THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Stochastic and Learning-Based Control Strategies for Electric Autonomous Mobility Systems

Sten Elling Tingstad Jacobsen

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2026

Stochastic and Learning-Based Control Strategies for Electric Autonomous Mobility Systems

Sten Elling Tingstad Jacobsen
ISBN 978-91-8103-384-7

Acknowledgements, dedications, and similar personal statements in this thesis, reflect the author's own views.

Selected material from the author's licentiate thesis: Sten Elling Tingstad Jacobsen, "Uncertain demand prediction for guaranteed automated vehicle fleet performance", *Chalmers*, Gothenburg, Sweden, Feb. 2023, is republished in this Ph.D. thesis.

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 5841
ISSN 0346-718X

Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
Phone: +46 (0)31 772 1000

Printed by Chalmers Digital Printing
Gothenburg, Sweden, March 2026

Stochastic and Learning-Based Control Strategies for Electric Autonomous Mobility Systems

Sten Elling Tingstad Jacobsen
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Electric Autonomous Mobility-on-Demand (E-AMoD) systems offer a path toward sustainable urban transportation through the coordinated operation of shared, zero-emission autonomous vehicles. Yet their deployment poses difficult operational challenges: fleet rebalancing, vehicle routing, and charging must be managed jointly, under uncertainty, and within tight computational budgets. A central argument of this thesis is that no single decision-making paradigm suffices. Optimization-based methods are well suited for strategic fleet control where uncertainty guarantees and feedback are essential, while learning-based methods become necessary at finer operational scales where real-time optimization is computationally prohibitive.

Three contributions are presented, each targeting a different operational level. The first introduces a chance-constrained model predictive control (MPC) framework for station-level fleet rebalancing, combining Gaussian Process Regression for probabilistic demand forecasting with a hierarchical architecture that separates strategic rebalancing from tactical matching. The second extends this framework to electric fleets operating under multiple interacting uncertainties, employing a tailored Nested Benders Decomposition to maintain metropolitan-scale tractability without sacrificing MPC's receding-horizon feedback. The third contribution shifts to node-level electric dial-a-ride routing, including pickup-delivery sequencing, time windows, and ride-time constraints, and proposes a deep reinforcement learning approach built on a Graph Edge Attention Network capable of handling hundreds of requests with seconds inference times. Taken together, the three contributions show that optimization and learning serve complementary roles in E-AMoD operations, with the appropriate paradigm determined by the granularity and real-time demands of the problem at hand.

Keywords: Electric Autonomous Vehicles, Mobility-on-Demand, Optimization Under Uncertainty, Stochastic Programming, Robust Optimization, Benders Decomposition, Deep Reinforcement Learning, Graph Neural Networks.

List of Publications

This thesis is based on the following publications:

[A] **Sten Elling Tingstad Jacobsen**, Anders Lindman, Balázs Kulcsár, "A Predictive Chance-Constrained Rebalancing Approach to Mobility-on-Demand Services". Published in Communications in Transportation Research, vol. 4, 100109, 2023. (2023).

[B] **Sten Elling Tingstad Jacobsen**, Balázs Kulcsár, Anders Lindman, "Combined Stochastic and Robust Optimization for Electric Autonomous Mobility-on-Demand with Nested Benders Decomposition". Revised manuscript under review in Special Issue on Innovations for Operation and Pricing of Public Mobility Services, Elsevier Transportation Research Part C: Emerging Technologies (2025).

[C] **Sten Elling Tingstad Jacobsen**, Attila Lischka, Balázs Kulcsár, Anders Lindman, "Learning to Dial-a-Ride: A Deep Graph Reinforcement Learning Approach to the Electric Dial-a-Ride Problem". Manuscript under review in Transportation Research Part C: Emerging Technologies (2026).

Contents

Acknowledgments

The work presented in this thesis marks the culmination of several years of learning, challenges, and growth. Throughout this journey, I have been fortunate to receive guidance, support, and encouragement from many people. I am sincerely grateful to all who have contributed, in different ways, to making this thesis possible.

First and foremost, I express my deepest gratitude to my main supervisor, Professor Balázs Kulcsár. His guidance, insight, and unwavering support throughout my doctoral studies have been invaluable. I am particularly grateful for his ability to combine high scientific standards with trust and encouragement, which allowed me to grow both as a researcher and as an independent thinker. His critical questions, patience, and belief in my work have profoundly shaped this thesis and my development as a researcher.

I am also deeply grateful to my industrial supervisor at Volvo Cars, Anders Lindman. His practical insights, industry perspective, and continuous engagement have greatly enriched this research. I particularly appreciate his ability to connect academic ideas with real-world challenges, as well as the trust and support he has shown throughout this collaboration.

My sincere thanks also go to my manager at Volvo Cars, Anders Björtin, for his strong support and belief in this project. His encouragement to share my research within the organization has meant a great deal to me. I am grateful for the confidence he placed in me and for fostering an environment where academic research and industrial practice could meaningfully intersect.

I would also like to express my appreciation to Professor Torsten Wik for the opportunity to serve as a teaching assistant in his courses at Chalmers. Being part of his teaching team has been both rewarding and inspiring. I am grateful for the chance to collaborate with such dedicated and skilled educators.

My sincere thanks also go to my fellow lab members and colleagues, many of whom I now have the privilege of calling friends. Thank you for the engaging discussions, the shared challenges, the afterworks, lunch breaks, and all the activities that made these years enjoyable and memorable. Special thanks to Maxi, Remi, Ivo, Ankit, Masoud, Albin, Angel, Ramin, Constantin, Mattias, Endre, Sabino, Rita, Ahad, Stefan, Anand, Huang, Yao, Yizhou, Ze, Ektor, Rikard, Carl-Johan, Sondre, Faris, Albert, Godwin, Gabriel, Ahmet, Alvin, Nishant, Johannes, Dan, Erik, Attila, Benedick, Lorenzo, Hannes, Erik, Filip, David, Kilian, and Francesco, as well as anyone I may have unintentionally

forgotten, for the support, the laughs, and the countless conversations, both academic and otherwise, that kept me going. I would also like to thank my colleagues and fellow VIPP students at Volvo Cars for the stimulating discussions, collaboration, and shared experiences throughout this journey.

A special thank you to my friends in Göteborg, Isak, Hjalmar, Maxi, Linus, Nils, David, Thomas, Arsam, and Mattias, for the many afterworks, concerts, trips, running, biking, skiing, and dinners.

Finally, I want to thank my parents Gry and Elling, and my sister Ingrid, whose love, support, and encouragement have meant more than I can express.

Acronyms

AM:	Attention Model
AMoD:	Autonomous Mobility-on-Demand
BNN:	Bayesian Neural Network
CNN:	Convolutional Neural Network
CVRP:	Capacitated Vehicle Routing Problem
DARP:	Dial-a-Ride Problem
DRL:	Deep Reinforcement Learning
E-AMoD:	Electric Autonomous Mobility-on-Demand
E-DARP:	Electric Dial-a-Ride Problem
EV:	Electric Vehicle
E-VRP:	Electric Vehicle Routing Problem
GNN:	Graph Neural Network
GPR:	Gaussian Process Regression
GREAT:	Graph Edge Attention Network

LP:	Linear Programming
LSTM:	Long Short-Term Memory
MC:	Monte Carlo
MDP:	Markov Decision Process
MILP:	Mixed-Integer Linear Programming
MoD:	Mobility-on-Demand
MPC:	Model Predictive Control
NBD:	Nested Benders Decomposition
OD:	Origin-Destination
POMO:	Policy Optimization with Multiple Optima
RL:	Reinforcement Learning
SoC:	State of Charge
TSP:	Travelling Salesman Problem
VRP:	Vehicle Routing Problem

Part I

Overview

CHAPTER 1

Background and Outline

1.1 Introduction

Transportation enables economic activity, social connection, and access to essential services in modern society. Over the past century, advances in transportation have dramatically improved connectivity between communities and facilitated urban growth. Reliable transportation access is now essential for individuals to participate fully in economic and social life. Yet the transportation sector faces challenges that threaten both sustainability and quality of life. Road transport accounts for nearly one quarter of global energy-related CO_2 emissions [1], [2]. Urban congestion costs billions in lost productivity and degrades air quality [3], [4]. As cities continue to grow, with urban populations projected to reach 68% globally by 2050 [5], these challenges intensify and demand solutions that reconcile environmental sustainability, operational efficiency, and equitable access to mobility.

Current transportation paradigms exhibit important limitations at the system level. Private car ownership, while providing flexibility, leads to low average vehicle utilization in urban contexts and requires extensive parking infrastructure. Conventional ride-hailing services reduce ownership burdens

but have also introduced new inefficiencies. Studies of ride-hailing operations in San Francisco reveal a 62% increase in Vehicle Hours of Delay (VHD), a measure of aggregate time lost to congestion, between 2010 and 2016, compared to a 22% increase in the absence of ride-hailing [6]. This evidence suggests that, without effective coordination, ride-hailing services can exacerbate system-level inefficiencies. A key contributing factor is widely understood to be spatial imbalances between vehicle supply and passenger demand, which lead to long pickup times, excessive empty travel, and increased congestion.

Electric Autonomous Mobility-on-Demand (E-AMoD) systems represent a promising response to these converging challenges. E-AMoD integrates three transformative technologies: electric propulsion, autonomous operation, and shared mobility. This integration enables transportation systems that are more sustainable, more efficient, and more accessible than current alternatives [7], [8]. Electric vehicles eliminate tailpipe emissions and, when powered by renewable energy, substantially reduce lifecycle carbon emissions [9]. Autonomous operation removes labor costs, the dominant expense in conventional taxi services, while enabling continuous availability and reducing variability in driving behavior [10]. Centralized fleet coordination enables proactive vehicle positioning based on anticipated demand, reducing empty travel and passenger wait times [11]. Shared mobility spreads infrastructure costs across many users, potentially providing high-quality service at lower cost than private ownership while reducing the total number of vehicles required to serve a given population [12].

The vision of E-AMoD systems is both compelling and increasingly achievable. Commercial autonomous ride-hailing services are already operating in several U.S. cities, with millions of autonomous miles logged under real-world conditions [13]. Autonomous vehicle technology is expected to mature sufficiently for broader deployment within the next decade, while many cities have committed to carbon neutrality goals that require large-scale transportation electrification. Shared mobility platforms have already demonstrated consumer acceptance, with the global shared mobility market accounting for approximately $150 billion in consumer spending and more than 40 million daily users in 2019 [14]. The integration of electrification, automation, and on-demand service creates synergies that could deliver transportation superior to existing alternatives. Studies project that E-AMoD could reduce operational costs by 70 to 84% compared to conventional taxi services [15], making shared

autonomous mobility a viable complement to public transportation in urban, suburban, and potentially rural contexts.

A central operational challenge is the spatial imbalance between vehicle supply and passenger demand. Passenger requests exhibit pronounced spatiotemporal variation: morning commutes concentrate demand from residential areas to employment centers, evening patterns reverse, and weekends and special events generate distinct demand profiles. Without intervention, vehicles naturally accumulate in destination areas while origin areas experience shortages, degrading service quality. Rebalancing addresses this imbalance by proactively repositioning empty vehicles toward areas of anticipated demand, but it incurs operational costs from energy consumption and non-revenue-generating vehicle time [11], [16]. The trade-off is clear: conservative rebalancing leads to vehicle shortages and poor service quality, while aggressive rebalancing increases operational costs. Determining effective rebalancing strategies therefore requires predicting future demand, which is inherently uncertain due to weather, events, and individual travel behavior.

For electric autonomous fleets, operational complexity extends beyond rebalancing to the integrated management of three tightly coupled decision problems [17], [18]. Fleet rebalancing determines where vehicles should be positioned over space and time. Vehicle–customer assignment determines which vehicle should serve which request while respecting service quality constraints and anticipating future demand. Charging management determines when and where vehicles should recharge to maintain fleet availability despite limited battery capacity and charging infrastructure [19], [20]. These decisions are interdependent. Rebalancing affects future vehicle availability, serving passengers depletes battery charge, and charging temporarily removes vehicles from service. Unlike conventional fleets, electric vehicles introduce hard operational constraints, including limited driving range, charging durations, and energy consumption that varies with load, speed, and road topology [21], [22].

Uncertainty further compounds the complexity of E-AMoD operations. Passenger demand fluctuates unpredictably [23], [24], travel times vary with traffic conditions [25], charging station availability is uncertain due to congestion and outages, and energy consumption depends on driving behavior, traffic, and environmental factors. Traditional deterministic optimization approaches that assume perfect knowledge of future conditions are therefore inadequate. They often lead to infeasible operations or degraded service quality when realized

conditions deviate from predictions [26].

These computational and operational challenges make evaluating E-AMoD systems through real-world experimentation at scale both impractical and risky. Cities cannot systematically test alternative fleet sizes, rebalancing policies, or charging strategies through trial and error on operational systems serving real passengers, where poor decisions directly affect users through long wait times, service disruptions, or stranded vehicles. This necessitates high-fidelity transportation simulators that can represent urban mobility systems with sufficient realism while remaining independent of the decision-making algorithms under evaluation. In this thesis, algorithmic decisions are evaluated using the AMoDeus simulation framework [27], built on the agent-based transportation simulator MATSim [28], which enables modeling of real urban networks with detailed road infrastructure and traffic dynamics. The simulation incorporates real-world transportation network data and historical demand patterns from San Francisco and Chicago, while modeling passenger behavior, vehicle dynamics, and infrastructure constraints to reflect realistic operating conditions. By decoupling the control, optimization, and learning algorithms from this more detailed and independent simulation environment, this approach enables rigorous assessment of system-level performance under heterogeneous travel demand, operational constraints such as time windows and battery limitations, and the effectiveness of fleet coordination strategies.

The computational requirements of E-AMoD operations are substantial. Passengers expect rapid responses to ride requests, while fleet-level decisions must adapt continuously to evolving conditions [29]. Yet the underlying optimization problems are computationally challenging. Even routing a single vehicle to serve multiple passengers, known as the dial-a-ride problem, is NP-hard [30]. For fleet-wide operations involving many vehicles, complex spatiotemporal constraints, battery limitations, and uncertainty, the resulting optimization problems become large-scale mixed-integer programs that are often intractable for real-time solution [31]. Importantly, different operational problems require fundamentally different computational structures. Station-level rebalancing permits aggregation that preserves tractability, while node-level routing involves combinatorial complexity that can render real-time optimization impractical regardless of algorithmic sophistication.

This thesis is motivated by the need to develop decision-making frameworks that address these challenges by explicitly accounting for uncertainty, compu-

tational constraints, and the heterogeneous structure of E-AMoD operational problems. In particular, it investigates how optimization-based and learning-based methods can be employed in a complementary manner across different decision layers and timescales to enable reliable, efficient, and scalable operation of electric autonomous mobility-on-demand systems.

1.2 Research Gaps

Despite substantial recent research on autonomous mobility-on-demand systems, several critical gaps remain in our ability to design and operate electric autonomous fleets effectively under real-world conditions. This section identifies four fundamental research gaps and formulates corresponding research questions that motivate the contributions of this thesis.

Systematic Modeling of Uncertainty

E-AMoD systems face multiple sources of uncertainty with different characteristics. Travel demand and charger availability exhibit strong spatiotemporal correlation and can be estimated from historical data using parametric models [32], [33], neural networks [34], [35], or Gaussian Process Regression [36]. In contrast, travel time and energy consumption variability depend on real-time conditions and may lack reliable probabilistic characterizations.

Traditional fleet operations rely on deterministic optimization with historical averages or point forecasts [11], [16], systematically underestimating the resources required to maintain service quality [37]. Uncertainty-aware approaches address this limitation but face trade-offs: scenario-based stochastic programming grows exponentially with scenarios and planning horizon [26], [38], and robust optimization can be overly conservative [39], [40]. Chance-constrained formulations offer a principled middle ground for uncertainties with known distributions [41], [42].

Existing research typically addresses different uncertainty sources in isolation. The gap lies in developing hybrid frameworks that combine stochastic or chance-constrained methods for demand and charger availability with robust methods for travel time and energy consumption, while remaining computationally tractable.

Research Question 1: *How can uncertainties be systematically modeled*

and exploited to improve decision-making in mobility-on-demand systems?

Computational Tractability at Scale

E-AMoD optimization problems are computationally challenging due to their combinatorial nature and the need to handle uncertainty. Even deterministic formulations of vehicle routing, charging scheduling, and fleet rebalancing result in large mixed-integer programs that are intractable at metropolitan scale [30], [43]. Incorporating uncertainty through scenarios or robust constraints compounds this difficulty [44].

Different operational decisions have fundamentally different computational requirements. Strategic decisions about fleet positioning may allow computation times of seconds to minutes, enabling optimization-based approaches with appropriate decomposition techniques [45], [46]. Operational decisions about individual vehicle routing must be made in milliseconds. The electric dial-a-ride problem exemplifies these challenges: it requires constructing pickup and delivery sequences respecting precedence, time windows, and battery constraints, and exact methods handle only tens of requests [47], [48].

The gap is twofold. First, for station-level problems, decomposition algorithms must exploit problem structure to enable tractability at scale while preserving solution quality under uncertainty. Second, for node-level routing where real-time optimization is infeasible, alternative paradigms must generate high-quality solutions within millisecond time budgets.

Research Question 2: *How can large-scale combinatorial optimization problems arising in E-AMoD operations be solved efficiently under real-time and uncertainty constraints?*

Fleet Specification and Operational Interactions

The operational performance of E-AMoD systems depends critically on fleet specifications, including battery capacity, charging infrastructure, and fleet size, yet interactions between these parameters and operational decision-making under uncertainty are poorly understood. Most research treats fleet specifications as fixed inputs or considers them in isolation from operational constraints [34], [49].

The parameters mentioned above do not act independently. Battery capacity affects charging frequency and operational flexibility; larger batteries

reduce interruptions but increase cost [50]. Fleet size interacts with service quality and the ability to accommodate demand peaks [12], [16]. Charging infrastructure creates spatial constraints that interact with both battery capacity and fleet operations [51]. A fleet with larger batteries may require fewer vehicles, but this depends on charging infrastructure density; more vehicles may compensate for smaller batteries if charging stations are strategically located.

The gap is understanding how fleet specifications interact with operational decision-making under uncertainty, enabling design choices that balance capital costs, operational performance, and energy efficiency.

Research Question 3: *How do fleet specifications, such as vehicle battery capacity, charging infrastructure, and fleet size, interact with operational decision-making under uncertainty?*

Optimization versus Learning Paradigms

A broader question concerns how optimization under uncertainty and learning-based approaches should be deployed within E-AMoD operations. Optimization-based methods offer theoretical guarantees, explicit constraint handling, and interpretable solutions but face computational scalability limits [26], [40]. Learning-based methods offer computational efficiency at deployment and can handle high-dimensional state spaces, but lack formal guarantees [52], [53].

We hypothesize that these paradigms address different operational problems rather than providing alternatives to the same problem. Strategic decisions about fleet positioning and charging benefit from the ability of optimization to provide probabilistic guarantees and adapt through model predictive control. Node-level vehicle routing may exceed real-time optimization capabilities, necessitating learned policies that generate solutions in milliseconds.

The gap is understanding when each paradigm is appropriate based on problem granularity (station-level vs. node-level), temporal scale (strategic vs. operational), and computational constraints, and how to combine both paradigms effectively.

Research Question 4: *Under what conditions should E-AMoD systems employ optimization-based versus learning-based methods, and how can these paradigms be combined effectively?*

1.3 Contributions

This thesis presents decision-making frameworks for the control and operation of Electric Autonomous Mobility-on-Demand systems, addressing problems that span from strategic fleet positioning to real-time vehicle routing. Each problem is formulated within an optimization or learning framework that explicitly accounts for uncertainty and computational constraints. Depending on the nature of each problem, the decision scale may be at the station level or at the node level. Station-level formulations divide the transportation network into zones and optimize flows of vehicles between these zones, rather than tracking individual customers and vehicles. This aggregation reduces problem size and enables tractable optimization. Node-level formulations, by contrast, construct detailed pickup and delivery sequences for individual vehicles while respecting constraints such as precedence requirements, time windows, and capacity limits. What all these problems have in common is that uncertainty is present throughout and must be handled carefully to ensure reliable operations. For the station-level problems studied, it is possible to formulate optimization problems that provide probabilistic service guarantees through chance constraints or robust constraints. Decomposition techniques can then be applied to solve problems at metropolitan scale. For the node-level routing problem, the combinatorial complexity makes real-time optimization infeasible, and learning-based approaches are needed to generate high-quality solutions in milliseconds. This thesis shows how optimization under uncertainty and deep reinforcement learning serve complementary roles in E-AMoD operations, with the appropriate method determined by problem structure.

Since the scope of the problems, algorithms, and methodologies treated in the thesis is rather wide, Table 1.1 presents an overview of the main features of each of the appended papers.

The main contributions of this thesis are:

- A chance-constrained Model Predictive Control (MPC) framework combining Gaussian Process Regression for demand prediction with Model Predictive Control is developed for AMoD systems, demonstrating how probabilistic demand models can be integrated into optimization to provide service level guarantees (**RQ1**).
- A hybrid stochastic-robust optimization framework is developed for E-AMoD operations that systematically treats different uncertainty sources

Table 1.1: Overview of the three contributions and their key distinguishing features.

Feature	**Paper I**	**Paper II**	**Paper III**
System	Mobility-on-Demand	Electric Mobility-on-Demand	Electric dial-a-ride routing
Decision scale	Station-level flows	Station-level flows	Node-level sequences
Method	Chance-constrained MPC	Stochastic-robust MPC	Deep Reinforcement Learning
Uncertainty	Demand	Demand, charger availability, energy consumption, travel time	Energy consumption, travel time
Combinatorial optimization problem	Mixed-Integer Linear Program (Totally Unimodular)	Mixed-Integer Linear Program	Mixed-Integer Linear Program
Feedback	Explicit (Model Predictive Control)	Explicit (Model Predictive Control)	No feedback
Solution method	Gurobi LP solver	Nested Benders Decomposition	RL inference
Computation	0.09 s (10 stations)	355 s (10 stations, 1035 scenarios)	< 1 s (500 nodes)
Research questions	RQ1, RQ3 (fleet size)	RQ1, RQ2, RQ3	RQ2, RQ3, RQ4

according to their characteristics: demand and charger availability through scenario-based stochastic optimization, and travel time and energy consumption through chance constraints derived from robust bounds (**RQ1**).

- A Nested Benders Decomposition algorithm is developed that exploits the temporal structure of E-AMoD operations to decompose large-scale stochastic-robust programs, enabling metropolitan-scale tractability within practical time limits (**RQ2**).

- An edge-based deep reinforcement learning approach using the Graph Edge Attention Network architecture is developed for the electric dial-a-ride problem, enabling real-time solutions for large-scale instances where optimization-based methods are computationally infeasible (**RQ2**).

- Extensive sensitivity analyses are conducted across fleet sizes, battery capacities, demand levels, and ride-sharing configurations to characterize how these service specifications interact with operational performance under uncertainty (**RQ3**).

- The complementary roles of optimization and learning in E-AMoD operations are established through systematic comparison, demonstrating that station-level decisions benefit from optimization-based MPC while node-level routing requires learning-based policies, with problem structure determining the appropriate paradigm (**RQ4**).

1.4 Thesis Outline and Scope

This is a compilation thesis of the appended papers, discussing aspects ranging from stochastic and robust optimization to deep reinforcement learning. We present several optimization and learning-based methods that tackle different operational challenges in electric autonomous mobility-on-demand systems. The thesis is organized as follows.

Chapter 2: Electric Autonomous Mobility-on-Demand Operations provides the technical foundation for E-AMoD systems. It introduces network flow formulations for AMoD operations, presents the mathematical model of E-AMoD that serves as the basis for subsequent chapters, discusses the sources and characteristics of uncertainty in E-AMoD operations, and describes the

electric dial-a-ride problem in detail. This chapter establishes notation, model assumptions, and problem formulations referenced throughout the thesis.

Chapter 3: Robust and Stochastic Optimization reviews the optimization under uncertainty techniques employed in this thesis. It covers the fundamentals of robust optimization, including uncertainty sets and the robust counterpart formulation, and the principles of stochastic programming, including scenario-based formulations and chance constraints. It also discusses decomposition methods, particularly Benders decomposition and its variants, which are essential for computational tractability. This chapter provides the theoretical background necessary to understand the optimization contributions in Papers I and II.

Chapter 4: Machine Learning for Spatio-Temporal Prediction develops probabilistic forecasting methods that provide the input distributions required for uncertainty-aware decision-making in E-AMoD systems. It motivates the need to capture complex urban demand patterns across space and time and emphasizes that reliable operations require distributional forecasts rather than point estimates. The chapter presents three complementary approaches: Gaussian Process Regression for Bayesian spatio-temporal modeling with principled uncertainty quantification and strong performance in data-sparse regimes; Bayesian Neural Networks for scalable forecasting of complex non-linear patterns with uncertainty estimates in data-rich operational settings; and scenario trees for discretizing uncertainty evolution over time into tractable representations for multi-stage stochastic optimization. The chapter concludes by connecting these predictive models to the chance-constrained and stochastic optimization frameworks used in later chapters.

Chapter 5: Deep Reinforcement Learning introduces the machine learning concepts underlying the third contribution. It covers the basics of reinforcement learning, including Markov Decision Processes, and graph neural networks, with emphasis on attention mechanisms and the Graph Edge Attention Network architecture. It discusses applications of these techniques to combinatorial optimization and vehicle routing problems, positioning the Electric Dial-a-ride (E-DARP) contribution within the broader literature.

Chapter 6: Summary of Appended Papers This chapter provides a concise overview of the papers appended to this thesis and summarizes their individual contributions.

Chapter 7: Conclusions and Future Directions synthesizes the con-

tributions, discusses their implications for E-AMoD operations, reflects on the relative merits of optimization and learning-based approaches, identifies limitations of the current work, and outlines promising directions for future research.

Scope and Delimitations

This thesis focuses on algorithmic decision-making for Electric Autonomous Mobility-on-Demand systems, with contributions spanning fleet rebalancing, charging and V2G scheduling, and vehicle routing under uncertainty. Papers I and II are evaluated on urban networks derived from San Francisco and Chicago using the AMoDeus simulation framework, while Paper III uses a custom simulation environment on the same network data.

The optimization frameworks in Papers I and II operate at the station level, aggregating individual vehicles into network flows over discretized zones. This enables tractable large-scale optimization but abstracts away individual vehicle identities and fine-grained road effects. Paper III addresses individual vehicle scheduling at the node level, though its integration with the full fleet coordination and charging framework remains an open problem. Prediction models, namely Gaussian Process Regression and Bayesian Neural Networks, are employed as inputs rather than primary contributions, and the reinforcement learning framework in Paper III is evaluated in simulation without addressing sim-to-real transfer or integration with live traffic systems. Travel times between nodes are treated as fixed or stochastically modeled inputs rather than quantities that respond endogenously to fleet behavior; the effect of the fleet on traffic congestion is not considered. Questions of pricing, market competition, and infrastructure investment planning are also outside the scope.

Ethical and Sustainability Aspects

The research in this thesis is directly motivated by one of the most pressing sustainability challenges of our time: decarbonizing urban transportation, a sector responsible for nearly one quarter of global energy-related CO_2 emissions [2]. By combining electric propulsion, shared mobility, and intelligent fleet management, E-AMoD systems have the potential to deliver high-quality transportation at a fraction of the energy and emissions cost of private vehicle

ownership. The optimization and learning methods developed here amplify these benefits, as reducing empty vehicle travel, improving charging coordination, and anticipating demand all translate into lower energy consumption per trip and less strain on charging infrastructure. As renewable energy penetration grows, the V2G capabilities modeled in Paper II position autonomous fleets as active participants in grid balancing, turning a fleet of vehicles into a distributed energy resource that supports the broader clean energy transition.

Beyond emissions, E-AMoD holds the promise of broadening access to mobility. Autonomous operation removes the labor cost that makes conventional on-demand services expensive, opening the possibility of high-quality shared transportation in areas currently underserved by both private car ownership and public transit. The sensitivity analyses conducted in this thesis across fleet sizes, battery capacities, and demand levels provide a foundation for understanding how system design choices affect service quality, which is a prerequisite for deploying systems that serve diverse communities equitably. Future extensions of this work that incorporate spatial equity constraints directly into the rebalancing optimization represent a natural and important direction, ensuring that efficiency gains benefit all users rather than concentrating service in the most profitable corridors.

Looking forward, the computational frameworks developed here are well positioned to support the transition from simulation to real-world deployment. The methods are designed with operational constraints in mind: solution times of under one second for node-level routing and under ten minutes for full stochastic fleet optimization make them compatible with the decision frequencies required in live systems. As autonomous vehicle technology matures and cities develop regulatory frameworks for shared autonomous mobility, tools like those developed in this thesis will be essential for demonstrating that safe, efficient, and equitable operations are achievable at metropolitan scale.

CHAPTER 2

Electric Autonomous Mobility-on-Demand Operations

An Electric Autonomous Mobility-on-Demand (E-AMoD) system consists of a centrally coordinated fleet of electric autonomous vehicles serving on-demand passenger requests on a road network. Unlike conventional ride-hailing, where drivers exercise independent judgment, all routing, rebalancing, and charging decisions are made explicitly by a central controller. This transforms what was once implicit human behavior into a large-scale optimization problem.

E-AMoD services can take different forms depending on the degree of sharing permitted. In ride-hailing, each vehicle serves one travel group at a time, prioritising short wait times at the cost of lower vehicle utilisation. In ride-pooling, multiple independent passengers share the same vehicle simultaneously, reducing total distance travelled but increasing individual ride times due to detours. This thesis addresses both modes: Papers I and II consider ride-hailing at the fleet level, while Paper III addresses ride-pooling at the level of individual vehicle routes.

This chapter introduces the operational and mathematical framework that underlies all three papers. It covers the graph representations used to model the network, the vehicle and energy models, the fleet-level optimization formulation, and the three distinct optimization problems addressed in the thesis.

The demand forecasting methods and the treatment of uncertainty are introduced here at a high level and developed in detail in Chapters 4 and 3 respectively. The simulation environment used to evaluate all three papers is described at the end of the chapter.

2.1 Graph Network Models

The movement of passengers and vehicles in all three papers is modelled as a directed graph, but at different levels of detail. A directed graph $G = (V, E)$ consists of a set of nodes V and a set of directed edges $E \subseteq V \times V$, where each edge (i, j) represents a feasible movement from node i to node j. The choice of what a node represents, what an edge represents, and which aspects of vehicle state are tracked has important consequences for both the expressiveness and the computational tractability of the resulting model.

The most detailed representation of an urban road network is one in which each node $v \in V$ corresponds to a road junction and each directed edge $(i, j) \in E$ corresponds to a road segment from junction i to junction j. Each edge carries attributes such as travel time, distance, and energy consumption. For real urban networks, the number of nodes is typically on the order of tens of thousands, with a comparable number of edges. Because the cost of traversing edge (i, j) is in general different from the cost of the reverse edge (j, i), the graph is *asymmetric*. One-way streets, turn restrictions, and road gradient all contribute to this asymmetry. For electric vehicles it is particularly pronounced: the energy consumed driving from junction i to junction j differs from the energy on the return trip because regenerative braking and gravitational forces depend on direction. Paper III operates on this asymmetric road network graph. Each passenger request specifies a precise pickup and delivery location, and each vehicle route is a sequence of junctions. Because the vehicles are electric, the state of a vehicle at any point in its route is described by three quantities: its location, the time at which it is there, and its remaining battery charge. The road network graph thus provides the spatial backbone of a *space-time-energy* representation, where a feasible route must simultaneously satisfy geographic, scheduling, and battery constraints.

Despite its detail, the road network graph is computationally intractable as the basis for fleet-level Model Predictive Control. Consider a fleet of K vehicles facing R outstanding passenger requests at a given decision point.

Even in the simplified case where each vehicle serves at most one request, the number of possible assignments is $\mathcal{O}(R^K)$, and finding the best assignment requires evaluating routes across the full network. Zhang et al. [54] showed that MPC formulated directly on the road network becomes intractable as the number of vehicles and customers grows, because the problem size scales with both simultaneously. Tracking the state of every individual vehicle and customer at every junction at every time step is simply not feasible for fleets of the size needed in urban operations.

Papers I and II therefore use a coarser representation: a discretised *station graph*. The city is divided into N regions, called stations, obtained by applying k-means clustering to historical trip origins and destinations so that each station forms a coherent travel zone. The resulting station graph is a *complete* directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with $\mathcal{N} = \{1, \ldots, N\}$ and $\mathcal{E} = \mathcal{N} \times \mathcal{N}$. Completeness means that every ordered pair of stations (i, j) is connected by a direct edge, so a vehicle can travel directly from any station to any other without an intermediate stop. This is in contrast to the road network graph, where a journey between two distant junctions follows a path through many intermediate nodes. In the station graph, the complexity of the underlying road topology is absorbed into the edge attributes: the average travel time κ_{ij} and energy consumption e_{ij} assigned to each edge are computed by routing over the full road network offline.

The key modelling shift is from tracking individual vehicles and passengers to optimising aggregate *flows*: the controller decides how many vehicles should move from station i to station j during time period t, rather than routing each vehicle individually. The dimensions of the graph differ between the two papers. In Paper I, which does not model electric vehicles, the vehicle state is fully described by its station and the current time step, giving a *space-time* network where each edge carries only the average travel time κ_{ij}, discretised into intervals of length Δt. Paper II introduces electric vehicles, so the graph becomes a *space-time-energy* network: each edge additionally carries an average energy consumption e_{ij}, and the vehicle state must also track the battery state of charge.

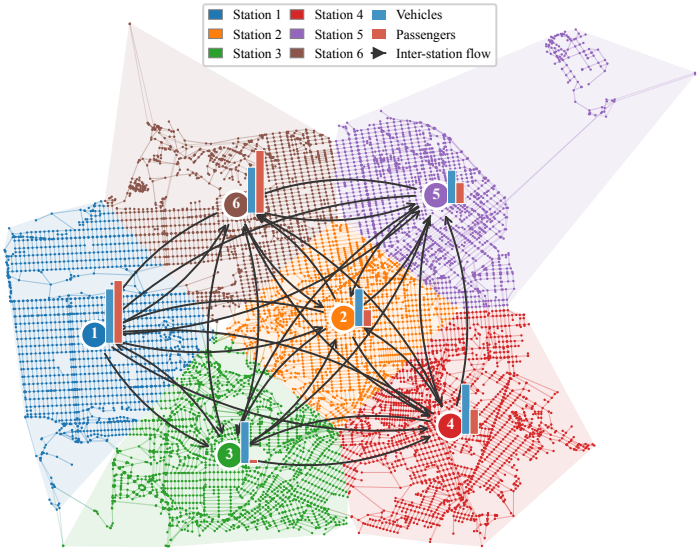
The station graph offers several practical advantages over the road network for fleet-level control. The number of decision variables in the flow formulation is $\mathcal{O}(N^2 T)$, where T is the prediction horizon length, regardless of fleet size or the number of outstanding requests. The aggregation also simplifies demand

forecasting: predicting how many passengers will travel between two stations is a much easier machine learning task than predicting demand for individual junction pairs. A single station typically encompasses many road junctions, so aggregation smooths out the sparse, noisy demand at junction level into a cleaner signal with clearer daily and weekly patterns that forecasting models can learn reliably.

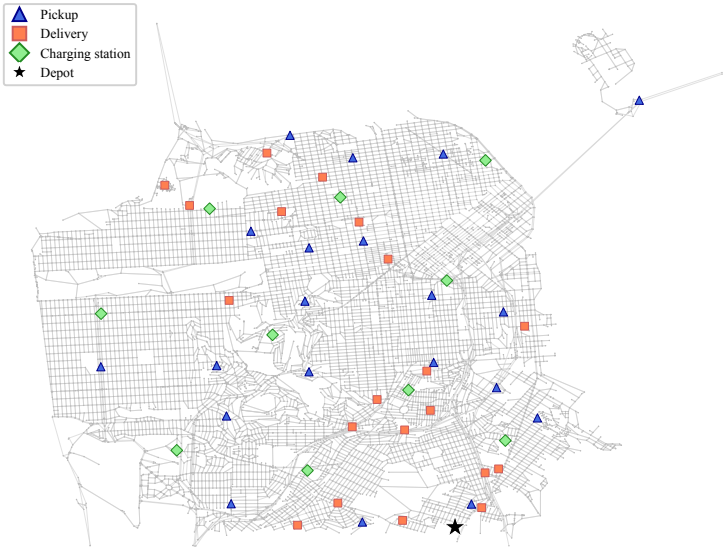
The station graph does, however, come with limitations. Because individual vehicle routes are not modelled explicitly, it is not possible to enforce passenger-specific constraints such as time windows, maximum ride times, and vehicle capacity as hard constraints. In Papers I and II this is addressed through a hierarchical decomposition: once the high-level optimiser has determined how many vehicles should serve requests between a given pair of stations, a lower-level controller matches specific vehicles to specific requests using a nearest-vehicle dispatching rule. Passenger-level service quality is therefore managed implicitly through service-level targets in the objective rather than as hard constraints. Furthermore, since stations can cover large geographic areas, the exact origin and destination within a station are unknown, meaning that travel times and energy consumption cannot be determined precisely and must be treated as uncertain quantities in the operational framework. This limitation motivates the vehicle-level formulation of Paper III, which retains the asymmetric road network graph and constructs detailed individual routes with explicit passenger constraints. The two representations are complementary: the station graph enables scalable strategic control, while the road network graph enables operationally precise routing.

2.2 Network Flow Formulation

Papers I and II share a common fleet-level optimization model based on network flows over the station graph. Table 2.1 summarises the notation used in the network flow formulation. The service area is partitioned into N stations indexed by $i \in \mathcal{N}$, and time is discretised into periods $t \in \mathcal{T}$ of duration Δt. The system is represented as a time-space network where nodes correspond to station-time pairs (i, t) and edges represent feasible vehicle movements between them.



(a) Station graph used in Papers I and II. The city is partitioned into $N = 6$ stations. Arrows indicate inter-station flows; bar charts show vehicle (blue) and passenger (orange) counts per zone.



(b) Road network graph used in Paper III. Nodes represent junctions and edges are directed road segments, with pickup, delivery, charging, and depot locations marked.

Figure 2.1: Two representations of the San Francisco network: a station graph (top) for fleet-level optimization and a road network (bottom) for individual vehicle routing.

Table 2.1: Notation used in the network flow formulation (Section 2.2).

Symbol	Description
$\mathcal{N}$	Set of stations, indexed by i, j
$\mathcal{T}$	Set of discrete time periods, indexed by t
$\mathcal{B}$	Set of discrete SoC levels, indexed by b
Δt	Duration of one time period
x^r_{ijbt}	Rebalancing vehicle flow from station i to j departing at time t with vehicle type b
x^c_{ijbt}	Customer service vehicle flow from station i to j departing at time t with vehicle type b
τ^r_{ij}	Travel time (in periods) for rebalancing from i to j
τ^c_{ij}	Travel time (in periods) for customer service from i to j
λ_{ijt}	Passenger demand from station i to j arriving during period t
s_{ijt}	Accumulated unserved demand (imbalance) from i to j at the end of period t
t_0	Initial time period of the optimisation horizon
T	Length of the prediction horizon (number of periods)

Vehicle Conservation

Two types of vehicle flow are modelled: rebalancing flows x^r_{ijbt} carry empty vehicles from station i to station j at time t for vehicle type b, and customer service flows x^c_{ijbt} carry vehicles serving passengers on the same edge. Vehicles that remain at a station between consecutive periods are modelled as self-rebalancing flows x^r_{iibt}, i.e. rebalancing edges with identical origin and destination. Vehicle flow must be conserved at each node:

$$\sum_{j\in\mathcal{N}} \left(x^r_{jibt-\tau^r_{ji}} + x^c_{jibt-\tau^c_{ji}}\right) = \sum_{j\in\mathcal{N}} \left(x^r_{ijbt} + x^c_{ijbt}\right), \quad \forall i \in \mathcal{N},\, b \in \mathcal{B},\, t \in \mathcal{T}, \tag{2.1}$$

where τ^r_{ji} and τ^c_{ji} are the travel times from station j to station i for rebalancing and customer service trips respectively. The self-rebalancing term x^r_{iibt} with $\tau^r_{ii} = 1$ naturally handles vehicles that wait one period before their next

dispatch, so no separate waiting variable is required.

Let λ_{ijt} denote the number of passenger requests from station i to station j at time t. Customer service flows must not exceed available demand:

$$\sum_{b \in \mathcal{B}} x^c_{ijbt} \leq \lambda_{ijt}, \quad \forall i, j \in \mathcal{N},\, t \in \mathcal{T}. \tag{2.2}$$

Demand Imbalance Dynamics

In practice, unserved passengers are not simply discarded. If more requests arrive than vehicles are available to serve them, the excess accumulates and must be served in a later time period. This behavior is not captured by the flow constraints alone and requires a separate state variable. The imbalance $s_{ijt} \geq 0$ tracks the number of outstanding unserved requests from station i to station j at the end of period t. At the initial time t_0 the imbalance is

$$s_{ijt_0} = \lambda_{ijt_0} - \sum_{b \in \mathcal{B}} x^c_{ijbt_0}, \quad \forall i, j \in \mathcal{N}, \tag{2.3}$$

and for subsequent periods it evolves as

$$s_{ijt+1} = s_{ijt} + \lambda_{ijt+1} - \sum_{b \in \mathcal{B}} x^c_{ijbt+1}, \quad \forall i, j \in \mathcal{N},\; t \in [t_0, T + t_0]. \tag{2.4}$$

The non-negativity constraint $s_{ijt} \geq 0$ ensures that customer service flows never exceed the total cumulative demand. Penalising accumulated imbalance in the objective rather than one-off rejections means that a passenger who waits k periods contributes k times to the cost, directly incentivising the controller to minimise waiting time rather than merely limit the number of rejections.

2.3 The Three optimization Problems

The three papers address three distinct optimization problems that differ in their network representation, decision granularity, and the nature of the uncertainty they face. Each problem is a natural extension of the previous one, motivated by limitations in the scope of what came before. This section describes the structure of each problem. The methods used to solve them are developed in Chapters 3 and 5.

Paper I: Fleet rebalancing under demand uncertainty. Paper I considers a non-electric MoD fleet operating on a space-time station graph. The central decisions are how many vehicles to rebalance between stations and how many passenger requests to serve or reject at each time step. The objective is to minimise a combination of rebalancing cost and the penalty for rejected requests over a finite planning horizon. The key challenge is that passenger demand is uncertain: the number of requests arriving at each station in each time period is not known in advance. Rather than requiring service-level constraints to hold under all possible demand realisations, Paper I requires them to hold with a specified probability. This makes the problem a chance-constrained programme, which must be solved repeatedly at each decision instance as new demand observations become available. The optimization problem is formulated over the network flow model of Section 2.2, and the fleet does not carry battery state as part of its state.

Paper II: Joint rebalancing and charging under multiple uncertainties. Paper II extends the scope to an electric fleet on a space-time-energy station graph. The decisions now include not only rebalancing flows but also charging schedules and vehicle-to-grid (V2G) energy sales at each station. The battery state of charge of each vehicle type enters the vehicle state, and the hard constraint that no vehicle may run out of charge must be satisfied at all times. The planning problem is therefore substantially more constrained than in Paper I. The uncertainty is also richer: travel demand and charger availability are stochastic and correlated across space and time, while energy consumption and travel time are subject to adversarial variation that is harder to characterise probabilistically. Paper II must handle all four uncertainty sources simultaneously while producing decisions that are both feasible and cost-effective. The resulting problem is a multi-stage stochastic mixed-integer programme, significantly larger than the chance-constrained LP in Paper I, and its tractability requires a decomposition strategy developed in Chapter 3.

Paper III: Individual vehicle routing under operational constraints. Papers I and II determine aggregate flows between stations but delegate the matching of specific vehicles to specific passengers to a lower-level dispatcher. This is adequate for fleet-level performance but cannot enforce passenger-level constraints such as time windows, maximum ride times, and vehicle capacity.

Paper III addresses this gap by formulating the Electric Dial-a-Ride Problem (E-DARP): constructing detailed pickup and delivery routes for individual electric vehicles over the asymmetric road network graph. Each request specifies a pickup location, a delivery location, a time window within which the pickup must occur, and a maximum allowable ride time. Each vehicle has a capacity constraint limiting the number of simultaneous passengers and a battery constraint requiring the vehicle to visit a charging station before its charge is depleted. The E-DARP is NP-hard and the search space grows combinatorially with the number of requests, making exact optimization intractable at the scales required for real-time deployment. The solution approach is therefore fundamentally different from Papers I and II, as described in Chapter 5.

2.4 Performance Metrics

Evaluating E-AMoD system performance requires metrics that capture the perspectives of passengers, operators, and society. These objectives often conflict, and operators must choose how to weight them in the objective function.

From the passenger perspective, the most important metrics are waiting time (from request submission to vehicle arrival), ride time (from pickup to delivery), and completion rate (the fraction of requests that are served rather than rejected). Short waiting times and high completion rates are the primary measures of service quality used across all three papers.

From the operator perspective, the key metrics are total vehicle distance travelled (including empty rebalancing), energy cost, and charging cost. Proactive rebalancing can reduce empty vehicle distance significantly by anticipating demand rather than reacting to it [11]. Paper II additionally tracks electricity costs under different time-of-use pricing schemes and V2G revenues.

From a societal perspective, the metrics of interest include emissions (which depend on the carbon intensity of the electricity grid), congestion effects, and accessibility. These are not directly optimised in the papers but inform the broader motivation for E-AMoD research.

2.5 Simulation Framework

Evaluating the performance of Electric Autonomous Mobility-on-Demand systems through real world experimentation at scale is both impractical and risky.

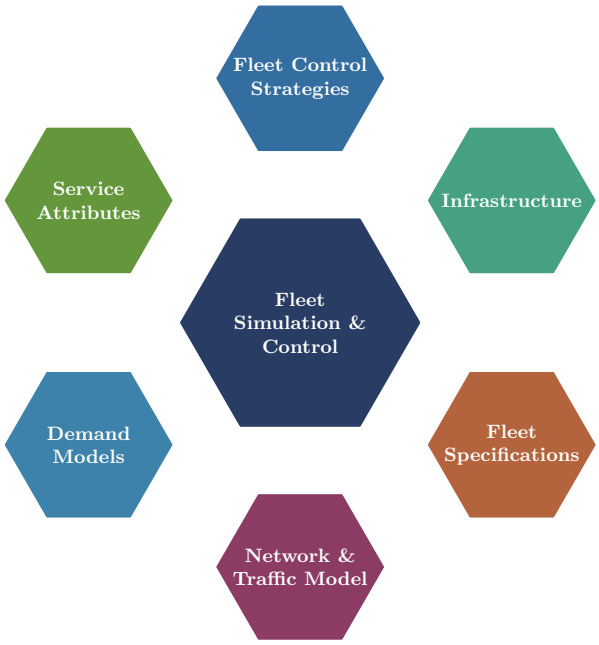


Figure 2.2: Key modelling components that define an E AMoD simulation environment.

Cities cannot systematically test alternative rebalancing policies, charging strategies, fleet sizes, or battery capacities through trial and error on operational systems serving real passengers. Large scale field experiments are costly and logistically complex, and poor decisions directly affect users through long waiting times or service disruptions. For these reasons, E-AMoD systems are evaluated in simulation, where assumptions can be controlled and varied systematically.

A simulation environment for E-AMoD evaluation is defined by six modelling components: fleet control strategies, infrastructure, fleet specifications, network and traffic model, demand models, and service attributes, as illustrated in Figure 2.2. Together, these components determine both the physical feasibility of operations and the interpretation of performance outcomes.

The network and traffic model specifies how vehicles move through the city. It defines the road topology, travel distances, and travel times that govern vehicle motion. Modelling choices include whether travel times are static or time dependent, whether congestion is endogenous or treated as fixed, and

the level of aggregation used to represent the network. In this thesis, real urban road networks are used, with travel times derived from a mesoscopic agent-based traffic simulation [28].The representation differs across decision scales. Station level graphs are used for fleet level optimization in Papers I and II,while Paper III operates on the full asymmetric road network.

Demand models describe when, where, and how many passengers request service. Because spatial imbalances between vehicle supply and demand drive rebalancing and charging decisions, the assumed demand process plays a central role in evaluation. In this thesis, demand is derived from real taxi trip data from San Francisco [55] and Chicago [56], preserving realistic spatiotemporal patterns of urban travel behavior. Demand is treated as stochastic and time varying, and its uncertainty directly influences operational decisions.

Fleet specifications define the physical characteristics of the vehicles, including fleet size, battery capacity, passenger capacity, and charging power. These parameters determine operational flexibility and resilience to demand fluctuations. Fleet size and battery capacity are varied systematically to study their interaction with operational decision making under uncertainty.

Infrastructure determines where vehicles may park and charge, and limits how many vehicles can charge simultaneously at a given location. Charging station placement and capacity introduce spatial and temporal constraints that interact strongly with fleet specifications and control strategies. Infrastructure is treated as fixed within each experiment, enabling controlled comparison of operational policies.

Service attributes define customer-facing constraints such as maximum waiting times, ride time limits, pricing assumptions, and whether ride pooling is permitted. These attributes influence both service quality and operational feasibility. In Papers I and II, service levels are incorporated through penalties or probabilistic constraints at the station level. In Paper III, time windows and ride time limits are enforced explicitly within the routing formulation.

Fleet control strategies constitute the decision layer of the system. They determine how vehicles are assigned, rebalanced, routed, and charged over time. This component is the primary object of algorithmic development in the thesis. Papers I and II employ optimization based Model Predictive Control to determine aggregate station-level flows under uncertainty, while Paper III uses deep reinforcement learning to construct vehicle-level pickup and delivery routes.

Transport simulation models are commonly classified into three levels. Macroscopic models treat traffic as a continuous flow and represent aggregate quantities such as density and average speed without tracking individual vehicles. Microscopic models simulate each vehicle individually with explicit car following and lane changing dynamics, but at significant computational cost. Mesoscopic models occupy the middle ground, representing vehicles as discrete agents while using simplified interaction rules. This balance between individual resolution and computational tractability makes mesoscopic simulation well suited to fleet-level E-AMoD studies across realistic urban networks.

Papers I and II are evaluated using AMoDeus [27], an open source simulation platform for autonomous mobility on demand systems built on top of the Multi Agent Transport Simulation MATSim [28]. MATSim is an activity based transport simulator operating at the mesoscopic level. It simulates the daily activity patterns of a synthetic population whose agents execute plans on a queue based road network. Road capacity constraints emerge from the interaction of competing agents rather than from explicit flow equations. Over successive iterations, the population converges toward a relaxed state approximating a stochastic user equilibrium, producing realistic demand patterns that reflect the spatial and temporal structure of urban travel.

AMoDeus integrates the on demand fleet service into the MATSim loop through a dispatcher interface. At each decision instance, the controller receives the current fleet state and outstanding requests, computes rebalancing and charging actions, and dispatches commands that are executed within the simulation. The simulator provides realized travel times, trip requests, and network information, and records waiting times, ride times, and energy consumption for performance evaluation. The San Francisco road network is used in Papers I and II, with synthetic demand derived from a MATSim population calibrated to match observed travel patterns.

Paper III is evaluated separately using benchmark instances from the electric dial-a-ride problem (E-DARP) literature as well as larger synthetic instances generated from the same San Francisco road network. This allows direct comparison with exact solvers and state of the art heuristics at scales where detailed traffic simulation is not the primary concern.

CHAPTER 3

Robust and Stochastic Optimization

Optimization under uncertainty is fundamental to E-AMoD operations, where decisions must be made with incomplete knowledge of future demand, travel times, energy consumption and resource availability. Two primary paradigms address this challenge. Robust optimization adopts a worst-case perspective, seeking solutions that remain feasible and near-optimal under all realizations of uncertainty within specified bounds [40], [57], and is particularly suitable when probability distributions are unknown or unreliable. Stochastic programming instead assumes knowledge of probability distributions and optimizes expected performance or probabilistic guarantees [26], [58], making it appropriate when historical data enables reliable distribution estimation. This chapter introduces both paradigms alongside the computational tools needed to apply them in E-AMoD contexts: chance constraints, Mixed Integer Linear Programming and the total unimodularity property that enables tractable network flow solutions, Model Predictive Control as the receding-horizon strategy that operationalizes these methods in real time, and Benders decomposition for large-scale multi-stage problems. Throughout, we emphasize techniques relevant to Papers I and II.

3.1 Robust Optimization

Robust optimization emerged from the recognition that optimal solutions to deterministic problems can be highly sensitive to parameter values, performing poorly when parameters deviate even slightly from their nominal values [40]. Rather than optimizing for a single scenario, robust optimization seeks solutions that hedge against uncertainty by remaining feasible and near-optimal across all scenarios in an uncertainty set.

Motivation and Fundamental Concepts

Consider a generic optimization problem with uncertain parameters $\xi \in \mathbb{R}^p$: minimize $f(x, \xi)$ subject to $g_i(x, \xi) \leq 0$ and $x \in \mathcal{X}$. The deterministic approach replaces ξ with a nominal value $\bar{\xi}$ and optimizes for this single scenario. However, the resulting solution may violate constraints or perform poorly in the true realization $\tilde{\xi} \neq \bar{\xi}$. For E-AMoD systems, this manifests as fleet positioning that appears optimal under expected demand but fails catastrophically when demand spikes unexpectedly [11].

A simple example illustrates this failure mode. Consider a rebalancing problem where x_i represents vehicles positioned at location i and ξ_i represents uncertain demand. The constraint $x_i \geq \xi_i$ ensures sufficient vehicles to meet demand. Using nominal demand $\bar{\xi}_i$, we position $x_i = \bar{\xi}_i$ vehicles. However, if actual demand $\tilde{\xi}_i > \bar{\xi}_i$, the constraint is violated and customers are rejected. Robust optimization prevents such failures by requiring feasibility for all plausible demand realizations [59].

Uncertainty Sets and Conservatism Control

Robust optimization requires specifying an uncertainty set $\mathcal{U} \subseteq \mathbb{R}^p$ containing all plausible realizations of uncertain parameters [39], [40]. The choice of uncertainty set involves a fundamental trade-off: larger sets provide greater protection against uncertainty but lead to more conservative solutions with higher costs, while smaller sets reduce conservatism but risk constraint violations. Figure 3.1 illustrates common uncertainty set geometries.

The simplest form is the box uncertainty set $\mathcal{U}_{\text{box}} = \{\xi : \underline{\xi}_i \leq \xi_i \leq \bar{\xi}_i\}$, which bounds each uncertain parameter independently. For E-AMoD demand uncertainty, this corresponds to assuming demand at each location-time pair lies

within bounds derived from historical data [16]. Box sets are computationally attractive but often overly conservative, as they assume all parameters simultaneously reach their worst-case values, a scenario that may be extremely unlikely [39].

Ellipsoidal uncertainty sets $\mathcal{U}_{\text{ellipsoid}} = \{\xi : (\xi - \bar{\xi})^\top \Sigma^{-1} (\xi - \bar{\xi}) \leq \Omega^2\}$ capture correlations between uncertain parameters through the covariance matrix Σ. For E-AMoD applications, Σ can be estimated from historical demand data, capturing spatiotemporal correlations between locations [24]. Ellipsoidal sets better reflect the joint distribution of uncertainties but lead to more complex robust counterparts.

Budget-constrained uncertainty sets, introduced by Bertsimas and Sim [39], provide an elegant mechanism for controlling conservatism. These sets limit the number of parameters that deviate significantly from nominal values:

$$\mathcal{U}_\Gamma = \left\{ \xi : \xi_i = \bar{\xi}_i + \hat{\xi}_i z_i,\ |z_i| \leq 1,\ \sum_{i=1}^{p} |z_i| \leq \Gamma \right\} \tag{3.1}$$

where $\bar{\xi}_i$ is the nominal value, $\hat{\xi}_i$ is the maximum deviation, and the uncertainty budget $\Gamma \in [0, p]$ controls conservatism. When $\Gamma = 0$, the set reduces to the nominal scenario; when $\Gamma = p$, all parameters can deviate maximally. Intermediate values reflect the realistic observation that while individual parameters frequently deviate from expectations, simultaneous worst-case deviations across all parameters are rare [39].

For E-AMoD demand uncertainty, this formulation is particularly appealing. Demand at individual locations often exceeds forecasts, but system-wide demand is more predictable due to averaging effects [23]. The parameter Γ can be calibrated using historical data to achieve desired service levels, such as ensuring ninety-five percent of demand is met [60]. This calibration procedure converts robust optimization into a data-driven approach with probabilistic performance guarantees.

Robust Counterpart Formulation

Given an uncertainty set $\mathcal{U}$, the robust counterpart requires feasibility and optimality for all realizations in $\mathcal{U}$ [40]. The objective minimizes the worst-

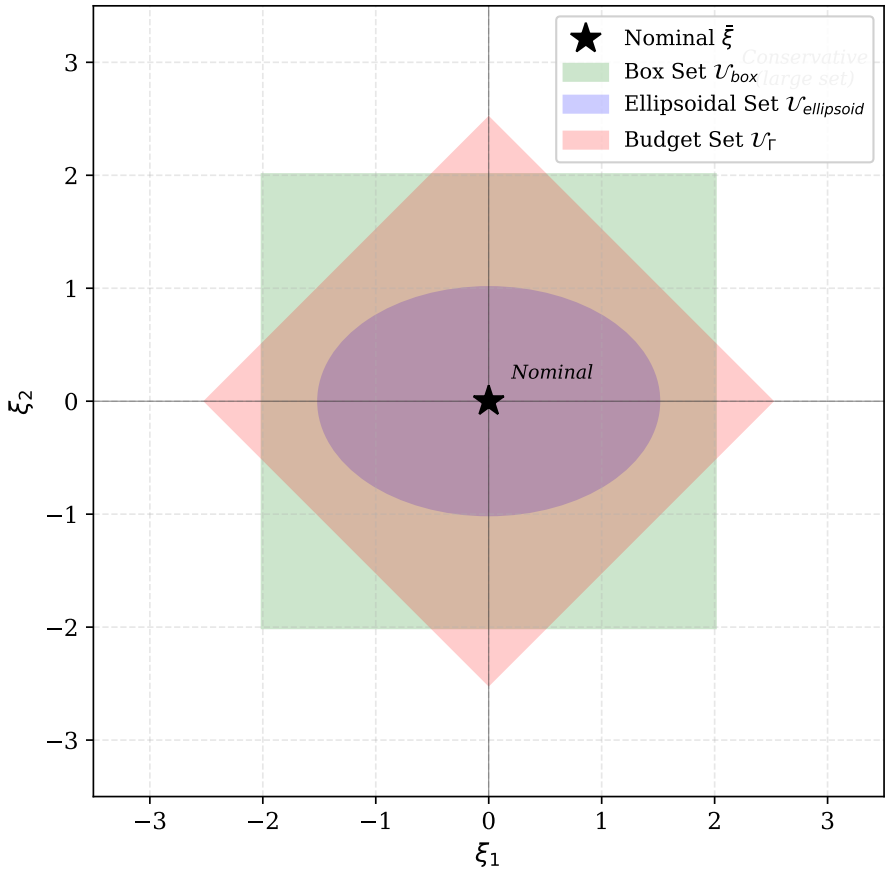


Figure 3.1: Comparison of uncertainty set geometries in two dimensions. The nominal parameter value (center point) is surrounded by three uncertainty sets: box sets (rectangles) bound each parameter independently, ellipsoidal sets (ellipses) capture correlations through quadratic constraints, and budget-constrained sets (diamonds) limit the total deviation across parameters. Larger sets provide greater protection but increase solution conservatism.

case cost, and constraints must hold for all possible parameter realizations:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \max_{\xi \in \mathcal{U}} f(x, \xi) \\ \text{subject to} \quad & g_i(x, \xi) \leq 0, \quad \forall \xi \in \mathcal{U} \end{aligned} \tag{3.2}$$

The constraints $g_i(x, \xi) \leq 0, \forall \xi \in \mathcal{U}$ are infinite-dimensional, as they must hold for infinitely many values of ξ. Remarkably, for many problem structures and uncertainty sets, these infinite constraints can be reformulated as finite sets of tractable constraints through duality theory [57].

3.2 Stochastic Optimization

While robust optimization hedges against worst-case scenarios, stochastic optimization explicitly models probability distributions of uncertain parameters and optimizes expected performance or probabilistic guarantees [26], [58]. This approach is suitable when historical data enables reliable estimation of uncertainty distributions, as is often the case for E-AMoD demand [23], [24].

Two-Stage Stochastic Programming

The canonical form of stochastic optimization is the two-stage stochastic program. In the first stage, decisions x are made before observing the random parameter $\tilde{\xi}$. In the second stage, after observing the realization ξ, recourse decisions $y(\xi)$ are made to adapt to the scenario. The objective minimizes the sum of first-stage costs and expected second-stage costs:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^\top x + \mathbb{E}_{\tilde{\xi}}[Q(x, \tilde{\xi})] \\ \text{subject to} \quad & Ax \leq b, \quad x \in \mathcal{X} \end{aligned} \tag{3.3}$$

where $Q(x, \xi)$ is the optimal value of the second-stage problem given first-stage decision x and realization ξ. This second-stage value function represents the best response to each scenario: $Q(x, \xi) = \min_y \{d(\xi)^\top y : W(\xi)y \leq h(\xi) - T(\xi)x,\ y \in \mathcal{Y}\}$. For E-AMoD applications, first-stage decisions represent fleet positioning based on demand forecasts, while second-stage decisions represent vehicle-to-request assignments after observing actual demand [11].

The expectation $\mathbb{E}_{\tilde{\xi}}[Q(x, \tilde{\xi})]$ is typically intractable to compute exactly because it requires integrating over the continuous distribution of $\tilde{\xi}$. Instead,

we approximate it using a finite set of scenarios $\{\xi^1, \ldots, \xi^S\}$ with probabilities $\{p^1, \ldots, p^S\}$ where $\sum_{s=1}^{S} p^s = 1$. These scenarios can be generated from predictive distributions provided by Gaussian processes or Bayesian neural networks as discussed in Chapter 4. The expectation is approximated by $\mathbb{E}_{\tilde{\xi}}[Q(x, \tilde{\xi})] \approx \sum_{s=1}^{S} p^s Q(x, \xi^s)$.

This scenario approximation leads to the extensive form formulation:

$$\begin{aligned} \underset{x, \{y^s\}}{\text{minimize}} \quad & c^\top x + \sum_{s=1}^{S} p^s d(\xi^s)^\top y^s \\ \text{subject to} \quad & Ax \leq b \\ & W(\xi^s) y^s \leq h(\xi^s) - T(\xi^s) x, \quad s = 1, \ldots, S \\ & x \in \mathcal{X},\ y^s \in \mathcal{Y} \end{aligned} \tag{3.4}$$

This formulation explicitly represents all scenarios with separate decision variables y^s for each scenario's recourse. The extensive form grows rapidly with the number of scenarios, but it can be solved by commercial solvers for problems with tens to hundreds of scenarios. For larger problems, decomposition methods discussed in Section 3.6 become essential.

Multi-Stage Stochastic Programming

Two-stage stochastic programming assumes that all uncertainty is revealed after the first-stage decision. In many operational settings, however, uncertainty unfolds gradually over time: demand at hour $t+1$ is revealed only after decisions at hour t have been implemented. Multi-stage stochastic programming extends the two-stage framework to capture this sequential structure, enabling decisions at each stage to exploit information revealed up to that point [26], [58].

Consider a planning horizon divided into T stages. At each stage $t = 1, \ldots, T$, a portion of uncertainty $\tilde{\xi}_t$ is revealed, and decisions x_t are made based on the history $\mathcal{F}_t = (\xi_1, \ldots, \xi_t)$. The non-anticipativity constraint formalizes this requirement: decisions at stage t may only depend on information available at that stage, not on future realizations. The multi-stage stochastic

program minimizes total expected costs across all stages:

$$\begin{aligned} \underset{x_1, x_2(\xi_1), \ldots, x_T(\xi_{1:T-1})}{\text{minimize}} \quad & c_1^\top x_1 + \mathbb{E}\left[\sum_{t=2}^{T} c_t^\top x_t(\xi_{1:t-1})\right] \\ \text{subject to} \quad & A_t x_t + B_t x_{t-1} \leq b_t(\xi_t), \quad t = 1, \ldots, T \\ & x_t = x_t(\mathcal{F}_t), \quad t = 1, \ldots, T \\ & x_t \in \mathcal{X}_t, \quad t = 1, \ldots, T \end{aligned} \tag{3.5}$$

where the first constraint enforces stage-wise feasibility linking decisions across consecutive stages, the second is the non-anticipativity constraint requiring that x_t depends only on the information $\mathcal{F}_t$ available at stage t, and $\mathcal{X}_t$ defines the feasible set at each stage. This structure is fundamentally more complex than two-stage programming because each decision function $x_t(\cdot)$ must be chosen from a space of mappings from observable histories to actions.

Scenario trees provide a tractable finite approximation of the continuous uncertainty process [61], [62]. A scenario tree is a directed tree where each node n at stage t represents a distinct history of uncertainty realizations, and each arc from a node to its children represents a possible transition in uncertainty. Each path from the root to a leaf corresponds to one complete scenario. The tree structure directly encodes non-anticipativity: all scenarios sharing the same ancestor node at stage t must make the same decisions at that stage.

A practical challenge is that the number of nodes grows exponentially with the number of stages, limiting the depth of the tree. Paper II addresses this through a *robust horizon*: the first stages of the planning horizon are modeled with stochastic branching, capturing distributional information from probabilistic forecasts, while the remaining stages replace scenario branching with one scenario per stage. This hybrid structure concentrates computational effort on the near-term stages where forecast quality is highest and decisions have the greatest impact, while the robust horizon maintains planning depth without requiring additional branching. Figure 3.2 illustrates this structure.

Formally, let $\mathcal{N}$ be the set of all nodes, $\mathcal{L} \subset \mathcal{N}$ the leaf nodes, and $a(n)$ the ancestor of node n. Each node n is associated with a probability p_n (the probability of reaching that node), a stage $t(n)$, and an uncertainty realization ξ_n. The extensive form of the multi-stage stochastic program across the

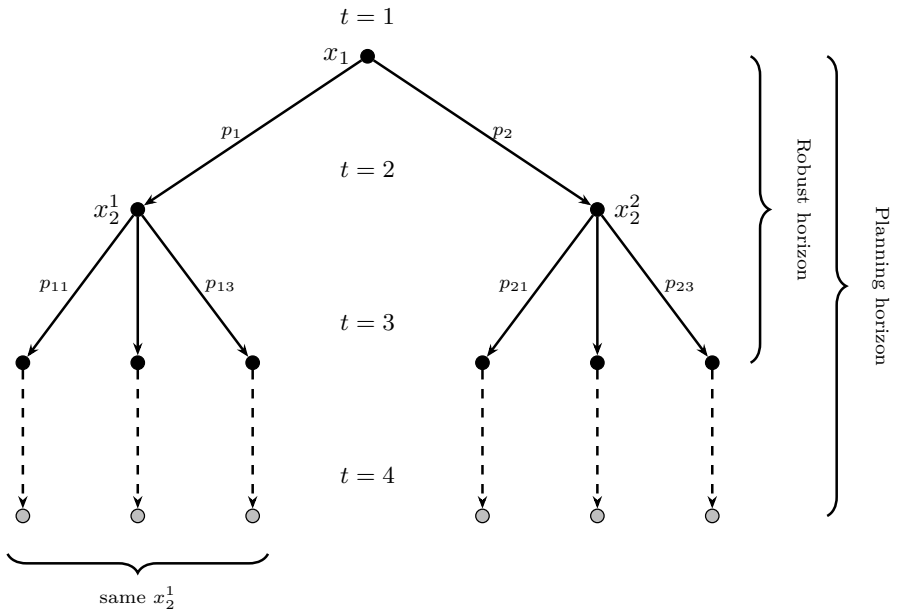


Figure 3.2: Combined stochastic-robust scenario tree. Solid arrows and black nodes denote the stochastic horizon ($t = 1$ to $t = 3$), where branching captures distributional information from probabilistic forecasts. Dashed arrows and gray nodes denote the robust horizon ($t = 4$), where each node continues with a single scenario, extending the planning depth without additional branching.

scenario tree is:

$$\begin{aligned} \underset{\{x_n\}_{n\in\mathcal{N}}}{\text{minimize}} \quad & \sum_{n\in\mathcal{N}} p_n \, c_{t(n)}^\top x_n \\ \text{subject to} \quad & A_{t(n)} x_n + B_{t(n)} x_{a(n)} \le b_n, \quad \forall n \in \mathcal{N} \\ & x_n \in \mathcal{X}_{t(n)}, \quad \forall n \in \mathcal{N} \end{aligned} \tag{3.6}$$

The coupling constraint $B_{t(n)} x_{a(n)}$ links decisions at node n to those made at its parent, capturing how resources (vehicles, battery charge, etc.) carry over across time periods. The tree structure makes the constraint matrix block-angular, a property exploited by Nested Benders Decomposition in Section 3.6.

A practical challenge is scenario tree construction: the tree must faithfully represent the underlying uncertainty process while remaining computationally tractable. Tree branching increases exponentially with depth, so trees are typically limited to a modest branching factor per stage. Scenario reduc-

tion techniques [63] prune or aggregate scenarios to produce compact trees that preserve key statistical properties such as moments and distributional distances. Paper II employs scenario trees generated from Bayesian neural network forecasts, where scenario fans are sampled from the predictive distribution at each stage and then clustered using k-means to determine the branching structure and assign scenario probabilities.

3.3 Chance Constraints

Chance constraints, also called probabilistic constraints, require probabilistic satisfaction rather than deterministic feasibility [41], [42]. Rather than requiring a constraint to hold for all scenarios, chance constraints require it to hold with specified probability:

$$\mathbb{P}\left[g_i(x, \tilde{\xi}) \leq 0\right] \geq 1 - \epsilon_i \tag{3.7}$$

For E-AMoD rebalancing, a natural chance constraint is $\mathbb{P}[\text{available vehicles} \geq \text{demand}] \geq 0.95$, requiring that vehicle availability is sufficient to meet demand with at least 0.95 probability. This provides a probabilistic service level guarantee while accepting that the constraint may be violated with probability up to ϵ, avoiding the excessive conservatism of robust optimization that requires feasibility under all scenarios in the uncertainty set [16].

Chance constraints are generally non-convex and difficult to handle computationally. However, under certain distributional assumptions, they admit tractable reformulations. When $g_i(x, \tilde{\xi})$ is affine in $\tilde{\xi}$ and $\tilde{\xi}$ follows a Gaussian distribution, the chance constraint can be reformulated as a deterministic second-order cone constraint [40]. Paper I exploits this property, using Gaussian process demand predictions to reformulate chance-constrained rebalancing problems as tractable convex programs.

An alternative approach is Sample Average Approximation, where the continuous distribution is approximated by the empirical distribution of a sample $\{\xi^1, \ldots, \xi^N\}$. Requiring feasibility for all sampled scenarios provides probabilistic guarantees. Calafiore and Campi [64] show that if $g_i(x, \xi^j) \leq 0$ for all $j = 1, \ldots, N$ sampled scenarios, then with probability at least $1 - \beta$, the constraint $\mathbb{P}[g_i(x, \tilde{\xi}) \leq 0] \geq 1 - \epsilon$ holds, provided the sample size satisfies:

$$N \geq \frac{2}{\epsilon} \ln \frac{1}{\beta} + \frac{2d}{\epsilon} \tag{3.8}$$

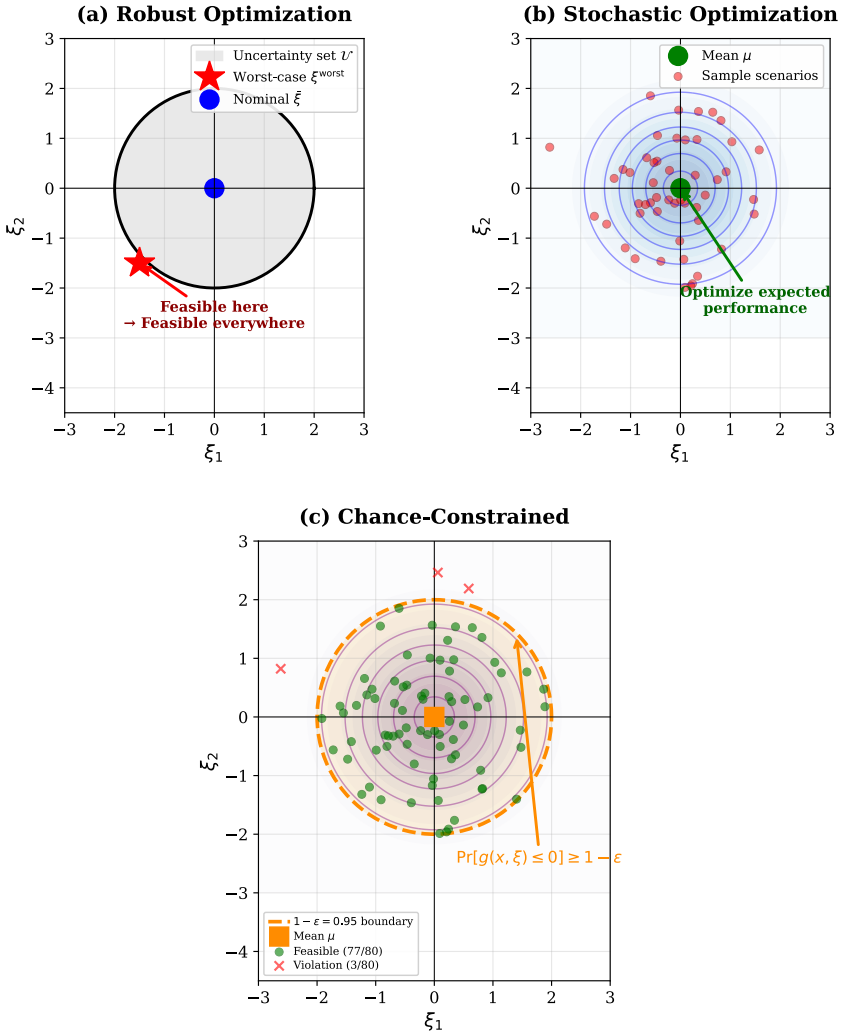


Figure 3.3: Comparison of optimization paradigms under uncertainty. Robust optimization (a) hedges against worst-case scenarios within an uncertainty set, providing feasibility guarantees but potentially high costs. Stochastic optimization (b) leverages probability distributions to minimize expected cost, providing better expected performance but only probabilistic guarantees. Chance-constrained optimization (c) bridges both approaches by enforcing that constraints are satisfied with a prescribed probability $1-\varepsilon$, allowing controlled constraint violation while avoiding excessive conservatism. The choice depends on data availability, risk tolerance, and operational requirements.

where d is the number of decision variables. This provides a data-driven approach to chance constraints without requiring specific distributional assumptions, though the required sample size grows linearly with problem dimension [65].

When only moments of the distribution are known, distributional robustness offers a complementary approach. Chebyshev's inequality guarantees that for any distribution with mean μ and variance σ^2, no more than $1/k^2$ of values lie beyond k standard deviations from the mean. This enables reformulation of chance constraints using only mean and variance information, though the resulting constraints are typically conservative. When the distribution is known to be Gaussian, tighter reformulations use the inverse cumulative distribution function directly, as exploited in Paper I where Gaussian process predictions provide Gaussian demand distributions.

3.4 Mixed Integer Linear Programming and Network Flow Tractability

Many E-AMoD decision problems involve discrete choices: the number of vehicles routed along a network arc, whether to activate a charger at a given station, or how many passengers to serve in a given period. Mixed Integer Linear Programs (MILPs) extend linear programming by restricting some variables to integer values. A general MILP takes the form:

$$\begin{aligned} \underset{x,z}{\text{minimize}} \quad & c^\top x + d^\top z \\ \text{subject to} \quad & Ax + Bz \leq b \\ & x \geq 0, \quad z \in \mathbb{Z}^m_{\geq 0} \end{aligned} \tag{3.9}$$

where $x \in \mathbb{R}^n$ are continuous variables and $z \in \mathbb{Z}^m$ are integer variables. Integer constraints make the feasible region non-convex and combinatorial, rendering MILPs NP-hard in general [66]. The standard solution framework is Branch and Bound (BnB): the algorithm maintains an upper bound by tracking the best feasible integer solution found so far, and a lower bound by solving LP relaxations at each node of a search tree in which integrality constraints are temporarily dropped. Branches whose lower bound exceeds the best known upper bound are pruned, progressively narrowing the search. Modern solvers such as Gurobi and CPLEX augment this with cutting planes,

presolve routines, and primal heuristics to handle large-scale problems in practice [67], [68].

Total Unimodularity and E-AMoD Tractability

For the network flow problems in Papers I and II, an important structural property means that the full BnB machinery is never required. A matrix $A \in \mathbb{Z}^{m \times n}$ is *totally unimodular* (TU) if every square submatrix has determinant 0, +1, or −1 [69]. The key consequence is that if A is TU and $b \in \mathbb{Z}^m$, every vertex of the polyhedron $\{x \geq 0 : Ax \leq b\}$ is integer-valued. The LP relaxation of the corresponding integer program therefore always yields an integer optimal solution, and expensive branch-and-bound is not required [70].

The node-arc incidence matrix of any directed graph is always TU [69], which means any minimum-cost flow problem on a directed network with integer capacities and demands has an integer optimal solution obtainable by solving the LP relaxation. The E-AMoD rebalancing problems in Papers I and II have precisely this structure: vehicles move between stations over a time-expanded network, flow conservation constraints define the node-arc incidence matrix, and all capacities and demands are integer-valued. Rather than solving integer programs, Paper I therefore solves the LP relaxation of the rebalancing subproblem directly, recovering integer solutions guaranteed by TU.

In Paper II, however, tracking battery state of charge as a continuous control variable introduces constraints that couple energy flows across arcs in a way that is not representable as a pure node-arc incidence structure. This destroys total unimodularity, and the problem must therefore be solved as a mixed-integer program using for example the BnB framework described above.

3.5 Model Predictive Control

The optimization and prediction methods developed in the preceding sections and Chapter 4 must be combined into an operational control framework. Model Predictive Control (MPC), also known as receding horizon control, provides this framework by repeatedly solving a finite-horizon optimization problem using the best available forecast and implementing only the first portion of the resulting plan [71], [72]. When new observations arrive, the horizon

shifts forward, a new problem is solved, and the cycle repeats.

Receding Horizon Principle

At each control step t, MPC solves an open-loop optimal control problem over a prediction horizon of H steps:

$$\begin{aligned} \underset{u_t,\ldots,u_{t+H-1}}{\text{minimize}} \quad & \sum_{\tau=t}^{t+H-1} \ell(x_\tau, u_\tau) + V_f(x_{t+H}) \\ \text{subject to} \quad & x_{\tau+1} = f(x_\tau, u_\tau, \xi_\tau), \quad \tau = t, \ldots, t+H-1 \\ & x_\tau \in \mathcal{X}, \quad u_\tau \in \mathcal{U} \end{aligned} \tag{3.10}$$

where x_τ is the system state (e.g., vehicle distribution across stations), u_τ is the control input (rebalancing decisions), $\ell(\cdot,\cdot)$ is a stage cost, $V_f(\cdot)$ is a terminal cost penalizing the end-of-horizon state, and ξ_τ represents uncertainty. Only the first control input u_t^* from the optimal sequence is implemented. At time $t+1$, the state is updated based on the observed transition, the horizon shifts forward to $\{t+1, \ldots, t+H\}$, and the problem is resolved with updated forecasts.

The closed-loop feedback mechanism is a defining strength of MPC. Even though the forecast $\hat{\xi}_{t:t+H}$ will differ from the true uncertainty realization, the system self-corrects at each step by re-optimizing with the observed state as the new initial condition. This feedback is particularly valuable in E-AMoD operations, where demand deviates from forecasts and vehicle availability evolves unpredictably [16], [29].

It is worth distinguishing this from the multi-stage stochastic programs introduced in Section 3.2. A multi-stage program is solved once over the full planning horizon, producing a contingency plan that maps every anticipated uncertainty path to a sequence of decisions. MPC instead re-solves a finite-horizon problem at every control step, implements only the first-stage decision, and discards the rest. Errors in later stages are never executed because the problem is re-solved before they are reached. In Papers I and II, the two ideas are combined: at each MPC step, a multi-stage stochastic program is solved over the prediction horizon, but only the root-node decisions are applied before the horizon shifts forward and a new scenario tree is constructed from updated forecasts. This repeated re-solving imposes a hard computational requirement, each optimization must complete within the control interval.

The decomposition methods discussed in Section 3.6 are therefore not merely convenient but necessary, as they enable the stochastic program to be solved within the minutes available between successive MPC steps.

Handling Uncertainty in MPC

The open-loop formulation in (3.10) uses a single forecast trajectory $\hat{\xi}_{t:t+H}$ and ignores uncertainty beyond feedback. Several extensions incorporate uncertainty more explicitly. Stochastic MPC replaces the deterministic objective with an expected cost over predicted scenarios:

$$\underset{u_t,\ldots,u_{t+H-1}}{\text{minimize}} \quad \mathbb{E}_{\tilde{\xi}}\left[\sum_{\tau=t}^{t+H-1} \ell(x_\tau, u_\tau, \tilde{\xi}_\tau)\right] \tag{3.11}$$

using scenario trees or sample-based approximations of the predictive distribution. This connects directly to the multi-stage stochastic programs described in Section 3.2, where the MPC horizon defines the tree depth.

Robust MPC instead minimizes worst-case cost over an uncertainty set $\mathcal{U}$, connecting to the robust counterpart formulations of Section 3.1. Chance-constrained MPC enforces probabilistic feasibility at each predicted step, as in (3.7), exploiting Gaussian predictive distributions from GP models to obtain closed-form constraint reformulations [73].

Two-Level Control Architecture

In Papers I and II, MPC operates as a higher-level controller within a two-level control architecture. The MPC rebalancing controller solves a network-level optimization problem over a horizon of several hours, computing desired vehicle distributions across stations at each time step. This high-level plan is then executed by a lower-level bipartite matching algorithm that assigns individual vehicles to specific passenger requests in real time, operating on a much shorter timescale of seconds to minutes [16], [29].

This separation of timescales is computationally motivated. Solving the joint problem of rebalancing and individual vehicle-to-request assignment simultaneously would require a large-scale combinatorial program that is intractable at operational frequencies. By decomposing into a high-level flow optimization (tractable as a network flow via total unimodularity) and a low-level assignment (tractable as a bipartite matching), both problems remain

solvable within their respective time budgets. Paper III revisits this decomposition, demonstrating that deep reinforcement learning can directly learn joint policies that implicitly solve both levels simultaneously, potentially capturing interactions that the two-level architecture misses.

The MPC framework also naturally accommodates the prediction models discussed in Chapter 4. Gaussian process predictions in Paper I provide both mean forecasts and uncertainty quantification for chance-constrained MPC. Bayesian neural networks in Paper II generate predictive distributions over demand and travel times that seed the scenario trees used in the multi-stage stochastic MPC formulation. In both cases, MPC serves as the operational layer that translates probabilistic forecasts into executable fleet management decisions.

3.6 Decomposition Methods

Two-stage stochastic programs in extensive form grow rapidly with the number of scenarios, quickly becoming intractable for direct solution. Benders decomposition [45] exploits problem structure by partitioning it into a master problem optimizing first-stage decisions and subproblems evaluating second-stage costs for each scenario. This decomposition enables parallel solution and dramatically reduces memory requirements.

Benders Decomposition for Stochastic Programs

The application of Benders decomposition to two-stage stochastic programs is known as the L-shaped method, introduced by Van Slyke and Wets [74]. The algorithm iteratively refines an approximation of the recourse function $Q(x, \xi)$ through optimality cuts. The master problem optimizes first-stage decisions x and an auxiliary variable θ representing the approximated expected recourse cost:

$$\begin{aligned} \underset{x,\theta}{\text{minimize}} \quad & c^\top x + \theta \\ \text{subject to} \quad & Ax \leq b \\ & \theta \geq \text{lower bound approximations} \\ & x \in \mathcal{X} \end{aligned} \tag{3.12}$$

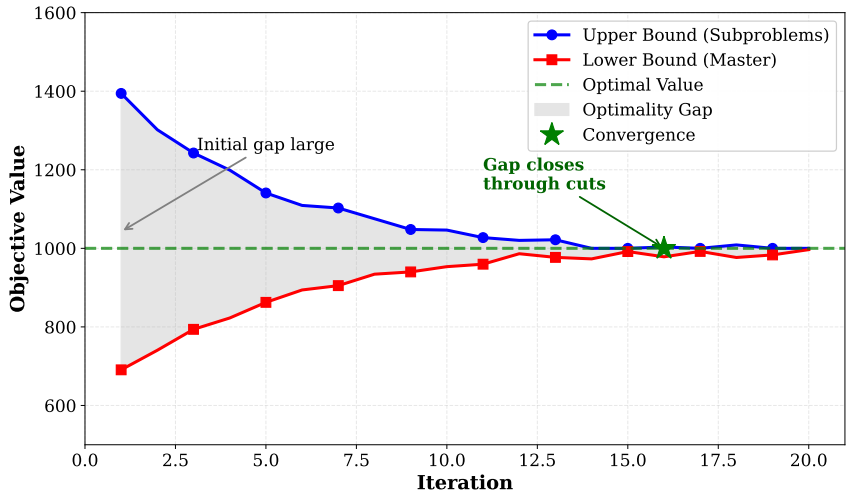


Figure 3.4: Benders decomposition convergence illustration. The algorithm iteratively refines upper bounds from subproblem evaluation and lower bounds from the master problem with accumulated cuts. Convergence occurs when the bounds meet, guaranteeing optimality.

Initially, the lower bound approximations are crude or absent. Given a candidate first-stage solution x^k from the master problem, we solve scenario subproblems in parallel to compute $Q(x^k, \xi^s)$ for each scenario $s = 1, \ldots, S$. These subproblems are independent and can be solved simultaneously, providing excellent parallel scalability. Solving the subproblems yields dual variables $\lambda^{s,k}$ that characterize how first-stage decisions affect second-stage costs.

Using these dual variables, we generate an optimality cut, a linear lower bound on the recourse function:

$$\theta \geq \sum_{s=1}^{S} p^s (\lambda^{s,k})^\top (h^s - T^s x) \tag{3.13}$$

Convergence is guaranteed when fixing the first-stage decisions yields a continuous linear subproblem for which strong duality holds [46]. Figure 3.4 illustrates the convergence behavior.

Several acceleration techniques improve practical performance. Cut management strategies remove inactive cuts that no longer affect the master problem solution, reducing problem size [75]. Multi-cut formulations generate separate cuts for each scenario rather than aggregating them, providing tighter

bounds at the cost of more master problem constraints.

Nested Benders Decomposition

For multi-stage stochastic programs where decisions and uncertainty revelations alternate over multiple time periods, Benders decomposition can be applied recursively at each stage. This Nested Benders Decomposition exploits the temporal decomposability of the problem, breaking it into a sequence of linked two-stage problems [26], [46]. At each stage, a cost-to-go function approximates the expected future cost from that point onward, and the algorithm iteratively refines this approximation through alternating forward and backward passes over the scenario tree. Paper II employs Nested Benders Decomposition to solve multi-stage E-AMoD rebalancing, charging and V2G problems over planning horizons of hours.

Forward pass. The forward pass traverses the scenario tree from the root to the leaves, simulating a candidate policy. Starting from the root node with a known initial state x_0, the algorithm solves each stage-t subproblem using $\hat{V}_{t+1}(\cdot)$, a piecewise-linear lower approximation of the true cost-to-go function built from Benders cuts accumulated in previous iterations, as a surrogate for future costs. At each node n at stage t, the subproblem is:

$$\underset{x_n}{\text{minimize}} \quad c_t^\top x_n + \hat{V}_{t+1}(x_n) \quad \text{subject to} \quad A_t x_n + B_t x_{a(n)} \leq b_n, \quad x_n \in \mathcal{X}_t \tag{3.14}$$

The optimal solution x_n^* is passed forward to all children of n, propagating the state through the tree until leaf nodes are reached. The forward pass thus produces a complete scenario-feasible policy and an upper bound on the true optimal cost.

Backward pass. The backward pass traverses the tree in reverse, from the leaves back to the root, generating Benders cuts that tighten the cost-to-go approximations. At each node n at stage t, the subproblem is solved with the state $x_{a(n)}^*$ fixed from the forward pass. This yields dual variables λ_n^* associated with the linking constraint $B_t x_{a(n)} \leq b_n - A_t x_n$. These duals characterize how the optimal cost at node n changes as a function of the incoming state, enabling construction of a supporting hyperplane:

$$V_t(x) \geq \alpha_n + (\lambda_n^*)^\top x \tag{3.15}$$

This cut is an optimality cut for the cost-to-go function V_t at the parent of n. All cuts generated across the children of a node are aggregated (weighted by their conditional probabilities) to produce a cut for the parent's cost-to-go approximation, and the process continues backward to the root. After the backward pass, every stage has a tighter piecewise-linear lower approximation of its cost-to-go function.

The algorithm iterates forward and backward passes until the gap between the upper bound (from the forward pass) and the lower bound (from the root node's cost-to-go approximation) falls below a tolerance δ. Algorithm 1 summarizes the procedure.

The optimization frameworks developed in this chapter, robust counterparts, chance constraints, stochastic programs, and Benders decomposition, share a common prerequisite: they require either uncertainty sets, probability distributions, or scenario trees as inputs. The quality of these inputs directly determines the quality of the resulting decisions. A robust counterpart is only as good as its uncertainty set, and a stochastic program is only as good as its scenario tree. Chapter 4 addresses this prerequisite directly, presenting the machine learning methods used in Papers I and II to construct these uncertainty representations from data. Gaussian process regression provides predictive distributions that seed the chance constraints in Paper I, while Bayesian neural networks generate the scenario fans that, after k-means clustering, form the scenario trees used in the Nested Benders formulation of Paper II.

Input: Scenario tree with nodes $\mathcal{N}$, root n_0, leaves $\mathcal{L}$, stages T, tolerance δ
Output: Optimal first-stage decision $x^*_{n_0}$
$UB \leftarrow +\infty$,
$LB \leftarrow -\infty$
$\hat{V}_{t+1}(\cdot) \leftarrow -\infty$ for all stages t
while $UB - LB > \delta$ **do**
  `/* Forward pass */`
  for $t = 1, \dots, T$ **do**
    foreach *node n at stage t* **do**
      $x^*_n \leftarrow \arg\min_{x_n \in \mathcal{X}_t} c_t^\top x_n + \hat{V}_{t+1}(x_n)$ s.t. $A_t x_n + B_t x^*_{a(n)} \le b_n$
      Record cost z^*_n
      Pass x^*_n to children of n
    end
  end
  $UB \leftarrow \sum_{n \in \mathcal{L}} p_n \sum_{\ell \in \mathrm{path}(n_0, n)} z^*_\ell$
  `/* Backward pass */`
  for $t = T, \dots, 2$ **do**
    foreach *node n at stage t* **do**
      Re-solve subproblem at n with $x^*_{a(n)}$ fixed
      Obtain dual multipliers λ^*_n for linking constraint
      $\alpha_n \leftarrow z^*_n - (\lambda^*_n)^\top B_t x^*_{a(n)}$
    end
    foreach *node m at stage $t-1$* **do**
      Add cut: $\hat{V}_t(x_m) \ge \sum_{n \in \mathcal{C}(m)} p_{n|m} \left(\alpha_n + (\lambda^*_n)^\top B_t x_m\right)$
    end
  end
  $LB \leftarrow$ optimal value of root subproblem
end
return $x^*_{n_0}$

Algorithm 1: Nested Benders Decomposition

CHAPTER 4

Machine Learning for Spatio-Temporal Prediction

Accurate prediction of future demand is fundamental to effective E-AMoD operations. Fleet positioning, rebalancing strategies, and charging decisions all rely on forecasts of where and when customers will request rides. However, demand in urban mobility systems exhibits complex spatio-temporal patterns: morning commutes flow from residential to business districts, evening patterns reverse this flow, weekends show different behaviors than weekdays, and special events create sudden demand surges [23], [24]. Capturing these intricate patterns while quantifying prediction uncertainty is essential for robust operational planning.

This chapter presents three complementary approaches to spatio-temporal prediction that support the optimization and learning frameworks developed in previous chapters. Gaussian Process Regression provide a principled Bayesian framework for learning smooth spatial and temporal patterns while naturally quantifying uncertainty through predictive distributions. Bayesian Neural Networks extend neural networks to capture uncertainty in learned representations, enabling more flexible function approximation while maintaining probabilistic predictions. Scenario Trees provide discrete representations of uncertainty evolution over time, essential for multi-stage stochastic optimiza-

tion. These prediction methods serve distinct but complementary roles in E-AMoD decision-making, and we emphasize throughout the connection between prediction and optimization under uncertainty.

4.1 Gaussian Process Regression

Gaussian Process Regression is a non-parametric Bayesian approach to function approximation that has proven highly effective for spatio-temporal prediction tasks [76]. Rather than assuming a specific parametric form, GPR defines a distribution over functions, allowing data to inform the shape of the learned function while naturally quantifying uncertainty through posterior distributions.

Fundamentals and Kernel Design

A Gaussian process is a collection of random variables, any finite subset of which follows a joint Gaussian distribution [76]. A function $f : \mathcal{X} \to \mathbb{R}$ follows a Gaussian process, written $f \sim \mathcal{GP}(m(x), k(x, x'))$, when it is completely specified by its mean function $m(x) = \mathbb{E}[f(x)]$ and covariance function $k(x, x') = \mathbb{E}[(f(x) - m(x))(f(x') - m(x'))]$. The kernel $k(x, x')$ measures similarity between inputs, determining how function values at nearby points influence each other. This is the key design choice in Gaussian process modeling.

For spatio-temporal demand prediction in E-AMoD systems, the kernel must capture both spatial smoothness, where nearby locations have similar demand, and temporal patterns reflecting time-of-day and day-of-week effects [23]. The squared exponential kernel $k_{\text{SE}}(x, x') = \sigma^2 \exp(-\|x-x'\|^2/2\ell^2)$ assumes smooth, infinitely differentiable functions controlled by variance σ^2 and length-scale ℓ. Small length-scales allow rapid variation while large length-scales enforce smooth functions. The Matérn kernel provides more flexibility through smoothness parameter ν, with $\nu = 3/2$ or $5/2$ often working well for mobility demand that exhibits local smoothness but not extreme smoothness.

Temporal patterns exhibit periodic structure superimposed on longer-term trends [77]. The periodic kernel $k_{\text{Per}}(t, t') = \sigma^2 \exp(-2\sin^2(\pi|t - t'|/p)/\ell^2)$ captures repeating patterns with period p, such as daily cycles with $p = 24$ hours. However, exact periodicity is unrealistic, motivating locally periodic

kernels $k_{\text{Quasi-Per}} = k_{\text{SE}} \times k_{\text{Per}}$ that allow periodic patterns whose amplitude and shape vary over time.

For joint spatio-temporal prediction, separable kernels $k((x,t),(x',t')) = k_{\text{space}}(x,x') \times k_{\text{time}}(t,t')$ assume independence between spatial and temporal patterns. While computationally convenient, this assumption is often violated since morning demand concentrates in business districts while evening demand shifts to residential areas. Nevertheless, separable kernels combining spatial Matérn and temporal quasi-periodic components provide a practical compromise, balancing expressiveness with computational tractability for datasets with thousands of observations.

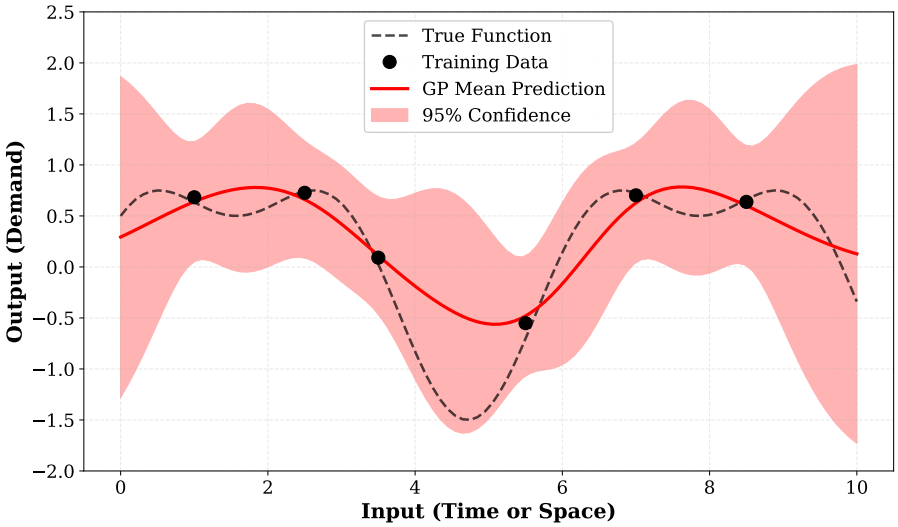


Figure 4.1: Gaussian Process predictive distribution illustration. Training observations (blue dots) inform the posterior mean (red line) and uncertainty bands (shaded regions). Uncertainty increases in regions far from training data and decreases near observations. The true underlying function (dashed black) lies within confidence intervals.

Inference and Prediction

Given training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ where $y_i = f(x_i) + \epsilon_i$ with Gaussian noise $\epsilon_i \sim \mathcal{N}(0, \sigma_n^2)$, we seek to predict $f(x_*)$ at test point x_*. Under the GP prior $f \sim \mathcal{GP}(m, k)$, the joint distribution of training outputs $\mathbf{y}$ and test output f_* is jointly Gaussian. Applying standard Gaussian conditioning yields

the posterior distribution:

$$f_*|\mathcal{D}, x_* \sim \mathcal{N}(\mu_*, \sigma_*^2) \tag{4.1}$$

$$\mu_* = m(x_*) + \mathbf{k}_*^\top (K + \sigma_n^2 I)^{-1}(\mathbf{y} - \mathbf{m}) \tag{4.2}$$

$$\sigma_*^2 = k(x_*, x_*) - \mathbf{k}_*^\top (K + \sigma_n^2 I)^{-1}\mathbf{k}_* \tag{4.3}$$

where K is the kernel matrix with $K_{ij} = k(x_i, x_j)$ and $\mathbf{k}_*$ is the vector of covariances between test and training points. The posterior mean provides the predictive estimate, and the posterior variance quantifies prediction uncertainty. Uncertainty is high far from training data and low near observations, providing automatic uncertainty quantification without manual tuning. Figure 4.1 illustrates this behavior.

Computing the posterior requires inverting the $n \times n$ matrix $(K + \sigma_n^2 I)$, costing $O(n^3)$ time and $O(n^2)$ memory. For large datasets with $n > 10,000$, this becomes prohibitive. Sparse approximations reduce complexity by approximating the full GP with a smaller set of $m \ll n$ inducing points, reducing complexity to $O(nm^2)$. Sparse Variational Gaussian Processes use variational inference to optimize inducing point locations and variational parameters jointly, enabling scalability to millions of data points through stochastic optimization [78].

Kernel hyperparameters critically affect prediction quality but are typically unknown. These are learned by maximizing the marginal likelihood $p(\mathbf{y}|X, \theta)$ where θ denotes all hyperparameters. The log marginal likelihood balances data fit against model complexity, automatically implementing Occam's razor [76]. Hyperparameters are optimized using gradient-based methods with gradients computed via automatic differentiation. The learned hyperparameters provide interpretable insights: spatial length-scales reveal demand correlation ranges, temporal periods identify dominant cycles, and noise variance indicates data quality. Figure 4.2 illustrates how length-scale selection affects predictions.

Application to E-AMoD

Gaussian processes have been successfully applied to mobility demand prediction [23], [79]. A typical workflow begins with data preparation, aggregating trip data into spatio-temporal grids such as one-kilometer cells at fifteen-minute intervals. Temporal features including hour, day-of-week, and holiday

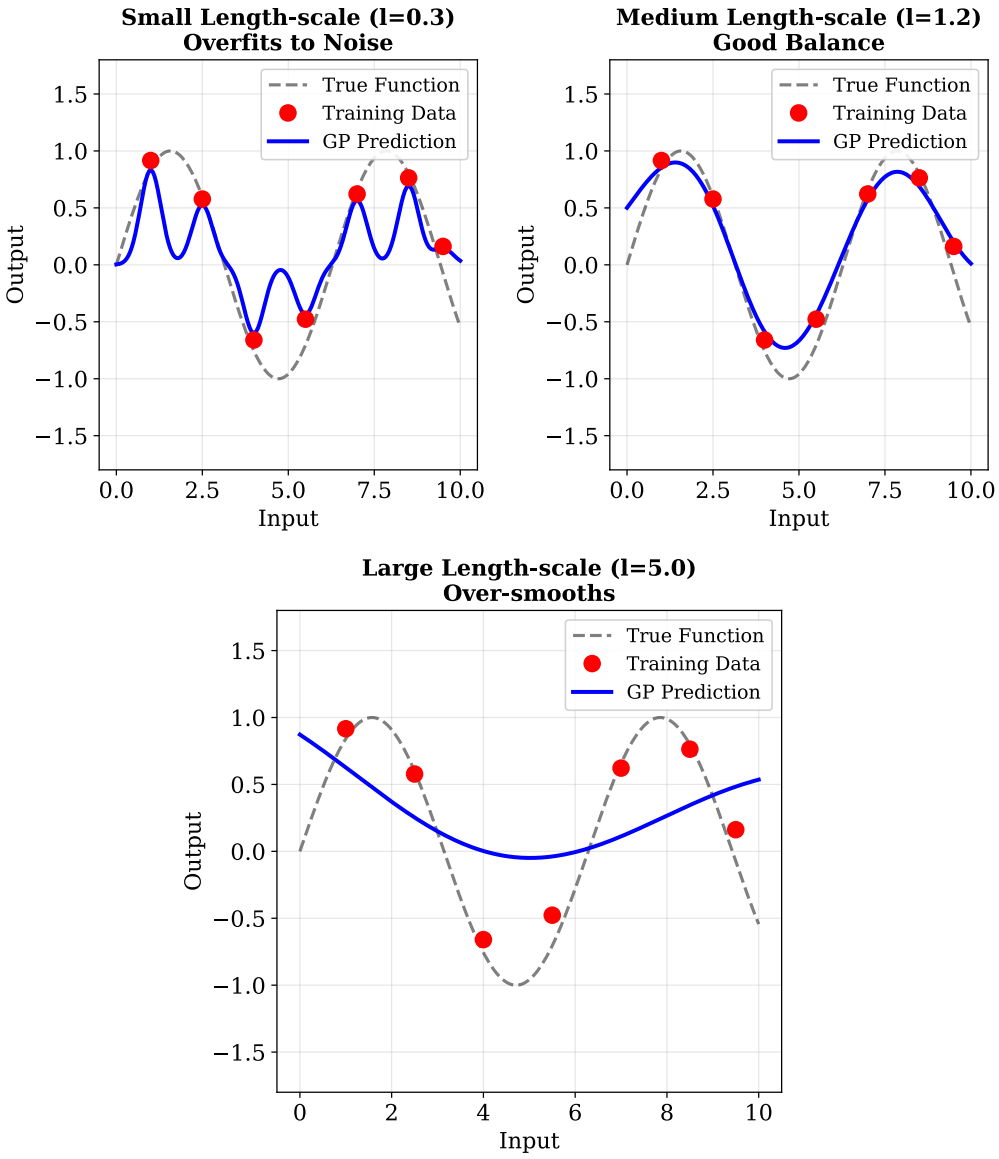


Figure 4.2: Effect of kernel length-scale on GP predictions. Small length-scales (left) allow rapid variation, potentially overfitting noise. Large length-scales (right) enforce excessive smoothness, potentially underfitting patterns. Intermediate length-scales (center) balance flexibility and smoothness, learned through marginal likelihood optimization.

indicators are extracted alongside spatial features like coordinates and point-of-interest density. A spatio-temporal kernel combining Matérn-QuasiPeriodic components is selected and hyperparameters are optimized via marginal likelihood maximization on training data. The trained GP then computes posterior mean and variance at test locations and times, with the predictive distribution used to generate demand scenarios for stochastic optimization as discussed in Section 4.3.

The advantages for E-AMoD are substantial. Predictive variance enables risk-aware rebalancing as developed in Paper I. Spatial interpolation allows prediction at locations without historical data. Few-shot learning enables good performance with limited training data, useful for new service areas. Learned length-scales and periods provide operational insights. However, computational cost limits scalability without sparse approximations, Gaussian likelihood assumptions may not capture heavy-tailed demand spikes, and stationarity assumptions with fixed hyperparameters may not hold across long time periods. These limitations motivate complementary approaches.

Neural Network Fundamentals

Neural networks (NNs) are parameterized functions composed of layers of simple computational units, loosely inspired by biological neural networks [80]. Each layer contains multiple nodes, called artificial neurons, and processes information sequentially from input to output, as illustrated in Figure 4.3.

Each neuron i in layer ℓ receives the activations $a^{(\ell-1)}$ of the previous layer and produces a scalar activation:

$$a_i^{(\ell)} = \sigma\left(\sum_j w_{ij}^{(\ell)}\, a_j^{(\ell-1)} + b_i^{(\ell)}\right), \tag{4.4}$$

where $w_{ij}^{(\ell)}$ and $b_i^{(\ell)}$ are the weight and bias of neuron i, collectively denoted as parameters θ, and σ is a nonlinear activation function such as the rectified linear unit $\sigma(z) = \max(0, z)$. The full network with L layers implements a function:

$$f(x;\theta) = a^{(L)}\Big(\cdots a^{(1)}(x)\cdots\Big), \tag{4.5}$$

where $a^{(0)}(x) = x$ is the input. This hierarchical composition allows deep networks to learn increasingly abstract representations of the input. Param-

eters θ are optimized by minimizing a loss function via gradient descent with backpropagation [80].

Neural networks are universal function approximators: with sufficient capacity, they can represent any continuous function [81]. In practice, they excel at capturing complex nonlinear patterns in high-dimensional data such as urban demand fields. However, standard training produces a single point estimate $\hat{\theta}$, yielding only point predictions $f(x;\hat{\theta})$ with no measure of uncertainty. For E-AMoD demand forecasting, where decisions under uncertainty require full predictive distributions, this is insufficient. Bayesian Neural Networks address this limitation by treating the weights θ as random variables rather than fixed parameters.

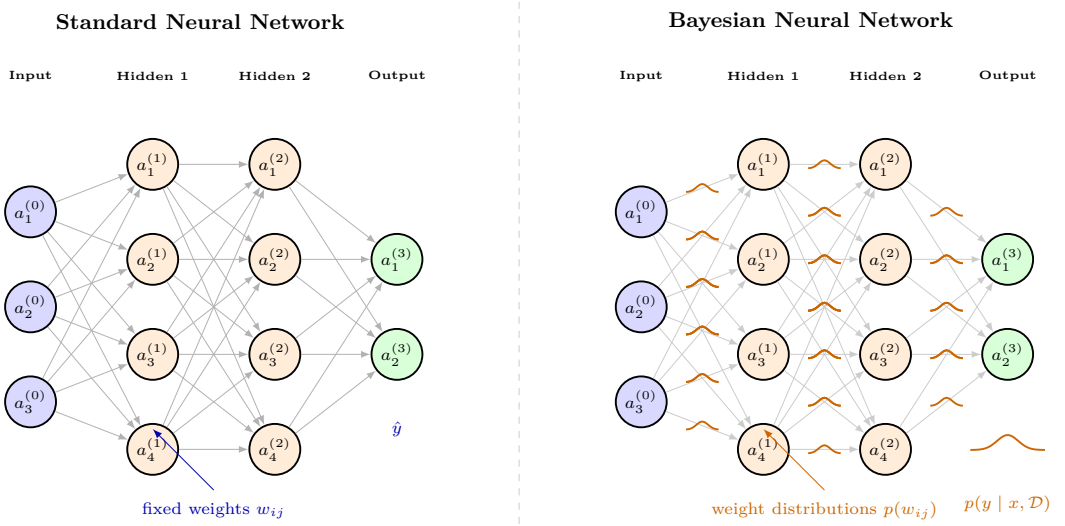


Figure 4.3: Comparison of a standard neural network (left) and a Bayesian Neural Network (right), both with two hidden layers. In a standard network, each connection carries a fixed scalar weight w_{ij} learned during training, yielding a single point prediction $\hat{y}$. In a BNN, weights are replaced by probability distributions $p(w_{ij})$ (orange curves), and the output is a full predictive distribution $p(y \mid x, \mathcal{D})$ that captures both model uncertainty about the weights and observation uncertainty in the data.

4.2 Bayesian Neural Networks

While Gaussian processes provide principled uncertainty quantification, they scale poorly to high-dimensional inputs and large datasets. Neural networks excel at learning complex patterns from massive data but typically provide only point predictions without uncertainty estimates. Bayesian Neural Networks (BNN) combine the flexibility of neural networks with probabilistic reasoning, learning distributions over network weights rather than single point estimates [82], [83].

Bayesian Framework and Approximate Inference

Consider a neural network $f(x; w)$ with weights $w \in \mathbb{R}^d$. Bayesian learning treats weights as random variables with a prior distribution $p(w)$ and computes the posterior distribution given data $\mathcal{D}$ via Bayes' rule: $p(w|\mathcal{D}) \propto p(\mathcal{D}|w)p(w)$. Predictions integrate over the posterior: $p(y_*|x_*, \mathcal{D}) = \int p(y_*|x_*, w)p(w|\mathcal{D})dw$. This predictive distribution captures two sources of uncertainty [83]. The first is model uncertainty, which reflects how confident the network is in its own weights. This type of uncertainty decreases as more training data becomes available. The second is observation noise, which reflects the inherent randomness in the data that cannot be reduced regardless of how much data is collected. For E-AMoD demand forecasting, model uncertainty is high in areas with sparse historical trip data, while observation noise captures the fundamental unpredictability of urban demand.

The posterior is intractable for neural networks due to non-conjugate, high-dimensional, non-linear relationships between weights and outputs. Variational inference approximates the intractable posterior with a simpler distribution $q_\theta(w)$ parameterized by θ, chosen to minimize $\mathcal{KL}$ divergence from the true posterior. This is equivalent to maximizing the evidence lower bound: $\mathcal{L}(\theta) = \mathbb{E}_{w\sim q_\theta}[\log p(\mathcal{D}|w)] - \mathcal{KL}(q_\theta(w)\|p(w))$. Using a factorized Gaussian variational distribution $q_\theta(w) = \prod_{j=1}^{d} \mathcal{N}(w_j; \mu_j, \sigma_j^2)$, we optimize variational parameters via stochastic gradient ascent using the reparameterization trick. This approach, called Bayes by Backprop [84], enables training BNNs with standard deep learning infrastructure.

A simpler approximation uses Monte Carlo (MC) Dropout, where dropout regularization during training approximates variational inference [82]. At test time, instead of disabling dropout, we keep it active and perform multiple

forward passes with different dropout masks, obtaining samples from the approximate posterior. The mean across samples provides the point prediction while the variance estimates uncertainty. MC Dropout requires no architectural changes, works with pre-trained networks, and adds minimal computational overhead. For E-AMoD demand forecasting, MC Dropout provides practical uncertainty estimates with standard deep learning architectures.

Ensemble methods provide an alternative, training multiple networks independently with different random initializations [85]. Predictions average across ensemble members and variance across predictions estimates uncertainty. While not strictly Bayesian, ensembles provide reliable uncertainty estimates in practice and are widely used.

Application to E-AMoD via Bayesian Neural Fields

While the methods described above apply Bayesian inference to general-purpose neural networks, they do not exploit the continuous spatiotemporal structure of E-AMoD demand data. A standard BNN takes arbitrary inputs and produces point or distributional outputs, but has no built-in notion of spatial location or temporal continuity, meaning it cannot interpolate predictions to unobserved locations or extrapolate smoothly in time. Paper II employs a specific instantiation called the Bayesian Neural Field (BayesNF), introduced by Saad et al. [86], which addresses these limitations by defining a probabilistic field over continuous space-time coordinates directly. Rather than taking demand observations as a fixed-dimensional vector, BayesNF takes the coordinates $(\mathbf{s}, t)$ themselves as input, enabling prediction at any novel location or time. Sinusoidal Fourier features appended to the inputs further allow the model to capture both slow demand trends and high-frequency periodic patterns such as daily commuting cycles, without manual kernel design. These properties make BayesNF particularly well-suited to E-AMoD demand forecasting, where demand must be predicted across a continuous urban area at irregular query locations and future time horizons.

BayesNF defines a probabilistic field $Y(\mathbf{s}, t)$ over continuous space-time coordinates $(\mathbf{s}, t) \in \mathbb{R}^d \times \mathbb{R}$, combining a deep neural network for high-capacity function approximation with hierarchical Bayesian inference for uncertainty quantification [86]. The network maps spatiotemporal coordinates directly to a latent field $F(\mathbf{s}, t)$, which parameterises an observation distribution $Y(\mathbf{s}, t) \sim \text{Dist}(F(\mathbf{s}, t), \Theta_y)$. All network weights are assigned prior distributions and

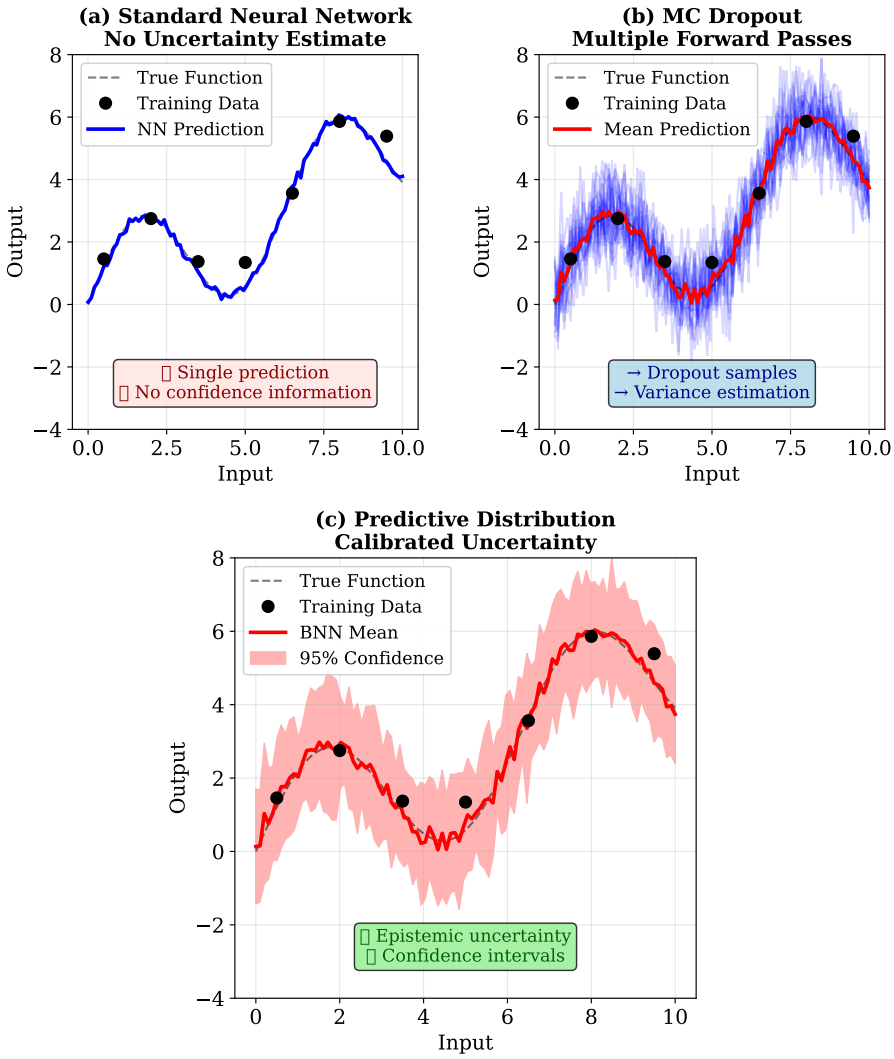


Figure 4.4: Bayesian Neural Network uncertainty quantification. A standard neural network (left) provides only point predictions. MC Dropout (center) generates multiple predictions from different dropout masks, with variance capturing model uncertainty. The predictive distribution (right) combines model uncertainty with observation noise, providing calibrated confidence intervals.

inferred via either maximum a-posteriori ensembles or variational inference, producing a posterior predictive distribution at any query location and time. The computational cost scales linearly with the number of observations rather than cubically, enabling application to the large urban datasets encountered in metropolitan E-AMoD systems.

In Paper II, BayesNF is trained on historical trip demand data across the service area. At each planning stage, the posterior predictive distribution $p(\tilde{d} \mid \mathcal{D})$ over future demand $\tilde{d}$ is queried at each zone and time horizon. Scenario fans are generated by sampling from this distribution, and the samples are subsequently clustered using k-means to determine the branching structure and assign scenario probabilities, as described in Section 4.3. The resulting scenario tree is passed directly to the Nested Benders formulation of Chapter 3, closing the loop between probabilistic prediction and optimization under uncertainty.

4.3 Scenario Trees

Stochastic optimization problems involve uncertainty that evolves over time, represented by continuous stochastic processes. Scenario trees offer a tractable discrete approximation of this uncertainty, discretizing continuous probability distributions into finite sets of scenarios that branch over time stages [61], [62]. Properly constructed scenario trees preserve key statistical properties of the underlying stochastic process while remaining computationally tractable for optimization. It has been demonstrated that increasing the number of scenarios improves the accuracy of the uncertainty representation, but also increases the computational complexity of the optimization problem [38]. Scenario reduction techniques address this tension by constructing a reduced tree that closely approximates the original one.

Structure and Properties

A scenario tree is a rooted tree where nodes represent possible states at different time stages and edges represent transitions between states. The root node represents the deterministic initial state known with certainty. At each node, multiple child nodes represent possible future realizations through branching. Leaf nodes represent terminal states at the planning horizon. Each node has

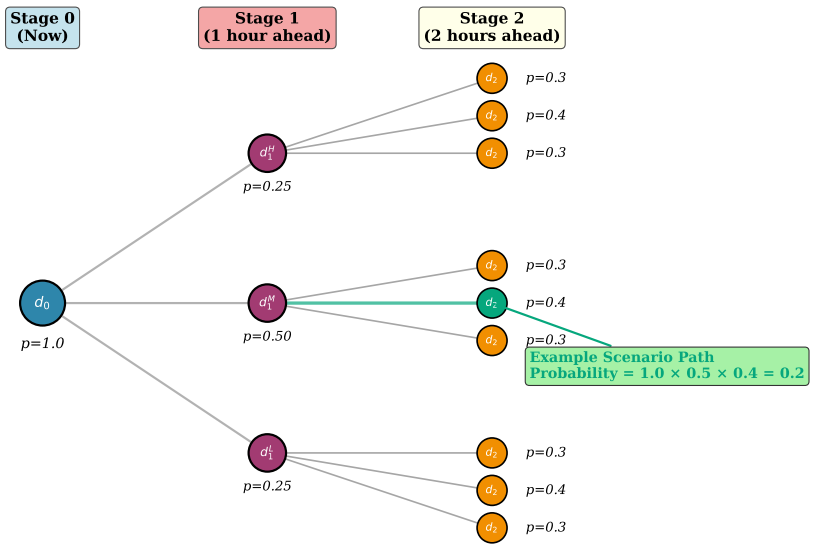


Figure 4.5: Scenario tree structure for E-AMoD demand forecasting. The root node represents current known demand. Stage 1 branches into three possibilities (low, medium, high demand) with associated probabilities. Each Stage 1 node further branches into three Stage 2 outcomes, creating nine leaf nodes and nine complete scenarios. One example scenario path is highlighted, showing the evolution from current state through intermediate demand realization to final outcome.

a stage indicating its time period, a state representing realized uncertain parameters such as demand vectors, a probability of reaching this node, and a conditional transition probability from its parent. A scenario is a path from root to leaf, representing one possible evolution of uncertainty over time, with scenario probability equal to the product of transition probabilities along its path.

An effective scenario tree for optimization should preserve low-order moments of the original distribution at each stage, minimize a suitable measure of distance between tree and true distribution, respect time-series correlations including auto-correlation and spatial cross-correlation, and remain computationally tractable by limiting tree size to enable optimization within time constraints. Figure 4.5 illustrates a typical tree structure.

Generation and Reduction Methods

A prevalent methodology for scenario tree construction is moment matching, whereby the reduced tree is constructed to preserve the first few statistical moments of the original stochastic process, such as mean, variance, skewness, and kurtosis [87]. This technique has been employed in several studies to generate scenario trees by matching the first four moments [88], [89]. However, it has been shown that two probability distributions with matching moments can still be significantly different [90].

An alternative metric is the Wasserstein distance, which quantifies the similarity between two probability distributions and has been used for scenario reduction [91]. However, the Wasserstein distance does not consider the filtration structure of the scenario tree, meaning it does not account for the temporal correlations between different stages. To address this, the Nested distance, also referred to as process distance, was introduced by Pflug and Pichler [92], explicitly accounting for tree structure and temporal dependencies. However, finding an optimally reduced scenario tree that minimizes the Nested distance is computationally challenging [93]. Paper II therefore employs multistage k-means clustering, which groups similar scenarios at each stage and replaces each cluster with its centroid, assigning branch probabilities proportional to cluster size [94]. This approach provides a practical balance between distributional accuracy and computational tractability, making it well-suited to the real-time requirements of E-AMoD rebalancing.

Application to E-AMoD

For E-AMoD rebalancing with uncertainty, the workflow in Paper II begins with forecasting demand distributions at each location and time using BayesNF, as described in Section 4.2. Scenario fans are sampled from the posterior predictive distribution at each stage and then clustered using multistage k-means to determine the branching structure and assign scenario probabilities. The resulting tree is used in the stochastic rebalancing optimization:

$$\min_{x} \; c^\top x + \mathbb{E}_{\text{tree}}[Q(x, \tilde{d})] \tag{4.6}$$

where x is the first-stage rebalancing decision and $Q(x, \tilde{d})$ is the recourse cost under demand realization $\tilde{d}$. The first-stage decision is executed, actual demand is observed, and the problem is re-solved in a rolling horizon fashion.

The prediction methods developed in this chapter, namely Gaussian processes, Bayesian Neural Fields, and scenario trees, share a common paradigm: uncertainty is first quantified through probabilistic forecasting, then propagated into optimization via scenario trees or chance constraints. This predict-then-optimize pipeline is well-suited to fleet rebalancing and charging decisions, where the planning horizon is long enough to warrant explicit uncertainty modeling. However, not all E-AMoD subproblems fit this paradigm. The electric dial-a-ride routing problem addressed in Paper III requires making thousands of sequential assignment decisions in real time as customer requests arrive continuously, a setting where the computational overhead of solving a stochastic program at each step is prohibitive. Chapter 5 introduces a fundamentally different approach to this challenge: rather than solving an optimization problem at deployment time, a reinforcement learning agent learns a routing policy offline through interaction with a simulated environment, and then executes that policy in milliseconds at runtime.

CHAPTER 5

Deep Reinforcement Learning

Deep reinforcement learning has emerged as a transformative approach to sequential decision-making, demonstrating remarkable success in domains ranging from game playing to robotics [95], [96]. Unlike the optimization-based methods discussed in Chapter 3, which require explicit mathematical formulations and solve for optimal decisions, reinforcement learning agents learn decision-making policies through interaction with their environment. This learning-based paradigm offers several advantages for complex problems like E-AMoD operations: the ability to handle high-dimensional state spaces, computational efficiency at deployment time, and adaptability to changing conditions through continued learning.

The appeal of reinforcement learning for E-AMoD stems from the structure of the decision problem. Electric dial-a-ride routing requires making thousands of sequential decisions about which customers to serve, in what order, and when to visit charging stations. Traditional optimization methods struggle with this combinatorial complexity, requiring hours of computation for problems with hundreds of requests. Reinforcement learning agents, once trained, generate high-quality solutions in milliseconds by learning from experience rather than solving mathematical programs. This speed enables real-

time operational deployment in dynamic environments where requests arrive continuously and decisions must be made immediately.

This chapter establishes the foundations of deep reinforcement learning necessary for understanding the electric dial-a-ride contribution presented in Paper III. We begin with the fundamentals of reinforcement learning, including Markov Decision Processes and key solution concepts. We then discuss policy gradient methods, which provide the training framework for learning routing policies. Next, we introduce graph neural networks and attention mechanisms, which enable learning on structured data like transportation networks. Finally, we examine applications of deep reinforcement learning to combinatorial optimization and vehicle routing problems, positioning our E-DARP contribution within the broader literature.

5.1 Reinforcement Learning Fundamentals

Reinforcement learning addresses the problem of learning optimal behavior through trial and error [52]. An agent interacts with an environment over time, taking actions that affect the environment's state and receiving rewards that evaluate action quality. The agent's goal is to learn a policy, a mapping from states to actions, that maximizes cumulative reward over time. This framework differs fundamentally from supervised learning, where the correct action is provided for each situation. In reinforcement learning, the agent must discover which actions lead to good outcomes through experimentation, balancing exploration of new actions against exploitation of known good actions.

Markov Decision Processes

The mathematical framework for reinforcement learning is the Markov Decision Process (MDP) [52], [97]. An MDP is defined by five components: a state space $\mathcal{S}$ containing all possible situations the agent might encounter, an action space $\mathcal{A}$ containing all possible decisions, a transition function $P(s'|s,a)$ specifying the probability of reaching state s' after taking action a in state s, a reward function $R(s,a)$ quantifying the immediate value of taking action a in state s, and a discount factor $\gamma \in [0,1)$ that weights future rewards relative to immediate rewards. The Markov property assumes that the next state

depends only on the current state and action, not on the history of previous states. While this assumption may seem restrictive, it can be satisfied by including relevant historical information in the state representation.

For E-DARP applications, the state s encodes the current vehicle locations, battery states, customer requests with their pickup and delivery locations and time windows, and charging station availability. The action a specifies which request to serve next or whether to visit a charging station. The transition function captures the deterministic evolution of vehicle positions and battery levels, though it may include stochastic elements representing traffic uncertainty or customer behavior. The reward function penalizes travel distance and time while rewarding served customers, encoding the operational objectives discussed in Chapter 2.

A policy $\pi : \mathcal{S} \to \mathcal{A}$ specifies the agent's behavior by mapping each state to an action. Policies can be deterministic, always selecting the same action in a given state, or stochastic, sampling actions from a probability distribution $\pi(a|s)$. Value functions quantify how good it is to be in a particular state under a given policy. The state value function $V^\pi(s)$ represents the expected cumulative discounted reward starting from state s and following policy π:

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s \right] \tag{5.1}$$

The state-action value function, also called the Q-function, represents the expected return from taking action a in state s and thereafter following policy π:

$$Q^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a \right] \tag{5.2}$$

These value functions satisfy recursive relationships known as Bellman equations, which express the value of a state in terms of the values of successor states. The optimal value functions $V^*(s)$ and $Q^*(s, a)$ correspond to the best possible policy, and they satisfy the Bellman optimality equations. Dynamic programming algorithms like value iteration and policy iteration exploit these equations to compute optimal policies for small to moderate-sized MDPs [52]. However, these classical algorithms require complete knowledge of the transition and reward functions and become computationally intractable for large state spaces, motivating the function approximation approaches in deep reinforcement learning. Figure 5.1 illustrates the agent-environment interaction

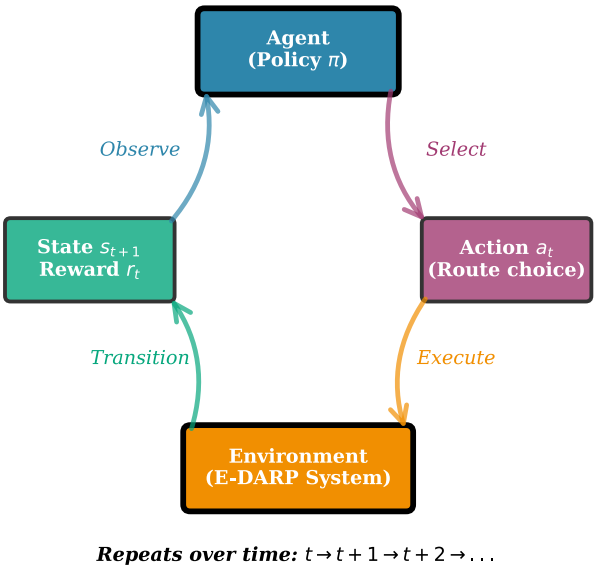


Figure 5.1: The reinforcement learning cycle. The agent observes the current state, selects an action according to its policy, receives a reward from the environment, and observes the resulting next state. This cycle repeats, with the agent learning to improve its policy based on accumulated experience. For E-DARP, states encode vehicle and customer information, actions specify routing decisions, and rewards reflect operational costs and service quality.

loop.

Function Approximation and Deep Learning

For problems with large or continuous state spaces like E-DARP, representing value functions or policies with lookup tables is infeasible. Function approximation uses parameterized functions to represent policies or values. Deep neural networks serve as powerful function approximators, capable of learning complex non-linear relationships from high-dimensional inputs [80]. A policy network $\pi_\theta(a|s)$ parameterized by weights θ maps states to action probabilities. Similarly, a value network $V_\phi(s)$ or Q-network $Q_\psi(s, a)$ approximates value functions.

Deep Q-Networks (DQN) introduced the combination of Q-learning with deep neural networks, achieving human-level performance on Atari games [95]. The key innovations included experience replay, which breaks temporal correlations in training data by sampling randomly from a buffer of past experiences, and target networks, which stabilize learning by using a slowly updated copy of the Q-network for computing target values. While DQN demonstrated the power of deep learning for reinforcement learning, Q-learning approaches face challenges in problems with large discrete action spaces or continuous action spaces, motivating policy gradient methods that directly learn policies.

5.2 Policy Gradient Methods

Policy gradient methods directly optimize the policy parameters by gradient ascent on expected return [52], [98]. Rather than learning value functions and deriving policies from them, these methods parameterize the policy explicitly and adjust parameters to increase expected cumulative reward. This direct optimization offers several advantages: it naturally handles continuous action spaces, it can learn stochastic policies that may be optimal in partially observable environments, and it often exhibits smoother convergence behavior than value-based methods.

The objective is to maximize the expected return $J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)]$ where τ denotes a trajectory, a sequence of states and actions generated by following policy π_θ. The policy gradient theorem [99] provides a fundamental result enabling gradient estimation. It shows that the gradient of expected return

with respect to policy parameters can be expressed as an expectation over trajectories:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot G_t \right] \tag{5.3}$$

where $G_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$ is the return following time t. This expectation can be approximated by sampling trajectories from the current policy and computing sample averages. The term $\nabla_\theta \log \pi_\theta(a_t|s_t)$ indicates the direction in parameter space that increases the probability of selecting action a_t in state s_t. The return G_t weights this direction: positive returns encourage the action, negative returns discourage it. This simple algorithm, known as REINFORCE [99], forms the foundation for modern policy gradient methods.

However, basic policy gradient methods suffer from high variance in gradient estimates, leading to slow and unstable learning. Several techniques reduce variance. Baselines subtract a state-dependent value estimate from returns without introducing bias: $\nabla_\theta J(\theta) = \mathbb{E}[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot (G_t - b(s_t))]$. The advantage function $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ represents how much better action a is compared to the average action in state s, providing a natural baseline. Generalized Advantage Estimation [100] provides a practical method for computing advantage estimates that trades off bias and variance through an exponentially-weighted average of temporal difference residuals.

REINFORCE and POMO Augmentation

Building on the REINFORCE algorithm introduced above, Policy Optimization with Multiple Optima (POMO) [101] augments the basic algorithm by exploiting the fact that combinatorial optimization problems such as vehicle routing have multiple optimal or near-optimal solutions that are equivalent by symmetry. Rather than sampling a single trajectory per problem instance, POMO samples N trajectories simultaneously from N different starting points, using the mean return across all trajectories as a shared baseline. This multi-start strategy simultaneously reduces gradient variance and encourages the policy to discover diverse high-quality solutions. The shared baseline is particularly effective because trajectories starting from different points on the same instance provide natural variance reduction without requiring a separate value network. Empirically, the low-variance baseline of POMO leads to fast and stable training, and greater resistance to local minima compared to

single-trajectory REINFORCE [101].

Paper III employs REINFORCE with POMO augmentation to train routing policies for E-DARP, leveraging the multi-start strategy to handle the combinatorial complexity of joint routing and charging decisions across large problem instances.

5.3 Graph Neural Networks

Transportation networks, including the road networks and customer-vehicle relationships in E-DARP, exhibit natural graph structure. Locations correspond to nodes, roads or potential routes correspond to edges, and routing decisions involve selecting paths through this graph. Standard neural network architectures like fully connected networks or convolutional networks cannot effectively exploit this structure. Graph neural networks extend deep learning to graph-structured data, enabling learning that respects and leverages graph topology [102], [103].

Graph Neural Networks (GNNs) operate through message passing, a computational paradigm where nodes iteratively exchange information with their neighbors to refine their representations [104]. A graph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. Each node v carries an initial feature vector, and the collection of these vectors forms the initial node representation $H^0 = X$. Each message passing layer executes two operations: an *aggregation* function, which consolidates information from a node's local neighborhood, and a *combine* function, which updates the node's representation by merging the aggregated neighbourhood data with its existing representation. This process is summarised in Algorithm 2.

Here, $\mathcal{N}(v)$ denotes the neighbours of node v, and the node representations from the final layer, H^K, serve as the definitive node embeddings. After K iterations, each node's representation incorporates information from its K-hop neighbourhood, capturing both local structure and global patterns. Different GNN architectures instantiate the AGGREGATE and COMBINE functions differently. Graph Convolutional Networks use mean aggregation with neural network updates [102]. GraphSAGE samples neighbours and supports mean, max, and LSTM aggregation [105]. Message Passing Neural Networks provide a general framework encompassing many GNN variants [104].

For E-DARP applications, graph structure naturally represents the routing

Input: Node features X, graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, depth K
Output: Node representations H^K
$H^0 \leftarrow X$;
for $k \leftarrow 1$ **to** K **do**
  for $v \in \mathcal{V}$ **do**
    $a_v^k \leftarrow \text{AGGREGATE}^k\big(\{H_u^{k-1} : u \in \mathcal{N}(v)\}\big)$;
    $H_v^k \leftarrow \text{COMBINE}^k\big(H_v^{k-1}, a_v^k\big)$;
  end
end
return H^K;

Algorithm 2: Framework graph neural network

problem. Nodes correspond to customer pickup and delivery locations, vehicle depots, and charging stations. Node features encode location coordinates, customer time windows, vehicle battery states, and other relevant attributes. Edges represent feasible routing connections between locations, with edge features encoding travel times, distances, and energy consumption. The GNN processes this graph to produce node embeddings that capture both local constraints, such as time window compatibility and battery feasibility, and global patterns, such as spatial clustering of customers and vehicle-customer proximity.

Graph Attention Networks

The GNN framework in Algorithm 2 leaves the AGGREGATE function unspecified. Graph Convolutional Networks treat all neighbours equally through mean aggregation, which implicitly assumes that every neighbour contributes equally to a node's representation. In many applications, however, some neighbours are more informative than others. Graph Attention Networks (GATs) [106] address this limitation by learning to assign different importance weights to different neighbours through an attention mechanism. At each layer k, GAT computes attention coefficients between node v and each neighbour $u \in \mathcal{N}(v)$:

$$e_{vu}^k = \text{LeakyReLU}\big(\mathbf{a}^\top \big[W^k H_v^{k-1} \,\|\, W^k H_u^{k-1}\big]\big) \tag{5.4}$$

where W^k is a learnable weight matrix, $\mathbf{a}$ is a learnable attention vector, and $\|$ denotes concatenation. These coefficients are normalized across neighbours using softmax to obtain attention weights:

$$\alpha_{vu}^k = \frac{\exp(e_{vu}^k)}{\sum_{u' \in \mathcal{N}(v)} \exp(e_{vu'}^k)} \tag{5.5}$$

The aggregation then becomes a weighted combination:

$$H_v^k = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{vu}^k \, W^k H_u^{k-1} \right) \tag{5.6}$$

To stabilize learning and increase representational capacity, GATs employ multi-head attention, running M independent attention heads in parallel and concatenating (or averaging) their outputs [106], [107]. This allows different heads to attend to different structural patterns in the graph.

A limitation of standard GATs is that attention coefficients depend only on node features, ignoring edge attributes. For routing problems, where edge features such as travel time, distance, and energy consumption carry critical information, this is a significant restriction. Edge-aware graph attention networks address this by incorporating edge features into the attention computation, enabling the model to weight neighbours based on both node and edge information [108]. The Graph Edge Attention Network (GREAT) architecture employed in Paper III adopts this approach, making it particularly suited to E-DARP where the relationship between locations, encoded in edge attributes, is as informative as the locations themselves.

5.4 Deep Reinforcement Learning for Combinatorial Optimization

Applying deep reinforcement learning (DRL) to combinatorial optimization has gained significant attention as an alternative to traditional exact methods and metaheuristics [109], [110]. Learned policies can generate high-quality solutions orders of magnitude faster than optimization-based methods once training is complete, enabling real-time decision-making in dynamic environments. The key challenge is designing RL formulations that respect problem constraints while efficiently exploring the vast solution space.

Pointer Networks [111] pioneered the use of attention mechanisms for learning to solve combinatorial problems, demonstrating success on the Traveling Salesman Problem (TSP) and other sequencing tasks. The key insight is that the output sequence should select from the input elements rather than from a fixed vocabulary, which attention mechanisms naturally enable. For TSP, the model reads city locations as input and outputs a sequence of pointers indicating visit order, with attention weights determining which city to select next based on already-visited cities and current location.

Subsequent work extended this framework to vehicle routing problems (VRP). The attention model for VRP [112] uses Transformer architectures to learn constructive heuristics that build solutions step by step, demonstrating performance competitive with classical operational research (OR) methods on standard benchmarks. This approach formulates routing as a sequential decision process where at each step the policy selects the next customer to visit, with the state encoding partial routes, remaining customers, and vehicle capacities. Training uses policy gradient methods, with the negative tour length serving as the reward signal.

Dynamic vehicle routing presents additional challenges because the problem instance evolves over time as new requests arrive [53]. The policy must make immediate assignment decisions for incoming requests without foreknowledge of future demand, balancing immediate service against maintaining flexibility for future requests. This dynamic setting more closely resembles real-world E-AMoD operations where demand arrives continuously and routing decisions must be made in real time. Reinforcement learning naturally accommodates this dynamic structure through its sequential decision framework, learning policies that perform well under the distribution of problem instances encountered during training. Figure 5.2 illustrates constructive routing with RL.

Paper III develops a deep reinforcement learning approach for E-DARP that combines graph attention networks with policy gradient training. The state representation uses a graph where nodes encode customer requests, vehicle states, and charging stations, with edges representing feasible routing connections. GREAT [108] processes this graph to produce embeddings capturing problem structure and constraints by explicitly incorporating edge features into the attention mechanism, which is particularly suited to routing problems where edge attributes such as travel time, distance, and energy consumption

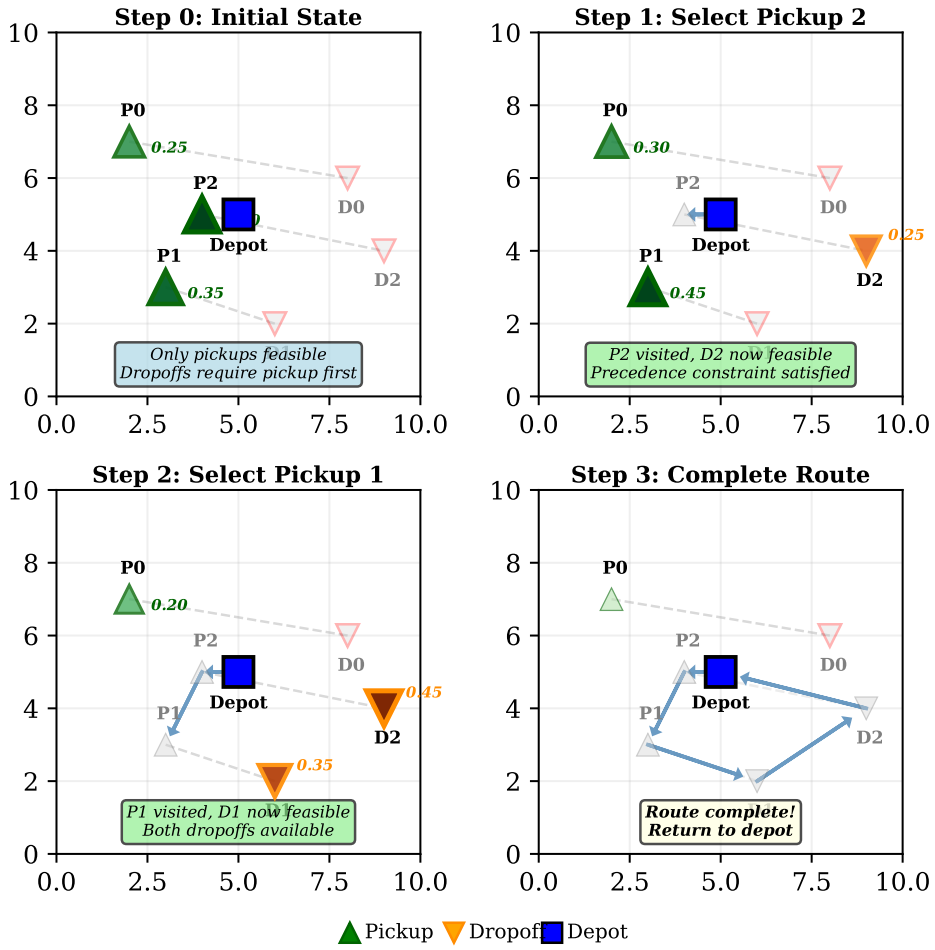


Figure 5.2: Reinforcement learning for the Electric Dial-a-Ride Problem. The agent constructs routes sequentially, selecting one node at a time. Pickups (triangles pointing up) must be visited before their corresponding dropoffs (triangles pointing down), enforced through masking. At each step, attention mechanisms compute scores over feasible nodes (indicated by color intensity), while infeasible options are grayed out. As pickups are completed, their dropoffs become available. The policy learns to select nodes that minimize total cost while respecting precedence, capacity, and time window constraints.

are as informative as node attributes. The policy network uses these embeddings to compute action probabilities over feasible next moves, with actions representing decisions about which customer to serve or whether to visit a charging station.

Constraint Handling

Handling hard constraints in reinforcement learning for combinatorial optimization is crucial, as violating constraints like time windows or vehicle capacity renders solutions infeasible. Three primary approaches address this challenge. Action masking prevents the policy from selecting infeasible actions by masking their probabilities to zero during action selection [53]. At each decision point, the environment determines which actions would violate constraints and masks them out, ensuring the policy only selects among feasible options. This approach guarantees constraint satisfaction but requires the environment to track feasibility, which can be computationally expensive for complex constraints.

Penalty shaping incorporates constraint violations into the reward signal as negative penalties [52]. The reward function includes large penalties for infeasible actions, encouraging the policy to learn to avoid them. However, this approach does not guarantee constraint satisfaction during deployment, as the policy may still select infeasible actions if the penalties are insufficient or if rare constraint violations are not adequately represented in training data. Feasibility repair uses learning to generate potentially infeasible solutions which are then repaired through post-processing heuristics [48]. This approach can explore a broader solution space during learning but requires effective repair procedures and may lead to inefficient solutions if the learning phase produces solutions that require extensive repair.

Paper III employs action masking to ensure hard constraints are never violated, combined with reward shaping that provides gradient information about constraint tightness. Time window violations are prevented through masking, while battery constraints are handled by including charging station visits as actions and masking infeasible assignments that would deplete battery before reaching a charger. This combination ensures feasibility while enabling effective learning.

Curriculum Learning

Training policies for complex combinatorial problems from scratch is challenging because the agent must simultaneously learn problem structure, constraint satisfaction, and optimization objectives. Curriculum learning addresses this by gradually increasing problem difficulty throughout training [113]. The core idea is that starting with simple problem instances where good solutions are easy to discover allows the agent to build foundational skills before encountering harder instances. As the policy improves, problem size and complexity increase progressively, introducing more customers, tighter constraints, and more challenging spatial distributions. This staged approach significantly accelerates training and improves final performance compared to training directly on complex instances, since the agent can build on previously learned behaviors rather than exploring randomly in a vast action space.

CHAPTER 6

Summary of included papers

This chapter provides a summary of the included papers.

6.1 Paper A

Sten Elling Tingstad Jacobsen, Anders Lindman, Balázs Kulcsár
A Predictive Chance-Constrained Rebalancing Approach to Mobility-on-Demand Services
Published in Communications in Transportation Research,
vol. 4, 100109, 2023.

This paper develops a predictive chance-constrained model predictive control framework for fleet rebalancing in mobility-on-demand systems under demand uncertainty. The core innovation lies in reformulating the rebalancing problem to explicitly account for the stochastic nature of passenger demand through chance constraints that ensure fleet positioning satisfies demand with a specified probability. Using Gaussian Process Regression with a locally periodic kernel, the approach captures non-strict periodicity in travel demand

patterns while providing uncertainty bounds that are integrated into the optimization through careful constraint reformulation. The framework employs a hierarchical architecture that separates strategic rebalancing decisions, solved at the station level via mixed-integer linear programming, from tactical vehicle-customer matching handled by the Hungarian algorithm. This decomposition exploits the different computational requirements and timescales of these decisions. The chance constraints are reformulated using a separable structure that enables the mixed-integer program to be relaxed to a totally unimodular linear program, ensuring computational tractability for real-time deployment. Validated in the AMoDeus simulator using San Francisco taxi data with 11,453 requests, the approach achieves a 4% reduction in median wait times compared to deterministic optimization using only mean demand predictions. Sensitivity analysis reveals that optimal confidence bounds occur at 0.65, balancing conservatism and service quality. The framework provides explicit control over the service-cost trade-off through tunable probabilistic guarantees, enabling operators to make informed decisions about operational parameters while maintaining sub-second computational performance.

- **Sten Elling Tingstad Jacobsen**: Conceptualization, methodology, algorithm development, software implementation, computational experiments, and manuscript writing.
- **Anders Lindman** and **Balázs Kulcsár**: Supervision, writing—review and editing.

6.2 Paper B

> **Sten Elling Tingstad Jacobsen**, Balázs Kulcsár, Anders Lindman
> Combined Stochastic and Robust Optimization for Electric Autonomous Mobility-on-Demand with Nested Benders Decomposition
> *Revised manuscript under review in Special Issue on Innovations for Operation and Pricing of Public Mobility Services, Elsevier Transportation Research Part C: Emerging Technologies*,
> arXiv preprint arXiv:2508.19933v2.

This paper presents an integrated optimization framework that combines stochastic programming with robust optimization to handle multiple inter-

acting uncertainties in electric autonomous mobility-on-demand operations. The methodological innovation lies in naturally separating stochastic and robust elements based on the character of different uncertainties. Passenger demand and charging station availability, which exhibit learnable patterns from historical data, are modeled stochastically through scenario trees generated from Bayesian Neural Network predictions. Operational uncertainties such as energy consumption and travel times, better characterized by bounds than precise distributions, are handled robustly through chance constraints with specified violation probabilities. This hybrid approach captures the heterogeneous nature of uncertainties while avoiding the excessive conservatism of pure robust optimization or the incomplete characterization of pure stochastic methods. To address computational intractability, the paper develops a tailored Nested Benders Decomposition algorithm that exploits the temporal and scenario tree structure of the problem. The algorithm decomposes the large-scale mixed-integer stochastic-robust program into a master problem for strategic fleet positioning and charging decisions, and parallelizable subproblems that handle tactical operations under specific uncertainty realizations. Novel valid inequalities and cut strengthening techniques accelerate convergence. Validated on realistic case studies from San Francisco and Chicago with up to 250 requests and 1035 scenarios, the approach solves problems in 355 seconds, achieving metropolitan-scale tractability where direct solution methods fail within hours. Compared to deterministic baselines, the framework reduces median wait times by 36%, rebalancing distance by 27%, and electricity costs by 35%. The model predictive control structure preserves adaptive feedback, enabling real-time correction of prediction errors at each receding horizon step. Comparative analysis demonstrates that pure stochastic optimization leads to frequent constraint violations when operational uncertainties exceed expectations, while pure robust optimization incurs 15 to 25% higher costs due to excessive buffering. The hybrid approach achieves service reliability comparable to robust optimization at costs approaching stochastic optimization.

- **Sten Elling Tingstad Jacobsen**: Conceptualization, methodology, algorithm development, software implementation, computational experiments, and manuscript writing.
- **Anders Lindman** and **Balázs Kulcsár**: Supervision, writing—review

and editing.

6.3 Paper C

Sten Elling Tingstad Jacobsen, Attila Lischka, Balázs Kulcsár, Anders Lindman
Learning to Dial-a-Ride: A Deep Graph Reinforcement Learning Approach to the Electric Dial-a-Ride Problem
Manuscript under review in Elsevier Transportation Research Part C: Emerging Technologies,
arXiv preprint arXiv:2601.22052 .

This paper studies the Electric Dial-a-Ride Problem (E-DARP), which extends classical dial-a-ride with battery capacity constraints and nonlinear charging dynamics, producing a combinatorial structure that renders real-time exact optimization intractable. To address this, a deep reinforcement learning approach is proposed, based on the Graph Edge Attention Network (GREAT) and an attention-driven route construction policy. By operating directly on edge representations, the method captures non-Euclidean, asymmetric, and energy-dependent routing costs in real road networks, while action masking enforces hard constraints throughout solution construction. The approach is evaluated on two case studies using ride-sharing data from San Francisco. On standard benchmark instances with up to 50 request pairs, the policy achieves a 0.4% optimality gap while reducing computation times by factors of up to 7,200 compared to branch-and-price and branch-and-cut methods. A second case study considers large-scale instances with up to 250 request pairs, physics-based energy models, and nonlinear charging dynamics; here, the learned policy outperforms Adaptive Large Neighborhood Search by 9.5% in solution quality, achieves 100% service completion, and infers solutions in under 10 seconds compared to hours for the metaheuristic. Sensitivity analyses examine the impact of battery capacity, fleet size, ride-sharing capacity, and reward weights, and robustness experiments confirm that deterministically trained policies generalize effectively under stochastic conditions.

- **Sten Elling Tingstad Jacobsen**: Problem formulation, overall methodology, software implementation, model training and evaluation, compu-

tational experiments, and manuscript writing.

- **Attila Lischka**: Neural architecture design, methodological input, and writing—review and editing.
- **Anders Lindman** and **Balázs Kulcsár**: Supervision and writing—review and editing.

CHAPTER 7

Conclusions and Future Work

This thesis investigates operational decision-making for Electric Autonomous Mobility-on-Demand systems under uncertainty. The central challenge addressed throughout the work is the need to make reliable large-scale decisions in environments characterized by stochastic demand, uncertain energy consumption, limited charging infrastructure, and strict operational constraints, all under severe computational time limits. By combining optimization-based and learning-based methodologies, the thesis demonstrates how uncertainty can be modeled, propagated, and acted upon in a principled manner, enabling Electric Autonomous Mobility-on-Demand systems that are both operationally efficient and practically deployable.

Rather than advocating a single modeling paradigm, the thesis adopts a layered perspective in which different decision levels are addressed using tools that are well matched to their structure, time horizon, and uncertainty characteristics. Optimization-based frameworks are shown to provide explicit guarantees, interpretability, and robustness when uncertainty must be handled formally, while learning-based approaches offer computational speed and adaptability when fine-grained operational decisions must be made in real-time. This chapter synthesizes the main insights that emerge from this perspective,

reflects on their implications for deployment, and outlines directions for future research.

7.1 Concluding Remarks

A unifying insight of this thesis is that uncertainty is not a peripheral complication in Electric Autonomous Mobility-on-Demand operations but a defining feature that fundamentally shapes model design, algorithmic choice, and system architecture. Travel demand, energy consumption, charger availability, and traffic conditions are all uncertain and often correlated, evolving over time in ways that cannot be captured adequately by deterministic planning. Effective operation therefore requires decision frameworks that do not merely react to realized outcomes but explicitly anticipate uncertainty and hedge against adverse realizations.

The thesis addressed this challenge through modeling abstractions at two complementary levels of granularity. At an aggregate, station-based level, vehicle flows and charging decisions are represented in a spatially and temporally aggregated manner. This abstraction enables optimization over extended horizons while retaining sufficient structure to incorporate probabilistic demand forecasts and electric vehicle constraints. At a finer, node-based level, individual vehicles and customer requests are modeled explicitly, capturing detailed routing constraints such as time windows, ride-time limits, precedence relations, and battery feasibility. This distinction reflects a fundamental trade-off between tractability and fidelity that recurs throughout the thesis.

At the aggregate level, the results demonstrate that uncertainty can be handled in a mathematically rigorous way without sacrificing computational efficiency. By combining probabilistic demand forecasts with chance-constrained optimization, service reliability can be enforced directly at the decision-making stage rather than evaluated only after execution. Crucially, the exploitation of network flow structure ensures that uncertainty-aware formulations remain solvable as linear programs, preserving polynomial-time complexity even for large metropolitan networks. This shows that explicit probabilistic guarantees, such as meeting demand with a specified confidence level, are not inherently incompatible with operational deployment, provided that model structure is carefully leveraged.

When multiple sources of uncertainty are considered simultaneously, scenario-

based stochastic programming emerges as a flexible and expressive framework. Representing uncertainty through scenario trees allows temporal evolution and correlation between demand, energy consumption, and charging infrastructure availability to be captured explicitly. Decomposition techniques exploit separability across time and scenarios, enabling large multi-stage problems to be solved that would otherwise be computationally infeasible. The results show that coordinated decision-making under uncertainty, particularly for charging and rebalancing, can substantially reduce operational cost while maintaining service quality, provided that uncertainty is modeled explicitly rather than absorbed into conservative safety margins.

A key strength of these optimization-based frameworks is their ability to provide formal guarantees. Chance constraints yield probabilistic assurances on service levels, while robust components ensure feasibility under adverse but plausible conditions. These guarantees are transparent, interpretable, and directly tied to model assumptions, allowing system designers and operators to reason explicitly about risk. In the context of safety-critical and publicly visible transportation systems, this capability is particularly important.

At the level of detailed operational routing, however, the thesis demonstrates that optimization-based methods face intrinsic scalability limits. As the number of vehicles and requests increases, the combinatorial structure of the dial-a-ride problem renders exact or near-exact optimization too slow for real-time use. In this regime, learning-based approaches provide a complementary solution by shifting computational effort from online decision-making to offline training. By embedding routing problems in graph-based representations and leveraging attention mechanisms, learned policies can exploit problem structure directly and generalize across diverse operational scenarios.

The principal advantage of learning-based routing lies in its responsiveness. Once trained, a policy can generate high-quality routing decisions in milliseconds, enabling dynamic ride-pooling in environments where requests arrive continuously and system state changes rapidly. However, unlike optimization-based methods, learning-based policies do not naturally provide explicit probabilistic or worst-case guarantees. Their reliability is established empirically rather than through formal constraints. This contrast highlights a central conclusion of the thesis: optimization-based frameworks are inherently better suited for handling uncertainty when guarantees are required, while reinforcement learning is most effective when speed and flexibility dominate and

empirical validation is sufficient.

Taken together, the findings support a layered architecture for Electric Autonomous Mobility-on-Demand operations. Strategic and tactical decisions, such as fleet positioning, charging scheduling, and capacity allocation, benefit from optimization-based formulations that explicitly model uncertainty and provide guarantees. Operational execution, such as real-time routing and request assignment, benefits from learning-based policies that can react instantly to changing conditions. Prediction and uncertainty quantification form the backbone of this architecture, linking data to decisions across all levels. Improvements in forecasting accuracy and uncertainty representation consistently translate into better operational outcomes, regardless of the downstream decision methodology. The proposed frameworks have been evaluated using simulation environments calibrated with real-world demand data and network characteristics, bridging the gap between algorithmic development and practical deployment. As these systems transition from research prototypes to operational infrastructure, architectures that integrate uncertainty-aware optimization with real-time learning will be essential for delivering reliable, efficient, and socially beneficial urban mobility.

7.2 Future Research Directions

Several research directions naturally follow from the contributions and limitations of this thesis.

Traffic congestion effects. All three papers assume that E-AMoD fleet operations do not affect background traffic conditions. In practice, rebalancing trips add vehicles to the road network, potentially increasing congestion and altering travel times in ways that feed back into operational decisions. Extending the optimization frameworks of Papers I and II to account for congestion effects, along the lines of Rossi et al. [114], would improve operational realism, particularly in dense urban environments where rebalancing intensity is high.

Mixed fleet configurations. The models developed in this thesis assume a homogeneous fleet of identical electric vehicles. Real-world deployments are likely to involve vehicles with different battery capacities, charging requirements, and passenger capacities. Extending the chance-constrained formulation of Paper I, SMPC in Paper II and the routing policy of Paper III to

heterogeneous fleets would significantly broaden their practical applicability.

Electricity grid interactions. Paper II coordinates charging decisions to minimize operational cost but does not model how large-scale EV charging affects electricity prices or grid stability. Jointly optimizing fleet operations and grid interactions, building on work by Rossi et al. [115], would enable more realistic cost modeling and could reduce both operational expenditure and grid stress during peak demand periods.

Competing operators. This thesis assumes a single E-AMoD operator with centralized control over the entire fleet. In markets with multiple competing operators, individual decisions affect shared infrastructure such as charging stations and road capacity. Game-theoretic extensions that model strategic interactions between operators would provide more realistic operational insights for deregulated mobility markets [116], [117].

Battery degradation. The energy models in all three papers treat battery capacity as fixed over time. In practice, repeated charging and discharging cycles degrade battery capacity, increasing long-term operational costs. As shown by Paparella et al. [118], battery degradation costs can substantially affect fleet economics and must be accounted for in joint design and operational optimization. Incorporating battery degradation models into the stochastic optimization framework of Paper II would improve long-term economic realism and enable smarter charging strategies that balance operational efficiency against battery longevity.

Dynamic demand and online learning. Paper III assumes deterministic demand arrival patterns, whereas real-world operations face uncertain request arrivals, customer no-shows, and cancellations. Extending the reinforcement learning framework to handle dynamic demand through online learning or anticipatory policies would enhance operational reliability and bring the methodology closer to real-time deployment.

Hybrid routing and charging optimization. The current framework in Paper III discretizes charging decisions by fixing charging duration at each station visit, which may contribute to the small optimality gaps observed on benchmark instances and could limit performance in scenarios requiring precise energy management. Future work could explore hybrid approaches that combine learned routing policies with optimization-based charging decisions, or action spaces that include multiple discrete charging duration options, bridging the optimization-based and learning-based paradigms developed in

this thesis.

References

[1] European Environment Agency, *Greenhouse gas emissions by source sector*, `https://ec.europa.eu/eurostat/databrowser/view/ENV_AIR_GGEDV_447/default/table?lang=en`, Accessed: Dec. 15, 2025, 2023.

[2] International Energy Agency, "Transport sector co2 emissions by mode in the sustainable development scenario, 2000-2030," International Energy Agency, Tech. Rep., 2020.

[3] Z. Zhu, Z. Li, Y. Liu, et al., "The impact of urban characteristics and residents' income on commuting in china," *Transportation Research Part D: Transport and Environment*, vol. 57, pp. 474–483, 2017.

[4] R. Godfrey and M. Julien, "Urbanisation and health," *Clinical Medicine*, vol. 5, no. 2, p. 137, 2005.

[5] United Nations Publications, *World Urbanization Prospects: The 2018 Revision*. UN, 2019, ISBN: 9789211483192.

[6] G. D. Erhardt, S. Roy, D. Cooper, B. Sana, M. Chen, and J. Castiglione, "Do transportation network companies decrease or increase congestion?" *Science Advances*, vol. 5, no. 5, eaau2670, 2019.

[7] D. J. Fagnant and K. Kockelman, "Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations," *Transportation Research Part A: Policy and Practice*, vol. 77, pp. 167–181, 2015.

[8] S. Shaheen and A. Cohen, "Shared mobility: Current practices and guiding principles," *US Department of Transportation, Federal Highway Administration*, 2016.

[9] A. Arvesen, S. Völler, C. R. Hung, V. Krey, M. Korpås, and A. H. Strømman, "Emissions of electric vehicle charging in future scenarios: The effects of time of charging," *Journal of Industrial Ecology*, vol. 25, no. 5, pp. 1250–1263, 2021.

[10] P. M. Boesch, F. Ciari, and K. W. Axhausen, "Autonomous vehicle fleet sizes required to serve different levels of demand," *Transportation Research Record*, vol. 2542, no. 1, pp. 111–119, 2016.

[11] M. Pavone, S. L. Smith, E. Frazzoli, and D. Rus, "Robotic load balancing for mobility-on-demand systems," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 839–854, 2012.

[12] K. Spieser, K. Treleaven, R. Zhang, E. Frazzoli, D. Morton, and M. Pavone, "Toward a systematic approach to the design and evaluation of automated mobility-on-demand systems: A case study in singapore," in *Road Vehicle Automation*, ser. Lecture Notes in Mobility, G. Meyer and S. Beiker, Eds., Springer, Cham, 2014, pp. 229–245.

[13] M. Schwall, T. Daniel, T. Victor, F. Favaro, and H. Hohnhold, *Waymo public road safety performance data*, 2020.

[14] K. Heineke, B. Kloss, T. Möller, and C. Wiemuth, *Shared mobility: Where it stands, where it's headed*, 2021.

[15] H. Becker et al., "Impact of vehicle automation and electric propulsion on production costs for mobility services worldwide," *Transportation Research Part A: Policy and Practice*, vol. 138, pp. 105–126, 2020.

[16] R. Zhang and M. Pavone, "Control of robotic mobility-on-demand systems: A queueing-theoretical perspective," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 186–203, 2016.

[17] F. Rossi, R. Iglesias, M. Alizadeh, and M. Pavone, "On the interaction between autonomous mobility-on-demand and public transportation systems," *IEEE Conference on Decision and Control (CDC)*, pp. 2262–2269, 2018.

[18] R. Iacobucci, B. McLellan, and T. Tezuka, "Optimization of shared autonomous electric vehicles operations with charge scheduling and vehicle-to-grid," *Transportation Research Part C: Emerging Technologies*, vol. 100, pp. 34–52, 2019.

[19] P. S. Klein and M. Schiffer, "Electric vehicle charge scheduling with flexible service operations," *Transportation Science*, vol. 57, no. 6, pp. 1605–1626, 2023.

[20] M. Hyland and H. S. Mahmassani, "Dynamic autonomous vehicle fleet operations: Optimization-based strategies to assign avs to immediate traveler demand requests," *Transportation Research Part C: Emerging Technologies*, vol. 92, pp. 278–297, 2018.

[21] M. Schneider, A. Stenger, and D. Goeke, "The electric vehicle-routing problem with time windows and recharging stations," *Transportation Science*, vol. 48, no. 4, pp. 500–520, 2014.

[22] M. Keskin and B. Çatay, "Partial recharge strategies for the electric vehicle routing problem with time windows," *Transportation Research Part C: Emerging Technologies*, vol. 65, pp. 111–127, 2016.

[23] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi-passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, 2013.

[24] J. Ke, H. Zheng, H. Yang, and X. M. Chen, "Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 85, pp. 591–608, 2017.

[25] M. Gendreau, G. Ghiani, and E. Guerriero, "Time-dependent routing problems: A review," *Computers & Operations Research*, vol. 42, pp. 1338–1355, 2015.

[26] J. R. Birge and F. Louveaux, *Introduction to stochastic programming*, 2nd. Springer, 1997.

[27] C. Ruch, S. Hörl, and E. Frazzoli, "Amodeus, a simulation-based testbed for autonomous mobility-on-demand systems," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 3639–3644.

[28] A. Horni, K. Nagel, and K. W. Axhausen, *The Multi-Agent Transport Simulation MATSim*. London, GBR: Ubiquity Press, 2016, ISBN: 1909188751.

[29] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, and D. Rus, "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment," *Proceedings of the National Academy of Sciences*, vol. 114, no. 3, pp. 462–467, 2017.

[30] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem: Models and algorithms," *Annals of Operations Research*, vol. 153, no. 1, pp. 29–46, 2007.

[31] M. Lokhandwala and H. Cai, "Dynamic ride sharing using traditional taxis and shared autonomous taxis: A case study of NYCd," *Transportation Research Part C: Emerging Technologies*, vol. 97, pp. 45–60, 2018.

[32] A. Braverman, J. Dai, X. Liu, and L. Ying, "Empty-car routing in ridesharing systems," *Operations Research*, vol. 67, 2019.

[33] C. Mao, Y. Liu, and Z.-J. Shen, "Dispatch of autonomous vehicles for taxi services: A deep reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 115, p. 102 626, 2020.

[34] R. Iglesias, F. Rossi, K. Wang, D. Hallac, J. Leskovec, and M. Pavone, "Data-driven model predictive control of autonomous mobility-on-demand systems," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 6019–6025.

[35] M. Tsao, R. Iglesias, and M. Pavone, "Stochastic model predictive control for autonomous mobility on demand," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 3941–3948.

[36] C. E. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2005, ISBN: 026218253X.

[37] D. K. George, "Stochastic modeling and decentralized control policies for large-scale vehicle sharing systems via closed queueing networks," Ph.D. dissertation, Ohio State University, 2012.

[38] F. Beltrán, E. C. Finardi, and W. de Oliveira, "Two-stage and multi-stage decompositions for the medium-term hydrothermal scheduling problem," *International Journal of Electrical Power & Energy Systems*, vol. 127, p. 106 659, 2021.

[39] D. Bertsimas and M. Sim, "The price of robustness," *Operations Research*, vol. 52, no. 1, pp. 35–53, 2004.

[40] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust optimization*. Princeton University Press, 2009.

[41] A. Charnes, W. W. Cooper, and G. H. Symonds, "Cost horizons and certainty equivalents: An approach to stochastic programming of heating oil," *Management Science*, vol. 4, no. 3, pp. 235–263, 1958.

[42] A. Prékopa, *Stochastic programming*. Springer, 1995.

[43] P. Toth and D. Vigo, *Vehicle routing: problems, methods, and applications*. SIAM, 2014.

[44] İ. Yanıkoğlu, B. L. Gorissen, and D. den Hertog, "A survey of adjustable robust optimization," *European Journal of Operational Research*, vol. 277, no. 3, pp. 799–813, 2019.

[45] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numerische Mathematik*, vol. 4, no. 1, pp. 238–252, 1962.

[46] R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei, "The benders decomposition algorithm: A literature review," *European Journal of Operational Research*, vol. 259, no. 3, pp. 801–817, 2017.

[47] J.-F. Cordeau, "A branch-and-cut algorithm for the dial-a-ride problem," *Operations Research*, vol. 54, no. 3, pp. 573–586, 2006.

[48] C. Bongiovanni, M. Kaspi, and N. Geroliminis, "The electric autonomous dial-a-ride problem," *Transportation Research Part B: Methodological*, vol. 122, pp. 436–456, 2019.

[49] M. M. Vazifeh, P. Santi, G. Resta, S. H. Strogatz, and C. Ratti, "Addressing the minimum fleet problem in on-demand urban mobility," *Nature*, vol. 557, no. 7706, pp. 534–538, 2018.

[50] S. Pelletier, O. Jabali, G. Laporte, and M. Veneroni, "Battery degradation and behaviour for electric vehicles: Review and numerical analyses of several models," *Transportation Research Part B: Methodological*, vol. 103, pp. 158–187, 2017.

[51] L. He, G. Ma, W. Qi, and X. Wang, "Charging an electric vehicle-sharing fleet," *Manufacturing & Service Operations Management*, vol. 23, no. 2, pp. 471–487, 2021.

[52] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.

[53] M. Nazari, A. Oroojlooy, L. Snyder, and M. Takáč, "Reinforcement learning for solving the vehicle routing problem," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[54] R. Zhang, F. Rossi, and M. Pavone, "Model predictive control of autonomous mobility-on-demand systems," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016, pp. 1382–1389.

[55] U. Technologies, *Uber anonymized gps logs - san francisco*, `https://github.com/dima42/uber-gps-analysis/tree/master/gpsdata`, Accessed: 2024.

[56] City of Chicago, *Taxi trips (2013 - 2023)*, `https://data.cityofchicago.org/Transportation/Taxi-Trips-2013-2023-/wrvz-psew/about_data`, Accessed: 2024-10-01, 2023.

[57] D. Bertsimas, D. B. Brown, and C. Caramanis, "Theory and applications of robust optimization," *SIAM Review*, vol. 53, no. 3, pp. 464–501, 2011.

[58] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on stochastic programming: modeling and theory*. SIAM, 2009.

[59] S. E. T. Jacobsen, A. Lindman, and B. Kulcsár, "A predictive chance constraint rebalancing approach to mobility-on-demand services," *Communications in Transportation Research*, vol. 3, p. 100 097, 2023.

[60] M. Goerigk and A. Schöbel, "Algorithm engineering in robust optimization," *Algorithm Engineering: Selected Results and Surveys*, pp. 245–279, 2016.

[61] H. Heitsch and W. Römisch, "Scenario tree modeling for multistage stochastic programs," *Mathematical Programming*, vol. 118, no. 2, pp. 371–406, 2009.

[62] G. C. Pflug, *Scenario tree generation for multiperiod financial optimization by optimal discretization*. Springer, 2001.

[63] H. Heitsch and W. Römisch, "Scenario reduction algorithms in stochastic programming," *Computational Optimization and Applications*, vol. 24, no. 2–3, pp. 187–206, 2003.

[64] G. C. Calafiore and M. C. Campi, "The scenario approach to robust control design," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 742–753, 2006.

[65] J. Luedtke and S. Ahmed, "A sample approximation approach for optimization with probabilistic constraints," *SIAM Journal on Optimization*, vol. 19, no. 2, pp. 674–699, 2008.

[66] L. A. Wolsey, *Integer Programming*. Wiley-Interscience, 1998.

[67] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, 2023.

[68] IBM, *Ibm ilog cplex optimization studio*, 2023.

[69] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Sons, 1986.

[70] G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization*. John Wiley & Sons, 1988.

[71] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*, 2nd. Nob Hill Publishing, 2017.

[72] C. E. Garcia, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice — a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[73] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems Magazine*, vol. 36, no. 6, pp. 30–44, 2016.

[74] R. M. Van Slyke and R. J.-B. Wets, "L-shaped linear programs with applications to optimal control and stochastic programming," *SIAM Journal on Applied Mathematics*, vol. 17, no. 4, pp. 638–663, 1969.

[75] W. Rei, J.-F. Cordeau, M. Gendreau, and P. Soriano, "Accelerating benders decomposition by local branching," *INFORMS Journal on Computing*, vol. 21, no. 2, pp. 333–345, 2009.

[76] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT Press, 2006, vol. 2.

[77] H. Yao, F. Wu, J. Ke, et al., "Deep multi-view spatial-temporal network for taxi demand prediction," *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[78] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for big data," in *Uncertainty in Artificial Intelligence*, 2013, pp. 282–290.

[79] J. Chen, K. H. Low, Y. Yao, and P. Jaillet, "Gaussian process decentralized data fusion and active sensing for spatiotemporal traffic modeling and prediction in mobility-on-demand systems," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 901–921, 2015.

[80] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[81] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[82] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," *International Conference on Machine Learning*, pp. 1050–1059, 2016.

[83] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[84] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *International Conference on Machine Learning*, PMLR, 2015, pp. 1613–1622.

[85] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[86] F. Saad et al., "Scalable spatiotemporal prediction with bayesian neural fields," *Nature Communications*, vol. 15, no. 1, p. 7942, 2024.

[87] K. Høyland and S. W. Wallace, "Generating scenario trees for multistage decision problems," *Management Science*, vol. 47, no. 2, pp. 295–307, 2001.

[88] A. H. Ansaripoor, F. S. Oliveira, and A. Liret, "Recursive expected conditional value at risk in the fleet renewal problem with alternative fuel vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 65, pp. 156–171, 2016.

[89] Y. Bao, K. J. Chan, A. Mesbah, and J. M. Velni, "Learning-based adaptive-scenario-tree model predictive control with improved probabilistic safety using robust bayesian neural networks," *International Journal of Robust and Nonlinear Control*, vol. 33, no. 5, pp. 3312–3333, 2023.

[90] R. Hochreiter and G. C. Pflug, "Financial scenario generation for stochastic multi-stage decision processes as facility location problems," *Annals of Operations Research*, vol. 152, pp. 257–272, 2007.

[91] J. Dupačová, N. Gröwe-Kuska, and W. Römisch, "Scenario reduction in stochastic programming," *Mathematical Programming*, vol. 95, pp. 493–511, 2003.

[92] G. C. Pflug, "Version-independence and nested distributions in multistage stochastic optimization," *SIAM Journal on Optimization*, vol. 20, no. 3, pp. 1406–1420, 2010.

[93] M. Horejšová, S. Vitali, M. Kopa, and V. Moriggia, "Evaluation of scenario reduction algorithms with nested distance," *Computational Management Science*, vol. 17, no. 2, pp. 241–275, 2020.

[94] R. M. Kovacevic and A. Pichler, "Tree approximation for discrete time stochastic processes: A process distance approach," *Annals of Operations Research*, vol. 235, no. 1, pp. 395–421, 2015.

[95] V. Mnih, K. Kavukcuoglu, D. Silver, et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[96] D. Silver, J. Schrittwieser, K. Simonyan, et al., "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[97] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[98] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[99] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3, pp. 229–256, 1992.

[100] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *International Conference on Learning Representations*, 2015.

[101] Y.-D. Kwon and J. Choo, "Pomo: Policy optimization with multiple optima for reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[102] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[103] P. W. Battaglia, J. B. Hamrick, V. Bapst, et al., "Relational inductive biases, deep learning, and graph networks," *arXiv preprint arXiv:1806.01261*, 2018.

[104] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International Conference on Machine Learning*, PMLR, 2017, pp. 1263–1272.

[105] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 1024–1034.

[106] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018.

[107] A. Vaswani, N. Shazeer, N. Parmar, et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[108] A. Lischka, F. Rydin, J. Wu, M. H. Chehreghani, and B. Kulcsár, *A great architecture for edge-based graph problems like tsp*, 2025.

[109] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: A methodological tour d'horizon," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.

[110] N. Mazyavkina, S. Sviridov, S. Ivanov, and E. Burnaev, "Reinforcement learning for combinatorial optimization: A survey," *Computers & Operations Research*, vol. 134, p. 105 400, 2021.

[111] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[112] W. Kool, H. van Hoof, and M. Welling, "Attention, learn to solve routing problems!" In *International Conference on Learning Representations (ICLR)*, 2019.

[113] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 41–48, 2009.

[114] F. Rossi, R. Zhang, A. Hindy, and M. Pavone, "Routing autonomous vehicles in congested transportation networks: Structural properties and coordination algorithms," *Autonomous Robots*, vol. 42, no. 7, pp. 1427–1442, 2018.

[115] F. Rossi, R. Iglesias, M. Alizadeh, and M. Pavone, "On the interaction between autonomous mobility-on-demand systems and the power network: Models and coordination algorithms," *IEEE Transactions on Control of Network Systems*, vol. 7, no. 1, pp. 384–397, 2019.

[116] F. Dandl and K. Bogenberger, "Autonomous mobility-on-demand real-time gaming framework," in *2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, IEEE, 2019.

[117] R. Engelhardt, P. Malcolm, F. Dandl, and K. Bogenberger, "Competition and cooperation of autonomous ridepooling services: Game-based simulation of a broker concept," *Frontiers in Future Transportation*, vol. 3, p. 915 219, 2022.

[118] F. Paparella, P. Labee, S. Wilkins, T. Hofman, S. Rasouli, and M. Salazar, "Multi-objective optimal trade-off between V2G activities and battery degradation in electric mobility-as-a-service systems," in *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2025, pp. 142–147.