf{x}^{\mathbf{w}*}$, $\xi$)
2:     Initialize: $x_{ij} \leftarrow d_{\mathbf{w}} \mathbb{1}_{(i,j)\in \text{shortest route for } \mathbf{w}}$
3:     while $\|\mathbf{x} - \mathbf{x}^{\mathbf{w}*}\| > \xi$ do
4:         $\mathcal{R}_{\mathbf{w}} \leftarrow$ append next shortest path
5:         $\boldsymbol{\pi}_{\mathbf{w}} \leftarrow$ solve (5.19)
6:     end while
7: end procedure
```

Origin-based model

Let $\mathbf{x}^{s*}$ be the solution of (5.14) and let $\mathcal{T}_s = \{j : \ \psi_s(j) < 0, \ \forall j \in \mathcal{V}\}$ be the set of destinations (targets) from origin s. Let $\psi_s(j)$ be the node imbalance of node j of the origin-based flows initialized at s. For each origin s, one can decompose its OD-flow solution by solving the following LP:

$$\min_{\{\mathbf{x}^t\}_{t\in\mathcal{T}_s}\geq 0} \quad \mathbf{t}^{0\prime}\mathbf{x} \tag{5.20a}$$

$$\text{s.t} \quad \sum_{i:(i,j)\in\mathcal{A}} x_{ij} - \sum_{k:(j,t)\in\mathcal{A}} x_{tj} = \psi_s(t), \quad \forall j \in \mathcal{V}, \tag{5.20b}$$

$$\sum_{i:(i,j)\in\mathcal{A}} x_{ij}^t - \sum_{k:(j,k)\in\mathcal{A}} x_{ij}^t \geq 0, \quad \forall j \in \mathcal{V}\backslash\{s\}, \tag{5.20c}$$

$$\sum_{i:(i,s)\in\mathcal{A}} x_{is}^t - \sum_{k:(s,k)\in\mathcal{A}} x_{sj}^t = \psi_s(t), \tag{5.20d}$$

$$\mathbf{x}^{s*} - \mathbf{x} = \mathbf{0}. \tag{5.20e}$$

In (5.20), $\mathbf{x}$ is the origin-based flow defined by $\mathbf{x} = \sum_{t \in \mathcal{T}_s} \mathbf{x}^t$. The first constraint, (5.20c), takes care of demand satisfaction and flow conservation. The second constraint, (5.20c), considers flow conservation but allows certain target nodes to have excess flow, allowing them to be a destination. Constraint (5.20d) ensures that the decision variables are designed for that specific origin $s \in \mathcal{S}$ and (5.20e) forces the solution to be equal to the origin-based flows. Finally, the objective (5.20a) is defined with the purpose of breaking ties in case multiple combinations of flows can satisfy the constraints (e.g. cycles).

Notice that as a result of Lemma 5.3.3, this problem is always feasible and recovers the OD-based solution. Once this is established, we could use Alg. 4 to find the path-based solution. Problem (5.20) is stated as a linear program that could be solved in parallel for each origin-based solution s, therefore, we expect this optimization process to be computationally efficient.

5.5.2 Rebalancing Flows

The problem of finding the paths of the rebalancing flows is more complex than that of finding the AMoD routes. This is because we have no information about their origin and destinations. Rather, the only information available is the aggregated link flows that the rebalancing vehicles are taking to minimize (5.4a) and comply with the load-balancing constraint (5.4c). Hence, a first step to recover the paths is to calculate the rebalancing node *imbalances* $\phi(j)$ for every node j defined over the available rebalancing solution x^r:

$$\phi(j) = \sum_{i:(i,j) \in \mathcal{A}_{\mathrm{R}}} x^r_{ij} - \sum_{k:(j,k) \in \mathcal{A}_{\mathrm{R}}} x^r_{jk}.$$

We define a rebalancing origin to be a deficit flow node, and its set $\mathcal{S}_r = \{j : \phi(j) < 0, \forall j \in \mathcal{A}_{\mathrm{R}}\}$; similarly, the rebalancing destination set is defined as $\mathcal{T}_r = \{j : \phi(j) >$

0, $\forall j \in \mathcal{A}_{\mathrm{R}}\}$. Notice that these definitions are made in $\mathcal{A}_{\mathrm{R}}$ and not in $\mathcal{A}$, as the rebalancing vehicles only exist in G_{R}. Then we aim to recover an OD rebalancing solution by solving

$$\min_{\{\mathbf{x}^s\}_{s\in\mathcal{S}_r}\geq 0} \quad \mathbf{t}^{0\prime}\mathbf{x} \tag{5.21a}$$

$$\text{s.t} \quad \sum_{i:(i,j)\in\mathcal{A}_{\mathrm{R}}} x_{ij} - \sum_{k:(j,k)\in\mathcal{A}_{\mathrm{R}}} x_{jk} = \phi(j), \quad \forall j \in \mathcal{V}_{\mathrm{R}}, \tag{5.21b}$$

$$\sum_{i:(i,j)\in\mathcal{A}_{\mathrm{R}}} x_{is}^s - \sum_{k:(s,k)\in\mathcal{A}_{\mathrm{R}}} x_{sj}^s = \phi(s), \quad \forall s \in \mathcal{S}_r, \tag{5.21c}$$

$$\sum_{i:(i,j)\in\mathcal{A}_{\mathrm{R}}} x_{ij}^s - \sum_{k:(j,k)\in\mathcal{A}_{\mathrm{R}}} x_{ij}^s \geq 0,\ \forall j \in \mathcal{V}_{\mathrm{R}}\backslash\{s\},\ \forall s \in \mathcal{S}_r, \tag{5.21d}$$

$$\mathbf{x} - \mathbf{x}^r = 0, \tag{5.21e}$$

where we define $\mathbf{x} = \sum_{s\in\mathcal{S}_r}\mathbf{x}^s$ and $\mathbf{x}^r$ is the available link flow solution of (5.4a). Notice that the model follows the same intuition as (5.20). Constraint (5.21b) takes care of the total flow conservation of the rebalancing flow, constraint (5.21c) ensures that, for each origin variables $\mathbf{x}^s$, the outflow of node s is equal to the excess of vehicles. Constraint (5.21d) allows any node different than s to be a potential destination of the rebalancing flow. Finally, (5.21e) ensures that the aggregated rebalancing flows by origin match the rebalancing flow obtained in the AMoD user problem.

Once we have decomposed the rebalancing flow by origins, we have for each rebalance origin s an origin-based rebalancing flow. Since now we have the flows available in an origin-based form, we can apply (5.20) in parallel for each $s \in \mathcal{S}_r$ to decompose to an OD-flow solution, and finally use Alg. 4 to recover the routes.

For both of (5.20) and (5.21) it is possible to dualize the last constraint (i.e., penalize $\|\mathbf{x} - \mathbf{x}^r\|$ on the cost function). This makes the optimization less restrictive and improves the solution time by lowering the quality of the solution. It is difficult to estimate exactly what the impact of this dualization would be in terms of efficiency.

However, for low-traffic networks, we expect (5.20) and (5.21) to be faster to solve as we expect the total flow on every link will belong to fewer OD pairs. Conversely, when dealing with high-traffic scenarios, the total flow on a link might be composed of many OD pairs, making the problem harder (slower) to decompose.

In practice we have observed that for low-traffic networks less than 3 routes per OD pair are enough to obtain an accurate solution, whereas for high-traffic cases, the number of routes required for good solutions are in the order of 6 to 8. Still, the problems as stated in this section can be solved to optimality.

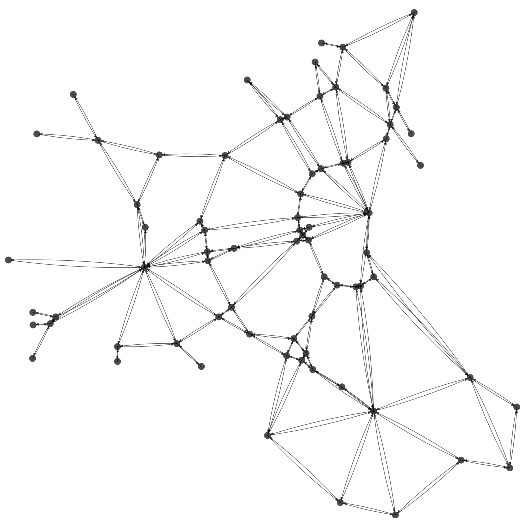
5.6 Numerical Results and Case Studies

To validate our proposed routing algorithms, we consider two data-driven case studies on sub-networks of Eastern Massachusetts interstate highways (EMA) and New York City (NYC). The EMA road network (Figure 5·4a) consists of 74 nodes, 258 links, and 1113 OD pairs, and it captures the dynamics in the context of suburban/urban mobility. Complementary to EMA, the NYC network focuses on urban mobility. The NYC topology was constructed using OpenStreetMap (2017) and contains 3317 arcs, 1351 nodes. The OD demand was built using historical data taxi rides (courtesy of the NYC taxi and Limousine Commission (2020)) that occurred on March 1, 2012, between 18:00 and 20:00 hrs which accounts for 8658 OD pairs.

5.6.1 Convergence of the approximated model

Our first experiment shows empirically our results of Theorem 1 and the observation that as n increases, the approximation of $\hat{t}(\cdot)$ to $t(\cdot)$ becomes tighter and therefore the QP and LP problems approximate the original problem more accurately.

To generate this experiment, we consider a problem with no rebalancing (not including the rebalancing constraints) and with no exogenous flow (i.e., $\mathbf{x}^c = 0$). This is exactly the SO formulation of the TAP for which we use the Frank-Wolfe algorithm


(a) EMA subnetwork

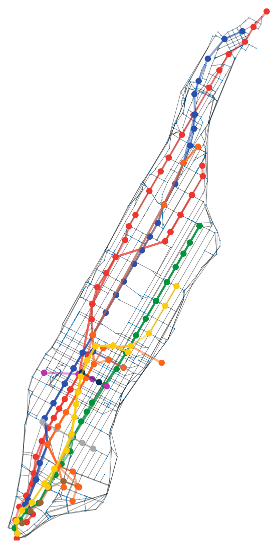

(b) NYC subnetwork

Figure 5·4: Subnetworks used for the experiments. Black lines indicate road links while colored lines indicate subway lines.

to find its solution (we solve the UC problem using the same method). Thereafter we solve the QP and LP versions of the CARSn model for different values of n and observe that, as we increase n, the objective of CARSn converges to the objective of the SO. For example, for both networks shown in Figure 5·5, we observe that for $n = 6$ the objective of the approximated models QP and LP are very close to the SO solution.

5.6.2 Joint vs. Disjoint Solution

This experiment aims to compare the solution of the joint and disjoint formulation of the problem. That is, solving (5.11) against the disjoint method in Section 5.3.4. We compare this by showing the improvement (ratio between the value of the objective functions) of the joint over the disjoint approach. For EMA and NYC we take account of an improvement in the objective of 3.85% and 0.91%, respectively. Moreover, we consider the case of NYC network with a higher demand, which we simulate by

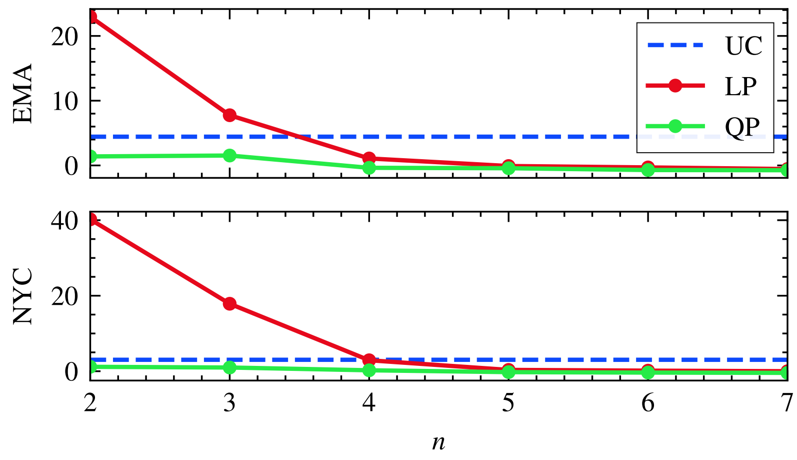


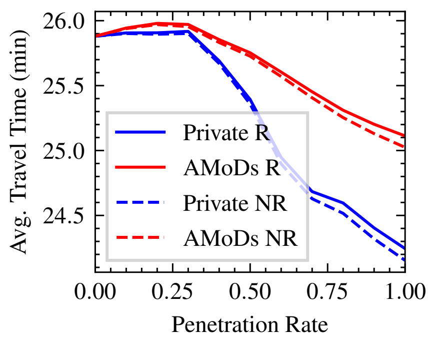
Figure 5·5: Deviation in percentage terms between the approximated model and the optimal solution of the non-rebalancing SO problem (baseline). UC indicates how much the solution of the UC deviates from the SO. This gap between the UC and SO models is referred to as the *Price of Anarchy* J. Zhang et al. (2018).

multiplying the demand vector $\mathbf{g}$ by 2. The improvement of the joint formulation over the disjoint model for this demand level is 5.85%. These results highlight the achievable benefits, especially for high demand scenarios, of jointly solving the routing and rebalancing problems, rather than separately.

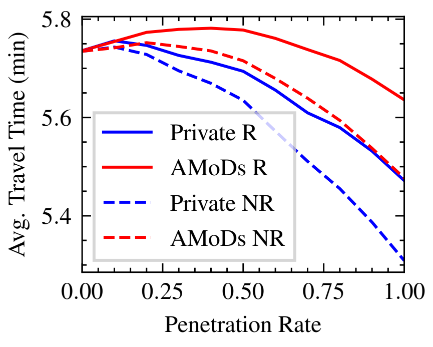
5.6.3 System-Optimal Routing and Rebalancing Trade-off

Considering the existence of selfish privately-owned vehicles and centrally-controlled AMoD vehicles, we analyze the trade-off that exists between system-optimal AMoD routing and the additional traffic due to AMoD rebalancing in terms of average travel times. We tackle the bilevel Problem (5.6) following the iterative methodology presented in Section 5.4. We use different *penetration rates* of AMoD customers with respect to the total demand, i.e., a penetration rate of 0.3 will indicate that 30% of the total demand uses the AMoD service while the rest use private vehicles. More specifically, we let $\gamma \in [0, 1]$ be the penetration rate and $\mathbf{g}$ the total OD demand. We assume that $\mathbf{g}^u = \gamma \mathbf{g}$ and $\mathbf{g}^p = (1 - \gamma)\mathbf{g}$ are the AMoD's and private vehicles' demand, respectively. However, different demand separation criteria can be readily implemented in this framework.

The general trend in Figure 5·6 shows that as the penetration rate of AMoD increases, its overall travel time decreases. More interestingly, not only the AMoD travel time is reduced, but also the private vehicles' travel time. This is because the collaborative routing decisions of the AMoD fleet lessen the traffic intensity on congested roads, which consequently allow private vehicles to travel faster. For low penetration rates, the addition of AMoDs could be detrimental as the negative effect of the new rebalancing flow on travel times is higher than the positive effect generated by better routing. For EMA, the impact of rebalancing is negligible, and increasing the percentage of AMoD users in the network allows to reduce travel time by up to 3%. For NYC, we observe that rebalancing indeed is detrimental to low penetration rates, but as the percentage of SO vehicles increases, social routing improves travel times for both AMoD users and private vehicles. Yet, in general, the impact of rebalancing on the system-level performance depends on the network topology, and on the symmetry and intensity of the OD demand distribution.



(a) EMA



(b) NYC

Figure 5·6: Travel times for AMoD users, private vehicles and all vehicles (total) for different penetration rates of AMoDs in the network."R" stands for an approach that considers rebalancing while "NR" does not.

5.6.4 Intermodal AMoD

We study the impact of intermodal SO routing against UC private vehicle routing for the NYC network. We consider high congestion levels and run the experiments by multiplying the demand distribution vector $\mathbf{g}$ by a factor of 1.5 (see details of the demand in the online repository[2]).

Similar to our last experiment, we run the analysis for different penetration rates. We assume that AMoD users are able to take public transit (subway), walk, or bike towards their destination and switch between modes in their route. In contrast to the AMoD users, we limit the flexibility of private vehicles to exclusively use the road network (no subway, biking or walking) due to parking constraints.

The top row of plots on Figure 5·7 display, on the left, the travel time for the two user types as the penetration rate of AMoD users increases and, on the right, the modal distribution of the total kilometers traveled. The top row shows the results when only taxi-type service is offered to AMoD users (no subway, walking or biking). We observe that the extra rebalancing flow increases the overall travel times of the system more than what SO routing can reduce. This result confirms the fact that pure vehicle-based MoD systems can have detrimental effects on the overall travel time (Fitzsimmons & Hu, 2017). The subsequent plots show that by considering the flexibility of other modes of transportation, AMoD mobility can reduce traffic congestion. The second row of plots in Figure 5·7 includes a public transit option, the third row adds a pedestrian option (6 km/h), and the last one also considers biking (10 km/h) as an option[3].

In general, we see that the more modes of transportation are offered, the lower the travel times for everyone. In addition, when new options for mobility are offered,

[2]https://github.com/salomonw/mixed-traffic-amod-route-rebalance

[3]For biking, we include a set of constraints in the same spirit as (5.4c) but for the bike layer. This ensures the balance between the incoming and outgoing flow of bikes at each node which goes in line with the dynamics of bike sharing systems Swaszek and Cassandras (2019b).

AMoD users could reach lower travel times than private vehicles, something which is impossible to achieve when only taxi-rides are available (due to the assumption on UC routing). This happens because they are more flexible and their overall transportation capacity is larger than the available capacity for private vehicles. However, at almost 100% penetration rates, it still seems that being selfish is benefited, raising interesting questions on how to incentivize users to act in a system-centric fashion. Finally, by comparing the first three rows of plots in Figure 5·7, we can make an important observation: If merely a tiny fraction of flow is accessible via subway or walking, travel times are reduced by almost 50%.

To account for more traffic intensities, Table 5.1 presents the results for an AMoD system with taxi-type (Veh), subway (Sub), pedestrian (Ped), and biking (Bike) layers when demand is multiplied by a factor of 1, 1.5, (corresponding to the last subplots of Fig 5·7) and 2. The Table shows results for the overall travel times and modal distributions of the I-AMoD kilometers traveled for penetration rates equal to 0, 50%, and 100%. In general we can claim that the higher the congestion, the higher the benefit in travel times due to the enlarged capacity resulting from intermodal options. In addition, we see that subway and biking options are critical to improve travel times.

Table 5.1: Intermodal AMoD results for different traffic intensities.

Demand	Pen. Rate	Avg. Travel Time (min)		I-AMoD Modal Distribution				
		I-AMoD	Private Veh	Veh	Reb	Bike	Sub	Ped
1	0	5.2	5.7	80%	15%	0%	5%	0%
	0.5	5.2	5.4	81%	14%	0%	5%	0%
	1	5.0	5.0	82%	13%	0%	4%	0%
1.5	0	7.5	8.8	69%	23%	2%	6%	0%
	0.5	7.0	6.9	74%	17%	4%	5%	0%
	1	6.3	5.7	78%	13%	4%	5%	0%
2	0	10.7	15.8	52%	28%	12%	7%	1%
	0.5	9.1	8.5	68%	13%	12%	6%	0%
	1	7.7	6.2	75%	8%	11%	6%	0%

In conclusion, we observe that while pure AMoD systems might decrease the system-level performance due to the additional congestion resulting from rebalancing, intermodal centralized-routing can significantly improve the overall travel times. Especially at high levels of demand, we see that, while SO intermodal routing can significantly improve travel times, it comes with the social dilemma that, from a UC perspective, being selfish would still be optimal.

5.6.5 Route Recovery Example

We show the applicability of our route-recovery strategies presented in Section 5.5. We implement the distribute version of the route-recovery algorithm described in Section 5.5.1 on the solution flows of the origin-based problem (5.14). We compute the routes using a commercial laptop with 8 cores for which we recover the routes in the order of 30 seconds to one minute, making it accessible for real-time implementation. Figure 5·8 shows the different SO routes connecting a single OD pair. The left plot shows the recommended routes which only include taxi-type service. Furthermore, the right plot shows an intermodal route composed of taking a taxi (solid lines) and the subway (dotted line).

5.7 Summary and Future Work

In this chapter we proposed a methodology to optimize the routes and rebalancing policies of a congestion-aware intermodal Autonomous Mobility-on-Demand (AMoD) system when it interacts with exogenous private traffic. To address the issue of non-convexity for this problem, we used a piecewise affine approximation of the travel latency function and proved that as the number of piecewise affine segments increases, the solution to the problem converges to the solution of the relaxed original problem. Using examples with the Eastern Massachusetts Area (EMA) and New York City (NYC) networks, (i) we empirically showed that the piecewise affine relaxation is asymptotically optimal, (ii) we captured the benefits of centrally controlling an intermodal AMoD system under mixed traffic conditions when different modes of transportation are available, (iii) we measured the advantage of using the approximated joint method versus a method that separately optimizes the routing and rebalancing policies, (iv) we revealed the existing trade-off between extra rebalancing flow and smart routing decisions, and (v) we tested the applicability of our proposed route-recovery algorithms in a real case study using the NYC network.

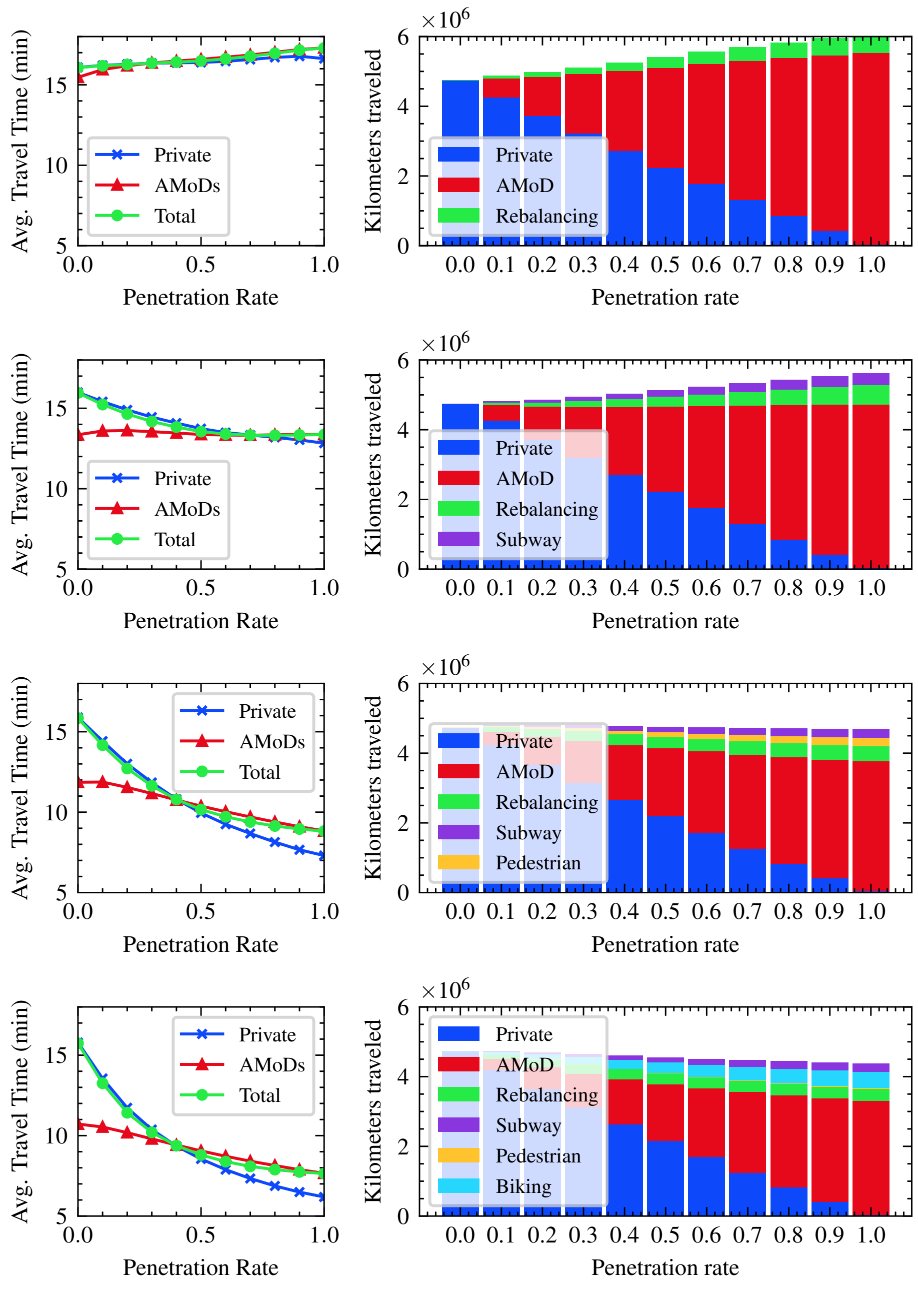


Figure 5·7: System performance with alternative modes of transport for a relatively high-demand scenario in NYC (we increase demand by a factor of 1.5). The first column of plots show the average travel time for different AMoD penetration rates while the second row depicts the miles traveled per mode of transportation for each penetration rate.

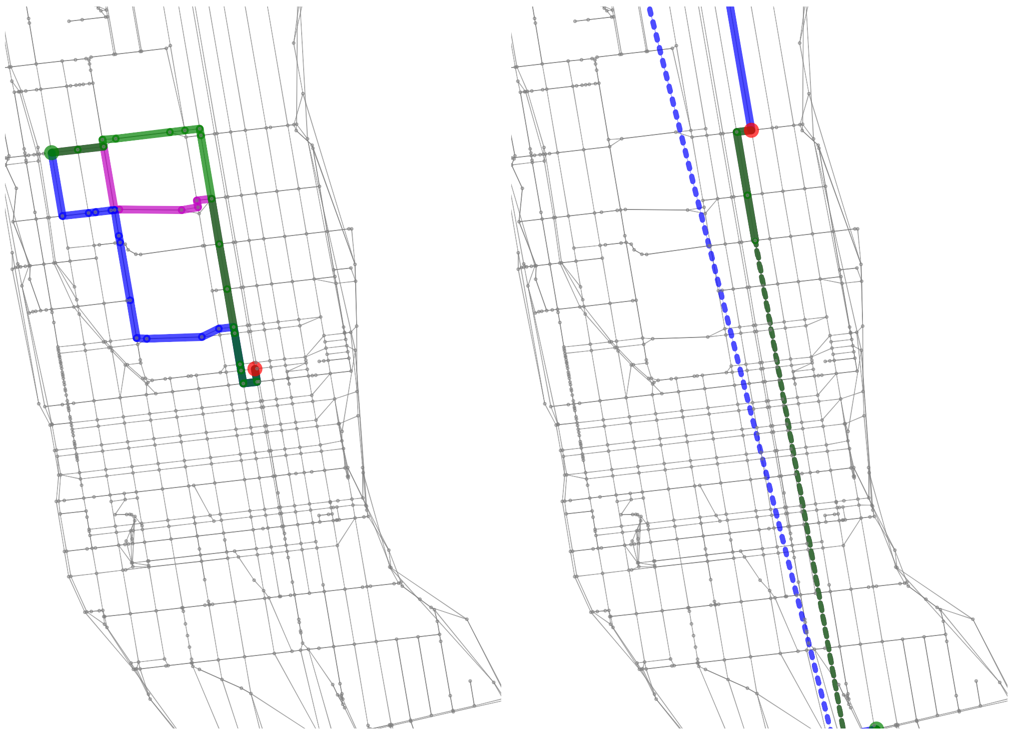


Figure 5·8: Example of the SO routes connecting an OD pair. Green and red dots represent origin and destinations, respectively. Solid lines portray traveling flow in the road network while dotted lines describe flow traveling via subway.

Chapter 6

Optimizing Pricing, Fleet Sizing, and Rebalancing of Autonomous Mobility-on-Demand systems

The emergence of the *sharing economy* in urban transportation networks has enabled new fast, convenient and accessible mobility services referred to as *Mobilty-on-Demand* systems (e.g. Uber, Lyft, DiDi). These platforms have flourished in the last decade around the globe and face many operational challenges in order to be competitive and provide good quality of service. A crucial step in the effective operation of these systems is to reduce customers' waiting time while properly selecting the optimal fleet size and pricing policy. In this chapter, we *jointly* tackle three operational decisions: (i) fleet size, (ii) pricing, and (iii) rebalancing, in order to maximize the platform's profit or its customers' welfare. To accomplish this, we first describe the system using a dynamic fluid model to show the existence and stability of an equilibrium (i.e., load balance) through pricing policies. Then, we devise an optimization framework which gives rise to a *static* policy. Then, we elaborate and propose *dynamic* policies that are more responsive to perturbations such as unexpected increases in demand. We test this framework in a simulation environment using three case studies and leveraging traffic flow and taxi data from Eastern Massachusetts, New York City and Chicago. Our results show that solving the problem jointly could increase profits between 1% and up to 50%, depending on the benchmark. Moreover, we observe that the proposed fleet size yield utilization of the vehicles in the fleet is around 75% compared to private vehicle utilization of 5%.

6.1 The Problem and Related Work

This chapter studies how to jointly optimize the prizing, fleet sizing and rebalancing policies to maximize a utility function. In particular, and in contrast with current methods, we are interested in leveraging information about the *destination* of the passenger (not only her origin) to select prices and to optimize the operations of the service. In addition to analytical results, in this chapter we incorporate real-time rebalancing strategies in conjunction with the static solutions and build a simulator to test the performance of the proposed policies.

6.1.1 Related Work

In the literature, the *pricing* problem has been addressed using two different perspectives: *one-sided*, or *two-sided* markets depending on whether the MoD controller has full or limited control over the supply. One-sided markets assume full control over the vehicles (Banerjee, Johari, & Riquelme, 2015; Turan, Pedarsani, & Alizadeh, 2019), whereas two-sided markets consider self-interested suppliers (drivers) (Banerjee et al., 2015; Bimpikis, Candogan, & Saban, 2019). To the best of our knowledge, all the optimal pricing policies presented in the previous papers, except Turan et al. (2019), do not rebalance proactively. Rather, they incentivize the supply (human drivers) to reallocate by the use of compensations.

Our model differs from Turan et al. (2019), which uses a microscopic model and Reinforcement Learning techniques, by the level of abstraction performed. Different than their microscopic model we employ a macroscopic (planning) model to assess the benefits of *jointly* solving the pricing and rebalancing problem over other approaches.

In contrast to pricing, *rebalancing* (without pricing) has been studied using simulation (Hörl et al., 2018; Levin et al., 2017; Swaszek & Cassandras, 2019a), queuing-

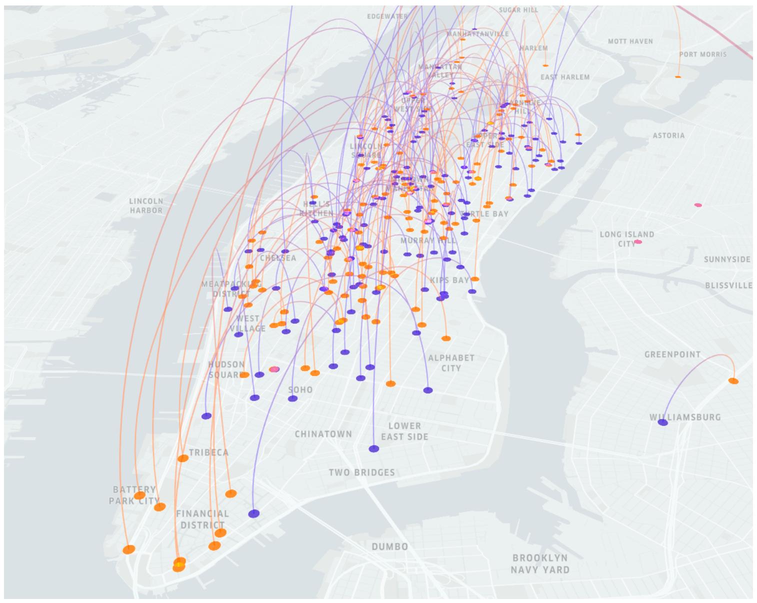


Figure 6·1: Requested taxi trips on January 15, 2015 10:38 a.m. in NYC. Blue and Orange circles represent origins and destinations respectively. One can observe that at this time, the Financial District (south) is an attractive destination but not origin. Hence, we expect taxis to rebalance to more attractive pickup locations.

theoretical (Iglesias, Rossi, Zhang, & Pavone, 2016; R. Zhang & Pavone, 2016), and network-flow (Pavone et al., 2012; Rossi et al., 2018) models. It has also been tackled jointly with routing schemes in Salazar et al. (2019); Wollenstein-Betech, Salazar, et al. (2021). In Swaszek and Cassandras (2019a), the rebalancing problem is addressed using a data-driven parametric controller suited for real-time implementation. Alternatively, Pavone et al. (2012) uses a steady-state fluid model. Both of these frameworks serve as the basis an building framework of this chapter.

The remainder of this chapter is organized as follows. In Section 6.2 we introduce the system model and the formal problem formulation. Section 6.3.1 provide a formal analysis the pricing dynamic system. In Section 6.4 we derive the optimal static policies for the pricing, rebalancing, and joint problems. In Section 6.5 we introduce real-time rebalancing strategies. In Section 6.6 we present our case studies and in

Section 6.7 we conclude.

6.2 Model and Problem Formulation

To model the MoD system, we use a closed Jackson queueing network as depicted in Figure 6·2. Formally, let the MoD fleet be composed of m vehicles who travel across the transportation network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = \{1, \ldots, N\}$ is the set of N *regions*, and $\mathcal{A} = \{(i, j) : i, j \in \mathcal{N} \times \mathcal{N}\}$ is the set of arcs connecting all regions. For every region i, we let $x_i(t) \in \{1, \ldots, m\}$ be a queue of available vehicles ready to serve a user request at time t, and $\mathbf{x}(t) = (x_i(t), \ldots, x_N(t))$ be the vehicle queue vector.

We model the arrival process of *potential* customers going from i to j using a time-invariant Poisson process with a rate λ_{ij}. Upon a customer arrival, she either (i) pays a fee p_{ij} and is served by one of the vehicles or (ii) leaves the system because the fee was above her willingness to pay, or because there were no available vehicles in region i to serve her. For every Origin-Destination (OD) in the network, the *fee* p_{ij} is formed by the product of a *base fee* p^0_{ij} and a *surge price* (or simply *price*) $u_{ij}(t)$. We assume the platform is not willing to charge less than the base fee for any trip, hence, we have that $u_{ij}(t) \geq 1$ for all i and j and all $t \geq 0$.

To determine the fraction of customer arrivals that are willing to pay the fee $p^0_{ij} u_{ij}(t)$, we assume there is a known *demand function* $\bar{F}_{ij}(u_{ij}(t)) : \mathbb{R}_{\geq 1} \mapsto [0, 1]$ which establishes this relationship. We assume this function $\bar{F}_{ij}(u_{ij}(t))$ to be (i) continuous; (ii) strictly decreasing, such that a higher price always results in a lower demand; and (iii) lower bounded by zero such that there exists a price $u^{\max}_{ij}$ that makes $\bar{F}_{ij}(u^{\max}_{ij}) = 0$ for all $i, j \in \mathcal{N}$ (see Figure 6·7). Consequently, the resulting arrival process considering the users' willingness to pay is described by the *modulated* demand $\Lambda_{ij}(u_{ij}(t))$ which follows a Poisson point process with rate $\lambda_{ij} \bar{F}_{ij}(u_{ij}(t))$.

After a customer arrival at region i who is willing to pay the fee, the MoD platform assigns her a vehicle at a service rate μ_i. Fot which we assume that $\mu_i > \sum_j \lambda_{ij}$, meaning that the platform assigns vehicles to customers faster than the rate at which customers arrive. Then, the travel time experienced by the customer and vehicle is an exponential random variable with rate $1/T_{ij}$, where T_{ij} is the average travel time from i to j. This is a standard assumption for queueing models, however, it can be replaced by a deterministic travel time if desired. Additionally, we let $y_{ij}(t)$ be the number of customer-carrying vehicles traveling from i to j at t.

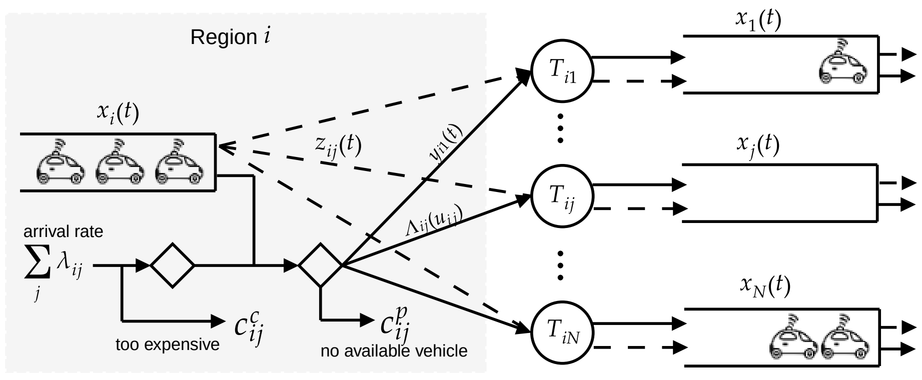


Figure 6·2: Prospective customers wishing to travel from i to j arrive at a rate λ_{ij}. Then, if they accept price u_{ij} and there is an available vehicle they travel to j in T_{ij} units of time. If the price u_{ij} is above their willingness to pay, the MoD incurs a cost composed by the loss of a trip and a cost c^c_{ij} for disappointing the customer. Moreover, if the customer is willing to pay u_{ij} but no vehicle in i is available, then the customer is rejected and the platform incurs a cost c^p_{ij}. The objective of the MoD provider is to plan a pricing policy $\mathbf{u}$ and a rebalancing policy $\mathbf{r}$ such that its profit (or other utility function) is maximized.

We assume the MoD service is capable of *rebalancing* the system, i.e., sending empty vehicles across regions to avoid having excess or fewer vehicles at every region. Hence, we let $r_{ij}(t)$ be a decision variable denoting the number of vehicles that the platform will send from i to j at time t. Finally, we let $z_{ij}(t)$ and c^r_{ij} be the number of empty vehicles *en-route* at t and the cost incurred for an empty trip from i to j, respectively.

The problem we are aiming to solve is how to properly select a fleet size m, the prices $\mathbf{u}(t) = (u_{ij}(t)\ ;\ i,j \in \mathcal{N})$ and a rebalancing policy $\mathbf{r}(t) = (r_{ij}(t)\ ;\ i,j \in \mathcal{N})$ so as to maximize a utility function. Examples of this utility function are profit maximization of the MoD service or, from a societal perspective, customer welfare maximization. We discuss these utility functions in Section 6.4 as well as *static* and *dynamic* strategies to solve this problem.

6.3 Analysis of Pricing Policies

We analyze the model using a relaxed steady-state deterministic fluid model of the described stochastic queueing. The reason for making this relaxation is the flexibility it provides to perform the mathematical analysis of the system. Moreover, since the analysis for rebalancing is described in Pavone et al. (2012), we develop the analysis only for the pricing decision. To make the model more robust in here, we allow customer queues to build, in other words, if a customer arrives but no vehicle is available, then the customer waits until a vehicle comes pick her up.

The idea is to analyze the problem at its steady-state and analyze pricing policies that are time-invariant, i.e. $\mathbf{u}(t) = \mathbf{u}$. To achieve this, it is convenient to use a *fluidic* abstraction of the real system. The main advantage of *fluidic* models is twofold. First, to relax the discrete system (vehicles and customers are discrete entities) to a continuous system in order to facilitate the optimization procedure. Second, to ensure that the system is *balanced*, which means that the total incoming flow of vehicles to any region $i \in \mathcal{N}$ is equal to the summation over the all modulated customers leaving i (we will give a formal definition in this section). We begin the analysis by making the following assumptions:

Assumption 1. The function $\Lambda_{ij}(\cdot)$ is monotonically decreasing $\forall i,j \in \mathcal{N}$, i.e., as price increases, the demand rate decreases.

Assumption 2. There exists a surge price $u_{ij}^{\max}$ for which $\Lambda_{ij}(u_{ij}^{\max}) = 0, \forall i, j \in \mathcal{N}$.

Customer Dynamics: Consider a customer queue $c_i(t)$ for each region $i \in \mathcal{N}$ in the network. The queue dynamics are:

$$\dot{c}_i(t) = \begin{cases} \sum_j \Lambda_{ij}(u_{ij}(t)), & \text{if } x_i(t) = 0, \\ 0, & \text{if } x_i > 0 \text{ and } c_i(t) = 0, \\ \sum_j \Lambda_{ij}(u_{ij}(t)) - \mu_i(t), & \text{if } x_i(t) > 0 \text{ and } c_i(t) > 0. \end{cases}$$

In order to express the customer dynamics with shorter notation we let $H(x) = \mathbb{1}_{x>0}$ be an indicator function for positive values of x, and we use the following shorthand notation:

$$\begin{aligned} \Lambda_{ij} &:= \Lambda_{ij}(u_{ij}); & \Lambda_i &:= \sum_j \Lambda_{ij}(u_{ij}), \\ x_i &:= x_i(t); & c_i &:= c_i(t), \\ x_j^i &:= x_j(t - T_{ji}), & c_j^i &:= c_j(t - T_{ji}) \end{aligned}$$

where Λ_i is the total endogenous outgoing flow from node i; and c_j^i, x_j^i are the customer and vehicle queue levels in region j at time $t - T_{ij}$, respectively. Then, we rewrite the customer dynamics in compact form as follows:

$$\dot{c}_i = \Lambda_i(1 - H(x_i)) + (\Lambda_i - \mu_i)H(c_i)H(x_i).$$

Note that as a result of using a fluid model, the variables denoting the number of customers in a region are real-valued.

Vehicle Dynamics: The outflow rate corresponding to vehicles departing station i is given by:

$$\dot{x}_i^- = \begin{cases} -\Lambda_i, & \text{if } x_i \geq 0 \text{ and } c_i = 0, \\ 0, & \text{if } x_i = 0, \\ -\mu_i, & \text{if } x_i \geq 0 \text{ and } c_i \geq 0. \end{cases}$$

which, by using the $H(x)$ notation above, can be written as $\dot{x}_i^- = -\Lambda_i H(x_i) + (\Lambda_i - \mu_i)H(x_i)H(c_i)$. In addition, the rate at which customer-carrying vehicles arrive at station i is given by: $\dot{x}_i^+ = \sum_j (\Lambda_{ji} H(x_j^i) - (\Lambda_{ji} - \mu_j)H(x_j^i)H(c_j^i))$. Hence, the vehicle dynamics is $\dot{x}_i = \dot{x}_i^- + \dot{x}_i^+$, which lead to

$$\begin{aligned} \dot{x}_i = &-\Lambda_i H(x_i) + (\Lambda_i - \mu_i)H(c_i)H(x_i) \\ &+ \sum\nolimits_j (\Lambda_{ji} H(x_j^i) - (\Lambda_{ji} - \mu_j)H(c_j^i)H(x_j^i)). \end{aligned}$$

Then, the global system dynamics are expressed by the differential equations

$$\dot{c}_i = \Lambda_i(1 - H(x_i)) + (\Lambda_i - \mu_i)H(c_i)H(x_i), \tag{6.1a}$$

$$\begin{aligned} \dot{x}_i = &-\Lambda_i H(x_i) + (\Lambda_i - \mu_i)H(c_i)H(x_i) \\ &+ \sum\nolimits_j (\Lambda_{ji} H(x_j^i) - (\Lambda_{ji} - \mu_j)H(c_j^i)H(x_j^i)), \end{aligned} \tag{6.1b}$$

which describe a non-linear, time-delayed, time-invariant, right-hand discontinuous system.

6.3.1 Well posedness, Equilibrium and Stability

Similar to Pavone et al. (2012), we say that the system (6.1) is *well posed* if two conditions are satisfied: (i) for any initial condition, there exists a solution of the differential equations in (6.1), and (ii), the number of vehicles in the system remain invariant over time. In order to analyze the model, we use the framework of Filippov solutions (Filippov, 2013).

Proposition 1 (Well-posedness of the fluid model)**.**

1. *For every initial condition in the fluid model represented in* (6.1)*, there exist continuous functions* $c_i(t) : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}$ *and* $x_i(t) : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}, \forall i \in \mathcal{N}$*, satisfying the system of equations in the Fillipov sense.*

2. *For all* $t > 0$*, the total number of vehicles is invariant and equal to* $m = \sum_{i\in\mathcal{N}} x_i(0)$*.*

Proof. See Appendix A.0.2 □

Equilibria: We say that the system is in equilibrium if customer queues (and therefore, waiting times) do not grow to infinity. We show the existence of an equilibrium in the fluid model (6.1) when we control the prices of every origin-destination pair. Additionally, we show that by having the ability to control the prices, one can have find multiple equilibria for a desired fleet size, giving the flexibility to AMoD managers to operate the system at different demand levels.

Theorem 3 (Existence of equilibria)**.** *Let* $\mathcal{U}$ *be a set of prices* $\mathbf{u}$*, such that when* $\mathbf{u} \in \mathcal{U}$ *we have*

$$\sum_{j} \Lambda_{ij}(u_{ij}) - \Lambda_{ji}(u_{ji}) = 0, \quad \forall i \in \mathcal{N}, \tag{6.2}$$

and let $m_{\mathbf{u}} := \sum_{ij} T_{ij}\Lambda_{ij}(u_{ij})$*. Then, if* $\mathbf{u} \in \mathcal{U}$*, and* $m > m_{\mathbf{u}}$*, an equilibrium exists with* $\mathbf{c} = 0$ *and* $\mathbf{x} > 0$*. Otherwise no equilibrium exists.*

Proof. See Appendix A.0.2 □

Lemma 6.3.1. *The set* $\mathcal{U}$ *is never empty, hence, at least one equilibrium exists.*

Proof. We use the fact that there exists a price $u_{ij}^{\max}$ for which $\Lambda_{ij}(u_{ij}^{\max}) = 0$ for all $i, j \in \mathcal{N}$. Then, setting $\mathbf{u} = \mathbf{u}^{\max}$, implies that an equilibrium exists. This strategy means that we are not providing service to any request, nevertheless the equilibrium exists. □

Lemma 6.3.2 (Infinite number of equilibria)**.** *If there is a positive demand tour in the graph, then there exists an infinite number of price vectors* $\mathbf{u}$ *which can steer the system to an equilibrium point.*

Proof. Assume that there exists at least one Eulerian tour (or `cycle`) in the graph for which $\lambda_{ij} > 0$ for all $(i,j) \in \texttt{cycle}$. Then, let $\boldsymbol{\lambda}^{\texttt{cycle}} = \{\lambda_{ij} \mid (i,j) \in \texttt{cycle}\}$ and the minimum rate on that tour be $\lambda^{\texttt{cycle}}_{\min} = \min\{\lambda_{ij}\}_{(i,j)\in\texttt{cycle}}$. Then by setting $u_{ij} = u^{\max}_{ij}$ for all $(i,j) \notin \texttt{cycle}$, we can express the equilibrium condition as

$$\sum_{j:(i,j)\in\texttt{cycle}} \Lambda_{ij}(u_{ij}) - \Lambda_{ji}(u_{ji}) = 0, \quad \forall i : (i,j) \in \texttt{cycle}. \tag{6.3}$$

Now, we use the fact that $\Lambda_{ij}(u_{ij})$ is a monotonically decreasing function and we focus on $(i,j) \in \texttt{cycle}$. Hence for all $\Lambda_{ij}(u_{ij}) > \lambda^{\texttt{cycle}}_{\min}$ we can find a u_{ij} such that $\Lambda_{ij}(u_{ij}) = \lambda^{\texttt{cycle}}_{\min}$. Then, extending this for higher prices on $\lambda^{\texttt{cycle}}_{\min}$ and using the same argument as before, we show that there exists a pricing strategy $\mathbf{u}$ for which we can obtain an equilibrium with a tour demand rate with any value in the range $(0, \lambda^{\texttt{cycle}}_{\min})$. □

These two lemmata imply that by incorporating an origin-destination pricing strategy, we can operate a MoD service at equilibrium for an infinite number of demand rate and for a *desired* selection of fleet size.

Corollary 1 (Minimum number of vehicles in equilibria)**.** *The minimum number of vehicles to operate in an equilibrium induced by policy $\mathbf{u}$ is at least $m > \underline{m} := \min_{\mathbf{u}} m_{\mathbf{u}}$ where $m_{\mathbf{u}} := \sum_{ij} T_{ij}\Lambda_{ij}(u_{ij})$.*

Proof. Follows from the last argument in the proof of Theorem 3. □

Stability: In this section we study local stability of the equilibria presented in the previous subsection. As an example, we look at cases when a disruptive change happens to the system, either because of an increase in customers or a decrease in the availability of vehicles. Let $\mathbf{u} \in \mathcal{U}$ and assume $m_{\mathbf{u}} > \underline{m}$. Then, we define the set of equilibria as

$$\Upsilon_{\mathbf{u}} := \{(\mathbf{c}, \mathbf{x}) \in \mathbb{R}^{2N} \mid c_i = 0, x_i > 0, \quad \forall i \in \mathcal{N}, \sum\nolimits_i x_i = m - m_{\mathbf{u}}\}. \tag{6.4}$$

Definition 1 (Locally asymptotically stable)**.** *A set of equilibria $\Upsilon_{\mathbf{u}}$ is locally asymptotically stable if for an equilibrium $(\underline{\mathbf{c}}, \underline{\mathbf{x}}) \in \Upsilon_{\mathbf{u}}$, there exists a neighborhood*

$\mathcal{B}^{\delta}_{\mathbf{u}}(\underline{\mathbf{c}},\underline{\mathbf{x}})$ *such that every evolution of* (6.1) *starting at* $(\mathbf{c}(\tau),\mathbf{x}(\tau)) = (\underline{\mathbf{c}},\underline{\mathbf{x}})$*, and with* $(\mathbf{c}(0),\mathbf{x}(0)) \in \mathcal{B}^{\delta}_{\mathbf{u}}(\underline{\mathbf{c}},\underline{\mathbf{x}})$ *has a limit which belongs to the equilibrium set* $\Upsilon_{\mathbf{u}}$ *i.e.,* $(\lim_{t\longrightarrow+\infty}\mathbf{c}(t), \lim_{t\longrightarrow+\infty}\mathbf{x}(t)) \in \Upsilon_{\mathbf{u}}$*, where* $\tau \in [-\max_{i,j} T_{ij}, 0)$ *and*

$$\mathcal{B}^{\delta}_{\mathbf{u}}(\underline{\mathbf{c}},\underline{\mathbf{x}}) := \{(\mathbf{c},\mathbf{x}) \in \mathbb{R}^{2N} \mid c_i > 0, x_i = \underline{x_i},\ \forall i \in \mathcal{N},\ ||(\mathbf{c}-\underline{\mathbf{c}},0)|| < \delta)\}. \qquad (6.5)$$

Theorem 4 (Stability of the equilibria)**.** *Let* $\mathbf{u} \in \mathcal{U}$ *and* $m_{\mathbf{u}} > \underline{m}$*; then, the set of equilibria* $\Upsilon_{\mathbf{u}}$ *is locally asymptotically stable.*

Proof. See Appendix A.0.2. □

All these results shed light to develop optimal pricing policies that are ensured to generate an equilibrium point when analyzed in the static deterministic model but can be applied in the stochastic Jackson queueing model.

6.4 Optimal Strategies

As pointed out by Swaszek and Cassandras (2019a) and Turan et al. (2019), one way in which we can derive optimal strategies for the pricing and rebalancing dynamic system is to frame it as a Markov Decision Process and use Dynamic Programming to solve it. Unfortunately, they observe that the problem suffers from the *curse of dimensionality* and becomes intractable even for small instances. One way in which we can address this complexity issue is to define *static* policies.

In general, the following mild assumptions are required to claim that the solution of the fluidic abstraction is a good solution for the original queueing system: (i) The solution is only optimal when analyzing the steady-state of the system, not during transient periods; (ii) the Poisson processes modeling the modulated customer demand and rebalancing are independent from each other.

Using these assumptions, all the stochastic processes in the queueing model are Poisson processes. For the modulated customer demand, a *thinning* (or *splitting*) Poisson process will result by interpreting the willingness-to-pay function for a given price as defining Bernoulli trials of customers accepting the price or leaving the system. Likewise, the process of vehicles leaving stations will result in a *superposition* of two independent Poisson processes (which remain a Poisson process) stemming from the customer-carrying and the rebalancing vehicles. This is equivalent to observing Bernoulli trials governed by the probability that the vehicle leaving a station is an empty (rebalancing) or a customer-carrying vehicle.

6.4.1 Static Pricing

As stated earlier, we aim to select time-invariant static prices $\mathbf{u}$ with the objective of maximizing the profit of a MoD provider while ensuring a balanced system. Similar to this approach, we also present an equivalent formulation that maximizes user social welfare rather than MoD profit.

Profit Maximization

Let c^o_{ij} be the operational cost of providing a transportation service from i to j and c^f be a fixed cost associated with the value of owning a vehicle for a period of time. Moreover, let c^c be an additional cost that the MoD service incurs when a costumer leaves the platform because of a high price. For example, a customer who thinks the service is too expensive might not consider to use this MoD platform in the future. With these definitions we write the profit maximization problem as follows:

$$\max_{\mathbf{u},m} \quad \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} \Lambda_{ij}(u_{ij})(u_{ij}p^0_{ij} - c^o_{ij}) - c^c(\lambda_{ij} - \Lambda_{ij}(u_{ij})) - c^f m, \tag{6.6a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{N}} (\Lambda_{ij}(u_{ij}) - \Lambda_{ji}(u_{ji})) = 0, \quad \forall j \in \mathcal{N}, \tag{6.6b}$$

$$\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} T_{ij}\Lambda_{ij}(u_{ij}) \leq m, \tag{6.6c}$$

$$1 \leq u_{ij} \leq u^{\max}_{ij}, \quad \forall i, j \in \mathcal{N}, \tag{6.6d}$$

where in the objective function $\Lambda_{ij}(u_{ij})$ is the modulated demand and $(u_{ij}p^0_{ij} - c^o_{ij})$ is the difference between the charged fee and the operational cost c^o_{ij}.

Contraint (6.6b) ensures the MoD system not to accumulate vehicles in any region, as well as to make sure that no region is being constantly rejecting customer due to a lack of vehicles (based on our results in the previous section). Constraint (6.6c) restricts the minimum number of vehicles the fleet has to have in order to provide such a service. Finally, (6.6d) ensures that the optimization process happens within prices range.

Note that in order for (6.6) to be tractable, we have to maximize a concave objective function (6.6a) in the range of $[1, \mathbf{u}^{\max}]$ over a convex feasible set. Both of these requirements are accomplished when using a linear willingness-to-pay function as the problem becomes to minimize a convex quadratic objective over linear equality constraints. Notice that in this fluidic formulation, we do not include a cost of losing a customer due to the shortage of vehicles. This is because constraint (6.6b) ensures a balanced system and since for the fluidic model we assume non-stochastic behavior, this cost is equal to zero.

Welfare Maximization

It is relevant for the discussion on smart cities to consider the case where social welfare is maximized instead of the platform's profit. To do this, we associate a utility with every customer arrival, which we model using a random variable U_{ij} with probability density function $f_{ij}(u_{ij})$ and support in $[1, u^{\max}_{ij}]$. If U_{ij} exceeds price u_{ij}, then the customer will accept the fee, which result in a modulated demand $\Lambda^{\mathrm{WM}}_{ij}(u_{ij}) = \lambda_{ij}\mathbb{P}[U_{ij} \geq u_{ij}]$. Consequently, the expected utility of a customer conditional on the fact that the customer is willing to pay the fee u_{ij} is $\mathbb{E}[U_{ij}|U_{ij} \leq u_{ij}]$. Hence, the welfare maximization problem is

$$\max_{\mathbf{u},m} \quad \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} \mathbb{E}[\Lambda_{ij}^{\mathrm{WM}}(u_{ij})\mathbb{E}[U_{ij}|U_{ij}\geq u_{ij}]] - c^f m, \tag{6.7a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{N}}(\Lambda_{ij}^{\mathrm{WM}}(u_{ij}) - \Lambda_{ji}^{\mathrm{WM}}(u_{ji})) = 0, \quad \forall j \in \mathcal{N}, \tag{6.7b}$$

$$\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} T_{ij}\Lambda_{ij}^{\mathrm{WM}}(u_{ij}) \leq m, \tag{6.7c}$$

$$1 \leq u_{ij} \leq u_{ij}^{\max}, \quad \forall i,j \in \mathcal{N}, \tag{6.7d}$$

Notice that the objective (6.7a) has a similar form as (6.6a), and thus, the two problems can be solved using the same optimization methods. Therefore, from now on, we will focus on the profit maximization problem.

6.4.2 Static Rebalancing

Following the rebalancing model developed in Pavone et al. (2012), we are interested in finding a static rebalancing policy that balances the system without adjusting prices. Let r_{ij} be the rebalancing flow from i to j, in other words, the rate (veh/h) at which we have empty vehicles traveling from i to j. We can formulate and solve this problem using a Linear Program (LP) that minimizes the empty travel time while ensuring a balanced system. Formally, this is stated as

$$\min_{\mathbf{r}\geq 0} \quad \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} T_{ij} r_{ij} \tag{6.8a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{N}}(\Lambda_{ij}(u_{ij}) + r_{ij} - \Lambda_{ji}(u_{ji}) - r_{ji}) = 0, \quad \forall j \in \mathcal{N}. \tag{6.8b}$$

Notice that in this formulation, $\mathbf{u}$ is a known parameter (not a decision variable) in the optimization problem. Therefore, we do not consider the possibility of decreasing the demand by adjusting prices. This LP is always feasible as one can always choose $r_{ij} = \Lambda_{ji}(u_{ji})$ for all $i,j \in \mathcal{N}$ which satisfies the set of constraints (6.8b). For more details about this formulation, we refer the interested reader to Pavone et al. (2012) and R. Zhang and Pavone (2016).

6.4.3 Joint Pricing and Rebalancing

We are interested in choosing the best policy that leverages multiple decisions that the MoD platform faces. In particular, we would like to optimize the pricing, rebalancing and fleet sizing problem. We write this joint optimization problem as the combination of (6.6) and (6.8) which leads to

$$\max_{\mathbf{u},\mathbf{r},m} \quad \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} \Lambda_{ij}(u_{ij})(p^0_{ij}u_{ij} - c^o_{ij}) - c^c(\lambda_{ij} - \Lambda_{ij}(u_{ij})) - c^r(r_{ij}T_{ij}) - c^f m \tag{6.9a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{N}}(\Lambda_{ij}(u_{ij}) + r_{ij} - \Lambda_{ji}(u_{ji}) - r_{ji}) = 0, \quad \forall j \in \mathcal{N}, \tag{6.9b}$$

$$\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} T_{ij}(\Lambda_{ij}(u_{ij}) + r_{ij}) \le m, \tag{6.9c}$$

$$1 \le u_{ij} \le u^{\max}_{ij} \quad \forall i, j \in \mathcal{N}, \tag{6.9d}$$

where c^r and c^f are the cost of rebalancing and the cost of owning and maintaining a vehicle per unit of time, respectively.

Problem (6.9) is always feasible as it can always admit the solution $\mathbf{u} = \mathbf{u}^{\max}, \mathbf{r} = 0$ and $m = 0$. However, in order to numerically solve (6.9) in polynomial time, and to ensure we have found the global maximum, we must validate that the objective function is concave for $\mathbf{u} \in [1, \mathbf{u}^{\max}]$ and that the constraints (6.9b)-(6.9d) form a convex set. If these conditions are satisfied, then (6.9) yields a solution with higher profits than the individual formulations (6.6) and (6.8), or the sequential approach of solving first the rebalancing problem (6.8) and then selecting optimal prices (6.6). This happens given that the problem is *jointly* solving for m, $\mathbf{u}$ and $\mathbf{r}$ rather than using an individual or a greedy sequential approach.

6.5 Real-Time Strategies

So far, we have discussed static pricing and rebalancing policies which are desirable economically (there is no better dynamic pricing policy that can exceed its static counterpart in steady-state) and socially (avoiding drastic fluctuation in prices generates a more desirable platform for users). However, one characteristic that static policies lack is their *responsiveness* to perturbations in the environment.

In this section, we introduce *real-time* (or *dynamic*) policies to optimize the operation of the MoD platform. The main idea is to exploit real-time information to operate the system more efficiently. For example, we can use the status of vehicle queues $\mathbf{x}(t)$ and traveling vehicles $\mathbf{y}(t)$ at time t to decide a rebalancing strategy $\mathbf{r}(t)$ or to adjust prices $\mathbf{u}(t)$.

Due to the desired theoretical static prices, we will focus on designing rebalancing policies $\mathbf{r}(t)$ that are dynamic in order to account for fluctuations, while we keep the prices static $\mathbf{u}(t) = \mathbf{u}$. Hence, from now on, we will focus on finding $\mathbf{r}(t)$ and we assume we use the optimal static pricing resulting from solving the joint problem (6.9).

To implement a dynamic controller we are required to define the *state* variables that will be available to control the system. Let us propose a state vector $\mathbf{s}(t) \in \{0, 1, \ldots, m\}^N$ composed of state variables indicating the *actual and prospective vehicles* at every region expressed by

$$s_i(t) = x_i(t) + \sum_{j \in \mathcal{N}} y_{ji}(t) + z_{ji}(t),$$

where we recall that $y_{ij}(t)$ and $z_{ij}(t)$ are the number of customer-carrying vehicles and empty vehicles traveling from i to j, respectively, and where $s_i(t)$ is the sum of all available vehicles at a region i and all vehicles traveling to i.

We let a *rebalancing event* be an event happening at a specific moment in time in which the platform decides to rebalance the system. Different from the fluidic

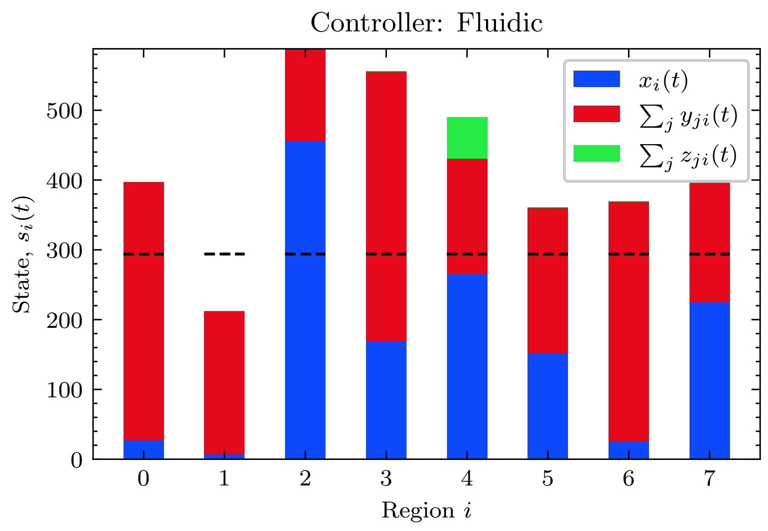


Figure 6·3: A diagram of the state of the system for the EMA network. Every bar in the plot represents the state of a region composed of the available vehicles (blue), the en-route customer-carrying vehicles (red), and the empty rebalancing vehicles (green). The dotted line represent the parameter θ_i for every region i which indicates the minimum desired level to be satisfied when performing a rebalancing action.

controller, in which vehicles are sent at a constant rate, the dynamic controllers herein trigger a rebalancing event once a condition is satisfied based on the state $\mathbf{s}(t)$ and a vector of d parameters denoted with $\mathbf{\Theta}(t) \in \mathbb{R}^d$.

Giving more structure to the dynamic policies presented here, let us define a set of parameters $\boldsymbol{\theta}(t) = (\theta_i; i \in \mathcal{N})$ corresponding to a *desired* level of vehicles in every region at time t. In other words, for region i at time t, we would like to have a number of $\theta_i(t)$ idle or prospective vehicles. The choice of $\boldsymbol{\theta}(t)$ is not known in advance and learning methods such as concurrent estimation (Cassandras & Lafortune, 2009) or RL (Sutton & Barto, 2018) can be leveraged to learn good choices of $\boldsymbol{\theta}(t)$.

With these parameters, we can define an optimization problem that rebalances the system to guarantee that every region has at least $\theta_i(t)$ prospective vehicles at the time of a rebalancing event. Therefore, at a fixed time t, we solve the *rebalancing* problem:

$$\min_{\mathbf{r}(t)} \quad \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} T_{ij} r_{ij}(t) \tag{6.10a}$$

$$\text{s.t.} \quad \theta_i(t) \leq s_i(t) + \sum_{j\in\mathcal{N}\setminus\{i\}} (r_{ji}(t) - r_{ij}(t)) \quad \forall i \in \mathcal{N}, \tag{6.10b}$$

$$\sum_{j\in\mathcal{N}} r_{ij}(t) \leq x_i(t) \quad \forall i \in \mathcal{N}, \tag{6.10c}$$

$$r_{ij}(t) \in \mathbb{N}, \quad \forall i,j \in \mathcal{N}, \tag{6.10d}$$

where $r_{ij}(t)$ is the number of rebalance vehicles to be send from i to j, (6.10b) ensures that the minimum number of current and prospective vehicles at every region is greater or equal than its corresponding parameter, (6.10c) allows rebalancing only idle vehicles in a region, and (6.10d) ensures that the solution is integer.

Solving general ILPs such as (6.10) is computational-expensive. Thus, we would like to write an alternative formulation which is faster to solve. To achieve this, we follow Pavone et al. (2012) which exploits the *total unimodularity* structure of the problem. Informally, total unimodularity implies that if the right hand side vector of a network flow problem is integer-valued and all submatrices of the constraint matrix have determinant $\{-1, 0, 1\}$, then, the solution to the linear program relaxation is guaranteed to return integer solutions. Note that in our case, the vectors $\boldsymbol{\theta}(t)$ and $\mathbf{s}(t)$ are integer-valued. Hence we rewrite (6.10) as a network flow model as follows:

$$\min_{\mathbf{r}(t)\geq 0} \quad \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} T_{ij} r_{ij}(t) \tag{6.11a}$$

$$\text{s.t.} \quad \min\{s_i(t) - \theta_i(t), x_i(t)\} \geq \sum_{j\in\mathcal{N}\setminus\{i\}} (r_{ij}(t) - r_{ji}(t)) \quad \forall i \in \mathcal{N}. \tag{6.11b}$$

Note that (6.11b) encompasses both equations (6.10b) and (6.10c). When $s_i(t) - \theta_i(t) < x_i(t)$ holds, then (6.11b) equals (6.10b). When $x_i(t) < s_i(t) - \theta_i(t)$, the constraint

(6.11b) indicates that region i has enough idle vehicles to send to other regions and it is identical to (6.10c). All the real time parametric controllers presented in the following subsections will be based on solving problem (6.11) at specific times t. Then, our next question is, when should we perform a rebalancing event?

6.5.1 Single parameter

We begin by considering the simple time-driven controller which triggers a rebalancing event every Ω units of time. The controller solves (6.11) by choosing the thresholds $\boldsymbol{\theta}(t)$ to be uniform and time-invariant, e.g., $\theta_i(t) = \lfloor \frac{m}{N} \rfloor$ for all t and for all i in $\mathcal{N}$. This is a single-scalar policy since it is based on a single parameter $\boldsymbol{\Theta} = \{\Omega \in \mathbb{R}_+\}$. This controller is quite effective, however, defining a uniform $\boldsymbol{\theta}$ vector can be inefficient since demand rates for different regions are not uniform. For example, a region with a high volume of requests would benefit for a higher θ_i.

6.5.2 N+1 parameter controller

To address the limitation of using only uniform thresholds, we can define a controller that chooses a good time-invariant vector $\boldsymbol{\theta}$. One way to select θ_i is to consider a number that is proportional to the outgoing flow from node i, i.e., $\theta_i = \lfloor m \frac{\sum_{j \in \mathcal{N}} \Lambda_{ij}(u_{ij})}{\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} \Lambda_{ij}(u_{ij})} \rfloor$. Another approach is to select $\boldsymbol{\theta}$ using simulation-based optimization methods such as concurrent estimation (see Swaszek and Cassandras (2019a)) or RL.

In addition to the N parameters coming from $\boldsymbol{\theta}$, we consider an additional parameter (this is why we called it the "N+1 controller") which triggers the rebalancing event. A natural option for this is to use a time-driven parameter as in the *single parameter* controller. However we consider triggering the rebalancing event using a metric of the total vehicle *imbalances* in the system. In this manner, if the system is balanced, the controller would not activate any rebalancing event, conversely, if the system is imbalanced, the rebalancing event will be triggered more often.

Formally, let Ω be a selected parameter which accounts for the minimum number of *negative imbalances* that the system is willing to tolerate before triggering a rebalancing event. That is, at every time t, the controller will solve problem (6.11) if $\Omega > \sum_{i \in \bar{\mathcal{N}}(t)} (\theta_i - s_i(t))$ where the set $\bar{\mathcal{N}}(t)$ is composed of the regions which have less than the desired number of vehicles, i.e., $\bar{\mathcal{N}}(t) = \{i \mid \theta_i - s_i(t) > 0\}$. Here, $\mathbf{\Theta}(t) = \{\boldsymbol{\theta}, \Omega\}$. Both of these parameters can be selected by using any non-convex global optimization approach or by leveraging concurrent estimation techniques as in Swaszek and Cassandras (2019a).

6.5.3 Dynamic N+1 controller

Up to this point, we have discussed polices where the parameters are time-invariant. The question now becomes: can we update the parameters in real-time to improve the overall performance? To do this, we introduce the notion of *episodes.*

We think of an episode as a interval of τ units of time for which we will gather information and will use the data to update our future decisions. By employing this approach, we assume that the last episode observation contains relevant information for our next decision.

For every episode $k = 1, \dots, K$, we would like to estimate the demand rate $\mathbf{\Lambda}(\mathbf{u}(t))$ with $\hat{\mathbf{\Lambda}}_k$ by counting the number of observed arrivals and dividing it by τ. Then, we update the $\boldsymbol{\theta}$ vector either by using a naive approach:

$$\theta_{i,k}(t) = \left\lfloor m \frac{\sum_{j \in \mathcal{N}} \hat{\Lambda}_{ij}}{\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} \hat{\Lambda}_{ij}} \right\rfloor,$$

or by taking a step in the direction of the new estimate, that is:

$$\theta_{i,k+1}(t) = \left\lfloor \theta_{i,k}(t) - \eta_k \big(\theta_{i,k}(t) - m \frac{\sum_{j \in \mathcal{N}} \hat{\Lambda}_{ij}}{\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} \hat{\Lambda}_{ij}} \big) \right\rfloor,$$

where η_k is a pre-specified *stepsize* or *learning rate* for all k in $1, \dots, K$.

6.6 Numerical Results

We perform experiments to showcase the advantages and disadvantages of the static and dynamic policies. To carry out the experiments, we employ the transportation networks of Eastern Massachusetts Area (EMA), Chicago (CHI) and New York City (NYC), shown in Figures 6·4-6·6.

The EMA network is composed of 8 regions and we retrieved its topological and demand information using speed data provided by the Central Transportation Planning Staff (CTPS) of the Boston Metropolitan Planning Organization (MPO) and processed as in Wollenstein-Betech, Sun, Zhang, Cassandras, and Paschalidis (2022).

The NYC network is composed of 70 regions distributed across the Manhattan area and we use open-source travel times and taxi trip data available in (NYC taxi and Limousine Commission, 2020).

Finally, the Chicago network is composed of 76 regions for which we retrieve the open-source data from (Chicago Data Portal, 2021).

To analyze the stable distributions of the demanded trips, we filter the data by only considering working days (Monday to Friday). Then, we focus on four time slots: Morning Peak (AM) from 7:00-10:00 hrs, Noon (MD) from 12:00-15:00 hrs, Afternoon Peak (PM) from 17:00-20:00 hrs and Night (NT) from 00:00-3:00 hrs. For every time slot we compute the average hourly demanded trips and travel times for every origin-destination pair and we use this information to preform our experiments.

Before stating our results, note that the static formulations (6.6)-(6.9) still require to define the demand functions $\Lambda_{ij}(u_{ij})$. We assume that customers within and across OD pairs are homogeneous (have the same demand function), and, since we are interested in explicitly solving (6.6)-(6.9) to optimality, we assume a linear willingness

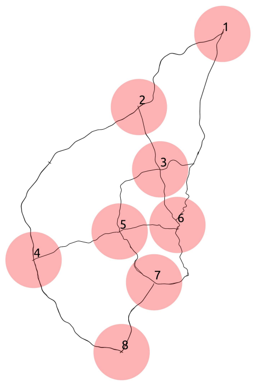


Figure 6·4: EMA

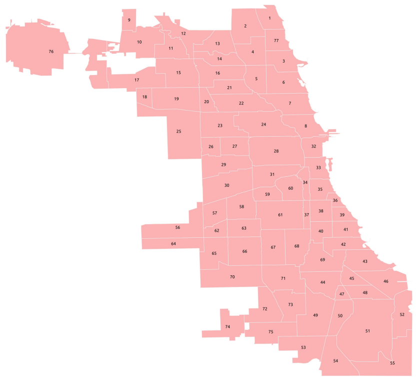


Figure 6·5: Chicago

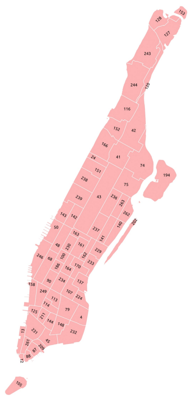


Figure 6·6: NYC

to pay function of the form:

$$\Lambda_{ij}(u_{ij}) = \frac{\lambda_{ij}}{u_{ij}^{\max} - 1}(u_{ij}^{\max} - u_{ij}), \tag{6.12}$$

where we select $u_{ij}^{\max} = 4$. We made this choice using the empirical results reported in (Cohen, Hahn, Hall, Levitt, & Metcalfe, 2016, Table 2) where the number of active customers in the platform looking for drivers for a surge price greater than 4 is negligible. See the shape of this function in Figure 6·7 Using (6.12), our static problem becomes a Quadratic Program (QP) with linear constraints. For all three

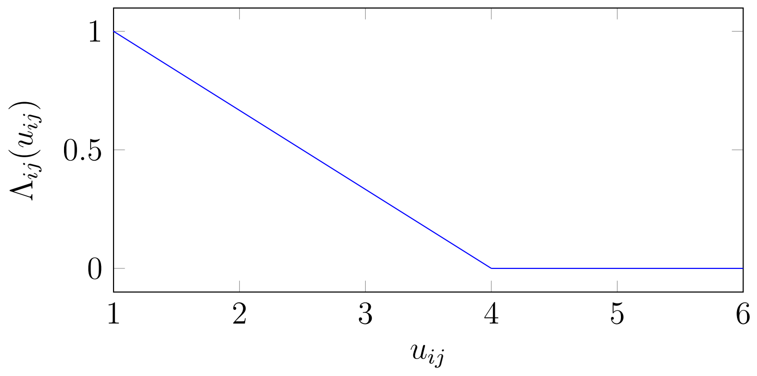


Figure 6·7: Willingness-to-pay function

experiments we let the operational cost and rebalancing cost be equal and proportional to the travel time, $c^o_{ij} = c^r_{ij} = \alpha T_{ij}$, where we select $\alpha = 0.72$ by transforming the distance-based cost suggested in (Bösch, Becker, Becker, & Axhausen, 2018, Section 2.1.2) for a midsize vehicle to a time-based cost (dollars per minute). Additionally, we let the *base price* be a multiplier of the operational cost $p^0_{ij} = \beta c^o_{ij}$ where we select $\beta = 1.75$ which can be interpreted as the minimum margin over the operation cost that the platform is willing to charge. We set the cost of losing customers due to the absence of vehicles in that region to be $c^c = \$5$, and the car ownership cost be $c^f = \$1.98$ per vehicle per hour as suggested in (AAA, 2019).

6.6.1 Joint solution

We are interested in understanding the achievable benefits of solving the joint problem over different static approaches. We refer to $\mathcal{P}_{ij} + \mathcal{R}_{ij}$ as the *joint* strategy stated in (6.9), which solves the pricing and rebalancing for every origin and destination.

First, we compare $\mathcal{P}_{ij} + \mathcal{R}_{ij}$ with an individual *pricing* policy $\mathcal{P}_{ij}$ that only adjusts prices without rebalancing the system, equivalent to solving (6.6). Second, we consider the policy $\mathcal{R}_{ij}$ of only solving the *rebalancing* problem (6.8) with a fixed set of prices, in particular for this policy we set $\mathbf{u} = \$2.66$ which is the maximizer of (6.12). Third, we compare against a *sequential* approach $\mathcal{R}_{ij} \rightarrow \mathcal{P}_{ij}$ which involves solving the the rebalancing and using its solution to solve the pricing problem. Our motivation for this methodology comes from the fact that current MoD platforms tend to separate their pricing and rebalancing processes. Note that the sequential policy $\mathcal{P}_{ij} \rightarrow \mathcal{R}_{ij}$ is not included because once the pricing problem is solved, the system is already balanced and the rebalancing problem becomes trivial (i.e., $\mathbf{r} = 0$). Finally, we also consider the *joint with fixed prices by origin* policy $\mathcal{P}_i + \mathcal{R}_{ij}$ which is motivated by the fact that current MoD services only use the origin (not the destination) when setting surge prices (see L. Chen, Mislove, and Wilson (2015) and Cohen et al. (2016)).

In Table 6.1 we report the *relative deviation* between a policy π and the joint policy $\mathcal{P}_{ij} + \mathcal{R}_{ij}$. Formally, let J_π be the optimal value of (6.9a) for a policy π. Then, the *relative deviation* is $(J_{\mathcal{P}_{ij}+\mathcal{R}_{ij}} - J_\pi)/J_\pi$, which measures the improvement in performance of $\mathcal{P}_{ij} + \mathcal{R}_{ij}$ relative to policy π.

Our results (in Table 6.1) show that $\mathcal{P}_{ij} + \mathcal{R}_{ij}$ outperforms all the other policies, highlighting the benefit of solving this problem using a joint approach. In particular, we observe that each of the individual strategies performs on average worse than strategies that optimize both pricing and rebalancing. It is relevant to stress the 2% to 3% deviation of the policy with *fixed surge price by origin*, as it reports the relevance of considering the destination when pricing. This happens because considering the destination in the pricing policy helps to balance the system via the selection of prices.

Table 6.1: Relative deviation in of each policy compared to the joint policy $\mathcal{P}_{ij} + \mathcal{R}_{ij}$ for different networks and time slots.

Network		$\mathcal{P}_i + \mathcal{R}_{ij}$	$\mathcal{P}_{ij}$	$\mathcal{R}_{ij}$	$\mathcal{R}_{ij} \to \mathcal{P}_{ij}$
EMA	AM	0.30%	7.05%	27.1%	1.20%
EMA mid		0.40%	55.9%	29.3%	6.56%
Anaheim		1.2%	50.3%	71.6%	7.24%
Chicago	AM	0.80%	58.5%	17.4%	4.07%
	MD	1.20%	1.27%	19.8%	0.16%
	PM	1.2%	4.07%	18.9%	0.50%
	NT	2.00%	27.5%	25.8%	2.7%
Manhattan	AM	1.00%	68.9%	40.3%	7.15%
	MD	1.00%	2.49%	38.3%	0.42%
	PM	1.0%	7.40%	40.5%	1.18%
	NT	1.80%	49.2%	68.4%	6.34%

To better understand the different approaches, we generated plots of the pricing distribution and trend. Figure 6·8 shows histograms comparing the value of the solution $\mathbf{u}$ for the individual pricing policy and the joint strategy. We observe the distribution of the individual approach to have higher variance than the joint method. This happens given the hard constraint to reach an equilibrium. When no rebalancing is considered as in $\mathcal{P}_{ij}$ the policy chooses prices to ensure (6.9b). In contrast, when solving the joint problem, the solution leverages rebalancing and pricing and gives the pricing decision more flexibility to concentrate to select values that maximize profits.

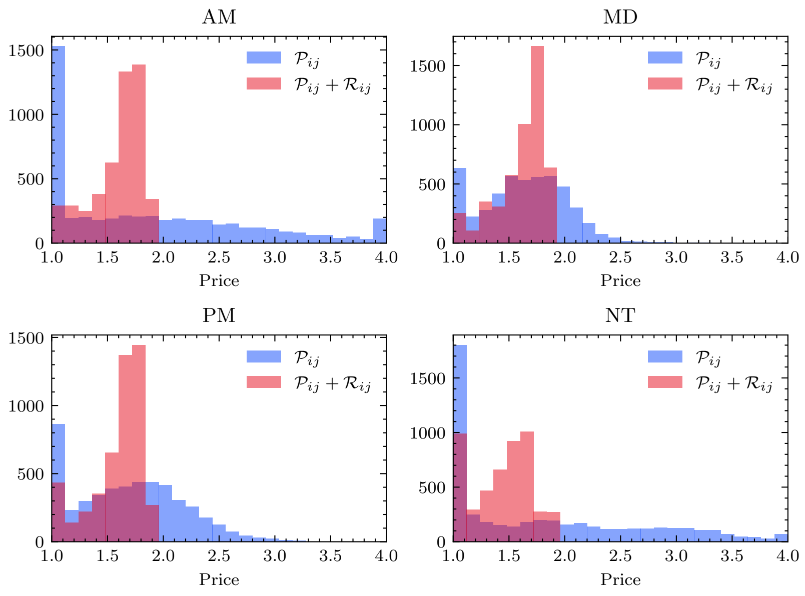


Figure 6·8: Distribution of prices $\mathbf{u}^*$ for different policies at different time slots

Finally, we quantify how relevant the pricing is relative to the rebalancing component when balancing the load of the system. Letting $\mathbf{r}^*$ and $\mathbf{u}^*$ be the solution of (6.9), we define a load dispersion metric as follows $\bar{\zeta}_0 = \frac{1}{N}\sum_i |(\sum_j \lambda_{ij} - \lambda_{ji})|$ when nothing is applied, $\bar{\zeta}_{\mathbf{r}} = \frac{1}{N}\sum_i |(\sum_j \lambda_{ij} + r_{ij} - \lambda_{ji} - r_{ij})|$ when the rebalancing component is applied, and $\bar{\zeta}_{\mathbf{u}} = \frac{1}{N}\sum_i |(\sum_j \Lambda_{ij}(u_{ij}) - \Lambda_{ji}(u_{ji}))|$ when the pricing component (but no rebalancing) is applied. Note that we do not define $\bar{\zeta}_{\mathbf{u},\mathbf{r}}$ as the result will be zero given that the system is at equilibrium by (6.9b). Table 6.2 shows this dispersion metric for the different time slots considered. Interestingly, we see that the pricing component of the policy reduces this metric in all cases, showing its relevance for load balancing the system while also maximizing profit.

Table 6.2: Dispersion on the average absolute value of potentials when components of the joint policy $\mathbf{u}^*$ and $\mathbf{r}^*$ are applied.

	AM	**MD**	**PM**	**NT**
ζ_0	57.03	16.62	34.77	17.64
$\zeta_{\mathbf{u}^*}$	20.44	4.10	6.80	6.24
$\zeta_{\mathbf{r}^*}$	36.71	13.23	28.36	11.49

6.6.2 Fleet size selection and system utilization

An important challenge for MoD systems is to properly select the correct number of drivers or autonomous vehicles to satisfy demand. One of the benefits of our static formulation is that its solution includes the variable m indicating the minimum fleet size to operate the system. However, this value is calculated assuming the steady-state solution of the system and does not account for the variance and perturbations that occur in the real world.

In this experiment, we are interested in analyzing how this fleet size suggestion behaves in a more dynamic environment where perturbations exist. To test this, we have built a simulator of the MoD system which is publicly available on an online repository[1]. The variance (or randomness) of the simulation comes from the Poisson processes modeling the modulated customer arrivals and rebalancing vehicle departures. The customer arrivals times come from a Poisson process with a rate estimated from the data and modulated by the static prices. Similarly, the rebalancing events arrive following a Poisson process with rate equal to the solution of the static problem.

We perform this experiment using the EMA network for which we first solve problem (6.9). Let the optimal solution to (6.9) be $\mathbf{u}^*, \mathbf{r}^*$ and m^*. Then, in our simulator we fix prices to $\mathbf{u}^*$ and vary the fleet size by selecting $m = \gamma m^*$ for $\gamma = [0, 3]$. Note that $\gamma = 1$ is equivalent to using the suggested fleet size m^*. We run our simulations until a steady-state is reached, using two different rebalancing policies: fluidic and N+1.

In Figure 6·9 we observe that the fleet size m^* performs very well. The top left plot shows that for $\gamma \leq 1$ the profit increases as we add vehicles to the fleet. This happens since the fleet size is too small to provide service to the platform's demand. In contrast, for $\gamma > 1$, we see that the profit decreases as γ increases. This is because

[1]https://github.com/salomonw/mobility-on-demand-control

the negative fixed cost of owning and maintaining an extra vehicle in the fleet is higher than profit it can produce (since most demand is already satisfied). As a result, this experiment suggests that solving (6.9) provides an automated procedure to determine a nominal fleet size.

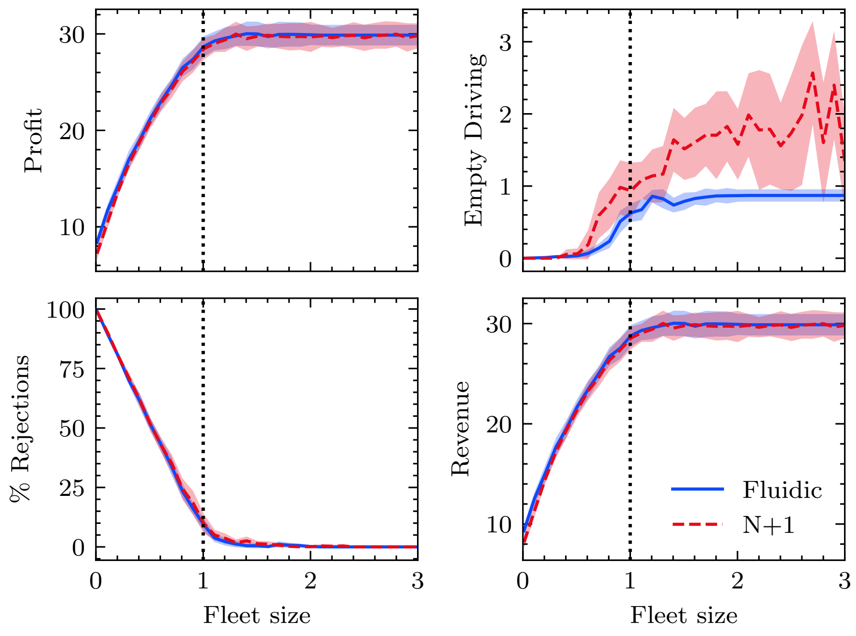


Figure 6·9: Performance indicators for different fleet sizes. Upper left plot shows the expected profit per minute. The upper right plot shows the percentage of time that vehicles are rebalancing. The lower left shows the percentage of customer requests that were rejected because there were no available vehicles to serve the customer. Finally, the lower right plot shows the revenue per minute.

In addition, we are interested in quantifying the utilization for different fleet sizes. In other words, we would like to measure how much time the fleet of vehicles spends waiting for customers, transporting a passenger, or driving to rebalance the system. Figure 6·10 shows the results for a simulation of the system for a total time of 10 hours where we observe that *vehicle utilization*, defined as the time that a vehicle is either transporting a customer or rebalancing, is around 75% compared with typical private vehicle utilization of 5%. Interestingly, the percentage of the total time that vehicles are rebalancing is practically negligible. We believe this value is small because the pricing policy is helping the system to be balanced, in other words, the "User" flow in Figure 6·10 is helping to balance the system. This is an interesting observation as there is an ongoing debate on the congestion effects that MoD rebalancing has caused in our cities (Fitzsimmons & Hu, 2017).

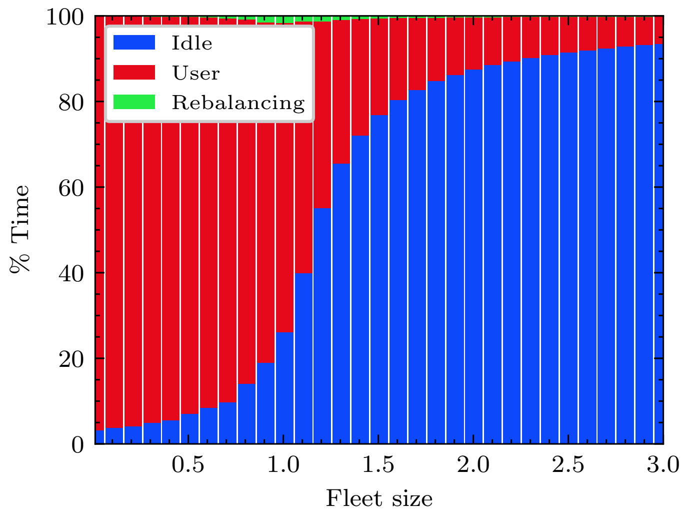


Figure 6·10: System utilization over different fleet sizes. At the suggested fleet size, corresponding to $\gamma = 1.0$, we observe that vehicle utilization is around 70% compared to 5% for private vehicles.

6.6.3 Responsiveness

One characteristic that static policies lack is their *responsiveness* to system perturbations or to changes in the environment. For example, consider the case when a sports event or concert finishes and all its attendees are requesting a transportation service to reach their destinations. In this situation, the steady demand is perturbed for a certain amount of time. We showcase this situation by running 15 simulations of the EMA network over a time period of 10 hours. For each of these simulations we intentionally perturb the system between minutes 300 and 380 by multiplying the demand from a particular region to all its destinations by a constant factor; in this example we use 3. We select the update rule of parameter $\boldsymbol{\theta}(t)$ to be the naive approach presented in Section 6.5.3 with $\tau = 10$ minutes and $\Omega = 15$.

Figure 6·11 shows the trajectory of the (i) profit per minute, (ii) the fraction of loss requests due to a lack of vehicles in the region, and (iii) the percentage of empty driving. We observe that the N+1 controller, which operates in real time, is capable of responding to demand perturbations in comparison with the fluidic controller. The second plot of Figure 6·11, we shows how the percentage of rejections for the N+1 controller is lower than the fluidic one within the perturbation range $[300, 380]$. Moreover, in the last plot we see how the the N+1 controller increases its empty driving minutes responding to the demand shift experienced by the system. Finally, in the upper plot we see minimal differences on profits as we have assumed that the cost of rejecting a customer is small.

In conclusion, Figure 6·11 shows that the N+1 controller provides service to more customers and hence incurs a higher rebalancing cost. In contrast, the fluidic model lowers its rebalancing costs by dropping more customers and incurs a reduction in profits and an additional cost for lost customers.

Either strategy could be efficient for a given cost function but, in general, the N+1 approach is more responsive and customer-oriented as it adjusts to customer demand in real-time. This is especially important at times when the system might be transitioning from one stationary distribution to another, for example from an AM to MD period.

6.7 Summary and Future Work

In this chapter we have addressed difficult operational decisions that shared mobility services face when operating their platforms. We discussed how to properly select the right number of vehicles to operate the platform, as well as how to choose prices to maximize a utility function while providing good service to customers.

Of particular interest, we have designed automated models that take as input the network topology, the estimated demand and a willingness-to-pay function of customers, and provide a framework to define the fleet size, the prices, and a real-time rebalancing policy for their proper operation.

We observe that it is of high value to design pricing together with rebalancing policies as well as to consider the customers' destination when defining prices. This modification in the pricing strategy achieves higher profits for the platform and helps rebalancing the system in a more equitable fashion, i.e., if a passengers' destination is helping to balance the system, her price will be lower compared with other passengers whose destination generate imbalances.

Arguably, our model deals with a simple linear willingness-to-pay function in order to provide tractable optimization models and be able to compare our results. Nevertheless more flexible willingness-to-pay functions such as logarithmic or exponential functions can be explored as part of future research in this area.

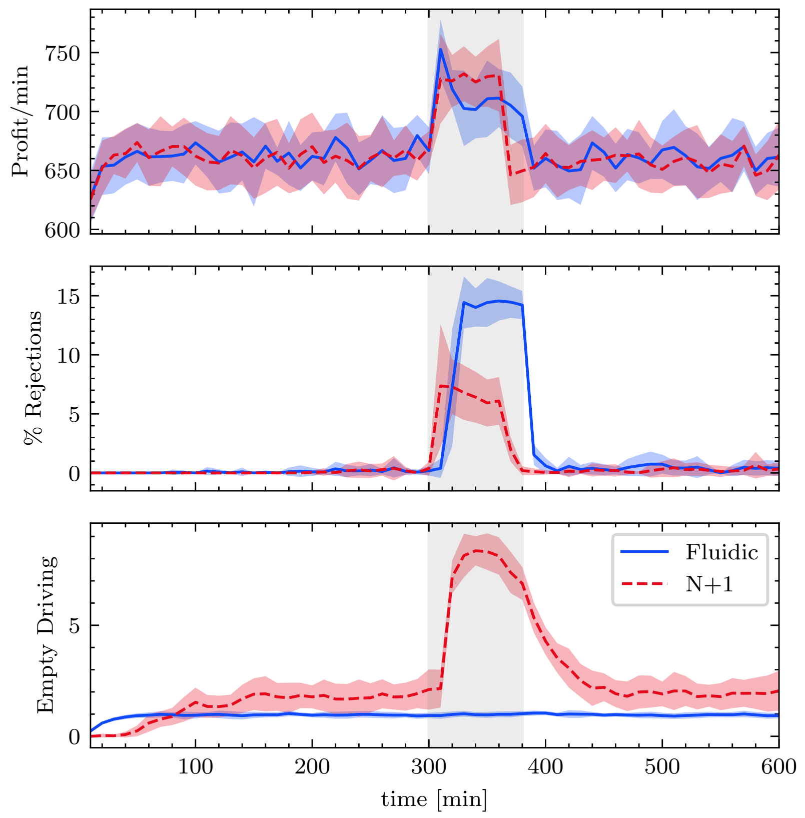


Figure 6·11: Responsiveness of the dynamic policy versus a static policy. Between minutes 300 to 380 (shaded region) we have perturbed the demand of a single region by multiplying it by a factor of three. We observe in the two lower plots how the real time controller N+1 responds to this by sending more vehicles to these region compared with the static fluidic controller. As a result, N+1 provides a better service as the number of customer rejections is lower compared to the fluidic controller.

Chapter 7

Conclusion and Future Work

In this dissertation we developed and tackled five network optimization problems that are of interest to transportation agencies and Autonomous Mobility-on-Demand (AMoD) platforms. A common objective of most problems is to come up with strategies to reduce traffic congestion in transportation networks while keeping the same volume of demand. We addressed the traffic congestion problem by exploiting mainly two strategies: (i) socially routing vehicles –as opposed to selfish routing–, and (ii) using contraflow lane reversals to the existing infrastructure. By applying both of these to case studies, they indicate overall travel time reductions of around 5%.

A key methodological and theoretical contribution of this thesis consists of approximating the travel latency function of the traffic assignment problem (TAP) with a piecewise affine function. This allows writing the TAP as a quadratic program (or linear program) resulting in faster solution times, the possibility to include additional linear constraints, and perform efficient sensitivity analysis. We applied this framework in Chapters 4 and 5 to solve the contraflow lane reversal problem and the joint optimization of routing and rebalancing policies of AMoDs, respectively, showing better and more efficient solutions than alternative methods.

Besides the piecewise affine approximation contribution, Chapter 2 develops a feasible-direction trust-region algorithm to tackle the joint estimation of the Origin Destination (OD) demand and travel latency functions of the TAP and it shows a faster convergence and better estimates compared with other algorithms. Chapter 3 uses the diagonalization scheme to study the interaction between socially-optimum and user-centric commuters traveling in a network and estimates inefficiencies due to user-centric routing of around 10%. We hope these results motivate traffic agencies to design incentives to steer users behavior toward the socially-optimum solution. Finally, Chapter 6 studies how pricing policies that consider origin and destinations can stabilize the AMoD system and find a balance between the load of customers and vehicles. Then, it proposes an optimization model that jointly selects pricing, rebalancing and fleet size decisions. The gains by jointly solving the problem could vary between 1% to 50% depending on the benchmark highlighting the importance on the role of the destination of passengers when optimizing the system.

Future Research

We identify two main research directions that are motivated from the work presented here: (1) Robust network control and optimization; and (2) Incentive design in networked systems.

Robust Network Optimization and Control

The goal is to build robust optimization and control algorithms for decision-making over networks using historical data and employing machine learning methods to predict the likelihood of disruption. Using *robust optimization*, the idea is to select a policy to *best* handle the *worst possible* situation out of a specified uncertainty set. Classically, the uncertainty set is chosen by the modeler, or it might be trivially estimated. A concrete research goal is to create methodologies to dynamically learn and adjust the uncertainty set based on observed data while managing the trade-off between efficiency and resilience. This could be potentially achieved by probing the vulnerabilities of the system's state space using adversarial bandits and Reinforcement Learning (RL) agents. This framework has immediate application in the area of supply chain networks.

One way to incorporate the work of this dissertation in this topic is by using the model developed in the Lane Reversal Problem in Chapter 4 and add uncertainty sets that are learned from the speed data.

Incentive Design for Resource Allocation

In general, a large class of problems can be framed as managing a set of resources over multiple time periods under uncertainty. Some examples include: streets and intersections for traffic management, doctors and nurses for personnel scheduling, and vaccines and tests during a pandemic. Similarly, the uncertainty may have different characterizations for different applications; car arrivals, weather conditions, number of new cases in a pandemic. A classical solution to improve resource allocation in networks is by using monetary incentives, for example congestion tolls in transportation. These approaches based on congestion pricing have two fundamental drawbacks: (i) they discriminate users with lower incomes; (ii) they may fail to consider individual preferences such as temporary urgency and needs. It is desirable to investigate incentive schemes based on non-monetary currencies to align the self-interest of users with a system-optimum allocation of resources while accounting for their temporal individual needs.

A starting point for this research could be accomplished along the lines of Salazar, Paccagnan, Agazzi, and Heemels (2021). In particular the idea is to use a currency (or tokens) that can neither be bought nor exchanged, but only spent or earned when using the resource either altruistically, neutrally, or selfishly. Since these tokens cannot be exchanged, there exist fewer opportunities to create black markets or other types of exchange methods. The key idea is that when users play this resource allocation game *repeatedly*, the system will converge to the system-optimal solution. In the context of transportation, this game will be played at each trip and the user will be able to determine if it will earn or spend tokens based on presumably the travel time. A longer route will result on earning tokens while a shorter route will require spending tokens.

Appendix A

Proofs

A.0.1 Chapter 5

Lemma 5.3.1

Proof. Notice that the maximum total error between the LP and QP is expressed by:

$$
\begin{aligned}
&\sum_{(i,j)\in\mathcal{A}} \max_{l=1,\ldots,n} \left\{a_l t^0_{ij}((\theta^{(l+1)}_{ij} - \theta^{(l)}_{ij})\varepsilon^{(l)}_{ij} - (\varepsilon^{(l)}_{ij})^2)\right\} && \text{(A.1a)}\\
&= \sum_{(i,j)\in\mathcal{A}} \max_{l=1,\ldots,n} \left\{a_l t^0_{ij}(\frac{\theta^n_{ij}}{n}\varepsilon^{(l)}_{ij} - (\varepsilon^{(l)}_{ij})^2)\right\} && \text{(A.1b)}\\
&= \sum_{(i,j)\in\mathcal{A}} \max_{l=1,\ldots,n} \left\{a_l t^0_{ij}(\frac{\theta^n_{ij}}{n}\frac{\theta^n_{ij}}{2n} - (\frac{\theta^n_{ij}}{2n})^2\right\} && \text{(A.1c)}\\
&= \sum_{(i,j)\in\mathcal{A}} a_n t^0_{ij}(\frac{(\theta^n_{ij})^2}{4n^2}) && \text{(A.1d)}\\
&= \frac{a_n(\theta^n)^2}{4n^2}\sum_{(i,j)\in\mathcal{A}} t^0_{ij} m^2_{ij} && \text{(A.1e)}
\end{aligned}
$$

where the first equality comes from the fact that the we use uniform distances for the piecewise regions, the second equality comes from the fact that $\varepsilon^{(l)*}_{ij} = \theta^2_{ij}/2n$ maximizes equation (A.1b). Finally the last steps selects the last segment n by observing that a_n has the steepest slope by assumption.

□

Theorem 2

Proof. Without loss of generality, let us select the thresholds $\boldsymbol{\theta}$ in a uniform manner as in Lemma 5.3.1. The proof follows immediately after observing that, for a bounded a_n and m_{ij}, the error goes to zero as $n \to \infty$. □

Lemma 5.3.2

Proof. We use contradiction. Assume that there is a cycle $\mathcal{C}$ where $(i_1, i_2), (i_2, i_3), \ldots, (i_k, i_1) \in \mathcal{A}^s$ and let h_i^s be the marginal cost (related to the SO solution) of the cheapest path from s to i. Further, consider any $(i, j) \in \mathcal{A}^s$, which implies that (i) there exists a positive flow path from s to i and (ii) $x_{ij}^s > 0$. Since we are minimizing a function where $t_{ij}(x)$ and $\hat{t}_{ij}(x)$ are strictly positive and monotonically increasing for all $(i, j) \in \mathcal{A}$, the path connecting s to i has to be a minimum cost path. Assume this cost to be T_{si} and since $x_{ij}^s > 0$, the cost to j is $T_{sj} = T_{si} + t_{ij}$. Note that by definition $T_{sj} \leq T_{si} + t_{ij}$. However, if $T_{sj} < T_{si} + t_{ij}$ then there must exist a lower-cost path to j than any of those passing through i. Hence, we must have $T_{sj} = T_{si} + t_{ij}$ for all the links in $\mathcal{A}^s$. Using the fact that all travel times are strictly positive, this implies that for the cycle $\mathcal{C}$ we have $T_1 < T_2 < \ldots < T_k < T_1$ which is logically inconsistent. □

Lemma 5.3.3

Proof. Similar to Rossi (2018), we prove this by construction and use the flow decomposition algorithms and results in (Ahuja, Magnanti, & Orlin, 1993, Thm. 3.5). We begin by decomposing the origin-based solution $\mathbf{x}^s$ of origin s to a set of acyclic paths $\mathcal{P}^s$ such that $\sum_{(s,t) \in \mathcal{W}} \mathbf{x}^{\mathbf{w}} = \mathbf{x}_t^s = d_{(s,t)}^u$, where $\mathbf{x}_t^s$ is the acyclic decomposed flow from $\mathbf{x}^s$ going from s to t, and $d_{(s,t)}^u$ is the demand from s to t. We conclude the proof by observing that the origin-based solution has no cycle (by Lemma 5.3.2) and the fact that it is possible to decompose the problem to flows using (Ahuja et al., 1993, Thm. 3.5). □

A.0.2 Chapter 6

Proposition 1

Proof. For 1), we use the framework in Haddad (1981). In particular, we check that all assumptions and conditions of (Haddad, 1981, Thm II-1) are satisfied. This theorem, ensures the existence of Fillipov solutions to the time-delayed differential equations with discontinuous right-hand sides.

To prove the second claim, we separate the vehicle dynamics in two parts: vehicles in transit $x_{ij}(t)$, and vehicles at a specific region $x_i(t)$. For the vehicles queued at i we know their dynamics are as in (6.1b). For the vehicles in transit, we let the total be

$$x_{ij}(t) = \int_{t-T_{ij}}^{t} \lambda_{ij} H(x_i(\tau)) + (\Lambda_{ij} - \mu_i) H(c_i(\tau)) H(x_i(\tau)) \, d\tau,$$

and their dynamics are

$$\begin{aligned} \dot{x}_{ij}(t) = \Lambda_{ij} H(v_i) &+ (\Lambda_{ij} - \mu_i) H(c_i) H(x_i) \\ &- (\Lambda_{ij} H(x_i^j) + (\Lambda_{ij} - \mu_i) H(c_i^j) H(x_i^j)). \end{aligned}$$

Hence, we let the total number of vehicles in the system be $m(t) = \sum_i x_i(t) + \sum_{ij} x_{ij}(t)$ with dynamics:

$$\begin{aligned} \dot{m}(t) &= \sum\nolimits_i \dot{x}_i(t) + \sum\nolimits_{ij} \dot{x}_{ij}(t), && \text{(A.2a)} \\ &= \sum\nolimits_i \Big(- \lambda_i H(x_i) + (\lambda_i - \mu_i) H(c_i) H(x_i) && \text{(A.2b)} \\ &\qquad + \sum\nolimits_j \Lambda_{ji} H(x_j^i) - (\Lambda_{ji} - \mu_j) H(c_j^i) H(x_j^i) \Big) + \sum\nolimits_{ij} \dot{x}_{ij}, \\ &= \sum\nolimits_{ij} -\Lambda_{ij} H(x_i) + (\Lambda_{ij} - \mu_i) H(c_i) H(x_i) && \text{(A.2c)} \\ &\qquad + \sum\nolimits_{ij} \Lambda_{ji} H(x_j^i) - (\Lambda_{ji} - \mu_j) H(c_j^i) H(x_j^i) + \sum\nolimits_{ij} \dot{x}_{ij}, \\ &= 0. && \text{(A.2d)} \end{aligned}$$

Note this result is obtained by expanding the first term in (A.2a) using (6.1b), rearranged terms and found that $-\sum_i \dot{x}_i(t) = \sum_{ij} \dot{x}_{ij}(t) \implies \dot{m} = 0$, which implies that the fleet size remains invariant over time. □

Theorem 3

Proof. Set $\dot{c}_i = 0$ and $\dot{x}_i = 0$ for all $i \in \mathcal{N}$. Then by using the customer system dynamics in (6.1a), we have:

$$\Lambda_i = \Lambda_i H(x_i) - (\Lambda_i - \mu_i) H(c_i) H(x_i), \tag{A.3}$$

and since $\Lambda_i < \mu_i$, the above equation just has a solution if $c_i = 0$ and $x_i > 0$ for all $i \in \mathcal{N}$. Setting $\dot{x}_i = 0$, and using the vehicle dynamics in (6.1b) we have

$$0 = -\Lambda_i H(x_i) + (\Lambda_i - \mu_i) H(c_i) H(x_i)$$

$$+\sum\nolimits_j \Lambda_{ji}H(x_j^i) - (\Lambda_{ji}-\mu_j)H(c_j^i)H(x_j^i). \tag{A.4}$$

Then. by combining (A.3) and (A.4) together with the fact that $\mathbf{c} = 0$ we get $0 = -\Lambda_i + \sum_j \Lambda_{ji}H(x_j)$. Which we used the fact that in the stationary equilibrium x_i and c_i are constants and there is no dependence on $t - T_{ij}$.

Recall that for every equilibrium solution, we require $\mathbf{x} > 0$ and thus $H(x_i) = 1,\ \forall i \in \mathcal{N}$. Therefore, a necessary condition for the existence of equilibria is that the prices $\mathbf{u}$ are chosen such that $0 = -\Lambda_i + \sum_j \Lambda_{ji},\ \forall i \in \mathcal{N}.$, which proves the first statement.

We now want to verify that the fleet size is large enough to maintain an equilibrium flow. Recall the fleet size dynamics $\dot{m}(t)$ when $\mathbf{c} = 0$ and $\mathbf{x} > 0$ in (A.2). Observe that to satisfy $\mathbf{x} > 0$, one needs to have a fleet size of at least $\sum_{ij} T_{ij}\Lambda_{ij}(u_{ij})$ vehicles which is the definition of $m_{\mathbf{u}}$. This, mixed with its invariant property ($\dot{m} = 0$), proves the claim. Conversely, if $m < m_{\mathbf{u}}$ no equilibrium exists. □

Theorem 4

Proof. We start by showing that $x_i(\tau) > c_i(\tau)$ for $\tau \in [-\max_{i,j} T_{ij}, t)$, which will serve as a key element to the analysis. To do so, we first assume $x_i(\tau) > 0$ for all $i \in \mathcal{N}$, and observe the system dynamics (6.1) at time t are

$$\dot{c}_i(t) = (\Lambda_i - \mu_i)H(c_i), \tag{A.5a}$$

$$\dot{x}_i(t) = -\Lambda_i + (\Lambda_i - \mu_i)H(c_i) + \sum_j (\Lambda_{ji} - (\Lambda_{ji} - \mu_j)H(c_j^i)), \tag{A.5b}$$

$$= (\Lambda_i - \mu_i)H(c_i) - \sum_j (\Lambda_{ji} - \mu_j)H(c_j^i), \tag{A.5c}$$

$$\geq (\Lambda_i - \mu_i)H(c_i), \tag{A.5d}$$

$$= \dot{c}_i(t), \tag{A.5e}$$

where all the $H(x_i)$ in (6.1) are replaced with 1 since we assume that $x_i(\tau) > 0$. The step (A.5c) is due to the fact that the system is at equilibrium, i.e. $\sum\limits_j \Lambda_{ji} - \Lambda_i = 0,\ \forall i \in \mathcal{N}$, and the step (A.5d), is a result of $\mu_i > \Lambda_i$ which means that $\Lambda_i - \mu_i < 0$. Given that $\dot{x}_i(t) \geq \dot{c}_i(t)$ and the fact that at the starting point $\mathbf{x} > \mathbf{c}$ (i.e., $\underline{\mathbf{x}} > \mathbf{0}$), we conclude that $x_i(\tau) > c_i(\tau)$ for $\tau \in [-\max_{i,j} T_{ij}, t)$ and $i \in \mathcal{N}$.

Two important consequences of this result are that c_i always reaches 0 before its corresponding x_i, and the vehicle time derivative $\dot{x}_i$ is greater than or equal to 0 after

c_i has reached 0. This follows by observing that all the terms in (6.1b) are all positive when $c_i = 0$.

Now we are in a position to show that $\mathbf{x}(t) > 0$ for $t \geq 0$. We do this by combining the second consequence in the previous paragraph with the assumption that the initial state of the system is $(\mathbf{0}, \underline{\mathbf{x}})$ and the fact that $\dot{x}_i(\tau) > \dot{c}_i(\tau)$. Thus, since $\mathbf{x}(t) > 0$ for $t \geq 0$, we have that $\dot{c}_i(t) = (\Lambda_i - \mu_i)H(c_i) \leq 0$ which will implies that c_i will for sure be 0 when $t \geq T^{\max}$ where $T^{\max} := \max_i\{c_i(0)/(\mu_i - \Lambda_i)\}_{i\in\mathcal{N}}$ and which implies that $\lim_{t\longrightarrow+\infty} \mathbf{c}(t) = \mathbf{0}$ since both $\dot{c}_i$ and c_i will be equal to 0 for all $i \in \mathcal{N}$.

To show that $\lim_{t\longrightarrow+\infty} \mathbf{x}(t) = \mathbf{x}$ we use the fact that $c_i = \dot{c}_i = 0$ for $t > 0$ and insert this into the vehicle dynamics in (6.1b) obtaining $x_i(t) = \Lambda_i(x_i) + \sum_j(\Lambda_{ji}H(x_j^i) - (\Lambda_{ji} - \mu_j)H(c_j^i)H(x_j^i))$. Since, $c_i(t) = 0$ we observe that after $T^{\max} + \max_{i,j} T_{ij}$ time units $H(c_j^i)$ will be equal to zero and therefore $\dot{x}(t) = 0$ for $t > T^{\max} + \max_{i,j} T_{ij}$. Moreover, since $\dot{x}_i(t) = 0$ the $\lim_{t\longrightarrow\infty} x_i(t)$ exists and can be retrieved using $x_i(t) = x_i(0) + \int_0^t \dot{x}_i(s)ds \geq x_i(0) + \int_0^t \dot{c}_i(s)ds = x_i(0) + c_i(t) - c_i(0)$. Given that we show that $x_i(0) > c_i(0)$, we conclude that $\lim_{t\longrightarrow\infty} x_i(t) > 0$. The property $\lim_{t\longrightarrow\infty} x_i(t) > 0 > m - m_{\mathbf{u}}$ holds straight from Proposition 1.

Finally, to characterize the ball $\mathcal{B}_{\mathbf{u}}^{\delta}$ we set $\psi_i := \underline{x_i}\sin(\pi/4)$ and $\psi^{\min} := \min_i \psi_i$ (see Figure A·1). Then, for $\delta = \psi^{\min}$ any path of the system (6.1) starting at $(\mathbf{c}(\tau), \mathbf{x}(\tau)) = (\underline{\mathbf{c}}, \underline{\mathbf{x}})$ for $\tau \in [-\max_{i,j} T_{ij}, 0)$ and satisfying $(\mathbf{c}(0), \mathbf{x}(0)) \in \mathcal{B}_{\mathbf{u}}^{\delta}(\underline{\mathbf{c}}, \underline{\mathbf{x}})$ has a limit which belongs to the equilibrium set $\Upsilon_{\mathbf{u}}$. $\square$

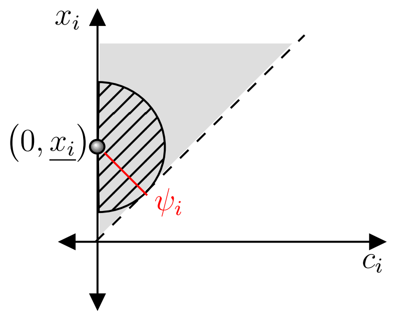


Figure A·1: Sketch of a variable of the initial solution $(\underline{\mathbf{c}}, \underline{\mathbf{x}})$ along with its neighborhood $\mathcal{B}_{\mathbf{u}}^{\delta}$. Shaded in grey is the feasible region $(c_i < x_i)$.

Appendix B

Description of datasets

B.0.1 Eastern Massachusetts Area (EMA)

The Boston Region Metropolitan Planning Organization (MPO) provided access to two datasets of the EMA network: (1) A dataset which includes average speeds (indistinguishable among different vehicle classes) for more than 13,000 road segments (see Fig. 2·2b). This contains the average speed for every minute of the year 2012 and 2015. For each road segment, the dataset provides information about instantaneous, average, and free-flow speeds (in *mph*), date, time, and travel time (in *minutes*). (2) A flow capacity (in *vehicles per hour*) dataset which includes capacity data for more than 100,000 road segments. For more detailed information of the two datasets, see J. Zhang, Pourazarm, Cassandras, and Paschalidis (2016).

B.0.2 New York City (NYC)

We built the NYC transportation network through two open source data sets: (1) OpenStreetMaps (OSM) OpenStreetMap (2017) from where we retrieved the network topology and road characteristics of NYC and (2) the *Uber Movement Speed Data set* Uber (2019) which provides average speeds of road segments on an hourly basis and which can be easily matched with OSM.

Appendix C

Data preprocessing

C.0.1 Selecting a sub-network

- *EMA:* To mitigate the computational complexity while still capturing the key elements of the EMA network, we considered a representative highway sub-network (Fig. 2·2b) where there are 701 road segments, composing a road network with 8 nodes and 24 links. These links and nodes compose a road network $\left(\tilde{\mathcal{V}}, \tilde{\mathcal{A}}, \tilde{\mathcal{W}}\right)$. Every node in the network is also considered as a zone (Origin-Destination candidate).

- *NYC:* Similar to EMA but in an urban environment we sought to reduce the dimensionality of the full network (Fig. C·2a). To do this, we first select the set of nodes from the full network to be the nodes of the NYC subnetwork (Fig. C·1a). Then, we create edges between this set of nodes. We did this in a way that matches our knowledge of the connectivity of NYC. Then, for each new edge connecting these nodes, we solve a shortest path problem on the larger network (Fig. C·2a) and report the travel time as the travel time in the smaller network. Based on this travel times we compute speeds on each link of the subnetwork. Finally, we select some of the nodes to serve just as intersections, and other to be Zones (see Fig. C·1d).

C.0.2 Calculating average speed and free-flow speed

We choose a time instances set $\mathcal{T}$ consisting of a set of minutes or hours (for EMA and NYC, respectively) of a time period of interest and calculate the average speed for each road segment. We match these values with the capacity dataset. Then, for each road segment we compute a proxy of the *free-flow speed* by using the 85th-percentile of the observed speeds on that segment.

C.0.3 Aggregating flows of the segments on each link

For $i \in [\![\tilde{\mathcal{A}}]\!]$, let $\{v_i^j, t_i^j, v_i^{0j}, t_i^{0j}, m_i^j;\ j = 1, \ldots, J_i\}$ denote the available observations (v_i^j, t_i^j), and parameters (v_i^{0j}, t_i^{0j}, m_i^j) of the segments composing the ith *physical* link, where, for each segment j, v_i^j (resp., v_i^{0j}) is the *average speed* (resp., *free-flow speed*; in *miles per hour*), t_i^j (resp., t_i^{0j}) is the *travel time* (resp., *free-flow travel time*; in *hours*), and m_i^j is the *flow capacity* (in *vehicles per hour*). Then, using Greenshield's model Greenshields (1935), we calculate the *traffic flow* (in *vehicles per hour*) on segment j by

$$\hat{x}_i^j = \frac{4m_i^j}{v_i^{0j}} v_i^j - \frac{4m_i^j}{(v_i^{0j})^2} (v_i^j)^2. \tag{C.1}$$

In our analysis, we enforce $v_i^j \leq v_i^{0j}$ to make sure that the flow given by (C.1) is non-negative. In particular, if for some time instance $v_i^j > v_i^{0j}$ (this rarely happens), we set $v_i^j = v_i^{0j}$ in (C.1), leading to a zero flow estimation for this time instance. Aggregating over all segments composed of link i we compute:

$$\hat{x}_i = \frac{\sum_{j=1}^{J_i} \hat{x}_i^j t_i^j}{\sum_{j=1}^{J_i} t_i^j}, \ t_i^0 = \sum\nolimits_{j=1}^{J_i} t_i^{0j}, \ m_i = \frac{\sum_{j=1}^{J_i} m_i^j t_i^{0j}}{\sum_{j=1}^{J_i} t_i^{0j}},$$

where $\hat{x}_i^j$ is given by (C.1) and $t_i^{0j} = v_i^j t_i^j / v_i^{0j}$, $j = 1, \ldots, J_i$.

C.0.4 Adjusting link flows to satisfy conservation

For $i \in [\![\tilde{\mathcal{A}}]\!]$, let $\hat{x}_i$ denote the original estimate of the flow on link i (see the last step), x_i its adjustment, and ξ_{iu} the flow percentage on link i for vehicle class $u \in [\![\tilde{\mathcal{U}}]\!]$ (note that $\xi_{iu} \geq 0$ and $\sum_{u=1}^{|\tilde{\mathcal{U}}|} \xi_{iu} = 1$). Then, $x_{iu} = \xi_{iu} x_i$ (recall that x_{iu} denotes the flow on link $a(i, u)$; i.e., x_{iu} is the flow on link i for vehicle class u). We solve the following *Least Squares* problem:

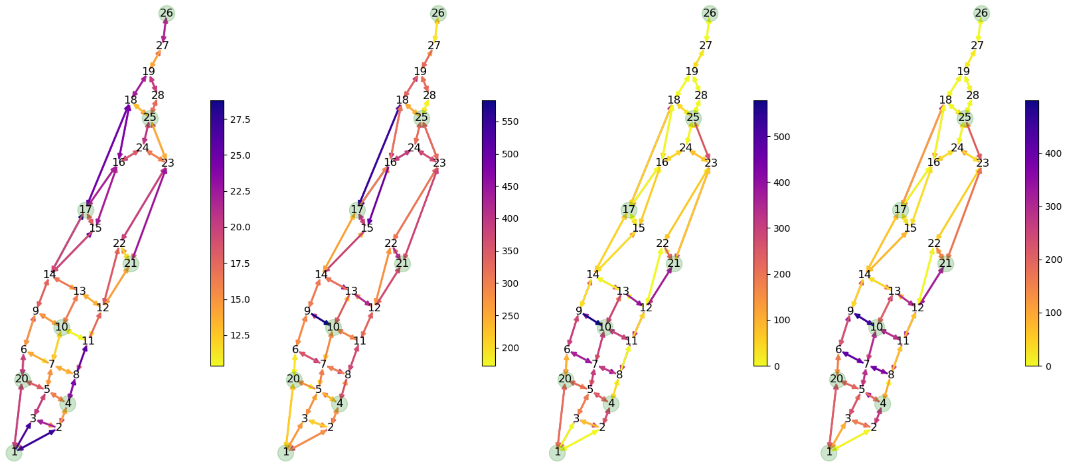
$$\begin{aligned} \min_{\mathbf{x}} \ & \sum_{i=1}^{|\tilde{\mathcal{A}}|} \sum_{u=1}^{|\tilde{\mathcal{U}}|} \xi_{iu}^2 \left(x_i - \hat{x}_i\right)^2 && \text{(C.2)} \\ \text{s.t.} \ & \sum_{i \in \mathcal{I}(v_j)} \xi_{iu} x_i = \sum_{i \in \mathcal{O}(v_j)} \xi_{iu} x_i, && \forall j \in [\![\tilde{\mathcal{V}}]\!],\ u \in [\![\tilde{\mathcal{U}}]\!], \\ & x_i \geq 0, && \forall i \in [\![\tilde{\mathcal{A}}]\!], \end{aligned}$$

where the first constraint enforces flow conservation for each node $v_j \in \tilde{\mathcal{V}}$ with $\mathcal{I}(v_j)$ (resp., $\mathcal{O}(v_j)$) denoting the set of links entering (resp., outgoing) to (resp., from) node v_j. Note that (C.2) generalizes its counterpart in J. Zhang et al. (2016); J. Zhang, Pourazarm, Cassandras, and Paschalidis (2017); J. Zhang et al. (2018); the latter only tackles the case where $|\tilde{\mathcal{U}}| = 1$. For the case where $|\tilde{\mathcal{U}}| = 2$, the datasets available to us do not contain exact information of the parameters ξ_{iu}.

C.0.5 OD Estimation

Given the observed data flows and the set of origin and destination pairs we want to compute the demand for each one of these pairs. To do this, we use the Generalized Least Squares method proposed by Hazelton (2000) which assumes an uncongested network. The basis of this assumption is the fact that the choice probabilities are independent of traffic flow. We begin by defining the following quantities:

- The k-th flow data instance: $\mathbf{x}^{(k)} = (x_a^{(k)}; a \in \mathcal{A}^{(k)})$;



(a) **(b)** **(c)** **(d)**

Figure C·1: NYC subnetwork: (a) Speed (miles/hr); (b) Density (veh/mile); (c) Flow (veh/hr); (d)Conserved flow (veh/hr)

- Sample mean vector: $\bar{\mathbf{x}} = (1/|\mathcal{K}|)\sum_{k=1}^{|\mathcal{K}|}\mathbf{x}^{(k)}$
- Covariance matrix: $\mathbf{S} = (1/(|\mathcal{K}|-1))\sum_{k=1}^{|\mathcal{K}|}(\mathbf{x}^{(k)}-\bar{\mathbf{x}})(\mathbf{x}^{(k)}-\bar{\mathbf{x}})'$
- Probability that commuter of O-D pair i uses route r: p_{ir}
- $\mathbf{g} = (g_i; i \in [[\mathcal{W}]])$ vectorized demand matrix.
- Link-route incidence matrix: $\mathbf{A}$

With all these elements we are able to establish the problem of minimizing the weighted sum of square errors in the flow observations as:

$$\begin{aligned} \min_{\mathbf{P}\succeq\mathbf{0},\mathbf{g}\succeq\mathbf{0}} \quad & \sum_{k=1}^{|\mathcal{K}|}(\mathbf{x^{(k)}} - \mathbf{AP'g})'\mathbf{S^{-1}}(\mathbf{x^{(k)}} - \mathbf{AP'g}) \\ \text{s.t.} \quad & p_{ir} = 0 \qquad \forall (i,r) \in \{(i,r) : r \notin \mathcal{R}_i\} \\ & \mathbf{P'1} = \mathbf{1} \end{aligned} \tag{C.3}$$

.

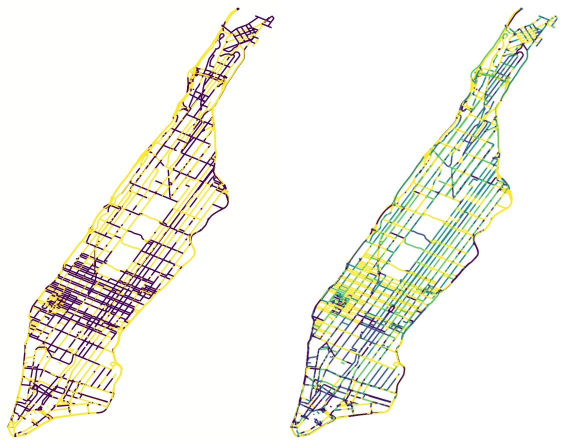


Figure C·2: New York network with speed data (February 13, 2019 at 9:00 a.m.) retrieved from Uber Movement. (a) Speed (miles/hr); (b) Flow (veh/hr)

Solving this problem is hard because of its complicated objective function. To solve it we decouple it into two sub-problems by introducing a new variable defined by $\xi = \mathbf{P}'\mathbf{g}$ and letting $h(\mathbf{P}, \mathbf{g})$ be a scalar valued function. We also define $\mathbf{Q} = \mathbf{A}'\mathbf{S}^{-1}\mathbf{A}$ and $\mathbf{b} = \sum_{\mathbf{k=1}}^{|\mathcal{K}|} \mathbf{A}'\mathbf{S}^{-1}\mathbf{x}^{(\mathbf{k})}$ Then we set the first subproblem to be

$$\min_{\xi \geq \mathbf{0}} \quad \frac{|\mathcal{K}|}{2}\xi'\mathbf{Q}\xi - \mathbf{b}'\xi \tag{C.4}$$

and the second subproblem with the objective of decoupling P and g as:

$$\begin{aligned} \min_{\mathbf{P}\succeq\mathbf{0},\mathbf{g}\succeq\mathbf{0}} \quad & h(\mathbf{P}, \mathbf{g}) \\ \text{s.t.} \quad & p_{ir} = 0 \qquad \forall (i, r) \in \{(i, r) : r \notin \mathcal{R}_\rangle\} \\ & \mathbf{P}'\mathbf{g} = \xi^{\mathbf{0}} \\ & \mathbf{P1} = \mathbf{1} \end{aligned} \tag{C.5}$$

The second sub-problem is equivalent to solving a system of equations and finding whether a solution exists.

Appendix D

Definition of matrices of inverse estimation problem

For the purpose of reproducibility, the detailed definition of the bilevel problem (2.22).
$H_1 = I, \quad H_2 = \operatorname{diag}\Big(\frac{1}{\binom{n}{1}c^{n-1}}, \dots, \frac{1}{\binom{n}{n}c^0}\Big)$.
For the constraint $e'_{iu}N'_k y^w \leq t^0_{iu} \sum_{j=0}^{n} \beta_j \Big(\frac{\theta' x^{(k)}_{a_i}}{m^{(k)}_i}\Big)^j$:

$$A_1 = \begin{bmatrix} \vdots \\ e'_{iu}N'_k \\ \vdots \end{bmatrix}, \quad B_1 = -\begin{bmatrix} \vdots \\ t^0_{iu}\left(\dots, \Big(\frac{\theta' x^{(k)}_{a_i}}{m^{(k)}_i}\Big)^j, \dots\right) \\ \vdots \end{bmatrix},$$

$$C_1 = \mathbf{0}, \qquad h_1 = \mathbf{0}.$$

For the constraint $\sum_{i=1}^{|\tilde{\mathcal{A}^{(k)}}|}\left(\sum_{j=0}^{n} \beta_j \Big(\frac{\theta' x^{(k)}_{a_i}}{m^k_i}\Big)^j\right) \sum_{u=1}^{|\tilde{\mathcal{U}}|} t^0_{iu}x_{iu} - \sum_{w \in \mathcal{W}_k} (d^w)' y^w \leq \epsilon_k$:

$$A_2 = -\begin{bmatrix} \vdots \\ d^w \\ \vdots \end{bmatrix}, \qquad B_2 = \begin{bmatrix} \vdots \\ \sum_{u=1}^{|\tilde{\mathcal{U}}|} t^0_{iu}x_{iu}(\dots, \sum_{i=1}^{|\tilde{\mathcal{A}^{(k)}}|} \Big(\frac{\theta' x^{(k)}_{a_i}}{m^k_i}\Big)^j, \dots) \\ \vdots \end{bmatrix},$$

$$C_2 = [\dots, -1, \dots], \qquad h_2 = \mathbf{0}.$$

For the constraint $\sum_{j=0}^{n} \beta_j \left(\frac{\theta' x_{a_i}^{(k)}}{m_i^k}\right)^j \leq \sum_{j=0}^{n} \beta_j \left(\frac{\theta' x_{a_{\tilde{i}}}^{(k)}}{m_{\tilde{i}}^k}\right)^j$:

$$A_3 = \mathbf{0}, \qquad B_3 = \begin{bmatrix} \vdots \\ \ldots, \left(\frac{\theta' x_{a_i}^{(k)}}{m_i^k}\right)^j - \left(\frac{\theta' x_{a_{\tilde{i}}}^{(k)}}{m_{\tilde{i}}^k}\right)^j, \ldots) \\ \vdots \end{bmatrix},$$

$$C_3 = \mathbf{0}, \qquad h_3 = \mathbf{0}.$$

For the constraint $\epsilon \geq 0$:

$$A_4 = \mathbf{0}, \quad B_4 = \mathbf{0}, \quad C_4 = -I, \quad h_4 = \mathbf{0}.$$

Then we obtain:

$$\begin{aligned} A &= [A_1', A_2', A_3', A_4']', \quad B = [B_1', B_2', B_3', B_4']' \qquad , \\ C &= [C_1', C_2', C_3', C_4']', \quad H = [h_1', h_2', h_3', h_4']'. \end{aligned}$$

Appendix E

Energy Consumption Modeling

In addition to solving the routing problem with the time objective, we are interested in solving the energy optimal routing problem (eco-routing). The first step to solve such a problem is coming up with an energy consumption model for vehicles. Energy consumption of vehicles depends on many different factors including velocity and acceleration as well as the power-train's architecture. Since in eco-routing we are making high level decisions, a low-fidelity model can be sufficient for our needs. We consider two energy models: (i) A charge depleting/charge sustaining energy model described in (Karabasoglu & Michalek, 2013); (ii) An empirical energy model for conventional vehicles estimated by Boriboonsomsin et al. (2012).

CD/CS Energy Model: Karabasoglu and Michalek (2013) proposed an approximated energy model which assumes two operational modes for vehicles: charge depleting (CD), and charge sustaining (CS). In CD, the main propulsion energy comes from the battery pack (electricity) while in CS the vehicle uses the internal combustion engine (gas). They calculated the average energy consumption (in mi/gal and mi/kWh) that a vehicle can travel through different standard drive cycles (NYC, UDDS, HWFET, etc.). These energy consumption values are referred to as μ_{CD} and μ_{CS} respectively and are reported in Table E.1.

Empirical Energy Model: Boriboonsomsin et al. (2012) estimated a model that assumes that the average energy consumption is polynomial function of the link's

Table E.1: Average Energy consumption values Karabasoglu and Michalek (2013)

Vehicle Type	Symbol	Unit	HWFET	UDDS	NYC
PHEV	μ_{CD}	mi/kWh	5.7	6.2	4.2
	μ_{CS}	mi/gal	58.6	69.4	45.7
HEV	μ_{CS}	mi/gal	59.7	69.5	48.0
EV	μ_{CD}	mi/kWh	5.2	4.8	3.1
CV	μ_{CS}	mi/gal	52.8	32.1	16.4

average speed. More specifically, they assume that

$$ln(e_a) = \sum_{i=0}^{4} \theta_i (v_a)^i + \theta_5 R_a, \tag{E.1}$$

in which e_a is the average energy consumption on link a; v_a is the average speed of each link in mph; R_a is the road grade (in percentage); and $\boldsymbol{\theta} = (\theta_i, i = 0, 1, ..., 5)$ is a coefficient vector. Their calibrated values for $\boldsymbol{\theta}$ are reported in Table E.2 and the function for these parameters is shown in Fig. E·1.

Table E.2: Energy cost coefficients Boriboonsomsin et al. (2012)

θ_0	θ_1	θ_2	θ_3	θ_4	θ_5
6.80	-1.4e-1	3.92e-3	-5.20e-5	2.57e-7	1.37e-1

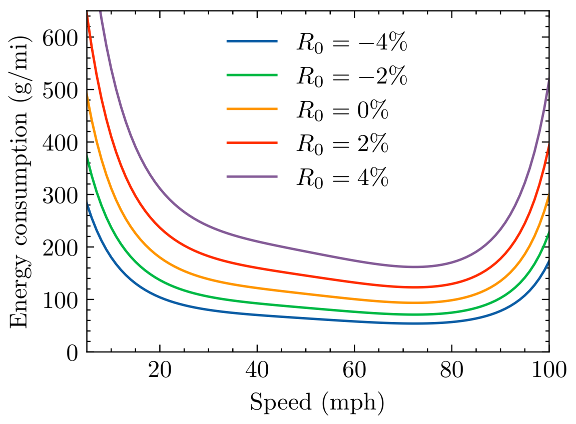


Figure E·1: Average fuel consumption of conventional vehicles using Boriboonsomsin model

Appendix F

MSA Algorithm

Algorithm 5 Method of Successive Averages (MSA)

Input: $(\tilde{\mathcal{V}}, \tilde{\mathcal{A}}, \tilde{\mathcal{W}})$: road network; $\tilde{\mathcal{U}}$: set of vehicle classes; $f(\cdot)$: cornerstone cost function in (2.5); $\mathbf{g}_u$, $u \in [\![\tilde{\mathcal{U}}]\!]$: demand vector for vehicle class u; $\varepsilon_3 > 0$: a real parameter; L: maximum number of iterations.

Step 0: Initialization. Initialize link flows $x_{iu}^{\ell} = 0$ for $i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]$; set iteration counter $\ell = 0$.

Step 1: Compute new extremal flows. Set $\ell = \ell + 1$.

1.1: Update link travel costs based on current link flows: $t_{iu}^{\ell} = t_{iu}(x_{i1}^{\ell-1}, \ldots, x_{i|\tilde{\mathcal{U}}|}^{\ell-1})$, $\forall i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]$.

1.2: Perform "all-or-nothing" assignment of the demands $\mathbf{g}_u$ on current shortest paths and obtain y_{iu}^{ℓ}.

Step 2: Update link flows via

$$x_{iu}^{\ell} = x_{iu}^{\ell-1} + \lambda^{\ell}\left(y_{iu}^{\ell} - x_{iu}^{\ell-1}\right),$$

where $\lambda^{\ell} = 1/\ell$.

Step 3: Stopping criterion. Compute the *Relative Gap* (*RG*)

$$\text{RG} = \frac{\sum_{u \in \tilde{\mathcal{U}}} \sum_{i \in \tilde{\mathcal{A}}} x_{iu}^{\ell} t_{iu}(x_{iu}^{\ell})}{\sum_{u \in \tilde{\mathcal{U}}} \sum_{i \in \tilde{\mathcal{W}}} \pi_{\mathbf{w}(i,u)} d^{\mathbf{w}(i,u)}} - 1.$$

If RG $< \varepsilon_3$ or $\ell \geq L$, terminate; otherwise, return to Step 1.

References

AAA. (2019). *Your driving costs: How much are you really paying to drive?* (Vol. 2717). AAA. Retrieved from `https://exchange.aaa.com/wp-content/uploads/2019/09/AAA-Your-Driving-Costs-2019.pdf`

Ahn, C. W., & Ramakrishna, R. S. (2002). A genetic algorithm for shortest path routing problem and the sizing of populations. *IEEE Transactions on evolutionary computation*, *6*(6), 566–579.

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: Theory, algorithms and applications.* Prentice Hall.

Alonso-Mora, J., Samaranayake, S., Wallar, A., Frazzoli, E., & Rus, D. (2017). On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, *114*(3), 462–467.

Ampountolas, K., dos Santos, J. A., & Carlson, R. C. (2019). Motorway tidal flow lane control. *IEEE Transactions on Intelligent Transportation Systems*, *21*(4), 1687–1696.

Arcand, R. (2019, July 9). Traffic Control Plans for the July 21st NASCAR Race at New Hampshire Motor Speedway in Loudon. *NH Department of Transportation.* Retrieved from `https://www.nh.gov/dot/media/nr2019/20190709-nascar-loudon.htm`

Banerjee, S., Johari, R., & Riquelme, C. (2015). Pricing in ride-sharing platforms: A queueing-theoretic approach. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation* (p. 639). doi: 10.1145/2764468.2764527

Beckmann, M. J., McGuire, C. B., & Winsten, C. B. (1955). *Studies in the economics of transportation.* Yale University Press. doi: 10.1057/jors.1980.83

Bei, X., & Zhang, S. (2018). Algorithms for trip-vehicle assignment in ride-sharing. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32(1)). Retrieved from `https://ojs.aaai.org/index.php/AAAI/article/view/11298`

Belakaria, S., Ammous, M., Smith, L., Sorour, S., & Abdel-Rahim, A. (2019). Multi-class management with sub-class service for autonomous electric mobility on-demand systems. *IEEE Transactions on Vehicular Technology*, *68*(7), 7155–7159.

Bell, M. G. (1983). The estimation of an origin-destination matrix from traffic counts. *Transportation Science*, *17*(2), 198–217.

Bellman, R. (1958). On a routing problem. *Quarterly of applied mathematics*, *16*(1), 87–90.

Ben-Tal, A., El Ghaoui, L., & Nemirovski, A. (2009). *Robust optimization.* Princeton university press.

Bertsekas, D. (2016). *Nonlinear programming* (3rd ed.). Athena Scientific.

Bertsimas, D., Gupta, V., & Paschalidis, I. C. (2015). Data-driven estimation in equilibrium using inverse optimization. *Mathematical Programming*, *153*(2), 595–633.

Bertsimas, D., & Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical Programming*, *98*(1), 49–71.

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM review*, *59*(1), 65–98.

Bimpikis, K., Candogan, O., & Saban, D. (2019). Spatial pricing in ride-sharing networks. *Operations Research*, *67*(3), 744–769.

Boriboonsomsin, K., Barth, M. J., Zhu, W., & Vu, A. (2012, December). Eco-Routing Navigation System Based on Multisource Historical and Real-Time Traffic Information. *IEEE Transactions on Intelligent Transportation Systems*, *13*(4), 1694–1704. doi: 10.1109/TITS.2012.2204051

Bösch, P. M., Becker, F., Becker, H., & Axhausen, K. W. (2018). Cost-based analysis of autonomous mobility services. *Transport Policy*, *64*, 76–91.

Boyles, S. D., Lownes, N. E., & Unnikrishnan, A. (2021). *Transportation network analysis* (.89 ed., Vol. 1). https://sboyles.github.io/blubook.html.

Braess, D. (1968). Über ein paradoxon aus der verkehrsplanung. *Unternehmensforschung*, *12*(1), 258–268.

Branston, D. (1976). Link capacity functions: A review. *Transportation Research*, *10*(4), 223–236. doi: 10.1016/0041-1647(76)90055-1

Brenninger-Göthe, M., Jörnsten, K. O., & Lundgren, J. T. (1989). Estimation of origin-destination matrices from traffic counts using multiobjective programming formulations. *Transportation Research Part B: Methodological*, *23*(4), 257–269.

Cascetta, E. (1984). Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimator. *Transportation Research Part B: Methodological*, *18*(4-5), 289–299.

Cascetta, E., & Nguyen, S. (1988). A unified framework for estimating or updating origin/destination matrices from traffic counts. *Transportation Research Part B: Methodological*, *22*(6), 437–455.

Cascetta, E., & Postorino, M. N. (2001). Fixed point approaches to the estimation of o/d matrices using traffic counts on congested networks. *Transportation Science*, *35*(2), 134–147.

Cassandras, C. G., & Lafortune, S. (2009). *Introduction to discrete event systems.* Springer Science & Business Media.

Chen, L., Mislove, A., & Wilson, C. (2015). Peeking beneath the hood of uber. In *Proceedings of the 2015 Internet Measurement Conference* (pp. 495–508). doi: 10.1145/2815675.2815681

Chen, R., & Cassandras, C. G. (2022). Optimal assignments in mobility-on-demand systems using event-driven receding horizon control. *IEEE Transactions on Intelligent Transportation Systems*, *23*(3), 1969-1983. doi: 10.1109/TITS.2020.3030218

Chicago Data Portal. (2021). *Taxi trips.* Retrieved from `https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psew#column-menu`

Chu, K.-F., Lam, A. Y., & Li, V. O. (2019). Dynamic lane reversal routing and scheduling for connected and autonomous vehicles: Formulation and distributed algorithm. *IEEE Transactions on Intelligent Transportation Systems*, *21*(6), 2557–2570.

Cohen, P., Hahn, R., Hall, J., Levitt, S., & Metcalfe, R. (2016). *Using big data to estimate consumer surplus: The case of uber* (Tech. Rep.). National Bureau of Economic Research. Retrieved from `https://www.nber.org/papers/w22627`

Colini-Baldeschi, R., Cominetti, R., Mertikopoulos, P., & Scarsini, M. (2020). When is selfish routing bad? the price of anarchy in light and heavy traffic. *Operations Research*, *68*(2), 411–434.

Cova, T. J., & Johnson, J. P. (2003). A network flow model for lane-based evacuation routing. *Transportation Research Part A: Policy and Practice*, *37*(7), 579–604.

Dafermos, S. C. (1972). The traffic assignment problem for multiclass-user transportation networks. *Transportation Science*, *6*(1), 73–87.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research*, *8*(1), 101–111.

Dijkstra, E. W. e. a. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269–271.

Doblas, J., & Benitez, F. G. (2005). An approach to estimating and updating origin–destination matrices based upon traffic counts preserving the prior structure of a survey matrix. *Transportation Research Part B: Methodological*, *39*(7), 565–591.

Etehad, M., & Nikolewski, R. (2016, December 23). Millennials and car ownership? it's complicated. *Los Angeles Times*. Retrieved from `https://www.latimes.com/business/autos/la-fi-hy-millennials-cars-20161223-story.html`

Evgeniou, T., Pontil, M., & Poggio, T. (2000). Regularization networks and support vector machines. *Advances in Computational Mathematics*, *13*(1), 1.

Fausset, R. (2018, September 10). As Hurricane Florence Threatens the Carolinas, 1 Million Ordered to Evacuate. *New York Times*. Retrieved from `https://www.nytimes.com/2018/09/10/us/hurricane-florence.html`

Filippov, A. F. (2013). *Differential equations with discontinuous righthand sides: control systems* (Vol. 18). Springer Science & Business Media.

Fisk, C. (1988). On combining maximum entropy trip matrix estimation with user optimal assignment. *Transportation Research Part B: Methodological*, *22*(1), 69–73.

Fisk, C. (1989). Trip matrix estimation from link traffic counts: the congested network case. *Transportation Research Part B: Methodological*, *23*(5), 331–336.

Fitzsimmons, E., & Hu, W. (2017, March 6). The Downside of Ride-Hailing: More New York City Gridlock. *New York Times*. Retrieved from `https://www.nytimes.com/2017/03/06/nyregion/uber-ride-hailing-new-york-transportation.html`

Florian, M., & Chen, Y. (1995). A coordinate descent method for the bi-level od matrix adjustment problem. *International Transactions in Operational Research*, *2*(2), 165–179.

Frank, M., & Wolfe, P. (1956). An algorithm for quadratic programming. *Naval research logistics quarterly*, *3*(1-2), 95–110.

García-Ródenas, R., & Marín, Á. (2009). Simultaneous estimation of the origin–destination matrices and the parameters of a nested logit model in a combined network equilibrium model. *European Journal of Operational Research*, *197*(1), 320–331. doi: 10.1016/j.ejor.2008.05.032

García-Ródenas, R., & Verastegui-Rayo, D. (2013). Adjustment of the link travel-time functions in traffic equilibrium assignment models. *Transportmetrica A: Transport Science*, *9*(9), 798–824.

Grange, L. d., Melo-Riquelme, C., Burgos, C., González, F., & Raveau, S. (2017). Numerical bounds on the price of anarchy. *Journal of Advanced Transportation*, *2017*(5062984). doi: 10.1155/2017/5062984

Greenshields, B. D. (1935). A study of traffic capacity. *Proceedings of the Fourteenth Annual Meeting of the Highway Research Board, Part 1*, 448–477.

Haddad, G. (1981). Monotone trajectories of differential inclusions and functional differential inclusions with memory. *Israel journal of mathematics*, *39*(1-2), 83–100.

Harker, P. T. (1988). Multiple equilibrium behaviors on networks. *Transportation Science*, *22*(1), 39–46.

Harker, P. T., & Pang, J.-S. (1990). Finite-Dimensional Variational Inequality and Nonlinear Complementary Problems: A Survey of Theory, Algorithms and Applications. *Mathematical Programming*, *48*(1-3), 161–220.

Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, *4*(2), 100–107.

Hausknecht, M., Au, T.-C., Stone, P., Fajardo, D., & Waller, T. (2011). Dynamic lane reversal in traffic management. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)* (pp. 1929–1934). IEEE Press. doi: 10.1109/ITSC.2011.6082932

Hazelton, M. L. (2000). Estimation of origin-destination matrices from link flows on uncongested networks. *Transportation Research Part B: Methodological*, *34*(7), 549–566. doi: 10.1016/S0191-2615(99)00037-5

Hertz, A., Laporte, G., & Mittaz, M. (2000). A tabu search heuristic for the capacitated arc routing problem. *Operations Research*, *48*(1), 129–135.

Hörl, S., Ruch, C., Becker, F., Frazzoli, E., & Axhausen, K. W. (2018). Fleet control algorithms for automated mobility: A simulation assessment for Zurich. In *Annual Meeting of the Transportation Research Board* (pp. 18–02171).

Houshmand, A., & Cassandras, C. G. (2018). Eco-Routing of Plug-In Hybrid Electric Vehicles in Transportation Networks. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (pp. 1508–1513). IEEE.

Houshmand, A., Wollenstein-Betech, S., & Cassandras, C. G. (2019). The Penetration Rate Effect of Connected and Automated Vehicles in Mixed Traffic Routing. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)* (pp. 1755–1760). doi: 10.1109/ITSC.2019.8916938

Iglesias, R., Rossi, F., Zhang, R., & Pavone, M. (2016). A BCMP network approach to modeling and controlling Autonomous Mobility-on-Demand systems. In *Workshop on Algorithmic Foundations of Robotics.*

Jha, M., Moore, K., & Pashaie, B. (2004). Emergency evacuation planning with microscopic traffic simulation. *Transportation Research Record: Journal of the Transportation Research Board*, *1886*(1), 40–48.

Josefsson, M., & Patriksson, M. (2007). Sensitivity analysis of separable traffic equilibrium equilibria with application to bilevel optimization in network design. *Transportation Research Part B: Methodological*, *41*(1), 4–31.

Karabasoglu, O., & Michalek, J. (2013). Influence of driving patterns on life cycle cost and emissions of hybrid and plug-in electric vehicle powertrains. *Energy Policy*, *60*, 445–461.

Kim, S., Shekhar, S., & Min, M. (2008). Contraflow transportation network reconfiguration for evacuation route planning. *IEEE Transactions on Knowledge & Data Engineering*, *20*(8), 1115–1129.

Korilis, Y. A., Lazar, A. A., & Orda, A. (1997). Achieving network optima using stackelberg routing strategies. *IEEE/ACM Transactions on Networking*, *5*(1), 161–173.

Koutsoupias, E., & Papadimitriou, C. (1999). Worst-case equilibria. *Annual Symposium on Theoretical Aspects of Computer Science*, 404–413.

Lazar, D. A., Coogan, S., & Pedarsani, R. (2017). Capacity modeling and routing for traffic networks with mixed autonomy. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)* (pp. 5678–5683). IEEE Press. doi: 10.1109/CDC.2017.8264516

Levin, M. W., & Boyles, S. D. (2016). A cell transmission model for dynamic lane reversal with autonomous vehicles. *Transportation Research Part C: Emerging Technologies*, *68*, 126–143.

Levin, M. W., Kockelman, K. M., Boyles, S. D., & Li, T. (2017). A general framework for modeling shared autonomous vehicles with dynamic network-loading and dynamic ride-sharing application. *Computers, Environment and Urban Systems*, *64*, 373 - 383. doi: 10.1016/j.compenvurbsys.2017.04.006

Lighthill, M. J., & Whitham, G. B. (1955). 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.

Liu, S., & Fricker, J. D. (1996). Estimation of a trip table and the θ parameter in a stochastic network. *Transportation Research Part A: Policy and Practice*, *30*(4), 287–305.

Lo, H.-P., & Chan, C.-P. (2003). Simultaneous estimation of an origin–destination matrix and link choice proportions using traffic counts. *Transportation Research Part A: Policy and Practice*, *37*(9), 771–788.

Lu, Z., Meng, Q., & Gomes, G. (2016). Estimating link travel time functions for heterogeneous traffic flows on freeways. *Journal of Advanced Transportation*, *50*(8), 1683–1698.

Lundgren, J. T., & Peterson, A. (2008). A heuristic for the bilevel origin–destination-matrix estimation problem. *Transportation Research Part B: Methodological*, *42*(4), 339–354.

Maher, M. J. (1983). Inferences on trip matrices from observations on link volumes: a bayesian statistical approach. *Transportation Research Part B: Methodological*, *17*(6), 435–447.

Maher, M. J., Zhang, X., & Van Vliet, D. (2001). A bi-level programming approach for trip matrix estimation and traffic control problems with stochastic user equilibrium link flows. *Transportation Research Part B: Methodological*, *35*(1), 23–40.

Malekzadeh, M., Papamichail, I., Papageorgiou, M., & Bogenberger, K. (2021). Optimal internal boundary control of lane-free automated vehicle traffic. *Transportation Research Part C: Emerging Technologies*, *126*, 103060.

Marcotte, P., & Wynter, L. (2004). A new look at the multiclass network equilibrium problem. *Transportation Science*, *38*(3), 282–292.

Martínez, V. (2021). Choca motociclista en carril reversible de Circuito Interior. *Reforma*.

Mehr, N., & Horowitz, R. (2020). How will the presence of autonomous vehicles affect the equilibrium state of traffic networks? *IEEE Transactions on Control of Network Systems*, *7*(1), 96-105. doi: 10.1109/TCNS.2019.2918682

Meng, Q., & Khoo, H. L. (2008). Optimizing contraflow scheduling problem: model and algorithm. *Journal of Intelligent Transportation Systems*, *12*(3), 126–138.

Meng, Q., Lee, D.-H., & Cheu, R. L. (2004). A bilevel programming approach to simultaneously estimate od matrix and calibrate link travel time functions from observed link flows. In *Applications of advanced technologies in transportation engineering (2004)* (pp. 56–60). American Society of Civil Engineers .

Mtoi, E. T., & Moses, R. (2014). Calibration and evaluation of link congestion functions. *Journal of Transportation Technologies*, *4*(2).

Neuhold, R., & Fellendorf, M. (2014). Volume delay functions based on stochastic capacity. *Transportation Research Record: Journal of the Transportation Research Board*, *2421*(1), 93–102.

Noriega, Y., & Florian, M. (2007). *Algorithmic approaches for asymmetric multi-class network equilibrium problems with different class delay relationships.* CIRRELT.

NYC taxi and Limousine Commission. (2020). *Taxi and limousine commission trip record data.* Retrieved from `https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page`

OpenStreetMap. (2017). *Planet dump.* Retrieved from `https://www.openstreetmap.org`

Patriksson, M. (1994). *The Traffic Assignment Problem: Models and Methods* (Vol. 54). Dover Publications.

Patriksson, M. (2004, August). Sensitivity analysis of traffic equilibria. *Transportation Science*, *38*(3), 258–281. doi: 10.1287/trsc.1030.0043

Pavone, M., Smith, S. L., Frazzoli, E., & Rus, D. (2012). Robotic load balancing for Mobility-on-Demand systems. *International Journal of Robotics Research*, *31*(7), 839–854.

Pigou, A. C. (1920). *The economics of welfare* (1st ed.). London: Palgrave Macmillan UK.

Pitombeira-Neto, A. R., Loureiro, C. F. G., & Carvalho, L. E. (2020). A dynamic hierarchical bayesian model for the estimation of day-to-day origin-destination flows in transportation networks. *Networks and Spatial Economics*, *20*(2), 499–527.

Ramalingam, G., & Reps, T. (1996). An incremental algorithm for a generalization of the shortest-path problem. *Journal of Algorithms*, *21*(2), 267–305.

Richard P. Feynman. (1998). Cargo cult science. In *The art and science of analog circuit design* (pp. 55–61). Elsevier.

Rossi, F. (2018). *On the interaction between Autonomous Mobility-on-Demand systems and the built environment: Models and large scale coordination algorithms* (Doctoral dissertation, Stanford University, Dept. of Aeronautics and Astronautics). Retrieved from `https://stanfordasl.github.io/wp-content/papercite-data/pdf/Rossi.PhD18.pdf`

Rossi, F., Iglesias, R., Alizadeh, M., & Pavone, M. (2020). On the Interaction Between Autonomous Mobility-on-Demand Systems and the Power Network: Models and Coordination Algorithms. *IEEE Transactions on Control of Network Systems*, *7*(1), 384–397. doi: 10.1109/TCNS.2019.2923384

Rossi, F., Zhang, R., Hindy, Y., & Pavone, M. (2018). Routing autonomous vehicles in congested transportation networks: Structural properties and coordination algorithms. *Autonomous Robots*, *42*(7), 1427–1442.

Roughgarden, T., & Tardos, E. (2002). How Bad is Selfish Routing? *Journal of the ACM*, *49*(2), 236–259. doi: 10.1145/506147.506153

Russo, F., & Vitetta, A. (2011). Reverse assignment: calibrating link cost functions and updating demand from traffic counts and time measurements. *Inverse Problems in Science and Engineering*, *19*(7), 921–950.

Salazar, M., Lanzetti, N., Rossi, F., Schiffer, M., & Pavone, M. (2020). Intermodal Autonomous Mobility-on-Demand. *IEEE Transactions on Intelligent Transportation Systems*, *21*(9), 3946-3960. doi: 10.1109/TITS.2019.2950720

Salazar, M., Paccagnan, D., Agazzi, A., & Heemels, W. M. (2021). Urgency-aware optimal routing in repeated games through artificial currencies. *European Journal of Control*, *62*, 22-32. doi: 10.1016/j.ejcon.2021.06.024

Salazar, M., Rossi, F., Schiffer, M., Onder, C. H., & Pavone, M. (2018). On the interaction between autonomous mobility-on-demand and the public transportation systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (pp. 2262–2269). IEEE Press. doi: 10.1109/ITSC.2018.8569381

Salazar, M., Tsao, M., Aguiar, I., Schiffer, M., & Pavone, M. (2019). A Congestion-aware Routing Scheme for Autonomous Mobility-on-Demand Systems. In *2019 18th european control conference (ecc)* (pp. 3040–3046). doi: 10.23919/ECC.2019.8795897

Skabardonis, A., & Dowling, R. (1997). Improved speed-flow relationships for planning applications. *Transportation Research Record: Journal of the Transportation Research Board*, *1572*(1), 18–23.

Smith, M. J. (1979). The existence, uniqueness and stability of traffic equilibria. *Transportation Research Part B: Methodological*, *13*(4), 295–304. doi: 10.1016/0191-2615(79)90022-5

Solovey, K., Salazar, M., & Pavone, M. (2019). Scalable and congestion-aware routing for autonomous mobility-on-demand via frank-wolfe optimization. In *Robotics: Science and Systems.* Retrieved from `http://www.roboticsproceedings.org/rss15/p66.pdf`

Spieser, K., Samaranayake, S., Gruel, W., & Frazzoli, E. (2016). Shared-vehicle mobility-on-demand systems: A fleet operator's guide to rebalancing empty vehicles. In *Annual Meeting of the Transportation Research Board* (p. 16-5987).

Spiess, H. (1987). A maximum likelihood model for estimating origin-destination matrices. *Transportation Research Part B: Methodological*, *21*(5), 395–412.

Spiess, H. (1990). A gradient approach for the od matrix adjustment problem. *Publication CRT-693, Centre de recherche sur les transports, Universite de Montreal, Montreal, Quebec, Canada*, *1*, 2.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction.* MIT Press.

Swaszek, R. M., & Cassandras, C. G. (2019a). Load balancing in mobility-on-demand systems: Reallocation via parametric control using concurrent estimation. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)* (pp. 2148–2153). doi: 10.1109/ITSC.2019.8917333

Swaszek, R. M., & Cassandras, C. G. (2019b). Receding horizon control for station inventory management in a bike-sharing system. *IEEE Transactions on Automation Sciences and Engineering*, *17*(1), 407–417.

Theodoulou, G., & Wolshon, B. (2004). Alternative methods to increase the effectiveness of freeway contraflow evacuation. *Transportation Research Record: Journal of the Transportation Research Board*, *1865*(1), 48–56.

Tobin, R. L., & Friesz, T. L. (1988, 1 1). Sensitivity analysis for equilibrium network flow. *Transportation Science*, *22*(4), 242–250. doi: 10.1287/trsc.22.4.242

Traffic Assignment Manual. (1964). Bureau of public roads. *US Department of Commerce*.

Turan, B., Pedarsani, R., & Alizadeh, M. (2019). Dynamic pricing and management for electric autonomous mobility on demand systems using reinforcement learning. *arXiv preprint arXiv:1909.06962*.

Uber. (2019). *Uber movement.* (Data retrieved from Uber Movement, `https://movement.uber.com`)

U.S. Census Bureau. (2019). *2019 Census.* U.S. Department of Commerce.

Van Zuylen, H. J., & Willumsen, L. G. (1980). The most likely trip matrix estimated from traffic counts. *Transportation Research Part B: Methodological*, *14*(3), 281-293.

Wang, Y., Ma, X., Liu, Y., Gong, K., Henricakson, K. C., Xu, M., & Wang, Y. (2016). A two-stage algorithm for origin-destination matrices estimation considering dynamic dispersion parameter for route choice. *PLoS ONE*, *11*(1), e0146850.

Wardrop, J. G. (1952). Some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers*, *1*(3), 325–362.

Wollenstein-Betech, S., Paschalidis, I. C., & Cassandras, C. G. (2021). Planning Strategies for Lane Reversals in Transportation Networks. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)* (pp. 2131–2136). doi: 10.1109/ITSC48978.2021.9564808

Wollenstein-Betech, S., Salazar, M., Houshmand, A., Pavone, M., Paschalidis, I. C., & Cassandras, C. G. (2021). Routing and Rebalancing Intermodal Autonomous Mobility-on-Demand Systems in Mixed Traffic. *IEEE Transactions on Intelligent Transportation Systems*. doi: 10.1109/TITS.2021.3112106

Wollenstein-Betech, S., Sun, C., Zhang, J., Cassandras, C. G., & Paschalidis, I. C. (2022). Joint Data-Driven Estimation of Origin-Destination Demands and Travel Latency Functions in Multi-Class Transportation Networks. *IEEE Transactions on Control of Network Systems*. doi: 10.1109/TCNS.2022.3161200

Wollenstein-Betech, S., Sun, C., Zhang, J., & Paschalidis, I. C. (2019). Joint Estimation of OD Demands and Cost Functions in Transportation Networks from Data. In *2019 IEEE 58th Conference on Decision and Control (CDC)* (pp. 5113–5118). doi: 10.1109/CDC40024.2019.9029445

Wächter, A., & Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, *106*(1), 25–57. doi: 10.1007/s10107-004-0559-y

Xie, C., & Turnquist, M. A. (2011). Lane-based evacuation network optimization: An integrated lagrangian relaxation and tabu search approach. *Transportation Research Part C: Emerging Technologies*, *19*(1), 40–63.

Xue, D., & Dong, Z. (2000). An intelligent contraflow control method for real-time optimal traffic scheduling using artificial neural network, fuzzy pattern recognition, and optimization. *IEEE Transactions on Control Systems Technology*, *8*(1), 183–191.

Yang, H., Meng, Q., & Bell, M. G. H. (2001). Simultaneous estimation of the origin-destination matrices and travel-cost coefficient for congested networks in a stochastic user equilibrium. *Transportation Science*, *35*(2), 107-123. doi: 10.1287/trsc.35.2.107.10133

Yang, H., Sasaki, T., Iida, Y., & Asakura, Y. (1992). Estimation of origin-destination matrices from link traffic counts on congested networks. *Transportation Research Part B: Methodological*, *26*(6), 417–434.

Yang, H., Zhang, X., & Meng, Q. (2007). Stackelberg games and multiple equilibrium behaviors on networks. *Transportation Research Part B: Methodological*, *41*(8), 841–861.

Youn, H., Gastner, M. T., & Jeong, H. (2008). Price of anarchy in transportation networks: efficiency and optimality control. *Physical Review Letters*, *101*(12), 128701.

Zhang, J., Pourazarm, S., Cassandras, C. G., & Paschalidis, I. C. (2016). The price of anarchy in transportation networks by estimating user cost functions from actual traffic data. *2016 IEEE 55th Conference on Decision and Control (CDC)*, 789–794. doi: 10.1109/CDC.2016.7798364

Zhang, J., Pourazarm, S., Cassandras, C. G., & Paschalidis, I. C. (2017). Data-driven estimation of origin-destination demand and user cost functions for the optimization of transportation networks. *IFAC World Congress*, *50*(1), 9680 - 9685.

Zhang, J., Pourazarm, S., Cassandras, C. G., & Paschalidis, I. C. (2018). The price of anarchy in transportation networks: Data-driven evaluation and reduction strategies. *Proceedings of the IEEE*, *106*(4), 538–553.

Zhang, R., & Pavone, M. (2016). Control of robotic Mobility-on-Demand systems: A queueing-theoretical perspective. *International Journal of Robotics Research*, *35*(1–3), 186–203.

Zhao, X., Feng, Z.-y., Li, Y., & Bernard, A. (2016). Evacuation network optimization model with lane-based reversal and routing. *Mathematical Problems in Engineering*, *2016*(1273508). doi: 10.1155/2016/1273508

Zhou, W., Livolsi, P., Miska, E., Zhang, H., Wu, J., & Yang, D. (1993). An intelligent traffic responsive contraflow lane control system. In *Proceedings of VNIS'93-Vehicle Navigation and Information Systems Conference* (pp. 174–181).

CURRICULUM VITAE

[redacted]

[redacted]