UNIVERSITY OF CALIFORNIA SAN DIEGO

Structured Models for Sample-Efficient Learning and Control in Energy Systems

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Electrical and Computer Engineering (Intelligent Systems, Robotics & Control)

by

Yuexin Bian

Committee in charge:

Professor Yuanyuan Shi, Chair
Professor Rajesh Gupta
Professor Patricia Hidalgo-Gonzalez
Professor Jan Kleissl
Professor Yang Zheng

2026

The Dissertation of Yuexin Bian is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2026

DEDICATION

To my family and friends,
whose love, support and encouragement made this thesis possible

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I feel deeply lucky to have spent five years at UC San Diego and in San Diego. UCSD gave me far more than a place to study. It gave me a community, a rhythm, and a home by the ocean, where I learned how to think, how to struggle, how to begin again, and how to become more fully myself. When I was younger, the California sunshine, the ocean, and this kind of beauty felt very far away. Even now, after five years here, I am still often struck silent by the sunset and the sea. Slowly, I learned how to rest inside an ordinary afternoon: to enjoy a patch of sunshine, a quiet walk, or the sight of seagulls crossing the campus as if they, too, belonged here. I am deeply grateful for the campus, the people, and the quiet moments that made these years unforgettable.

These have been the best five years of my life. Looking back now, I realize how much of this path began with one conversation five years ago. I still remember that interview, when I was a senior undergraduate student presenting my advisor's Ph.D. work and imagining the future directions that might grow from it. At that time, I was only beginning to look toward research, while my advisor was beginning her own path toward becoming a tenured professor. Among all the people and opportunities that have shaped my Ph.D., the luckiest, and the one for which I am most grateful, is the chance to have been advised by Professor Yuanyuan Shi. I am especially grateful to her for her guidance, trust, patience, and support throughout my Ph.D.

Whenever I felt stressed about the next step or stuck in my research, the first warmth I received often came from her, asking whether I needed help. Because of her support, I was able to truly enjoy these five years: to explore research ideas freely, to work without unnecessary constraints on where I was located, and to have the freedom to pursue my own path.

I would also like to thank the professors at UC San Diego who taught and encouraged me along the way. I am deeply grateful to my committee members: Professor Rajesh Gupta, for helping shape the ideas in this dissertation; Professor Patricia Hidalgo-Gonzalez, for her kind and thoughtful suggestions during my qualifying exam and dissertation process; Professor Jan Kleissl, for generously sharing his time and feedback; and Professor Yang Zheng, who

introduced me to the world of optimization through one of the best courses I have ever taken, and whose work has been a constant source of inspiration. I would also like to thank Professor Sicun Gao for his insightful feedback on our collaborative research; Professor Oliver Schmidt, for his generous help and collaboration throughout my research; and Professor Bolun Xu and Professor Yize Chen, for their guidance and support during the early years of my Ph.D.

Beyond the work itself, some of my most cherished memories at UC San Diego were shaped by the people around me. My lab has been an inspiring and warm place to grow as a researcher, and I have been fortunate to work with wonderful collaborators and labmates, including Jie Feng, Luke Bhan, and many others. Many meaningful moments unfolded on the second floor of Franklin Antonio Hall, where ideas, conversations, encouragement, and countless memories were shared. I hope these connections will continue to grow in the years ahead.

UC San Diego also brought me many amazing coauthors and friends, including Ningkun Zheng, Xiaohan Fu, Tao Wang, Yijiang Li, Yufan Zhang, Ruisi Zhang, Tian Liang, Louis Liu, and Yuan Zhuang. Their help, companionship, and generosity have made these years richer and more memorable.

Outside of research, San Diego became brighter because of my dear friends Benjie Miao, Amy Cao, Xiangyu Zhang, Aiwei Yin, and Alice Wei. Thank you for filling these years with laughter, care, food, walks, conversations, and countless small moments of light. In a city far from where I first began, your friendship made San Diego feel like home. I am also grateful to Linhong Dai, Xuyang Shen, and Boyu Chen, whose kindness and joy have made them such lovely and unforgettable parts of these years.

Some friendships began long before this chapter of my life and have stayed with me through every change that followed. To my longtime friends, Xinyun Mao, Jialin Ye, Lilin Xu, Dingqi Zhang, Ayan Mao, Yizhou Chen, Shuchen Ren, and Haiwen Yu: for more than twelve years, our lives have continued to meet across different cities, countries, seasons, and chapters. Thank you for the long flights you have taken to visit me, the late-night conversations, the messages that crossed oceans, and the love that has remained steady through time and distance.

Finally, I owe my deepest gratitude to my family, who have supported me not only financially, but also with constant encouragement and love. Thank you for believing in me, for giving me the freedom to pursue my own path, and for standing behind me through every step.

Most importantly, I am deeply grateful to my boyfriend, Jiaqi Liu. I have always felt that life is more colorful because of you. From Hangzhou to California, you have been with me through so many chapters, turning unfamiliar places into home and ordinary days into memories I will always treasure. Because of your companionship, my days in Southern California have become the best years of my life so far. Thank you for being my best friend, my closest companion, and the person who has stood beside me through every season of this path. I look forward to all the years ahead of us, and to the life we will continue building together.

Chapter 2, in full, contains material from Y. Bian, N. Zheng, Y. Zheng, B. Xu, and Y. Shi, "Demand Response Model Identification and Behavior Forecast with OptNet: A Gradient-Based Approach," *Proceedings of the Thirteenth ACM International Conference on Future Energy Systems (e-Energy '22)*, 2022, pp. 268–279; and Y. Bian, N. Zheng, Y. Zheng, B. Xu, and Y. Shi, "Predicting Strategic Energy Storage Behaviors," *IEEE Transactions on Smart Grid*, 2024. The dissertation author was the primary investigator and author of these papers.

Chapter 3, in full, contains material from Y. Bian, X. Fu, R. K. Gupta, and Y. Shi, "Ventilation and Temperature Control for Energy-efficient and Healthy Buildings: A Differentiable PDE Approach," *Applied Energy*, 2024. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in full, contains material from Y. Bian, O. Schmidt, and Y. Shi, "Operator Learning for Energy-Efficient Building Ventilation Control with Computational Fluid Dynamics Simulation of a Real-World Classroom," *Applied Energy*, 2026. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in full, contains material from Y. Bian, J. Feng, and Y. Shi, "DiffOP: Reinforcement Learning of Optimization-Based Control Policies via Implicit Policy Gradi-

ents," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. The dissertation author was the primary investigator and author of this paper.

Chapter 6, in full, contains material from Y. Bian*, J. Feng*, T. Wang, Y. Li, S. Gao, and Y. Shi, "RN-D: Discretized Categorical Actors with Regularized Networks for On-Policy Reinforcement Learning," *International Conference on Machine Learning*, 2026 (*equal contribution). The dissertation author was a co-primary investigator and co-author of this paper.

VITA

2021 Bachelor of Engineering in Electrical Engineering, Zhejiang University, China

2023 Master of Science in Electrical and Computer Engineering, University of California San Diego, USA

2026 Doctor of Philosophy in Electrical and Computer Engineering, University of California San Diego, USA

PUBLICATIONS

Y. Bian, N. Zheng, Y. Zheng, B. Xu, and Y. Shi, "Demand Response Model Identification and Behavior Forecast with OptNet: A Gradient-Based Approach," *Proceedings of the Thirteenth ACM International Conference on Future Energy Systems (e-Energy '22)*, 2022, pp. 268–279.

Y. Bian, N. Zheng, Y. Zheng, B. Xu, and Y. Shi, "Predicting Strategic Energy Storage Behaviors," *IEEE Transactions on Smart Grid*, 2024.

Y. Bian, X. Fu, R. K. Gupta, and Y. Shi, "Ventilation and Temperature Control for Energy-efficient and Healthy Buildings: A Differentiable PDE Approach," *Applied Energy*, 2024.

Y. Bian, O. Schmidt, and Y. Shi, "Operator Learning for Energy-Efficient Building Ventilation Control with Computational Fluid Dynamics Simulation of a Real-World Classroom," *Applied Energy*, 2026.

Y. Bian, J. Feng, and Y. Shi, "DiffOP: Reinforcement Learning of Optimization-Based Control Policies via Implicit Policy Gradients," *Proceedings of the AAAI conference on artificial intelligence*, 2026.

Y. Bian*, J. Feng*, T. Wang, Y. Li, S. Gao, and Y. Shi, "RN-D: Discretized Categorical Actors with Regularized Networks for On-Policy Reinforcement Learning," *International Conference on Machine Learning*, 2026. (* Equal contribution.)

ABSTRACT OF THE DISSERTATION

Structured Models for Sample-Efficient Learning and Control in Energy Systems

by

Yuexin Bian

Doctor of Philosophy in Electrical and Computer Engineering (Intelligent Systems, Robotics & Control)

University of California San Diego, 2026

Professor Yuanyuan Shi, Chair

Modern energy systems are increasingly shaped by flexible demand, distributed storage, and building infrastructure. These systems generate rich data and require fast, adaptive decisions, but they are also governed by physical laws, safety constraints, and strategic behavior that are difficult to capture with generic black-box learning. This creates a disconnect between the structure-rich models used in classical energy systems analysis and the flexible learning pipelines favored in modern data-driven control. Methods that ignore known structure often suffer from poor sample efficiency, weak interpretability, and unreliable constraint handling, while purely model-based approaches struggle with hidden objectives and computational scale.

The goal of this thesis is to bridge this disconnect by developing structured learning and control methods that integrate optimization, physics, and control-theoretic structure directly into trainable decision-making architectures.

First, the dissertation develops differentiable optimization models for forecasting strategic demand response behavior from historical observations, allowing latent agent objectives and constraints to be learned from price-responsive actions. Second, it develops a PDE-constrained framework for building ventilation and temperature control based on the Navier–Stokes and convection–diffusion equations, enabling energy-efficient control while explicitly enforcing air-quality and thermal constraints under the modeled dynamics. Third, it addresses the computational burden of PDE-based control by introducing an ensemble neural operator surrogate trained on CFD simulations of a real classroom, yielding over five orders-of-magnitude speedup while preserving the spatial structure needed for closed-loop optimization. Fourth, it extends structured modeling to reinforcement learning through DiffOP, in which the policy is defined by a parameterized optimal control problem and trained end-to-end from reward feedback. Finally, it shows that structure can also enter through policy representation, where discretized categorical actors with regularized networks improve stability and performance in on-policy continuous control.

Taken together, these results show that structured learning offers a systematic way to combine model-based rigor with learning-based adaptability, yielding methods that are more sample-efficient, interpretable, and reliable for modern energy and control systems.

Chapter 1

Introduction

1.1 Background

The global transition toward a sustainable and decarbonized energy future is reshaping how energy systems are operated, controlled, and designed. Buildings consume nearly 40% of global energy [3], with heating, ventilation, and air conditioning (HVAC) systems accounting for roughly half of that use [4]. At the same time, the rapid growth of distributed energy resources, behind-the-meter storage, and price-responsive loads [5, 6, 7] is transforming the grid edge from a relatively passive layer into a large-scale network of heterogeneous, strategic, and dynamically coupled decision-makers. These developments create substantial opportunities. Flexible demand, distributed storage, and responsive buildings can absorb renewable variability, reduce peak demand, relieve congestion, and defer costly infrastructure expansion. They also create equally substantial operational challenges. System operators, market designers, and building managers must increasingly coordinate assets whose behavior depends not only on physical dynamics, but also on hidden objectives, economic incentives, occupancy patterns, and exogenous uncertainty. In such settings, effective decision-making requires algorithms that are simultaneously high-performing, sample-efficient, interpretable, and safe.

The energy applications considered in this dissertation sit at the intersection of several difficulties that rarely occur in isolation. First, they are *cyber-physical*: control actions propagate through real physical systems whose dynamics are constrained by conservation laws, actuator

limitations, and safety requirements. Second, they are increasingly *data-rich*: high-frequency sensors, market records, building telemetry, and simulation pipelines provide much more information than classical control pipelines were originally designed to exploit. Third, they are often *partially observed and strategic*: system operators must act without direct access to all relevant states, and many participants, such as storage owners or flexible loads, respond according to internal objectives that are not fully known. Finally, they are *high stakes*: failures in modeling or control can manifest as wasted energy, loss of comfort, unsafe air quality, equipment stress, or grid instability. These characteristics make energy systems an especially demanding domain for modern learning and control.

Historically, two broad paradigms have been used to address such problems. The first is the *model-based* paradigm, which includes optimization-based control, model predictive control (MPC) [8, 9], inverse optimization, and physics-based simulation [10]. These methods leverage prior knowledge of system dynamics, physical laws, and operational constraints. Their appeal lies in their transparency and rigor: the designer can specify an explicit objective, encode hard state and action constraints, analyze stability and feasibility using control-theoretic tools, and often achieve strong sample efficiency because much of the problem structure is supplied by the model rather than inferred from data. For this reason, model-based methods remain the dominant paradigm in many safety-critical infrastructure settings, including building HVAC control and power-system operation. Yet their limitations are increasingly evident as modern energy systems become larger, more heterogeneous, and more data-rich. Accurate first-principles models are expensive to build and frequently misspecified in practice [11]; high-fidelity spatiotemporal models such as computational fluid dynamics or detailed network dynamics are often too expensive to deploy inside real-time control loops [12]; and purely structural models are poorly suited to representing partially observed human or market behavior.

The second is the *learning-based* paradigm, including deep neural networks and deep reinforcement learning (RL) [13, 14]. These methods offer a different set of advantages: they can approximate highly nonlinear mappings, scale to high-dimensional observations and action

spaces, and adapt directly from data or interaction without requiring a complete hand-crafted model. Recent successes in robotics, games, and language model post-training suggest that sufficiently expressive learning systems can discover effective decision rules in domains previously thought to require extensive manual structure. Yet energy systems also expose several weaknesses of purely black-box learning. Training data-hungry policies through real-world interaction is often infeasible; for example, an RL controller may require years of equivalent interaction to match the performance of relatively simple engineered controllers [15]. Black-box policies also struggle to satisfy hard physical and safety constraints, which is unacceptable in infrastructure settings where violations may lead to equipment damage, blackouts, occupant discomfort, or health hazards. Their opacity complicates verification, regulatory approval, and operator trust [16]. Finally, their performance often depends strongly on choices of architecture, parameterization, and optimization procedure that remain insufficiently understood.

These competing strengths and weaknesses are especially pronounced in the applications studied in this dissertation. A building controller must reason about thermodynamics, airflow, air quality, and actuator limits while accounting for uncertain occupancy and disturbances. A market-facing storage resource must respond to prices while respecting private constraints and degradation costs that are not directly visible to the system operator. A grid controller must optimize network performance under safety requirements, nonlinear dynamics, and incomplete information. In each case, the decision-maker faces a problem that is richly structured, yet only partially modeled and only partially observed. This combination motivates the central perspective of this dissertation: learning algorithms for energy systems should exploit structure not as an afterthought, but as a core ingredient of representation, training, and control.

1.2 Motivation

The central motivation of this dissertation is that the properties required in real-world energy applications, including sample efficiency, interpretability, constraint satisfaction, and

strong closed-loop performance, are difficult to achieve with either paradigm in isolation. Purely data-driven methods ignore important prior structure that practitioners already possess: physical conservation laws, optimization-based decision rules of strategic agents [17], and the inherently constraint-aware nature of infrastructure operation. This omission wastes information, inflates sample complexity, and often produces models that generalize poorly outside the range of observed training data. Conversely, purely model-based methods struggle to absorb the uncertainty, heterogeneity, and scale of contemporary energy systems [18], and they often fail to exploit the large volumes of high-resolution data that are now routinely available from sensors, markets, and building management systems. The resulting gap is therefore not simply a gap in performance, but a gap in *representation*: classical models often encode the right structure but lack adaptability, whereas modern learners adapt readily but too often fail to encode the structure that matters.

This dissertation is motivated by the view that the most promising path forward is not to choose between model-based and learning-based paradigms, but to integrate them in a principled manner. Recent advances in differentiable optimization layers [19] and neural operators [20, 21] suggest that structural knowledge can be preserved *inside* a trainable pipeline rather than imposed only before or after learning. In this formulation, optimization problems, physical simulators, and structured control laws are not merely external tools used to generate data, labels, or constraints. They become differentiable components of the decision-making architecture itself. This shift in viewpoint is subtle but important. Once structure is embedded in the learning pipeline, it can shape the hypothesis class, regularize the search space, improve data efficiency, and expose quantities that remain meaningful to human operators and domain experts.

The notion of "structure" in this dissertation is intentionally broad. In Chapter 2, the relevant structure is the latent optimization problem solved by a strategic energy resource. In Chapters 3 and 4, the relevant structure is the spatiotemporal physics governing indoor airflow, temperature, and CO_2 transport. In Chapter 5, the structure appears in the control policy itself, which is defined implicitly as the solution to an optimal control problem and trained through reinforcement learning. In Chapter 6, the emphasis shifts again: rather than embedding external

physics or optimization into the policy, the work studies how the *representation* of the policy can itself impose useful structure on the learning problem. Across these settings, the technical forms differ, but the underlying design principle is shared: choose a representation that reflects what is already known about the system, and then learn within that structured space.

This perspective raises a sequence of research questions that motivate the chapters of the dissertation. How can one recover the hidden optimization logic of strategic agents from observed market behavior alone? How can high-fidelity PDE models of building physics be incorporated into control without sacrificing tractability? If direct PDE-based control is too expensive for real-time deployment, how can one learn surrogates that preserve the relevant physical structure rather than collapse the problem to low-dimensional averages? Can optimization-based policies be trained directly from environmental feedback through reinforcement learning while retaining interpretability and theoretical guarantees? More broadly, what kinds of policy representations produce optimization landscapes that are better matched to stable, sample-efficient RL? Each chapter addresses one part of this broader agenda.

The central thesis of this dissertation is that embedding *structured models* derived from optimization, physics, and control directly into learning pipelines can substantially improve sample efficiency, interpretability, and closed-loop performance in energy systems. Structure is treated here not as an auxiliary regularizer or a post hoc correction, but as a first-class design principle that shapes the model class, the training objective, and the resulting controller. Taken together, the chapters argue that when the relevant domain knowledge is incorporated into the learning architecture itself, one can obtain algorithms that are not only more accurate, but also more reliable, more data-efficient, and more suitable for deployment in real energy systems.

1.3 Research Questions and Thesis-Level Contributions

The dissertation can be read as addressing five linked research questions, each corresponding to a different manifestation of structure in learning and control.

First, when an energy resource behaves strategically by solving an internal optimization problem, can that hidden decision model be inferred directly from observations? This question motivates Chapter 2, which develops differentiable optimization layers for strategic behavior prediction and agent model identification. The key contribution is to show that a private optimization problem can be embedded inside a gradient-based learning loop, allowing cost parameters and constraints to be identified directly from behavior data while preserving interpretability.

Second, when the true dynamics of a control problem are governed by spatially distributed physics, can these dynamics be used directly for control rather than replaced by coarse aggregated surrogates? This question motivates Chapter 3, which formulates building HVAC learning and control through PDE-constrained optimization. The contribution here is both methodological and conceptual: it shows that explicitly modeling airflow and contaminant transport at the PDE level can produce materially better control decisions than low-dimensional approaches that ignore spatial structure.

Third, if high-fidelity physics is valuable but too expensive for real-time decision-making, can one learn surrogates that preserve the right structure rather than discarding it? This question motivates Chapter 4, where neural operators are trained on CFD data to approximate high-dimensional spatiotemporal dynamics. The contribution is to bridge physical fidelity and computational tractability: the surrogate remains compatible with optimization-based control and preserves the spatial structure needed for air-quality-aware decisions, while reducing computation by several orders of magnitude.

Fourth, can the idea of structured modeling be pushed beyond supervised identification and open-loop control into reinforcement learning itself? This question motivates Chapter 5, which introduces DiffOP, a framework in which the control policy is defined as the solution of a parameterized optimal control problem and trained end-to-end from environmental feedback. The contribution is twofold: an optimization-based policy class that remains interpretable and

constraint-aware, and a theoretical analysis establishing a non-asymptotic convergence guarantee for policy-gradient learning in this setting.

Fifth, even when the learning algorithm is fixed, how much can be gained by re-thinking the structure of the policy representation itself? This question motivates Chapter 6, which studies discretized categorical actors and regularized actor architectures for on-policy reinforcement learning. The contribution is to show that representation is not a minor implementation detail, but a central determinant of optimization behavior, gradient variance, and ultimately downstream control performance.

At the thesis level, these chapters contribute a broader methodological viewpoint. They demonstrate that structure can enter the learning pipeline through multiple routes: through the latent model of a strategic agent, through the governing equations of a physical system, through a learned surrogate that preserves operator structure, or through the parameterization of the policy itself. The broader research direction of this dissertation is therefore not limited to any single algorithmic family. Instead, it advocates a general program of *structure-aware learning for control*, in which the right inductive bias is chosen to reflect the semantics, geometry, and constraints of the underlying decision problem.

1.4 Dissertation Overview

The chapters follow a deliberate progression from identifying hidden structure in observed behavior, to controlling through high-fidelity physical structure, to scaling those structured methods, and finally to embedding structure directly in policies and policy representations for reinforcement learning. This progression is summarized in Figure 1.1, which serves as a roadmap for the dissertation and highlights how the individual chapters connect through the common theme of structure-aware learning and control.

Chapter 2 studies strategic demand response behavior forecasting in electricity markets, where flexible loads, aggregators, and storage resources respond to prices according to private

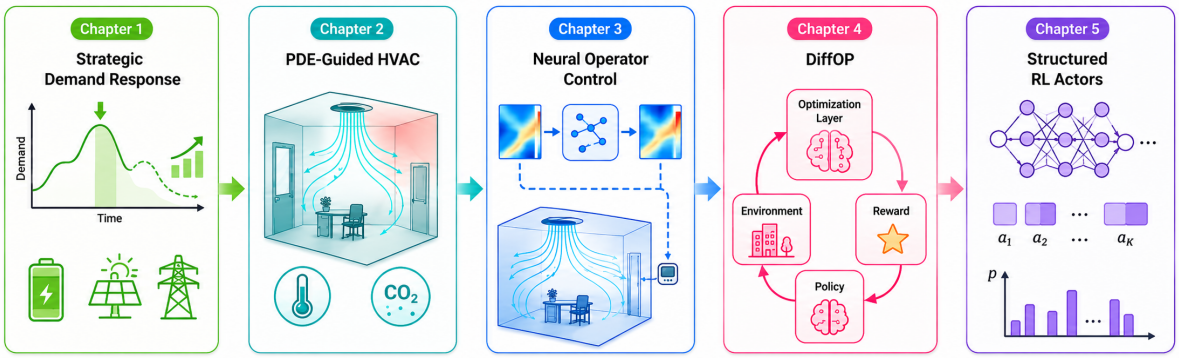


Figure 1.1. Dissertation roadmap. The five chapters address complementary manifestations of structure in learning and control, progressing from optimization-guided behavior prediction (Chapter 2), through physics-based and surrogate-based building control (Chapters 3–4), to structured policy learning via reinforcement learning (Chapters 5–6).

decision rules that are not directly observed by the system operator [22, 23]. Rather than treating forecasting and model identification as separate tasks, the chapter formulates prediction as learning from *observed decisions* generated by a latent optimization problem. It develops differentiable optimization layers [24, 25] that embed an agent's private optimization model inside a gradient-based training loop, so that unknown cost parameters and operational constraints can be inferred in a way that directly supports behavior forecasting. The framework extends from quadratic agent models with exact implicit gradients to more general convex formulations via sequential convex programming, and it also handles partially observed settings such as aggregate load forecasting. Across synthetic and real-world price-responsive behavior data, the resulting approach achieves strong forecasting accuracy with high sample efficiency. Conceptually, this chapter establishes the first pillar of the thesis: optimization structure can serve as a powerful inductive bias for forecasting and learning strategic behavior.

Chapter 3 turns to building HVAC and ventilation control, where the central challenge is to simultaneously maintain indoor air quality and thermal safety while minimizing energy consumption. Existing approaches based on ordinary differential equation (ODE) models [26, 27] often ignore the spatial structure of airflow and contaminant transport, which can lead to over-ventilation or undetected dead zones. To address this limitation, Chapter 3 develops a PDE-constrained

framework based on the Navier–Stokes and convection–diffusion equations [28], and uses adjoint-based optimization to learn and control the resulting spatiotemporal dynamics. The contribution of this chapter is not only a new control framework, but also a broader methodological point: when the quantities of interest are fundamentally spatial, aggregated models can obscure the very phenomena the controller must regulate. This chapter thus establishes the second pillar of the thesis: physical structure matters, and it can be exploited directly through differentiable physics.

Chapter 4 addresses the computational bottleneck of the PDE-based approach by replacing the expensive online PDE solve with a learned surrogate. Using computational fluid dynamics (CFD) simulations of a real-world classroom as training data, the chapter develops an ensemble of neural operator transformers [29, 20, 21] that learn the mapping from control inputs and historical CO_2 fields to future CO_2 fields. The learned operator preserves the spatial structure that makes PDE-based control valuable, while providing over five orders-of-magnitude speedup relative to the underlying CFD solver. Embedded in closed-loop optimization and validated against CFD, the resulting controller achieves substantial energy savings while consistently satisfying air-quality requirements [30]. Read together, Chapters 3 and 4 illustrate a recurring theme of the dissertation: when exact structure is too expensive to use online, one should approximate it in a way that preserves its essential semantics rather than replacing it with an unrelated black-box proxy.

Chapter 5 considers the broader problem of learning optimization-based control policies directly from interaction with the environment. Whereas Chapter 2 used differentiable optimization for supervised identification, and Chapters 3 and 4 used structured models inside open-loop or receding-horizon control, Chapter 5 closes the loop by training an optimization-defined policy directly from closed-loop rewards. The chapter introduces DiffOP [31], a reinforcement learning framework in which the policy is defined *implicitly* as the solution to a parameterized optimal control problem. By combining implicit differentiation through Pontryagin's Maximum Principle [2, 32] with standard policy-gradient methods [33], DiffOP jointly learns cost and dynamics models end-to-end from environmental rewards. The chapter

further establishes, to the best of our knowledge, the first non-asymptotic $\mathcal{O}(\varepsilon^{-1})$ convergence guarantee for policy-gradient learning of this form, and validates the framework on nonlinear control benchmarks and a constrained voltage regulation task. This chapter extends the thesis narrative from "learning through structure" to "learning *as* structured control."

Chapter 6 revisits policy representation itself as a first-class design choice in on-policy reinforcement learning for continuous control. Modern algorithms such as PPO are highly sensitive to policy parameterization and typically rely on diagonal Gaussian actors, which often lead to fragile optimization dynamics. Motivated by the success of classification-style objectives in value learning [34, 35] and language model post-training [36], the chapter studies factorized discretized categorical actors [37, 38] together with regularized actor networks. This structured change to the policy class reduces gradient variance, stabilizes on-policy learning, and yields substantial improvements over standard PPO across a range of continuous-control benchmarks, including locomotion and both state- and vision-based manipulation tasks. Within the broader narrative of the dissertation, this chapter shows that even when one does not explicitly encode physics or optimization constraints, learning can still benefit greatly from well-chosen structure at the representational level.

Chapter 7 concludes the dissertation by synthesizing the common themes across these contributions and outlining several directions for future work, including robustness under uncertainty, broader theoretical guarantees for structured learning architectures, and real-world deployment in large-scale cyber-physical energy systems.

Chapter 2

Optimization-Guided Models for Learning Strategic Behavior in Energy Systems

This chapter develops the first technical component of the dissertation's structure-aware learning agenda by studying strategic demand response behavior forecasting in electricity markets. The central premise is that price-responsive actions should not be modeled as arbitrary black-box outputs, but as decisions generated by latent optimization problems with private objectives and operational constraints. Building on this view, the chapter shows how differentiable optimization can be used to learn forecasting models directly from observed responses, thereby connecting behavior prediction and model identification within a unified framework.

2.1 Introduction

The rapid expansion of demand-side flexibility resources – price-arbitrage energy storage, thermostatically controlled buildings, and aggregated responsive loads – is reshaping modern power systems. U.S. utility-scale storage capacity tripled in 2021, reached 7.8 GW by October 2022, and is projected to exceed 30 GW by 2025 [5]; behind-the-meter (BTM) storage is growing even faster, increasing 67% year-over-year in 2022. FERC Orders 841 [6] and 2222 [7] require system operators to allow storage and distributed energy resource portfolios to participate in all market services. Across all these resources, price arbitrage is becoming a primary revenue mechanism.

These resources are operated by private entities whose decision logic is unknown to system operators: a storage operator solves a private look-ahead optimization weighing prices against degradation costs and SoC constraints; a building operator optimizes HVAC consumption subject to thermal dynamics and comfort bounds; a load aggregator schedules heterogeneous loads while the operator sees only the aggregate. In all cases, the operator's cost parameters, physical constraints, and efficiency are private. This opacity creates a significant challenge for clearing markets efficiently, designing tariffs that align private incentives with social objectives, and monitoring for strategic market power abuse [22, 23].

Two broad families of methods have been applied. *Model-free* approaches (feed-forward and recurrent neural networks) treat prediction as a black-box sequence task; they capture nonlinear patterns but need large datasets and generalize poorly off-distribution because they do not encode the underlying optimization structure. *Model-based* inverse optimization [17] casts identification as a bi-level problem solved with successive linear programming or nonlinear solvers; it incorporates structural knowledge but struggles with scale, noise, and partial observability.

This chapter presents a unified gradient-based framework for *strategic demand response behavior forecasting* that bridges these two families. The core idea is to treat the agent's optimal response to a price signal as a differentiable function of unknown model parameters, thereby enabling end-to-end training of a forecasting model through the agent's private optimization problem. Rather than viewing forecasting and model identification as separate tasks, we cast them as a single learning problem: learn the latent optimization model that best predicts future price-responsive behavior. For agents with quadratic objective functions, which cover common cases such as quadratic storage degradation cost and building temperature violation penalties, we derive closed-form gradients via implicit differentiation of the KKT conditions and establish convergence for the class of equality-constrained quadratic agent models. For agents with general, possibly non-quadratic objective functions, we extend the framework using sequential convex programming (SCP) paired with input convex neural networks (ICNNs) to model the unknown disutility function. Critically, the framework also handles partially observable forecasting settings

by propagating gradients through any known observation function, such as summing individual schedules into an aggregate load.

2.1.1 Contributions

The main contributions of this chapter are as follows:

- We propose a unified gradient-based framework that jointly identifies agent model parameters and forecasts price-responsive behavior for a broad class of demand response resources, including energy storage systems, thermostatically controlled buildings, and load aggregators, by embedding each agent's private optimization as a differentiable layer and training end-to-end via gradient descent.
- We establish a formal convergence guarantee for the gradient descent algorithm on equality-constrained quadratic agent models, and characterize the existence of local minima for the inequality-constrained case.
- We conduct comprehensive experiments across three application domains, namely energy storage (synthetic and real-world), building demand response, and load aggregation, demonstrating consistent improvements over black-box neural networks and inverse optimization baselines.

The remainder of this chapter is organized as follows. Section 2.2 reviews related work. Section 2.3 presents the demand response agent models and problem formulation. Section 2.4 develops the gradient-based algorithms for both quadratic and generic agent models. Section 2.5 describes the experimental results. Section 2.6 summarizes the chapter.

2.2 Related Work

Energy storage market participation.

Under FERC Order 841 [6], energy storage participates via either self-scheduling or economic bids; the latter offers greater efficiency under price uncertainty [39, 40]. Behind-the-

meter (BTM) storage on the customer side instead makes charge/discharge decisions in response to real-time price signals [41, 42, 43]. We model both wholesale and BTM storage as price-takers and seek to capture their strategic price-responsive behavior from data.

Market power mitigation.

With surging storage capacity, market power mitigation extends classical generator-side analysis [44, 45] to storage, where opportunity costs replace fuel costs and the operator's degradation parameters are typically private [46, 47]. By identifying these latent parameters from historical participation data, the framework developed here can support market behavior monitoring; [23] demonstrates an application to marginal offer price recovery for conventional generation.

Demand response behavior forecasting.

Prior work falls into two camps. *Model-free* approaches use Gaussian models [48] or neural networks [49] to predict price-responsive behavior, but require large datasets and generalize poorly off-distribution. *Model-based* inverse optimization [50, 17, 51] formulates identification as a bi-level optimization, exploiting agent structure but struggling with large or noisy data. This chapter combines the two, using gradient-based learning through a structured optimization layer for strategic demand response behavior prediction.

2.3 Problem Formulation

We consider a general setting in which a demand response agent responds to a time-varying price signal by solving a private optimization problem. The agent's decision model, including its cost parameters and physical constraints, is unknown to the system operator. The system operator observes historical price signals and the corresponding agent responses (possibly partial), and seeks to learn a forecasting model that can accurately predict future behavior under unseen price scenarios. In our framework, this forecasting model is parameterized implicitly through the agent's latent optimization problem.

We model the agent's decision-making as a parametric convex optimization problem:

$$\min_{y} \quad f(y;\lambda,\theta) \tag{2.1a}$$

$$\text{subject to} \quad Ay = b, \tag{2.1b}$$

$$Gy \leq h, \tag{2.1c}$$

where y is the agent's decision variable (e.g., charging/discharging power, building HVAC load, or scheduled load quantities), $\lambda \in \mathbb{R}^T$ is the observed price signal, θ parameterizes the agent's private cost function $f(\cdot)$, and matrices A, G with right-hand sides b, h encode the equality and inequality constraints. The full parameter set is $\Theta = \{\theta, A, b, G, h\}$. In general, the system operator may not observe the decision variable y directly but instead observes $z = \phi(y)$ for some known function ϕ; for example, ϕ is the identity for energy storage and building loads, but is a summation operator when only the aggregate of multiple loads is observable.

Figure 2.1 illustrates the proposed framework using energy storage as a representative example. We use storage as a concrete running example because its strategic response to prices is easy to visualize, but the same forecasting architecture applies to all agent types considered in this chapter.

2.3.1 Demand Response Agent Models

We present four concrete instantiations of (2.1) across different demand response applications. These examples are not separate methods; rather, they are different forecasting settings that share the same optimization-guided learning framework. We begin with energy storage, where the decision variable is the charging and discharging power schedule, and then extend to building thermal control and load aggregation.

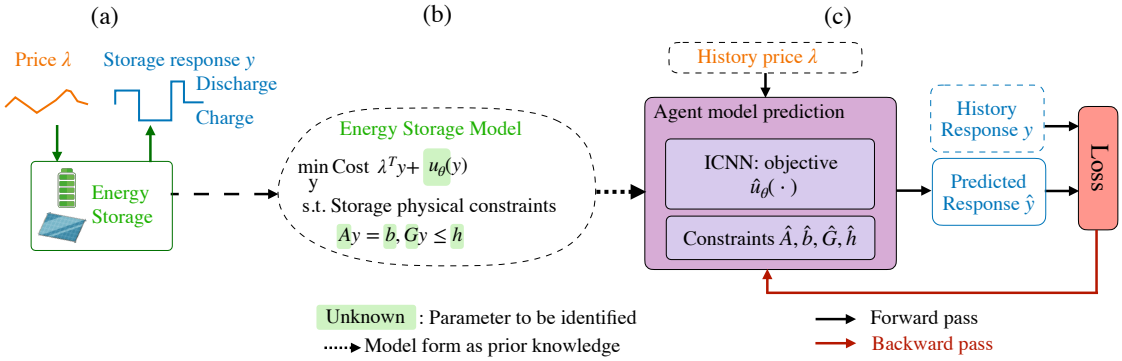


Figure 2.1. Illustration of the proposed demand response behavior forecasting framework, shown here for energy storage. The agent's private optimization is embedded as a differentiable layer, and the unknown model parameters are trained by minimizing the forecasting error between predicted and observed responses on historical price-response data. The same framework generalizes to building demand response and load aggregation.

Energy Storage

We start by considering a price-responsive energy storage agent subject to a time-varying electricity price $\lambda \in \mathbb{R}^T$ and a disutility cost function:

$$\min_{p,d} \quad \sum_{t=1}^{T} \lambda_t (p_t - d_t) + u_\theta(p,d) \tag{2.2a}$$

$$\text{subject to} \quad \underline{P} \le d_t, p_t \le \overline{P}, \tag{2.2b}$$

$$e_t = e_0 + \sum_{\tau=1}^{t} p_\tau \eta_c - \frac{d_\tau}{\eta_d}, \tag{2.2c}$$

$$\underline{E_m} \le e_t \le \overline{E_m}, \tag{2.2d}$$

where decision variables $p_t, d_t \ge 0$ are the energy storage charging and discharging power at time t, e_0 is the initial state of charge level, e_t is the SoC at operating time t, and $u_\theta(\cdot)$ models the storage disutility cost, which is a generic function of the storage charging/discharging power. Here $\underline{P}, \overline{P}, \underline{E_m}, \overline{E_m}$ are the maximum and minimum storage power and energy constraints, and η_c, η_d are the charging and discharging efficiencies. The expression $\sum_{\tau=1}^{t} p_\tau \eta_c - \frac{d_\tau}{\eta_d}$ represents the accumulated relative state of charge level from time 1 to time t. We assume e_0 is known.

The full parameter set of each energy storage agent is $\Theta = \{\theta, \underline{P}, \overline{P}, \underline{E_m}, \overline{E_m}, \eta_c, \eta_d\}$, where θ parameterizes the disutility function $u_\theta(\cdot)$.

We re-write (2.2c) and (2.2d) as:

$$\underline{E_m}\eta_d \leq e_0\eta_d + \sum_{\tau=1}^{t} p_\tau \eta_c \eta_d - d_\tau \leq \overline{E_m}\eta_d\,, \tag{2.3}$$

which makes the storage constraints affine in the decision variables. The new parameters are $\{\eta_c\eta_d, (\underline{E_m} - e_0)\eta_d, (\overline{E_m} - e_0)\eta_d\}$. When charging and discharging efficiencies are symmetric ($\eta_c = \eta_d = \eta$), the round-trip efficiency $\eta_c\eta_d$ can be parameterized as a single scalar.

We consider two concrete instantiations of the disutility function:

Example 1: Price Arbitrage with Quadratic Storage Degradation Cost. Here we consider u_θ to be a parametric quadratic function:

$$\min_{p,d} \quad \sum_{t=1}^{T} \lambda_t(p_t - d_t) + c_1 d_t + c_2 d_t^2 \tag{2.4a}$$

$$\text{subject to} \quad \text{(2.2b), (2.3)} \tag{2.4b}$$

where $c_1, c_2 \geq 0$ are the linear and quadratic degradation cost coefficients.

Example 2: Price Arbitrage with SoC-Dependent Storage Degradation Cost. Here we consider u_θ to be an SoC-related storage degradation cost, which includes a quadratic degradation cost with respect to charging power and an SoC-cost related to the SoC level. The SoC-cost is lower when operating around 50% SoC and higher when SoC approaches 0% or 100%:

$$\min_{p,d} \quad \sum_{t=1}^{T} \lambda_t(p_t - d_t) + c_1(e_t - 0.5)^2 + c_2 d_t^2 \tag{2.5a}$$

$$\text{subject to} \quad \text{(2.2b), (2.3)} \tag{2.5b}$$

Both examples are special cases of the general formulation (2.1), with $y = [p_1, \ldots, p_T, d_1, \ldots, d_T]^\top$ and affine constraints encoding power and energy limits.

Building Demand Response

Example 3: Building Demand Response with Thermal Dynamics. We consider a price-responsive building demand response model [50] subject to a linear thermal dynamics model:

$$\min_{p,y,\sigma} \quad \sum_{t=1}^{T} (\lambda_t p_t + \rho y_t^2) \tag{2.6a}$$

$$\text{subject to} \quad \sigma_t = \alpha_1 \sigma_{t-1} + (1-\alpha_1)[\sigma_t^{\text{amb}} - \alpha_2 p_t], \tag{2.6b}$$

$$-y_t + \underline{\sigma} \le \sigma_t \le \overline{\sigma} + y_t, \tag{2.6c}$$

$$0 \le p_t \le \overline{p}, \quad y_t \ge 0, \quad t = 1, \dots, T, \tag{2.6d}$$

where p_t is the power consumption, y_t is a slack variable penalizing temperature constraint violations, ρ is the penalty coefficient, σ_t is the indoor temperature, σ_t^{amb} is the ambient temperature, and $\underline{\sigma}, \overline{\sigma}$ are the thermal comfort bounds. To maintain linear constraints with respect to the parameters, we define $\alpha_3 = (1-\alpha_1)\alpha_2$, so that (2.6b) becomes $\sigma_t = \alpha_1 \sigma_{t-1} + (1-\alpha_1)\sigma_t^{\text{amb}} - \alpha_3 p_t$. The parameters to be identified are $\{\rho, \overline{\sigma}, \underline{\sigma}, \overline{p}, \alpha_1, \alpha_3\}$.

Load Aggregation

Example 4: Aggregator with Multiple Responsive Loads. We consider a load aggregator model [17] with K types of price-responsive loads. The system operator observes only the aggregate load $z_t = \sum_{k=1}^{K} y_{k,t}$ rather than individual schedules:

$$\min_{y} \quad \sum_{k=1}^{K} \sum_{t=1}^{T} \lambda_t y_{k,t} - \alpha_{k,t} y_{k,t} \tag{2.7a}$$

$$\text{subject to} \quad 0 \le y_{k,t} \le P_{k,t}, \quad k = 1, \dots, K,\ t = 1, \dots, T, \tag{2.7b}$$

$$\sum_{t=1}^{T} y_{k,t} = M_k, \quad k = 1, \dots, K, \tag{2.7c}$$

where $y_{k,t}$ is the scheduled load for type k at time t, $\alpha_{k,t}$ is the utility coefficient capturing the economic value of scheduling load type k into period t, $P_{k,t}$ is the per-period power limit, and

M_k is the total consumption requirement for load type k. Since only the aggregate $z_t = \sum_k y_{k,t}$ is observed, the identification loss is evaluated on the aggregated prediction rather than individual schedules. The parameters to be identified are $\{P_{k,t}, M_k, \alpha_{k,t}\}$.

Remark 1. The four examples above illustrate the breadth of the proposed framework. Examples 1 and 2 target energy storage price arbitrage with quadratic and SoC-dependent degradation costs, respectively. Examples 3 and 4 demonstrate that the same gradient-based identification approach generalizes naturally to building demand response with thermal dynamics and to load aggregators where only aggregate consumption is observable. The key structural requirements are: (i) the agent's decision problem is a convex optimization with affine constraints and a convex (possibly non-quadratic) cost function; and (ii) the observation is a known function of the decision variable (identity for storage and buildings; summation for the aggregator). These requirements are satisfied across a broad class of price-responsive demand response models.

2.3.2 Agent Model Identification Problem

The system operator has access to a dataset of N historical price signals and the corresponding agent responses: $\{(\lambda_i, z_i)\}_{i=1}^N$, where $z_i = \phi(y_i)$ is the observed response (possibly a partial observation of the true decision y_i). The true agent parameters $\bar{\Theta}$ are unknown. The forecasting problem is to learn parameters Θ such that the induced optimization model accurately predicts the observed price-response behavior:

$$\begin{aligned} \min_{\Theta} \quad & L := \frac{1}{N} \sum_{i=1}^{N} \|\phi(y_i^\star) - \phi(y_i)\|^2 \\ \text{where} \quad & y_i^\star \in \arg\min_{y} \; f(y; \lambda_i, \theta) \quad \text{s.t. (2.1b), (2.1c)}, \end{aligned} \tag{2.8}$$

in which the training inputs include

- λ_i: the price signal for scenario i;
- $z_i = \phi(y_i)$: the observed agent response (full or aggregate);

and the problem variables are

- $\Theta = \{\theta, A, b, G, h\}$: learnable parameters of the agent model;
- $y_i^\star$: the predicted agent decision under estimated parameters Θ.

This is a bi-level optimization problem: the outer problem minimizes forecast error over the unknown parameters Θ, while the inner problem represents the agent's private optimization (2.1). The key technical challenge is computing $\frac{\partial L}{\partial \Theta}$, which requires differentiating through the solution of the agent's optimization problem.

2.4 Algorithm Design

We propose a gradient-descent approach to solve the demand response behavior forecasting problem (2.8). The key idea is to train the forecasting model by differentiating through the agent's optimization layer. We first present the generic gradient-based training perspective, then derive the required gradients for quadratic agent models via implicit differentiation of the KKT conditions, and finally extend the method to generic convex agent models via sequential convex programming. We conclude with a convergence analysis for the equality-constrained quadratic case.

2.4.1 Gradient-Based Training for Behavior Forecasting

To explain the algorithmic development concretely, we use the storage model as a running example. This choice is purely expository: the same training logic applies to any agent model in Section 2.3 once it is expressed in the generic form (2.1). Without loss of generality, the storage

model with a generic convex objective and affine constraints can be written as:

$$\min_{y} \quad c(y,\lambda) := \lambda^{\top} y + u_{\theta}(y) \tag{2.9a}$$

$$\text{subject to} \quad Ay = b\,, \tag{2.9b}$$

$$Gy \leq h\,, \tag{2.9c}$$

where $\lambda = [\lambda_1,\ldots,\lambda_T,-\lambda_1,\ldots,-\lambda_T]^{\top} \in \mathbb{R}^{2T}$ encodes the price signal, $y = [p_1,\ldots,p_T, d_1,\ldots,d_T]^{\top} \in \mathbb{R}^{2T}$ stacks the charging and discharging schedules, and matrices $A \in \mathbb{R}^{m\times 2T}$, $b \in \mathbb{R}^m$, $G \in \mathbb{R}^{p\times 2T}$, $h \in \mathbb{R}^p$ encode the equality and inequality constraints. The full parameter set is $\Theta = \{\theta, A, b, G, h\}$. In forecasting terms, solving (2.9) produces the predicted response $y^{\star}$ under a given price input λ.

At iteration t, parameters are updated via:

$$\Theta^{(t+1)} \leftarrow \Theta^{(t)} - \eta \sum_{i=1}^{N} \frac{\partial L}{\partial y_i^{\star}} \frac{\partial y_i^{\star}}{\partial \Theta}\,, \tag{2.10}$$

where η is the learning rate, $\frac{\partial L}{\partial y_i^{\star}}$ is the gradient of the forecast loss with respect to the predicted response, and $\frac{\partial y_i^{\star}}{\partial \Theta}$ is the sensitivity of the optimal solution to the model parameters. Computing $\frac{\partial L}{\partial y_i^{\star}}$ is straightforward once the observation function ϕ is specified. The main technical challenge is computing $\frac{\partial y_i^{\star}}{\partial \Theta}$ efficiently and robustly, which is the focus of the following subsections.

Remark 2. We use a gradient descent approach over other descent-based techniques, such as Newton or Quasi-Newton methods, primarily because deriving the first-order gradient involves differentiating KKT conditions, which requires solving a costly Jacobian-based equation. Computing the Hessian matrix and its inverse needed for second-order methods is more expensive. Investigating the use of second-order optimization methods is an interesting direction for future work.

2.4.2 Special Case: Differentiating Quadratic Agent Models

We first consider the important special case in which the agent objective is quadratic. This class covers common forecasting models for storage arbitrage and building demand response, and it admits efficient closed-form differentiation. Using the storage model for concreteness, let $u_\theta(y) = q^\top y + y^\top Q y$, so that the parameters to be identified are $\Theta = \{q, Q, A, b, G, h\}$:

$$\min_{y} \quad c(y,\lambda) := \lambda^\top y + q^\top y + y^\top Q y \tag{2.11a}$$

$$\text{subject to} \quad Ay = b\,, \tag{2.11b}$$

$$Gy \leq h\,. \tag{2.11c}$$

The agent model (2.11) is convex since it has a convex objective and affine constraints. For the running storage example, it is reasonable to assume strict feasibility, i.e., there exists $y \in \mathbb{R}^{2T}$ such that $Ay = b$, $Gy < h$. Then strong duality holds for (2.11) since Slater's condition holds. The KKT conditions, which are both necessary and sufficient for optimality, are:

$$\lambda + q + Qy^\star + A^\top \nu^\star + G^\top \mu^\star = 0 \tag{2.12a}$$

$$Ay^\star - b = 0 \tag{2.12b}$$

$$D(\mu^\star)(Gy^\star - h) = 0\,, \tag{2.12c}$$

where $D(\cdot)$ creates a diagonal matrix from a vector, and $\nu^\star, \mu^\star$ are the optimal dual variables for equality and inequality constraints, respectively.

Taking total differentials of these conditions and collecting terms yields the linear system:

$$\underbrace{\begin{bmatrix} Q & G^\top & A^\top \\ D(\mu^\star)G & D(Gy^\star - h) & 0 \\ A & 0 & 0 \end{bmatrix}}_{K} \begin{bmatrix} \mathrm{d}y \\ \mathrm{d}\mu \\ \mathrm{d}\nu \end{bmatrix} = - \begin{bmatrix} \mathrm{d}Qy^\star + \mathrm{d}q + \mathrm{d}G^\top \mu^\star + \mathrm{d}A^\top \nu^\star \\ D(\mu^\star)\,\mathrm{d}Gy^\star - D(\mu^\star)\,\mathrm{d}h \\ \mathrm{d}A\,y^\star - \mathrm{d}b \end{bmatrix}. \tag{2.13}$$

Algorithm 1. Quadratic Demand Response Forecast Training

Require: Dataset $D = \{(\lambda_i, z_i)\}$, $i = 1, \ldots, N$, initial parameters $\Theta^{(0)}$
  for $ite = 0, 1, \ldots, T_{\max} - 1$ **do**
    Sample batch $B \subseteq D$
    Forward: Solve $y_i^\star = \arg\min_y c(y; \lambda_i, \Theta^{(ite)})$ for each $i \in B$
    Evaluate: $L = \frac{1}{|B|} \sum_{i \in B} \|\phi(y_i^\star) - z_i\|^2$
    Backward: Compute $\frac{\partial y_i^\star}{\partial \Theta}$ via (2.13)–(2.14)
    Update: $\Theta^{(ite+1)} \leftarrow \Theta^{(ite)} - \eta \sum_{i \in B} \frac{\partial L}{\partial y_i^\star} \frac{\partial y_i^\star}{\partial \Theta}$
  end for

Solving for different right-hand sides gives all required partial derivatives:

$$\begin{bmatrix} \frac{\partial y^*}{\partial Q} & \frac{\partial \mu^*}{\partial Q} & \frac{\partial \nu^*}{\partial Q} \end{bmatrix}^\top = -K^{-1} \begin{bmatrix} \frac{\mathrm{d}Q}{\mathrm{d}Q} y^\star & 0 & 0 \end{bmatrix}^\top, \tag{2.14a}$$

$$\begin{bmatrix} \frac{\partial y^*}{\partial q} & \frac{\partial \mu^*}{\partial q} & \frac{\partial \nu^*}{\partial q} \end{bmatrix}^\top = -K^{-1} \begin{bmatrix} I & 0 & 0 \end{bmatrix}^\top, \tag{2.14b}$$

$$\begin{bmatrix} \frac{\partial y^*}{\partial h} & \frac{\partial \mu^*}{\partial h} & \frac{\partial \nu^*}{\partial h} \end{bmatrix}^\top = -K^{-1} \begin{bmatrix} 0 & -D(\mu^\star) & 0 \end{bmatrix}^\top, \tag{2.14c}$$

$$\begin{bmatrix} \frac{\partial y^*}{\partial b} & \frac{\partial \mu^*}{\partial b} & \frac{\partial \nu^*}{\partial b} \end{bmatrix}^\top = -K^{-1} \begin{bmatrix} 0 & 0 & -I \end{bmatrix}^\top, \tag{2.14d}$$

$$\begin{bmatrix} \frac{\partial y^*}{\partial A} & \frac{\partial \mu^*}{\partial A} & \frac{\partial \nu^*}{\partial A} \end{bmatrix}^\top = -K^{-1} \begin{bmatrix} \frac{\mathrm{d}A^\top}{\mathrm{d}A} \nu^\star & 0 & \frac{\mathrm{d}A}{\mathrm{d}A} y^\star \end{bmatrix}^\top, \tag{2.14e}$$

$$\begin{bmatrix} \frac{\partial y^*}{\partial G} & \frac{\partial \mu^*}{\partial G} & \frac{\partial \nu^*}{\partial G} \end{bmatrix}^\top = -K^{-1} \begin{bmatrix} \frac{\mathrm{d}G^\top}{\mathrm{d}G} \mu^\star & y^{\star\top} \otimes D(\mu^\star) & 0 \end{bmatrix}^\top, \tag{2.14f}$$

where $\frac{\mathrm{d}Q}{\mathrm{d}Q} \in \mathbb{R}^{2T \times 2T \times 2T \times 2T}$, $\frac{\mathrm{d}A^\top}{\mathrm{d}A} \in \mathbb{R}^{2T \times m \times 2T \times m}$, $\frac{\mathrm{d}A}{\mathrm{d}A} \in \mathbb{R}^{m \times 2T \times m \times 2T}$, $\frac{\mathrm{d}G^\top}{\mathrm{d}G} \in \mathbb{R}^{2T \times p \times p \times 2T}$. The derivative of a parameter with respect to itself (e.g., $\frac{\mathrm{d}q}{\mathrm{d}q}$) is an identity matrix while the cross-parameter derivative (e.g., $\frac{\mathrm{d}q}{\mathrm{d}b}$) is zero. Each gradient requires solving one linear system with the same coefficient matrix K, making the backward pass computationally efficient through LU factorization.

The full training procedure for the quadratic forecasting model is summarized in Algorithm 1.

Convergence Analysis.

We provide a formal convergence guarantee for the equality-constrained quadratic case. Consider the simplified problem with only equality constraints and scalar parameters (α, b):

$$\hat{y}_i = \arg\min_y \ \lambda_i^\top y + \frac{\alpha}{2} y^\top y \quad \text{s.t. } Ay = b\,. \tag{2.15}$$

Using the block-matrix inversion lemma, the optimal solution has the closed form:

$$\hat{y}_i = \frac{1}{\alpha}\left(A^\top (AA^\top)^{-1} A - I\right)\lambda_i + A^\top (AA^\top)^{-1} b\,. \tag{2.16}$$

Defining $k_{1i} = (A^\top (AA^\top)^{-1} A - I)\lambda_i$ and $k_2 = A^\top (AA^\top)^{-1}$, the loss becomes $L(\alpha, b) = \frac{1}{N}\sum_i \| \frac{1}{\alpha} k_{1i} + k_2 b - y_i \|^2$.

Theorem 2.1 (Convergence for equality-constrained quadratic forecast training)**.** *Consider the quadratic agent model with only equality constraints. Let $\alpha > \delta > 0$. The loss function $L(\alpha, b)$ satisfies a gradient dominance property (also known as the Polyak-Łojasiewicz condition) on any compact sublevel set $\mathscr{G}_{10\varepsilon^{-1}} = \{(\alpha, b) : L(\alpha, b) - L(\alpha^\star, b^\star) \le 10\varepsilon^{-1}\Delta_0\}$, where Δ_0 is the initial optimality gap. Furthermore, ∇L is Lipschitz continuous on $\mathscr{G}_{10\varepsilon^{-1}}$. Consequently, gradient descent with step size $\eta \le 1/L_0$ (where L_0 is the Lipschitz constant) converges linearly:*

$$L(\alpha^{(t)}, b^{(t)}) - L(\alpha^\star, b^\star) \le \left(1 - \frac{\mu_\varepsilon}{2L_0}\right)^t \Delta_0\,, \tag{2.17}$$

and $\lim_{t\to\infty} \alpha^{(t)} \to \alpha^\star$, $\lim_{t\to\infty} b^{(t)} \to b^\star$, where $(\alpha^\star, b^\star)$ are the true parameters generating the training data.

Proof sketch. Let $h(\alpha) = 1/\alpha$ and $g(x, b) = \frac{1}{N}\sum_i \| x k_{1i} + k_2 b - y_i \|^2$. Since g is strongly convex in (x, b) with constant $\mu > 0$, strong convexity implies $2\mu(g(q) - g(q^\star)) \le \|\nabla g(q)\|^2$ for all q. Applying the chain rule through h and bounding the Jacobian $\mathscr{J}_h$ on the compact sublevel set establishes the gradient dominance inequality $\mu_\varepsilon(L(\alpha, b) - L(\alpha^\star, b^\star)) \le \|\nabla L(\alpha, b)\|^2$. Combined

with Lipschitz continuity of ∇L, this yields the claimed linear convergence rate by standard gradient dominance arguments [33]. □

For the inequality-constrained case (identifying h in addition to α), the loss function admits multiple local minima, and global convergence guarantees do not hold in general. In practice, good initialization heuristics (e.g., $h_j \leftarrow \max_i(G_j y_i)$) and random restarts are effective at finding solutions with low prediction error.

2.4.3 Differentiating Generic Agent Models

While the quadratic case is important, real-world demand response agents may exhibit more complex costs, and a system operator may not know the parametric form of the disutility function in advance. We therefore extend the forecasting framework to accommodate a generic convex disutility $u_\theta(\cdot)$ by combining sequential convex programming (SCP) with input convex neural networks (ICNNs). We again use the storage formulation as the running example, but the same idea applies to other convex agent models in this chapter.

A common approach to solve problem (2.9) when u_θ is non-quadratic is sequential convex programming [52]. The basic idea is to *iteratively* form a convex quadratic approximation problem and optimize over it:

$$y^{(k+1)} = \arg\min_y \lambda^\top y + \hat{u}_\theta^{(k)}(y), \tag{2.18a}$$

$$\text{s.t. } Ay = b,\ Gy \le h, \tag{2.18b}$$

until convergence, where the cost function $u_\theta(\cdot)$ is approximated via a second-order Taylor expansion around the k-th solution $y^{(k)}$:

$$\hat{u}_\theta^{(k)}(y) = u_\theta(y^{(k)}) + {q^{(k)}}^\top (y - y^{(k)}) + (y - y^{(k)})^\top Q^{(k)} (y - y^{(k)}), \tag{2.19}$$

where $q^{(k)} = \nabla_y u_\theta\big|_{y^{(k)}} \in \mathbb{R}^{2T}$ and $Q^{(k)} = \nabla_y^2 u_\theta\big|_{y^{(k)}} \in \mathbb{R}^{2T \times 2T}$.

To guarantee that the approximation $\hat{u}_\theta^{(k)}$ is always convex (i.e., $Q^{(k)} \succeq 0$), we model the unknown disutility function u_θ using an input convex neural network (ICNN) [19]. ICNNs are neural network architectures whose output is provably convex in the input, achieved by constraining intermediate weight matrices to be non-negative and using non-decreasing convex activations. This ensures that the Hessian $Q^{(k)}$ is always positive semi-definite, so the approximated subproblem (2.18) is always a well-posed convex quadratic program.

Once the SCP iterations converge to a fixed point $y^\star = y^{(k)}$, we differentiate through the converged quadratic approximation using the same implicit differentiation approach as in Section 2.4.2. The gradient of the predicted response $y^\star$ with respect to parameters A, b, G, h in affine constraints follows directly from (2.14). To obtain the gradient with respect to the unknown parameters in the objective function, $\frac{\partial y^\star}{\partial \theta}$, we apply the chain rule:

$$\frac{\partial y^\star}{\partial \theta} = \frac{\partial y^\star}{\partial \hat{q}} \frac{\partial \hat{q}}{\partial \theta} + \frac{\partial y^\star}{\partial \hat{Q}} \frac{\partial \hat{Q}}{\partial \theta}, \tag{2.20}$$

where $\hat{q} = \nabla_{y^\star} u_\theta \in \mathbb{R}^{2T}$ and $\hat{Q} = \nabla^2_{y^\star} u_\theta \in \mathbb{R}^{2T \times 2T}$ are the gradient and Hessian of the ICNN evaluated at the fixed point. The terms $\frac{\partial \hat{q}}{\partial \theta}$ and $\frac{\partial \hat{Q}}{\partial \theta}$ are computed via automatic differentiation through the ICNN.

The complete training procedure for generic agent models is summarized in Algorithm 2. In practice, directly solving the approximate convex problem has been found to work well and is computationally efficient [52]. In the experimental section, we demonstrate that sequential convex approximation consistently converges within 20 iterations and yields accurate demand response forecasts across multiple applications.

2.5 Experimental Results

We evaluate the proposed forecasting framework on five strategic demand response settings: (1) storage behavior forecasting with a quadratic degradation cost function, (2) storage behavior forecasting with an SoC-dependent degradation cost function, (3) forecasting on

Algorithm 2. Generic Demand Response Forecast Training

Require: Dataset $D = \{(\lambda_i, z_i)\}$, $i = 1, \ldots, N$, initial parameters $\Theta^{(0)}$
  for $ite = 0, 1, \ldots, T_{\max} - 1$ **do**
    Sample batch $B \subseteq D$
    Initialize $y_i^{(1)}$ for each $i \in B$
    for $k = 1, 2, \ldots$ until convergence **do**
      Approximate $\hat{u}_\theta^{(k)}$ around $y_i^{(k)}$ for each $i \in B$ via (2.19)
      Obtain $y_i^{(k+1)}$ by solving approximated QP (2.18) for each $i \in B$
    end for
    Set $y_i^\star \leftarrow y_i^{(k)}$ (fixed point)
    Evaluate: $L = \frac{1}{|B|} \sum_{i \in B} \|\phi(y_i^\star) - z_i\|^2$
    Backward: Compute $\frac{\partial y_i^\star}{\partial \Theta}$ via (2.14) and (2.20)
    Update: $\Theta^{(ite+1)} \leftarrow \Theta^{(ite)} - \eta \sum_{i \in B} \frac{\partial L}{\partial y_i^\star} \frac{\partial y_i^\star}{\partial \Theta}$
  end for

a real-world price-arbitrage storage dataset, (4) forecasting of a building demand response agent with linear thermal dynamics, and (5) forecasting of a load aggregator with multiple responsive load types. We compare our approach with existing data-driven approaches including feedforward neural networks (MLP) and recurrent neural networks (RNN), as well as inverse optimization [17], and demonstrate that the proposed framework consistently outperforms these baselines across all five settings.

2.5.1 Quadratic Energy Storage Model

Experiment setting. We start with storage arbitrage with a quadratic disutility function following model (2.4). We consider a storage model with power rating $\overline{P} = 0.5$ MW and $\underline{P} = 0$ MW, starting with 50% SoC, and vary the energy storage duration H from $\{1\text{h}, 2\text{h}, 3\text{h}, 4\text{h}\}$ to calculate $\{\underline{E_m}, \overline{E_m}, e_0\}$. We conduct 10 experiments with different true parameter sets, sampled independently from uniform distributions:

$$c_1 \sim U[0, 20], \quad c_2 \sim U[0, 20], \quad \eta \sim U[0.8, 1.0]. \tag{2.21}$$

For each true parameter set, we take $T = 24$ with training data $N = 20$ and test data $N = 10$ using hourly averaged real-time price data randomly selected from 2019 New York City (Zone-J of New York Independent System Operator) real-time price data [53]. We report the average performance across the 10 experiments.

For the proposed gradient-based forecast training, we consider two scenarios: (i) the system operator has knowledge that the objective function is quadratic, thus using Algorithm 1 (Ours: quadratic); (ii) the system operator does not know the cost function form, thus using the generic training Algorithm 2 with ICNN (Ours: generic). Baseline methods include a four-layer multi-layer perceptron with ReLU activations (MLP) and a recurrent neural network (RNN), trained with $10\times$ more data ($N = 200$).

Forecast results. The average mean absolute error (MAE) for parameter identification across 10 experiments is shown in Table 2.1. The power limits $\overline{P}, \underline{P}$ can be inferred from dispatch data by finding the maximum/minimum values of d_t, p_t. For the quadratic identification algorithm, all parameters are directly identified using Algorithm 1. For the generic identification algorithm, we use ICNN $u_\theta(d_t)$ to model the disutility function and estimate c_1, c_2 via the first- and second-order Taylor approximations around the solution d:

$$c_1 = \frac{1}{T}\sum_t \nabla_{d_t} u_\theta(d_t), \quad c_2 = \frac{1}{T}\sum_t \nabla^2_{d_t} u_\theta(d_t).$$

The average test MSE across 10 experiments is $\mathbf{8.74 \times 10^{-5}}$ for Ours (quadratic), $\mathbf{7.37 \times 10^{-4}}$ for Ours (generic), 0.018 for MLP, and 0.017 for RNN. The quadratic forecast model recovers the exact parameters of the ground-truth storage model and predicts unseen price-response sequences nearly perfectly (Figure 2.2). The generic approach has higher error than the quadratic approach but still significantly outperforms the neural network baselines using one-tenth the training data, highlighting the value of incorporating agent structure into data-driven models for strategic demand response forecasting.

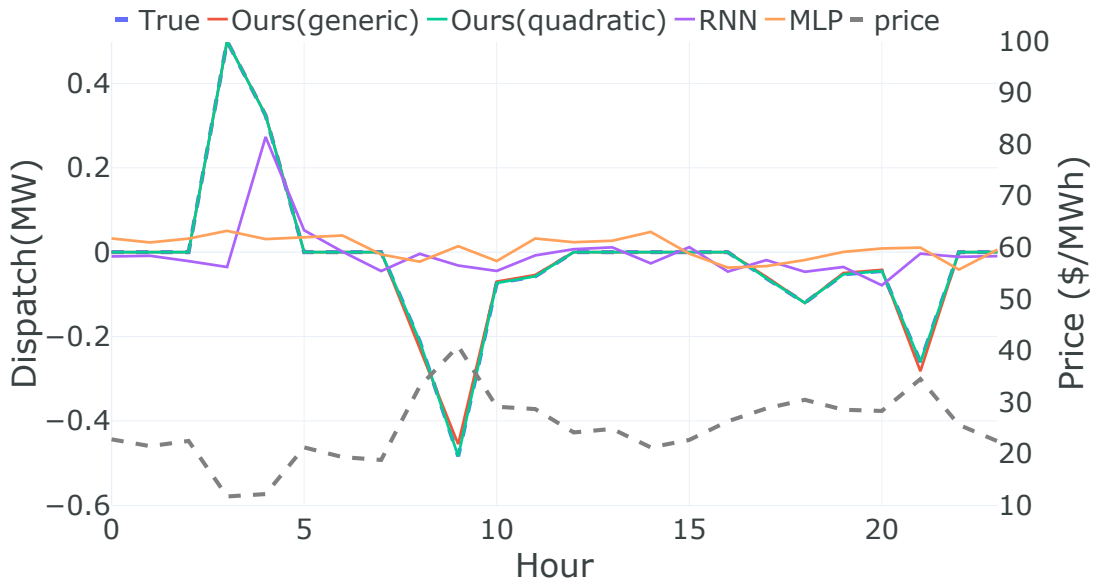


Figure 2.2. Comparison of energy storage dispatch using our methods and baselines (MLP and RNN). Our methods use 20 samples for training while MLP and RNN use 200. Ours (quadratic) identifies the true model and predicts behavior exactly, so the green line overlaps with the true response (blue).

Table 2.1. Average MAE of parameters identified compared to true parameters across 10 experiments.

	c_1	c_2	$\overline{E_m}$	$\underline{E_m}$	η
Ours (quadratic)	0.39	0.20	0.007	0.004	0.007
Ours (generic)	3.25	3.70	0.021	0.037	0.022

2.5.2 SoC-Dependent Energy Storage Model

Experiment setting. Next, we consider storage arbitrage with the SoC-dependent degradation model in (2.5). We conduct 10 experiments with different true parameter sets, using the same physical parameters and sampling distributions as the quadratic case. We compare Algorithm 1 (Ours: quadratic), Algorithm 2 (Ours: generic), MLP, and RNN, using $N = 40$ training samples for our methods and $N = 200$ for the baselines.

Forecast results. The average test MSE loss across 10 experiments is 0.032 for Ours (quadratic), **0.016** for Ours (generic), 0.039 for MLP, and 0.039 for RNN. The generic model

accurately captures the non-quadratic SoC dependence and achieves the lowest test error. The quadratic identification method incurs a model-mismatch penalty since the true storage model is non-quadratic, yet still beats the data-driven baselines. Both of our methods correctly reproduce the charge-idle-discharge patterns driven by price arbitrage, whereas MLP and RNN fail to recover the structured behavior.

2.5.3 Real-World Dataset: Queensland Battery Project

Experiment setting. We evaluate the framework on a real-world storage arbitrage dataset from the University of Queensland (UQ), which operates a 1.1 MW/2.22 MWh Tesla Powerpack 2.5 for price arbitrage and operating reserves [54]. Real-time charge/discharge behavior is dominated by arbitrage rather than rare contingency events. Using January–June 2020 data downsampled to half-hourly resolution, we form 42 sequences of length $T = 80$ (40 hours each, matching UQ's MPC look-ahead), excluding dates with regulation events or manual intervention; 22 samples are used for training and 20 for testing. We compare against MLP, RNN, and a threshold-based baseline that triggers charge/discharge when prices fall outside the training-median ratio relative to the per-sample mean.

Forecast results. Table 2.2 reports training MSE, test MSE, and test correlation. Both proposed methods outperform the baselines in test MSE and correlation. Figure 2.3 shows that MLP and RNN fail to capture optimization-driven dispatch behavior, and the threshold method only partially does. The quadratic model achieves better test accuracy than the generic model, consistent with UQ's own use of a sequential quadratic programming dispatch algorithm [54]. Despite acknowledged mismatches in the UQ report between actual and simulated battery responses, our data-driven approach forecasts the observed behavior accurately.

2.5.4 Building Demand Response

We evaluate the framework on the building demand response model from Example 3 (2.6), which captures price-responsive HVAC control subject to linear thermal dynamics.

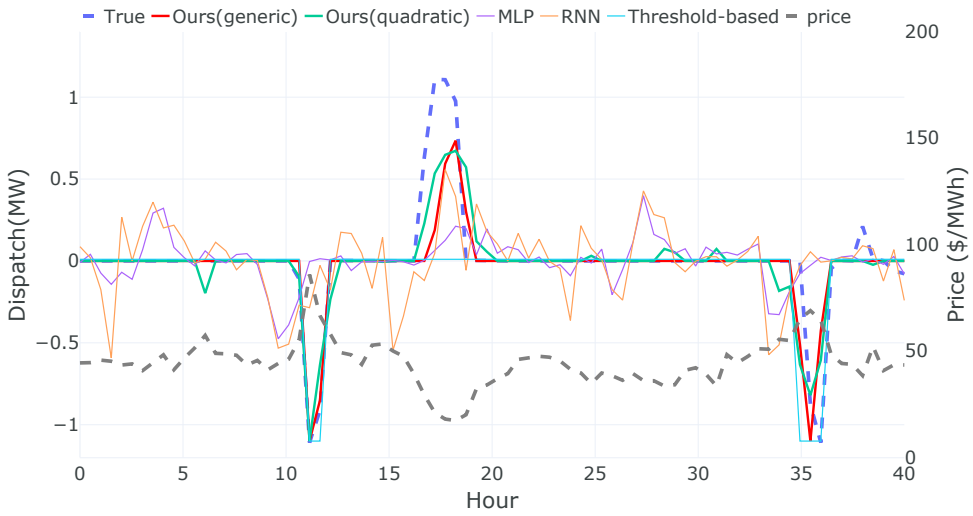


Figure 2.3. Comparison of energy storage dispatch using our methods and baselines (RNN, MLP, Threshold-based) on a 40-hour test example. Positive values represent charging and negative values represent discharging.

Table 2.2. Mean square error (MSE) and correlation score.

	Training MSE	Test MSE	Test Correlation
Ours (quadratic)	**0.044**	**0.042**	**0.74**
Ours (generic)	0.054	0.053	0.68
Threshold-based	0.088	0.070	0.57
MLP	0.047	0.096	0.34
RNN	0.092	0.109	0.27

Experiment setting. The building is characterized by thermal resistance $R = 2\,^{\circ}$C/kW, thermal capacitance $C = 10$ kWh/°C, set-point $\sigma_r = 20\,^{\circ}$C, deadband half-width $\Delta = 1\,^{\circ}$C, coefficient of performance $\tau = 2.5$, and rated power $P = 5.4$ kW. Following [50], we perturb all physical parameters by a multiplicative uniform factor $U(1-h, 1+h)$ with $h = 0.1$ to generate per-building variation. The identification parameters $\Omega^2 = \{\rho, \overline{\sigma}, \underline{\sigma}, \overline{p}, \alpha_1, \alpha_3\}$ are computed as:

$$\underline{\sigma} = \sigma_r - \Delta, \quad \overline{\sigma} = \sigma_r + \Delta, \quad \overline{p} = P, \quad \alpha_1 = 1 - \tfrac{1}{RC}, \quad \alpha_3 = (1-\alpha_1)\tau R, \tag{2.22}$$

with temperature violation penalty $\rho = 0.02$. We use European electricity price data from ENTSO-E and ambient temperature traces to generate price-demand pairs by solving (2.6).

The dataset covers 20 days; we use 10 days for training and 10 for testing. We consider two observation scenarios: OptNet[1], where both power usage p_t and indoor temperature σ_t are observed; and OptNet[2], where only power usage p_t is observed.

Identification results. Across 10 independent experiments, the six identified building parameters are recovered with low median MAPE: ρ at 0.32%, $\overline{\sigma}$ at 0.001%, α_1 at 0.008%, α_3 at 0.04%, and $\overline{p}$ at 0.009%. The single exception is the lower temperature bound $\underline{\sigma}$ (median MAPE 3.7%), which is expected: the experiments use summer data, so the indoor temperature never reaches the lower comfort bound, leaving that constraint non-binding and its parameter unidentifiable from the data alone.

Forecast results. The average test MSE across 10 experiments for power prediction is 0.538 for MLP, 0.535 for RNN, 0.016 for OptNet[2], and **0.004** for OptNet[1]. The proposed method achieves the lowest forecast error in all experiments. Neural networks perform somewhat better on this application than on the storage task because building power consumption has strong correlation with outdoor temperature (correlation 0.77 in the dataset), and neural networks can partially capture this pattern when temperature is provided as an input. However, the optimization-guided approach still outperforms the baselines by a large margin by exploiting the thermal dynamics structure. When only power usage is observed (OptNet[2]), accuracy degrades modestly but remains far superior to the neural network baselines.

2.5.5 Aggregator with Multiple Responsive Loads

We evaluate the framework on the aggregator model from Example 4 (2.7), where K types of price-responsive loads are coordinated and only the aggregate load $z_t = \sum_k y_{k,t}$ is observable.

Experiment setting. We follow the experimental setup of [17] with $K = 3$ load types and $T = 12$ time steps. We generate $N = 50$ training price-demand pairs by solving (2.7) with known true parameters $\{\alpha_{k,t}, M_k, P_{k,t}\}$. To test robustness to measurement noise, the observed aggregate consumption is perturbed as $z_t^i = U(1-\pi, 1+\pi)\sum_k y_{k,t}^i$ with noise levels $\pi \in \{0, 0.1, 0.2\}$. We compare against inverse optimization (IO) [17], MLP, and RNN. To assess the full potential

Table 2.3. RSE comparison for different training data sizes on the aggregator model ($K = 3$, noise level $\pi = 0.2$).

Training samples	Ours	MLP	RNN
50	**0.078**	0.705	0.904
100	0.083	0.616	0.851
1000	—	0.324	0.201
2000	—	0.316	0.108
3000	—	0.311	**0.104**

of neural network baselines, MLP and RNN are given up to 3000 training samples, while our approach uses only 50.

Forecast accuracy is measured by the relative squared error (RSE):

$$\text{RSE} = \frac{\sum_{t=1}^{T}(z_t - \hat{z}_t)^2}{\sum_{t=1}^{T}(z_t - \bar{z})^2}, \tag{2.23}$$

where $\hat{z}_t$ is the predicted aggregate load and $\bar{z}$ is the mean of the true aggregate load over the test set.

Results. Table 2.3 summarizes RSE as a function of training data size. With only 50 samples, our approach achieves RSE of 0.078, while MLP and RNN require 3000 samples to reach their best RSE of 0.311 and 0.104, respectively. As the noise level π increases, the RSE of all baselines degrades substantially, whereas our framework maintains low error with small variance, demonstrating its robustness to noisy aggregate observations.

An important distinction for the aggregator case is that exact parameter recovery is not possible: because only the aggregate z_t is observed and the objective is linear in the loads, multiple parameter combinations can produce the same aggregate behavior. Nevertheless, the gradient-based approach finds a set of parameters that minimize aggregate prediction loss, yielding accurate forecasts even when individual schedules cannot be identified.

2.6 Chapter Summary

This chapter developed an optimization-guided framework for forecasting price-responsive demand response behavior from historical observations. The framework is grounded in a key structural observation: real-world demand response agents, such as energy storage systems, HVAC-controlled buildings, and responsive load aggregators, make decisions by solving structured optimization problems rather than by following arbitrary black-box mappings. For storage and building demand response, degradation, comfort, or temperature-violation penalties are naturally modeled as quadratic or smooth convex costs. In these settings, the optimal response is uniquely determined and varies smoothly with the cost parameters. This smoothness is precisely what enables the implicit-differentiation approach: the KKT-based Jacobian in (2.13) is well-defined, non-degenerate, and admits efficient computation via a single LU factorization. Thus, the method succeeds here not despite the optimization layer, but *because of* the convex structure that these energy-system agents naturally possess. For load aggregators and other demand response agents whose objectives are closer to linear programs, the role of the framework is more limited. It remains useful for *forecast-oriented identification*: by embedding known energy, power, and aggregation constraints, the learned model can reproduce aggregate responses from few samples even when the underlying private parameters are not uniquely recoverable. However, the resulting parameters should be interpreted as predictive latent parameters, not necessarily as the true economic coefficients of the agent.

Boundaries of the approach.

The framework has two principal limitations that demarcate the class of agents to which it applies.

Pure linear-program agents. The most fundamental boundary arises when the agent's objective is linear in the decision variables, i.e., $f(y;\lambda,\theta) = c(\lambda,\theta)^\top y$ with affine constraints. In this case the optimal solution sits at a vertex of the feasibility polytope and is piecewise constant as a function of the objective coefficients. When the objective coefficients vary without changing

the optimal basis, the optimizer remains locally constant, leading to zero gradients almost everywhere with respect to the objective parameters. Consequently, gradient descent receives no informative signal for identifying those parameters, even though the forward optimization problem itself may still produce accurate forecasts. The KKT matrix K in (2.13) can also become singular at degenerate vertices, breaking the implicit-differentiation formula. This is the core reason the chapter's strongest identification claims apply to agents with quadratic or strictly convex disutility costs: strict convexity regularizes the solution map and yields informative, well-conditioned gradients. Demand response agents with purely linear costs — for example, a price-taking load that simply maximizes net revenue with no degradation, utility curvature, or comfort penalty — fall outside the regime where objective coefficients are identifiable by this gradient signal. For such agents, our approach should be used primarily for response prediction under the embedded operational constraints, or augmented with surrogate smoothing, regularization, perturbation-based gradient estimation, or additional observations that reveal basis changes.

Non-convex agent objectives. The second boundary is non-convexity. The framework requires the agent's forward problem to be convex so that KKT conditions are both necessary and sufficient for optimality and so that the SCP approximation converges reliably. Agents with non-convex costs (e.g., unit-commitment-style fixed charges, combinatorial on/off decisions, or battery aging models with non-convex cycle-depth terms) violate this requirement. In such cases the implicit-differentiation step may correspond to a local rather than global optimum, and multiple KKT points may coexist, making the gradient signal ambiguous.

These boundaries suggest natural directions for future work, including surrogate-smoothing techniques for LP-structured agents and mixed-integer extensions for combinatorial demand response resources.

Acknowledgments

The authors would like to thank the University of Queensland for making the Tesla Battery Project dataset publicly available.

Chapter 2, in full, contains material from Y. Bian, N. Zheng, Y. Zheng, B. Xu, and Y. Shi, "Demand Response Model Identification and Behavior Forecast with OptNet: A Gradient-Based Approach," *Proceedings of the Thirteenth ACM International Conference on Future Energy Systems (e-Energy '22)*, 2022, pp. 268–279; and Y. Bian, N. Zheng, Y. Zheng, B. Xu, and Y. Shi, "Predicting Strategic Energy Storage Behaviors," *IEEE Transactions on Smart Grid*, 2024. The dissertation author was the primary investigator and author of these papers.

Chapter 3

Dynamics-Guided Models for Energy-Efficient Building Control I: PDE-based Identification and Control

This chapter develops a physics-informed learning and control framework for building HVAC systems. The spatiotemporal dynamics of airflow, temperature, and CO_2 concentration are modeled using partial differential equations (PDEs), specifically the Navier-Stokes and convection-diffusion equations. Both system identification and optimal control are formulated as PDE-constrained optimization problems and solved via gradient descent using the adjoint method. On a representative case study, the framework reduces energy consumption by 52.6%, 36.4%, and 7.9% relative to a maximum airflow policy, reinforcement learning, and MPC with ODE models, respectively, while satisfying all modeled safety-critical air quality and thermal comfort constraints in the reported experiments.

3.1 Introduction

Buildings account for over 40% of global energy consumption [55], with HVAC systems responsible for up to 50% of a building's usage [4]. Because occupants spend roughly 90% of their lives indoors [56], control strategies must balance energy efficiency against thermal comfort and air quality [57, 58]. Post-COVID guidelines amplified this tension: high airflow reduces pathogen exposure [59, 60], but operating HVAC at maximum airflow [61, 62] drives

energy consumption 2–2.5$\times$ above nominal. We use CO_2 as a tractable proxy for indoor air quality [63, 64, 26, 65], since direct viral concentration is rarely measurable.

Existing HVAC control methods fall broadly into rule-based control [66, 67, 68], optimization-based control such as Model Predictive Control (MPC) [69, 26, 65, 70], and learning-based control via reinforcement learning (RL) [71, 72, 59, 73, 74, 27]. RL is appealing for hard-to-model dynamics but is sample-hungry – as much as 47.5 simulated years to match a traditional controller [15] – and lacks hard-constraint guarantees. We take an optimization-based approach in which HVAC dynamics are modeled as partial differential equations with data-fitted parameters, and control actions are derived by solving a PDE-constrained optimization. This reduces data requirements while enabling explicit enforcement of hard health/safety constraints under the modeled dynamics.

3.1.1 Contributions

To the best of our knowledge, this is the first PDE-based learning and control framework for buildings that simultaneously optimizes energy consumption while explicitly enforcing air-quality and thermal constraints. Compared to ODE-based work [26, 65, 27, 74], the PDE formulation captures spatial-temporal airflow. Compared to existing PDE/CFD work [75, 76, 1, 59, 77], which focuses on prediction, occupancy detection, or building design, we focus on real-time optimal control with accurate airflow modeling. The closest prior work [59] minimizes infection risk under simplified uniform flow and optimizes the velocity field directly, which is impractical for real buildings.

The proposed framework is illustrated in Figure 3.1: the system state $(\vec{v}, C, T_e)$[1] is governed by the Navier–Stokes equations for airflow and convection–diffusion PDEs for CO_2 and temperature. The system-learning task estimates unknown building parameters; the control task minimizes energy consumption while ensuring thermal comfort and air quality. We solve both tasks with a gradient-descent approach based on the adjoint method, achieving a 52.6%

[1] We use CO_2 as an air-quality proxy; the framework extends to PM2.5, PM10, and airborne pathogens [78].

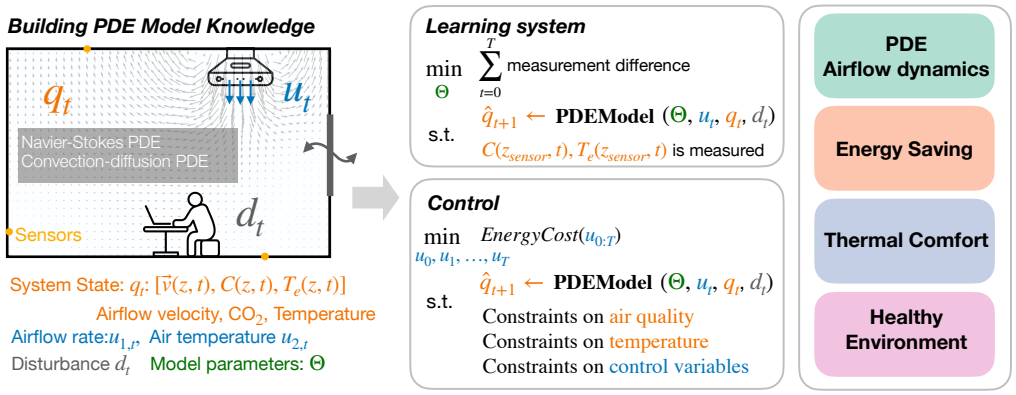


Figure 3.1. Schematic of the proposed PDE-based learning and control framework for healthy and energy-efficient buildings. Left: we model the airflow, CO_2, and temperature dynamics with PDEs, including Navier-Stokes and convection-diffusion PDEs, with unknown model parameters denoted by Θ. HVAC control and external disturbances are represented as u_t and d_t, respectively. Right: we present the formulation of the learning and control problems as PDE-constrained optimization. The proposed framework models spatiotemporal dynamics with PDEs, leading to improved energy efficiency, ensured thermal comfort, and a healthy indoor environment.

energy reduction relative to the maximum-airflow policy and 36.4%/7.9% over RL and MPC with ODE models, while RL and MPC-ODE occasionally violate safety constraints and our approach does not.

We organize the remainder of this chapter as follows. Section 3.2 surveys related work. Section 3.3 describes the building PDE models. Section 3.4 presents the problem formulation. Section 3.5 provides the solution method. Section 3.6 presents case studies. Section 3.7 concludes this chapter.

3.2 Related Work

While HVAC control research has overwhelmingly focused on thermal comfort and energy efficiency, only a small fraction of studies addresses indoor air quality [57]. Post-COVID work has begun to close this gap, primarily through ODE-based models of CO_2 or particulate concentrations. Li et al. [26] modeled CO_2 dynamics with ODEs and applied MPC to optimize the supply airflow for energy efficiency under air-quality constraints. Zhang et al. [65] similarly combined ODE-based CO_2 modeling with MPC solved via a genetic algorithm. Shang et al. [27]

used an ODE model for PM2.5 concentrations and applied RL to control the air purifier exchange rate. Heo [74] proposed an RL-based smart ventilation system with PM10 modeled by an ODE. However, ODE models cannot represent the spatial structure of airflow and can mask localized high-concentration zones [59, 79].

To capture spatial dynamics, prior work has used PDE-based models, including the Navier–Stokes and convection–diffusion equations and full computational fluid dynamics (CFD) simulations. Lau et al. [75] used advection–diffusion–reaction equations to predict spatiotemporal infection risk. Narayanan et al. [76] employed CFD to model airborne pathogen concentration in a classroom and inform air purifier placement. He et al. [77] integrated CFD to investigate the effect of room geometry on ventilation performance. Jin et al. [1] applied a convection PDE to model CO_2 for occupancy detection at a fixed airflow rate. Koga et al. [80] employed the Navier–Stokes and convection–diffusion equations for spatiotemporal airflow and temperature modeling, though their study focused solely on parameter identification. Hosseinloo et al. [59] modeled pathogen concentration with convection–diffusion equations and optimized the velocity field via RL; however, optimizing the entire velocity field is impractical for real buildings, where control is limited to the boundary airflow velocity at the supply air vents. These PDE/CFD studies focus on pathogen prediction, occupancy detection, or building design rather than real-time optimal control.

3.3 Building PDE Models

In this section, we describe the building dynamic models that characterize the airflow velocity, temperature field, and CO_2 concentration field, which are modeled by PDEs. The notations are summarized in Table 3.1.

3.3.1 Models for Airflow, Temperature, and CO_2

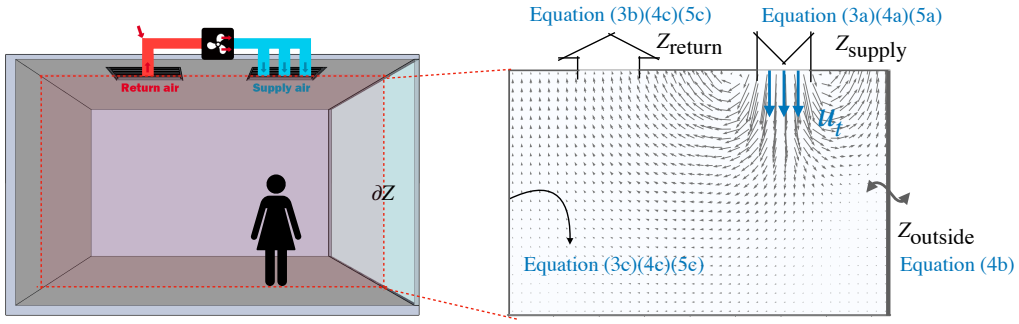
Let us consider a typical office environment where the temperature and CO_2 levels dynamically evolve due to a set of contributing factors including the physical layout of the

Table 3.1. Notation used for PDE models.

Notation	Description
$\mathscr{Z}$	domain of PDEs
$z = (x, y)$	spatial coordinate
$\partial \mathscr{Z}$	boundary of the field
$\mathscr{Z}_{\text{supply}}$	air supply vent positions
$\mathscr{Z}_{\text{return}}$	air return vent positions
$\mathscr{Z}_{\text{outside}}$	wall adjacent to the exterior
$\mathscr{T} = [0, \overline{T}], t$	the time set and time index
$\vec{v}(z,t)$	airflow velocity field
$p(z,t)$	pressure field
$T_e(z,t), d_{T_e}(z,t)$	temperature field and heat source
$C(z,t), d_C(z,t)$	CO_2 field and CO_2 source
$g(z,t)$	the number of people in a room
q_t	system state $q_t = [\vec{v}(z,t), C(z,t), T_e(z,t)]$
$T_{\text{ambient}}(t)$	ambient temperature
ρ	fluid density
C_{fresh}	CO_2 level for fresh air in (3.6a)
Model parameters	
ν	kinematic viscosity
k_{T_e}	diffusion coefficient of temperature
k_C	diffusion coefficient of CO_2
α	the re-circulation rate in (3.6a)
α_{T_e}	heat source coefficient in (3.3)
α_C	CO_2 source coefficient in (3.3)
Control variables	
$u_{1,t} \in \mathbb{R}$	supply airflow rate
$u_{2,t} \in \mathbb{R}$	supply air temperature

space, HVAC control actions (supply airflow rate and temperature), human occupancy, and external weather conditions. The physical representation of the considered room is illustrated in Figure 3.2. This room bears close resemblance to other typical indoor spaces, with a ventilation system including air supply and air return vents on the ceiling.

We employ PDEs to model the spatiotemporal dynamics of the airflow velocity field, and the resulting temperature and CO_2 fields. We denote the domain of the PDE by $\mathscr{Z} \subset \mathbb{R}^2$ (or $\mathbb{R}^3$), which represents a confined region, e.g., a meeting room. The time domain is $\mathscr{T} = [0, \overline{T}] \subset \mathbb{R}^+$. Spatial coordinate and time are denoted by $z \in \mathscr{Z}$ and $t \in \mathscr{T}$. We denote the airflow velocity



(a) Physic representation (b) Airflow velocity and boundary conditions

Figure 3.2. (a) The physical representation of the simulation testbed: a typical room with a ventilation system including air supply and air return vents on the ceiling. All the walls are insulated, and the right side features a glass window wall. (b) We define a 2D region and model it using 2D PDEs. The figure visualizes the airflow velocity, governed by the Navier-Stokes equations. All boundary conditions are highlighted in blue text.

field $\vec{v}(z,t) : \mathscr{Z} \times \mathscr{T} \to \mathbb{R}^2$, the temperature field $T_e(z,t) : \mathscr{Z} \times \mathscr{T} \to \mathbb{R}$ and the CO_2 field $C(z,t) : \mathscr{Z} \times \mathscr{T} \to \mathbb{R}$.

The airflow velocity field is modeled by the Navier-Stokes equations,

$$\nabla \cdot \vec{v} = 0, \qquad \text{Continuity equation} \qquad (3.1a)$$

$$\frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \vec{v} + \vec{g}, \qquad \text{Momentum equation} \qquad (3.1b)$$

where $\vec{v}$ is the velocity vector, ν is the kinematic viscosity of the airflow, ρ is the fluid density, p is the pressure field, and $\vec{g} = [0, -9.8]^\mathsf{T}$ is the gravitational force.

The thermal and CO_2 dynamics are modeled by the convection-diffusion equations,

$$\frac{\partial T_e}{\partial t} + \vec{v} \cdot \nabla T_e = k_{T_e} \nabla^2 T_e + d_{T_e}(z,t), \qquad \text{Temperature} \qquad (3.2a)$$

$$\frac{\partial C}{\partial t} + \vec{v} \cdot \nabla C = k_C \nabla^2 C + d_C(z,t), \qquad CO_2 \qquad (3.2b)$$

where k_{T_e}, k_C are the diffusion coefficients for the temperature and CO_2 fields, and $d_{T_e}(z,t), d_C(z,t)$ are the heating source and CO_2 source, respectively. We employ proportional

models to represent both the heat and CO_2 sources in a room [59, 81, 24],

$$d_{T_e}(z,t) = \alpha_{T_e} g(z,t)\,, \quad d_C(z,t) = \alpha_C g(z,t)\,, \tag{3.3}$$

where $g(z,t)$ denotes the number of people at position z at time step t and α_{T_e}, α_C are the respective coefficients for heat and CO_2 source per occupant. Our framework can also take more sophisticated models of occupants' effects, such as polynomial models [71].

3.3.2 Boundary Conditions

In this work, we consider HVAC control as boundary control [59, 79], with the specific boundary conditions detailed in Figure 3.2(b). The boundary of the field is denoted by $\partial\mathscr{Z}$, which represents the walls, the ceiling, and the ground of the room. Within this boundary, the positions of the air supply vent and air return vent are specified as $\mathscr{Z}_{\text{supply}}$ and $\mathscr{Z}_{\text{return}}$. We denote the control variable $u_t \in \mathbb{R}^2$, where $u_{1,t} \in \mathbb{R}$ represents the airflow rate at the supply vent (pointing in a downward direction), and $u_{2,t} \in \mathbb{R}$ represents the supply air temperature at the supply vent.

Boundary conditions for airflow velocity field.

$$\vec{v}(z,t) = -u_{1,t} \cdot e_y, \forall z \in \mathscr{Z}_{\text{supply}}, \tag{3.4a}$$

$$\vec{n} \cdot \nabla \vec{v} = 0, \forall z \in \mathscr{Z}_{\text{return}}, \tag{3.4b}$$

$$\vec{v} = 0, \forall z \in \partial\mathscr{Z} \setminus (\mathscr{Z}_{\text{supply}} \cup \mathscr{Z}_{\text{return}}). \tag{3.4c}$$

Constraint (3.4a) specifies that the airflow velocity at the supply vent $\mathscr{Z}_{\text{supply}}$ is controlled by $u_{1,t}$ (m/s), where $e_y = [0,1]^\top$ is the unit vector in the y-upward direction. Constraint (3.4b) sets the Neumann boundary conditions at the return vent, and constraint (3.4c) applies Dirichlet conditions to all other boundaries by setting the airflow velocity as zero [79].

Boundary conditions for temperature field.

$$T_e(z,t) = u_{2,t}, \forall z \in \mathscr{Z}_{\text{supply}}, \tag{3.5a}$$

$$T_e(z,t) = T_{\text{ambient}}(t), \forall z \in \mathscr{Z}_{\text{outside}}, \tag{3.5b}$$

$$\vec{n} \cdot \nabla T_e = 0, \forall z \in \partial \mathscr{Z} \setminus (\mathscr{Z}_{\text{supply}} \cup \mathscr{Z}_{\text{outside}}). \tag{3.5c}$$

Constraint (3.5a) states that the temperature at the supply air vent is controlled as $u_{2,t}$ (°C). Constraint (3.5b) represents the temperature of the right window wall $\mathscr{Z}_{\text{outside}}$ influenced by the ambient temperature $T_{\text{ambient}}(t)$. Constraint (3.5c) sets the Neumann boundary conditions for solid walls as insulated surfaces [82].

Boundary conditions for CO_2 field.

$$C(z,t) = \alpha \cdot C_{\text{fresh}} + (1-\alpha) \cdot \frac{1}{\mathscr{Z}} \int_{\mathscr{Z}} C(z,t) \mathrm{d}z, \forall z \in \mathscr{Z}_{\text{supply}}, \tag{3.6a}$$

$$\vec{n} \cdot \nabla C = 0, \forall z \in \partial \mathscr{Z} \setminus \mathscr{Z}_{\text{supply}}. \tag{3.6b}$$

Constraint (3.6a) dictates that the CO_2 concentration at the supply air vent is a *mixture* of the CO_2 concentration of fresh air C_{fresh} and the CO_2 concentration of recirculated air within the building $\frac{1}{\mathscr{Z}} \int_{\mathscr{Z}} C(z,t) \mathrm{d}z$. The parameter α represents the re-circulation rate that can vary among different buildings. In this work, $C_{\text{fresh}} = 400\,\text{ppm}$ [26] and α is a parameter to be identified from data. Constraint (3.6b) establishes the Neumann boundary conditions for all boundaries except the supply vent [59].

3.4 Problem Formulation

Given the PDE model knowledge described in Section 3.3, we formulate the system learning and control problems as PDE-constrained optimization problems. The goal is to learn the unknown parameters in the PDE system and develop control algorithms to optimize energy efficiency, thermal comfort, and indoor air quality.

3.4.1 Learning the System Model

A fundamental challenge in controlling building systems is that the relationship between airflow, temperature, and CO_2 concentration and the HVAC control actions is governed by a set of nonlinear PDEs whose parameters depend on detailed building characteristics that are difficult to measure in practice. We consider the set of system model parameters to be identified: $\Theta = \{\nu, k_{T_e}, k_C, \alpha, \alpha_{T_e}, \alpha_C\}$. It is important to note that for incompressible flow, the fluid density ρ in (3.1b) is considered constant, thus it has no impact on the velocity, temperature, and CO_2 concentration values.

Given historical temperature and CO_2 records obtained from sensors placed at specific locations, our goal is to learn the unknown system parameters Θ that minimize the difference

between actual historical records and predicted values based on estimated parameters,

$$\min_{\Theta} \quad \sum_{t} \Big(\|T_e(z_{sensor},t) - \widehat{T}_e(z_{sensor},t)\|^2 \tag{3.7a}$$

$$+\|C(z_{sensor},t) - \widehat{C}(z_{sensor},t)\|^2 \Big) \tag{3.7b}$$

$$\text{s.t. } \Theta = \{\nu, k_{T_e}, k_C, \alpha, \alpha_{T_e}, \alpha_C\}, \tag{3.7c}$$

$$\nabla \cdot \widehat{\vec{v}} = 0, \tag{3.7d}$$

$$\frac{\partial \widehat{\vec{v}}}{\partial t} + \widehat{\vec{v}} \cdot \nabla \widehat{\vec{v}} = -\frac{1}{\rho}\nabla \hat{p} + \nu \nabla^2 \widehat{\vec{v}} + \vec{g}, \tag{3.7e}$$

$$\frac{\partial \widehat{T}_e}{\partial t} + \widehat{\vec{v}} \cdot \nabla \widehat{T}_e = k_{T_e} \nabla^2 \widehat{T}_e + \alpha_{T_e} g(z,t), \tag{3.7f}$$

$$\frac{\partial \widehat{C}}{\partial t} + \widehat{\vec{v}} \cdot \nabla \widehat{C} = k_C \nabla^2 \widehat{C} + \alpha_C g(z,t), \tag{3.7g}$$

$$\widehat{C}(z,t) = \alpha \cdot C_{\text{fresh}} + (1-\alpha) \cdot \frac{1}{\mathscr{Z}} \int_{\mathscr{Z}} \widehat{C}(z,t) \mathrm{d}z, \forall z \in \mathscr{Z}_{\text{supply}}, \tag{3.7h}$$

(3.4)(3.5)(3.6b) (other boundary conditions)

where $z_{sensor} \subset \mathscr{Z}$ represents the set of sensor positions. The predicted temperature and CO_2 $\widehat{T}_e(z_{sensor},t), \widehat{C}(z_{sensor},t)$ are driven by the PDEs with estimated parameters Θ, and $T_e(z_{sensor},t)$, $C(z_{sensor},t)$ are the ground-truth measurements recorded by sensors.

3.4.2 Control Problem Formulation

Now we are able to formulate the HVAC control problem. The airflow dynamics (3.1), temperature and CO_2 dynamics (3.2), and equation (3.6a) are driven under the estimated param-

eters obtained by solving the system learning problem (3.7).

$$\min_{u_t} \quad \int_{t=0}^{\overline{T}} l_{\text{control}}(t)\,\mathrm{d}t \tag{3.8a}$$

$$\text{s.t.} \quad (3.1) \qquad \text{(Airflow dynamics)}$$

$$(3.2) \qquad \text{(Temperature and } CO_2 \text{ dynamics)}$$

$$(3.4)\ (3.5)\ (3.6) \qquad \text{(Boundary conditions)}$$

$$\underline{u_1} \le u_{1,t} \le \overline{u_1}, \underline{u_2} \le u_{2,t} \le \overline{u_2}, \forall t, \tag{3.8b}$$

$$C(z,t) \le C_{\max}, \forall z \in \mathscr{Z}, t \in \mathscr{T}, \tag{3.8c}$$

$$T_{\min} \le \frac{1}{\mathscr{Z}} \int_z T_e(z,t)\mathrm{d}z \le T_{\max}, \forall t \in \mathscr{T}. \tag{3.8d}$$

We define the loss objective as an integral of $l_{\text{control}}(t)$ over time t. $l_{\text{control}}(t)$ can include costs such as energy consumption, comfort score reflecting indoor air quality and temperature, and operational expenses. Constraint (3.8b) specifies control action bounds. Constraints (3.8c) and (3.8d) ensure that CO_2 concentration levels do not exceed healthy limits at *all* locations and that the average room temperature remains within occupants' thermal comfort range. Average temperature is a practical measure for occupant comfort since people are less sensitive to minor temperature variations. However, CO_2 is strictly controlled everywhere due to higher sensitivity and infection risks associated with poor air quality [64].

To handle the control constraint (3.8b), we use a projected gradient method. To deal with the state constraints (3.8c)(3.8d), we utilize log barrier functions for the inequality constraints,

$$\begin{aligned} \bar{L}_{\text{control}} = \int_{t=0}^{\overline{T}} \Big[& l_{\text{control}}(t) - \alpha_1 \log\left(C_{\max} - \max_z(C(z,t))\right) \\ & - \alpha_2 \log\left(T_{\max} - \frac{1}{\mathscr{Z}} \int_z T_e(z,t)\mathrm{d}z\right) \\ & - \alpha_2 \log\left(\frac{1}{\mathscr{Z}} \int_z T_e(z,t)\mathrm{d}z - T_{\min}\right)\Big]\,\mathrm{d}t \end{aligned} \tag{3.9}$$

where α_1, α_2 are the weight factors. In this study, we define $l_{\text{control}}(t)$ to account for both energy consumption and deviations in control variables:

$$l_{\text{control}}(t) = Energy(t) + \alpha_3 \dot{u}_t \tag{3.10}$$

where α_3 is the weight factor. For energy consumption, we use the L1 norm of the airflow rate $u_{1,t}$ and the difference between the supply air temperature $u_{2,t}$ and the default air temperature $u_{2,\text{default}}$ as proxies for energy consumption [83, 84, 67]:

$$Energy(t) = w_1 \|u_{1,t}\|_1 + w_2 \|u_{2,t} - u_{2,\text{default}}\|_1 \tag{3.11}$$

where w_1, w_2 are weight factors that vary with building types, and we set $u_{2,\text{default}} = 14.4°\text{C}$ according to [67]. Other objective functions (e.g., time-of-use electricity price, peak demand charge) can also be flexibly included in the control problem based on the building system operation criteria.

3.5 Solution Approach

We propose a gradient descent method for solving the system learning problem (3.7) and HVAC control problem (3.8), based on the adjoint method [85]. An overview of the overall solution algorithm is illustrated in Figure 3.3.

3.5.1 Adjoint Method for PDE-Constrained Optimization

We consider the HVAC control problem in (3.8) as an illustrative example for explaining the proposed solution method. Let us introduce a short-hand notation to represent the PDE dynamics in (3.1)–(3.2),

$$\frac{\partial q_t}{\partial t} = \mathscr{P}\left(q_t, \frac{\partial q_t}{\partial z}, \frac{\partial^2 q_t}{\partial z^2}, u_t, d_t; \Theta\right)$$

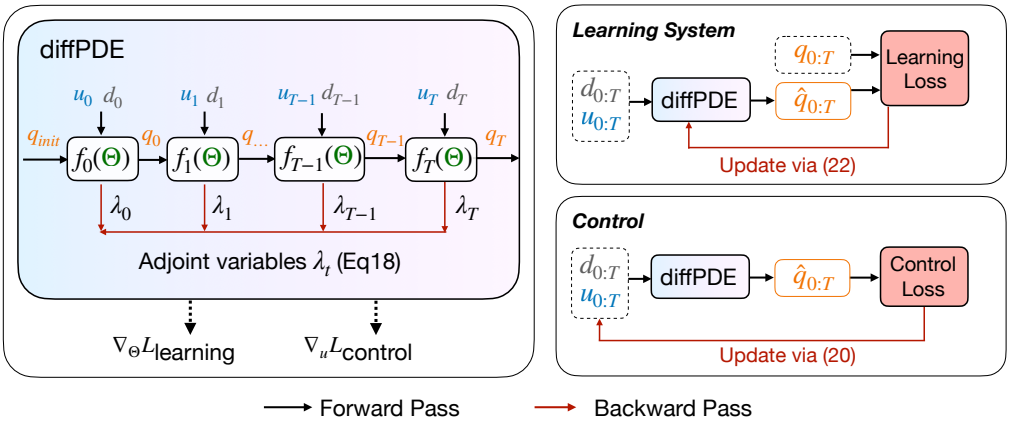


Figure 3.3. Algorithm for the PDE-based learning and control framework. For the "diffPDE" block, "diff" signifies "differentiable", denoting that it allows the computation of gradients. Left: we first solve the PDEs in a forward pass, then solve the adjoint variables $\lambda_t, t = 0, \ldots, T$ in a backward pass with (3.17). The gradient of model learning loss w.r.t. system parameters, and the gradient of control loss w.r.t. control actions are computed using (3.21a) and (3.18) using the adjoint variables. Right: updates of the model parameters and control via the obtained gradients.

where u_t specifies the control action at time t, $q_t = [\vec{v}(z,t), C(z,t), T_e(z,t)]$ denotes the system state at time t, $d_t = [g(z,t), T_{\text{ambient}}(t)]$ represents the disturbance including occupancy and ambient temperature, and $\mathscr{P}$ models the physical behavior of the PDE system defined in (3.1)–(3.2) for a given set of system parameters Θ.

We consider a temporal discretization interval Δt, with the total number of discretization steps as $T = \frac{\overline{T}}{\Delta t}$, and denote the initial system state as $q_{\text{init}} = q_{-1}$. The discrete-time version of optimal control problem (3.8) with the log barrier functions for the state constraints is

$$\min_{u_0, u_1, \ldots, u_T} \quad L_{\text{control}} := \sum_{t=0}^{T} l(q_t, u_t) \tag{3.12a}$$

$$\text{s.t.} \quad q_t = q_{t-1} + \Delta t \cdot \mathscr{P}\left(q_{t-1}, \frac{\partial q_{t-1}}{\partial z}, \frac{\partial^2 q_{t-1}}{\partial z^2}, u_t, d_t; \Theta\right), \tag{3.12b}$$

$$\text{(3.4) (3.5) (3.6)} \qquad \text{(Boundary conditions)}$$

$$\underline{u} \le u_t \le \overline{u}, \quad t = 0, \ldots, T, \tag{3.12c}$$

where $l(q_t, u_t) := \bar{l}(q_t, u_t)\Delta t$, with $\bar{l}(q_t, u_t)$ as the integral term in (3.9). Equation (3.12b) describes the discrete-time system evolution using the first-order Euler method. For further simplicity, we define the right hand side of (3.12b) as f_t:

$$q_t = f_t\left(q_{t-1}, u_t, d_t; \Theta\right). \tag{3.13}$$

We use a projected gradient descent method to solve the constrained optimization in (3.12). The critical challenge is computing $\nabla_{u_t} L_{\text{control}}, \forall t$ since all state and control variables are constrained by the system dynamics (3.13). We leverage the adjoint method [85, 86], which uses adjoint variables to enforce the dynamics constraints and then applies a projection operator to enforce the control constraints (3.12c).

Taking the total derivatives of both sides of (3.12a),

$$\delta L_{\text{control}} = \sum_{t=0}^{T} \left(\frac{\partial l}{\partial q_t} \delta q_t + \frac{\partial l}{\partial u_t} \delta u_t \right). \tag{3.14}$$

By taking the total derivative of equation (3.13) and re-arranging,

$$\delta q_t - \frac{\partial f_t}{\partial q_{t-1}} \delta q_{t-1} - \frac{\partial f_t}{\partial u_t} \delta u_t - \frac{\partial f_t}{\partial d_t} \delta d_t = 0. \tag{3.15}$$

The left side of equation (3.15) is always zero. The adjoint method multiplies the adjoint variables λ_t by the left side of equation (3.15) and adds to equation (3.14). As a result,

$$\begin{aligned}
\delta L_{\text{control}} = &\sum_{t=0}^{T}\left(\frac{\partial l}{\partial q_t}\delta q_t + \frac{\partial l}{\partial u_t}\delta u_t\right) + \\
&\sum_{t=0}^{T}\lambda_t^{\mathsf{T}}\left(\delta q_t - \frac{\partial f_t}{\partial q_{t-1}}\delta q_{t-1} - \frac{\partial f_t}{\partial u_t}\delta u_t - \frac{\partial f_t}{\partial d_t}\delta d_t\right) \\
= &\left(\frac{\partial l}{\partial q_T} + \lambda_T^{\mathsf{T}}\right)\delta q_T + \sum_{t=0}^{T}\left(\frac{\partial l}{\partial u_t} - \lambda_t^{\mathsf{T}}\frac{\partial f_t}{\partial u_t}\right)\delta u_t \\
&+ \sum_{t=0}^{T-1}\left(\frac{\partial l}{\partial q_t} + \lambda_t^{\mathsf{T}} - \lambda_{t+1}^{\mathsf{T}}\frac{\partial f_{t+1}}{\partial q_t}\right)\delta q_t \\
&- \lambda_0^{\mathsf{T}}\frac{\partial f_0}{\partial q_{-1}}\delta q_{-1} - \sum_{t=0}^{T}\lambda_t^{\mathsf{T}}\frac{\partial f_t}{\partial d_t}\delta d_t.
\end{aligned}$$

Dividing both sides by δu_t,

$$\begin{aligned}
\frac{\delta L_{\text{control}}}{\delta u_t} = &\left(\frac{\partial l}{\partial q_T} + \lambda_T^{\mathsf{T}}\right)\underbrace{\frac{\delta q_T}{\delta u_t}} + \sum_{t=0}^{T}\left(\frac{\partial l}{\partial u_t} - \lambda_t^{\mathsf{T}}\frac{\partial f_t}{\partial u_t}\right)\underbrace{\frac{\delta u_t}{\delta u_t}} \\
&+ \sum_{t=0}^{T-1}\left(\frac{\partial l}{\partial q_t} + \lambda_t^{\mathsf{T}} - \lambda_{t+1}^{\mathsf{T}}\frac{\partial f_{t+1}}{\partial q_t}\right)\underbrace{\frac{\delta q_t}{\delta u_t}} \\
&- \lambda_0^{\mathsf{T}}\frac{\partial f_0}{\partial q_{-1}}\underbrace{\frac{\delta q_{-1}}{\delta u_t}} - \sum_{t=0}^{T}\lambda_t^{\mathsf{T}}\frac{\partial f_t}{\partial d_t}\underbrace{\frac{\delta d_t}{\delta u_t}}.
\end{aligned} \tag{3.16}$$

The five underbraced terms correspond to gradients relevant to our system analysis. We have $\delta u_t/\delta u_t = 1$, $\frac{\delta d_t}{\delta u_t} = 0$, and $\frac{\delta q_{-1}}{\delta u_t} = 0$, as the initial system state and disturbance are not affected by control actions.

To eliminate the terms $\frac{\delta q_t}{\delta u_t}$ for $t = 0, \ldots, T$, the adjoint method sets the adjoint variables λ_t backwards as follows,

$$\lambda_T = -\frac{\partial l}{\partial q_T}^{\mathsf{T}}, \tag{3.17a}$$

$$\lambda_t = \left(\frac{\partial f_{t+1}}{\partial q_t}\right)^{\mathsf{T}}\lambda_{t+1} - \left(\frac{\partial l}{\partial q_t}\right)^{\mathsf{T}}, \quad t = 0, \ldots, T-1. \tag{3.17b}$$

By plugging in the adjoint variable values satisfying (3.17), and setting $\frac{\delta d_t}{\delta u_t} = \frac{\delta q_{-1}}{\delta u_t} = 0$ and $\frac{\delta u_t}{\delta u_t} = 1$, the desired gradient of the control loss L_{control} w.r.t. the control variables u_t is

$$\nabla_{u_t} L_{\text{control}} = \frac{\delta L_{\text{control}}}{\delta u_t} = \sum_{t=0}^{T} \left(\frac{\partial l}{\partial u_t} - \lambda_t^{\mathsf{T}} \frac{\partial f_t}{\partial u_t} \right). \tag{3.18}$$

We optimize the control variables with projected gradient descent:

$$u_t \leftarrow u_t - \eta \cdot \nabla_{u_t} L_{\text{control}}, \tag{3.19a}$$

$$u_t \leftarrow \text{Proj}(u_t, \underline{u}, \overline{u}), \tag{3.19b}$$

with the projection operator $\text{Proj}(x, a, b)$ defined as,

$$\text{Proj}(x, a, b) := \begin{cases} a, & \text{if } x < a \\ b, & \text{if } x > b \\ x, & \text{else} \end{cases} \tag{3.20}$$

where η represents the learning rate.

For the learning task, the decision variable changes to the system parameter $\Theta = \{\nu, k_{T_e}, k_C, \alpha, \alpha_{T_e}, \alpha_C\}$. The system parameter estimates are updated as follows,

$$\Theta \leftarrow \Theta - \eta \cdot \nabla_{\Theta} L_{\text{learning}}, \tag{3.21a}$$

$$\nabla_{\Theta} L_{\text{learning}} = \sum_{t=0}^{T} -\lambda_t^{\mathsf{T}} \frac{\partial f_t}{\partial \Theta}, \tag{3.21b}$$

where the adjoint variable is computed using (3.17). When the PDE operations are implemented in a differentiable manner, automatic differentiation tools [87, 88] can chain the derivatives of these operations to compute $\frac{\partial f_t}{\partial q_t}$, $\frac{\partial f_t}{\partial u_t}$, and $\frac{\partial f_t}{\partial \Theta}$ accordingly. In our implementation, we built our differentiable solver based on the PhiFlow framework [87].

3.5.2 Learning and Control Algorithms

The system learning procedure and control procedure are summarized in Algorithm 3 and Algorithm 4, respectively.

For the learning phase (Algorithm 3), we require the offline dataset with recordings $g(z,t), T_{\text{ambient}}(t)$, the history control sequences u_t, and measured CO_2 concentrations and temperatures $C(z_{sensor},t), T_e(z_{sensor},t)$. At each iteration k, we solve the PDEs based on estimated building parameter $\Theta^{(k)}$ and update it following the update law in (3.21) to minimize the learning loss.

Once Algorithm 3 has estimated the system parameters Θ, we are then able to optimize the control variables using these estimated parameters. For the control phase (Algorithm 4), we require the initial system state $q_{init} = q_{-1}$, predicted disturbance $g(z,t), T_{\text{ambient}}(t)$, and estimated system parameters Θ. We first initialize the control action sequence $u^{(0)}$. At each iteration k, we solve the PDEs based on control actions $[u_0^{(k)}, \ldots, u_T^{(k)}]$ and update the actions following the update law in (3.19) to minimize the control loss.

Algorithm 3. Algorithm for Learning System Phase

Require: Dataset $D = \{g(z,t), T_{\text{ambient}}(t), u_t, C(z_{sensor},t), T_e(z_{sensor},t), t = 0,\ldots,T\}$.
Ensure: $\Theta^{(0)} = \{\nu^{(0)}, k_{T_e}^{(0)}, k_C^{(0)}, \alpha^{(0)}, \alpha_{T_e}^{(0)}, \alpha_C^{(0)}\}$ ▷ initial parameters
1: **for** $k = 0, 1, \ldots, K$ **do**
2: **sample** data $B = \{(C(z_{sensor},t), T_e(z_{sensor},t)), t = 0,\ldots,T\}$ from D
3: **compute** $\widehat{\vec{v}}(z,t)$ for $t = 0,\ldots,T$ using (3.1) with $\nu^{(k)}$
4: **compute** $\widehat{T}_e(z,t)$ and $\widehat{C}(z,t)$ for $t = 0,\ldots,T$ using (3.2) with $k_{T_e}^{(k)}, k_C^{(k)}, \alpha^{(k)}, \alpha_{T_e}^{(k)}, \alpha_C^{(k)}$
5: **evaluate** the learning loss via (3.7a)(3.7b)
6: **update**: $\Theta^{(k+1)} \leftarrow \Theta^{(k)} - \eta \cdot \nabla_{\Theta^{(k)}} L_{\text{learning}}$
7: **end for**
8: **Return** Θ

3.6 Numerical Experiments

In this section, we present the capabilities of the proposed framework in system learning and optimal HVAC control. We illustrate how our framework accurately learns unknown system parameters using historical data and demonstrate significant outperformance over existing control

Algorithm 4. Algorithm for Control Phase

Require: Initial state $q_{init} = q_{-1}$, predicted disturbance $g(z,t), T_{\text{ambient}}(t), t = 0,\dots,T$, and learned parameters Θ

Ensure: $u^{(0)} = [u_0^{(0)},\dots,u_T^{(0)}]$ ▷ initial control variables

  for $k = 0,1,\dots,K$ **do**

    compute $\widehat{\vec{v}}(z,t), t = 0,\dots,T$ using (3.1) with $u^{(k)}$

    compute $\widehat{T}_e(z,t)$ and $\widehat{C}(z,t)$ for $t = 0,\dots,T$ using (3.2) with $u^{(k)}$

    evaluate the control loss via (3.9)

    update: $u^{(k+1)} \leftarrow u^{(k)} - \eta \cdot \nabla_{u^{(k)}} L_{\text{control}}$, then $u^{(k+1)} \leftarrow \text{Proj}(u^{(k+1)}, \underline{u}, \overline{u})$

  end for

  Return u

methods including maximum airflow policy, reinforcement learning [14], and optimization-based control with ODE models [26, 65, 67, 81].

We build the testbed shown in Figure 3.2. For all simulations unless otherwise specified, we consider a room with length $z_x = 4.2\,\text{m}$ and height $z_y = 2.7\,\text{m}$, which has dimensions identical to the conference room discussed in [1]. The computational mesh is defined by $(n_x, n_y) = (42, 27)$ with PDE discretization time step $\Delta t = 1$ minute. The occupancy positions are $\{z = (x,y) \mid 2.5 \le x \le 3.1, 0.7 \le y \le 0.9\}$, and the supply, return, and exterior wall regions follow Figure 3.2. The airflow rate is bounded by $\underline{u_1} = 0.12\,\text{m/s}$ and $\overline{u_1} = 1.2\,\text{m/s}$.

3.6.1 System Model Learning

Case 1: Joint temperature and CO_2 field learning.

Here, the model parameters are $\Theta = \{\nu, k_C, k_{T_e}, \alpha, \alpha_C, \alpha_{T_e}\}$. We have chosen specific values as presented in Table 3.2, following works [82, 89, 90]. The fan speed is set to be consistent during the simulation, $\vec{v}(z,t) = -\overline{u}_1 \cdot e_y, \forall z \in \mathscr{Z}_{\text{supply}}$. The simulation runs for a duration of $\overline{T} = 60$ minutes. Sensors are placed at the ground, supply vent, return vent, wall, and table; specific coordinates follow the geometry in Figure 3.2.

The training process is conducted over 60 epochs. Both the learning loss $\sum_t \|T_e(z_{sensor},t) - \widehat{T}_e(z_{sensor},t)\|^2 + \|C(z_{sensor},t) - \widehat{C}(z_{sensor},t)\|^2$ and the parameter esti-

Table 3.2. Model parameter learning for the joint temperature and CO_2 field learning.

Parameter	True value	Estimated value
ν	0.00108	0.00106
k_C	0.00108	0.00106
k_{T_e}	0.002	0.002
α	0.65	0.048
α_C	0.83	0.83
α_{T_e}	0.00500	0.00496

mation loss $\|\Theta - \hat{\Theta}\|^2$ decrease monotonically with epochs. True and estimated parameters are listed in Table 3.2, indicating that the system model can be learned precisely with low error.

Case 2: A real-world dataset for CO_2 field learning.

We now test the performance of the proposed approach using a real-world dataset in [1]. The dataset is collected in a conference room on the UC Berkeley campus, which shares the same dimensions and ventilation system as depicted in Figure 3.2. The sensors are placed on both vents and near the blackboard on the sidewall to sense CO_2 concentrations. The sensor locations for the return vent are set as $\{(x = 1\text{m}, y = 2.7\text{m}), (x = 0.9\text{m}, y = 2.7\text{m})\}$, the supply vent sensor is at $(x = 3.0\text{m}, y = 2.7\text{m})$, and the blackboard area is defined by points $\{(x = 0.1\text{m}, y = 0.7\text{m}), (x = 0.2\text{m}, y = 0.8\text{m})\}$.

Occupancy was simulated via a CO_2 pump, operated periodically (on for 30 minutes, off for 30 minutes). The simulation runs for a duration of $\overline{T} = 480$ minutes. Figure 3.4 (bottom) shows the pump status: when the pump is open (value of 1), $g(z,t) = 1$, and when closed (value of 0), $g(z,t) = 0$.

Figure 3.4(top, solid line) illustrates the CO_2 measurements at the supply vent, return vent, and blackboard. We observe that the measurements exhibit periodic patterns driven by the on/off status of the CO_2 pump, and the concentration varies both spatially and temporally. For instance, the CO_2 concentration at the blackboard will increase before the CO_2 concentration at the return vent increases. Additionally, the air supply vent has a smaller magnitude of CO_2 concentration compared to the blackboard and the air return vent.

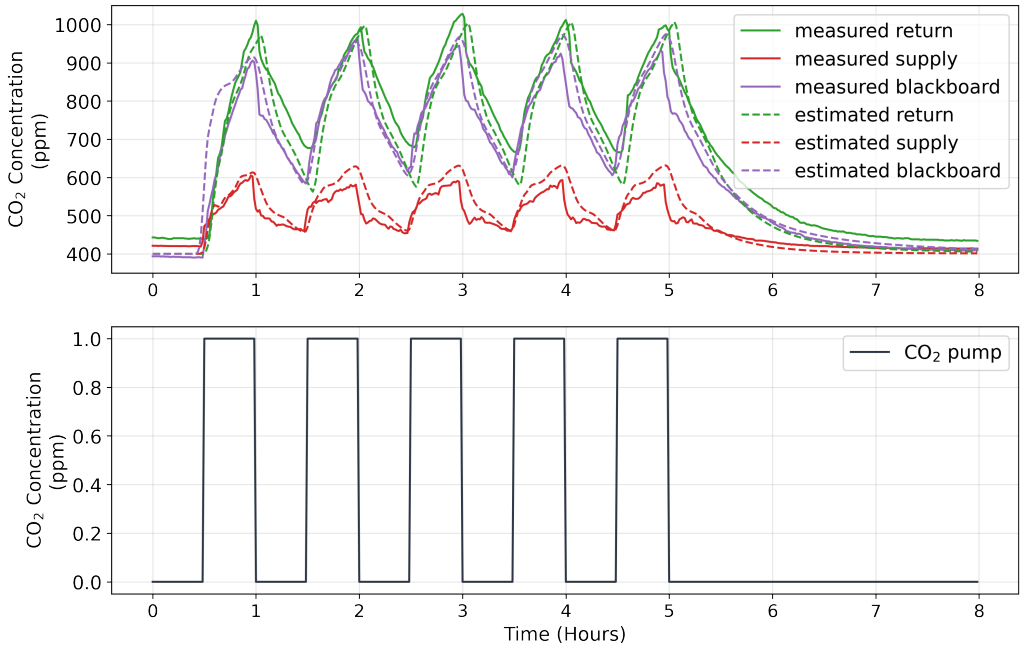


Figure 3.4. Comparison of measured CO_2 concentrations from a real-world dataset [1] and estimated CO_2 concentrations based on the learned PDE model (top). The open status of the CO_2 pump in the real-world experiment (bottom). An open CO_2 pump (indicated by a value of 1) corresponds to $g(z,t) = 1$, while a closed CO_2 pump (indicated by a value of 0) corresponds to $g(z,t) = 0$.

For this real-world dataset, the learning parameter is a subset of Θ as $\{\nu, k_C, \alpha, \alpha_C\}$ since there is no temperature measurement available for estimating $\{k_{T_e}, \alpha_{T_e}\}$. The loss function is the mean-squared error between actual CO_2 measurements and predicted values at the sensor locations: $L(\Theta) = \sum_{t=0}^{T} \|C(z_{sensor,t}) - \widehat{C}(z_{sensor}, t)\|^2$.

The estimated CO_2 concentrations are shown in Figure 3.4 (top, dashed line). The experiment demonstrates that the proposed method can capture CO_2 concentration patterns. The mean absolute percentage error (MAPE) between the measured and estimated CO_2 concentrations is 5.4%. It is important to highlight the complexity of this task: sensor recordings may contain noise, and we may not have precise information regarding sensor locations and CO_2 sources. Importantly, our method requires only the initial condition $C(z,t=0)$ and the disturbance information $g(z,t)$ for all times, allowing us to solve the PDEs and retrieve the forecast CO_2 data for all subsequent time steps.

3.6.2 Building Control

Experiment setting.

We assume that the building management system has learned the building model from historical data and is now focused on solving the control problem (3.8). The model parameters are established as described in Section 3.6.1. The management system determines the control action $\{(u_{1,t}, u_{2,t}), t \in [0, \ldots, T]\}$, where $u_{1,t}$ is the supply airflow rate (m/s) and $u_{2,t}$ indicates the supply air temperature (°C).

The CO_2 limit is $C_{\max} = 1200\,\text{ppm}, \forall z, t$, recommended by [91] as the maximum CO_2 level for human health. The temperature limits are $T_{\min} = 21.0°C$ and $T_{\max} = 22.0°C$, chosen based on the existing building controller range in [84]. The simulation runs for a duration of $\overline{T} = 360$ minutes using San Francisco's temperature data on July 1, 2023 (12pm to 6pm). We assume the building occupancy schedule and ambient temperature profiles are known. The control cost function parameters are set as $\alpha_1 = 0.1, \alpha_2 = 0.2, \alpha_3 = 0.5$, $w_1 = 30, w_2 = 3$.

Baselines.

We compare the proposed algorithm against a set of baselines including both traditional control methods and state-of-the-art control strategies: maximum airflow policy (Maxflow control) [67], minimum airflow policy (Minflow control), MPC with ODE models (MPC-ODE), and reinforcement learning (RL). The comparison with MPC-ODE is motivated by its widespread use in the literature [26, 65, 67, 81] using ODE models such as RC models, which often overlook spatial interrelationships.

The baselines are: *Maxflow* (maximum supply airflow at $u_{2,t} = T_{\min} = 21°\text{C}$), *Minflow* (minimum supply airflow at the same temperature), *MPC-ODE* (ODE models for averaged CO_2 and temperature, solved with MPC and deployed in the PDE environment), and *RL* (a policy trained against the PDE environment that maps the current $T_e(z,t)$ and $C(z,t)$ fields to actions u_t).

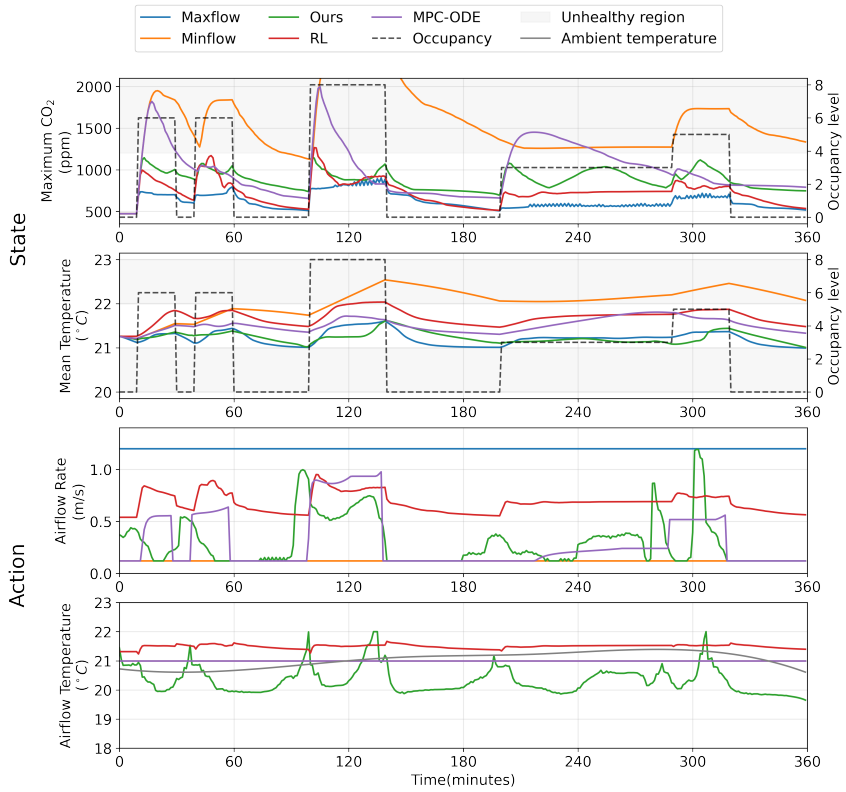


Figure 3.5. Comparison of building control performance using the proposed approach and baselines, including Maxflow control, Minflow control, MPC with ODE models (MPC-ODE), and reinforcement learning (RL). The figure shows one example of control actions and indoor temperature and CO_2 dynamics. In the top two figures, the black dashed line represents the occupancy and the grey area denotes the unhealthy region. In the bottom figure, the grey line represents the ambient temperature.

Control results.

Figure 3.5 illustrates an example of control actions along with the dynamics of indoor temperature and CO_2. Table 3.3 presents the performance of our proposed approach and baseline methods.

Notably, the Minflow control results in significant violations of CO_2 and indoor temperature constraints. During periods of high occupancy, inadequate ventilation rates fail to introduce sufficient fresh air, leading to an inability to effectively reduce CO_2 concentrations and control temperature. In contrast, our approach adapts to changes in occupancy and external temperature conditions by increasing the airflow rate when effective ventilation control is needed,

Table 3.3. HVAC control: performance of the proposed approach in comparison with Maxflow control, Minflow control, MPC with ODE models (MPC-ODE), and reinforcement learning (RL).

Method	Energy Consumption (kWh)	Temperature violation (°C)		CO_2 violation (ppm)	
		average	maximum	average	maximum
Maxflow control	334.1	0	0	0	0
Minflow control	139.7	0.16	0.54	377.1	1164.2
MPC-ODE	172.0	0.0	0.0	57.0	800.0
RL	249.1	0.001	0.04	0.35	65.6
Ours	**158.4**	**0**	**0**	**0**	**0**

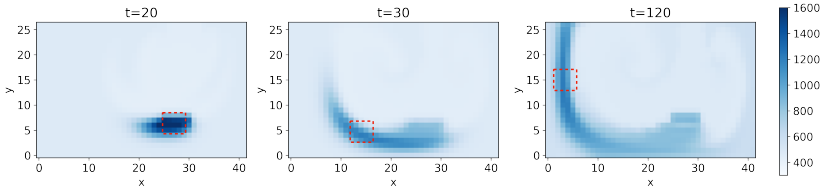


Figure 3.6. The distribution of CO_2 levels for different t (minute) with the maximum CO_2 concentration marked by red dashed boxes. The locations of the maximum CO_2 concentrations vary over time.

ensuring that state constraints are satisfied while maintaining low energy costs. The Maxflow control ensures a healthy environment but leads to high energy consumption. Our proposed approach demonstrates a 52.6% reduction in energy costs compared to the Maxflow policy. The RL approach manages to save energy but fails to meet temperature and CO_2 constraints.

MPC with ODE dynamic models performs better than RL in terms of energy usage. However, it neglects spatial variations of CO_2 distribution and leads to constraint violations: while the ODE-averaged CO_2 stays within healthy limits, the actual maximum CO_2 concentration in the room occasionally exceeds these limits.

Figure 3.6 illustrates how CO_2 distributions change over time under MPC-ODE. The maximum CO_2 levels are shown in the red dashed boxes, and we observe that the locations of maximum CO_2 can differ at different times. The results demonstrate that the ODE approach fails to capture the rich spatial-temporal dynamics of airflow velocity fields and "dead zones" of high CO_2/aerosol concentrations, thus leading to violations of health constraints. ODE modeling is insufficient for effective building ventilation design. In contrast, our approach not only adheres

to all constraints but also shows a 7.9% and 36.4% energy consumption reduction compared to MPC-ODE and RL, respectively.

3.7 Chapter Summary

This chapter introduced a PDE-based framework for building HVAC control that solves the Navier–Stokes and convection–diffusion equations for airflow, temperature, and CO_2. System learning was validated by a synthetic joint-field study with precise parameter recovery and a real-world CO_2 application with 5.4% MAPE. For control, the approach delivers 52.6%/36.4%/7.9% energy savings over Maxflow, RL, and MPC-ODE, respectively, while satisfying all modeled air-quality and control constraints. Unlike MPC-ODE and RL, which average over the room and miss localized CO_2 dead zones, the PDE formulation captures spatial dynamics directly and combines data-driven parameter learning with physically grounded control.

Several limitations merit attention. First, the current implementation considers only a 2D field and ignores common room objects such as furniture. Second, the framework relies on known disturbance factors (occupancy schedules and outdoor temperatures) and may be less effective under unpredictable conditions. Third, the same PDE dynamics that provide the main advantage of the approach also introduce a clear computational cost: resolving spatial airflow, temperature, and CO_2 fields enables the controller to capture localized dead zones and enforce modeled safety constraints more faithfully than simple ODE-based approaches, but each control update requires forward and adjoint PDE solves, which increases runtime relative to low-dimensional ODE models. Follow-up work leverages neural operator learning to retain the spatial-temporal modeling benefits of PDE dynamics while replacing repeated PDE solves with a faster learned surrogate. Future work will also extend the PDE solve to 3D with realistic geometries and integrate predictive models for disturbance variables.

Acknowledgments

This work was funded by the Hellman Fellowship and the University of California, San Diego.

Chapter 3, in full, contains material from Y. Bian, X. Fu, R. K. Gupta, and Y. Shi, "Ventilation and Temperature Control for Energy-efficient and Healthy Buildings: A Differentiable PDE Approach," *Applied Energy*, 2024. The dissertation author was the primary investigator and author of this paper.

Chapter 4

Dynamics-Guided Models for Energy-Efficient Building Control II: Operator Learning for Control

Chapter 3 demonstrated that partial differential equations coupled with adjoint-based optimization can deliver energy-efficient building ventilation control that is grounded in physical laws and can explicitly enforce air quality constraints under the modeled dynamics. That framework, however, pays a steep computational price. Each control update requires a full forward solve of the Navier–Stokes and convection-diffusion equations followed by a backward solve of the adjoint system, which confines the approach in practice to two-dimensional geometries with simplified room layouts. Real classrooms and offices are three-dimensional, have multiple ceiling vents with controllable airflow angles, and exhibit airflow patterns that cannot be faithfully reproduced by a 2D reduction.

This chapter develops a complementary approach in which the high-fidelity fluid dynamics model is retained but the expensive PDE solve is replaced by a learned neural surrogate. Specifically, we train an ensemble of neural operator transformers to approximate the mapping from ventilation control actions and historical CO_2 fields to future CO_2 fields, using high-resolution computational fluid dynamics (CFD) simulations of a real classroom as the source of ground truth. Once trained, the neural operator can evaluate a 3-minute future CO_2 rollout in a few milliseconds, roughly 2.5×10^5 times faster than the corresponding

CFD simulation, and it is differentiable by construction, so it can be embedded in the same optimization-based control loop that Chapter 3 used for the PDE dynamics. On a real-world classroom testbed with 18 inlet vents, the proposed approach achieves significant energy savings compared to maximum airflow control, rule-based control, and data-driven control methods using spatially averaged CO_2 prediction and deep learning-based reduced-order models, while consistently maintaining safe indoor air quality.

4.1 Introduction

Buildings account for nearly 40% of global energy consumption [3], with HVAC systems among the primary contributors. Effective ventilation is critical for both reducing energy use and maintaining indoor air quality [92], a tension amplified by post-COVID public health guidance [59]. Most deployed HVAC systems still rely on fixed or rule-based strategies [93, 62, 67]: maximum-fresh-air policies implemented at UC San Diego, for example, raised energy usage 2–2.5$\times$ [67].

Recent data-driven approaches, including MPC [70, 69, 94, 95] and reinforcement learning [96, 97, 98, 27, 99], have improved on rule-based methods. However, most represent indoor air states using spatially aggregated variables – typically a point measurement or zone average – modeling CO_2 via ODEs [27, 26] or neural networks [70, 69, 99] that predict mean values. These representations ignore spatial variations in airflow and pollutant distribution, leading controllers to either over-ventilate the entire space to satisfy a single sensor or miss under-ventilated regions. This motivates *spatiotemporal* airflow modeling for energy-efficient HVAC operation that respects local air-quality constraints.

Solving the governing PDEs with finite element or finite volume methods is computationally expensive [12], which makes embedding CFD solvers in real-time control loops impractical and confines most CFD-based studies [100, 101, 102] to offline analysis. The PDE-constrained

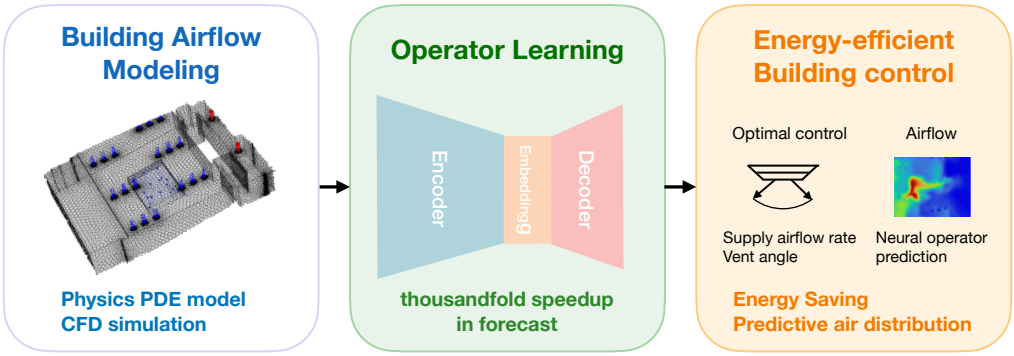


Figure 4.1. Schematic of the proposed data-driven operator learning framework for energy-efficient ventilation control. Computational fluid dynamics (CFD) simulations are used to model complex 3D airflow and CO_2 spatiotemporal dynamics. An ensemble of neural operator transformer models is trained to learn the mapping between ventilation control actions and airflow field evolution. Leveraging high-fidelity simulation data, the approach enables real-time optimization of ventilation strategies to minimize energy consumption while maintaining indoor air quality.

framework of Chapter 3 bridges this gap via adjoint differentiation, but still requires a full PDE solve per control iteration and was demonstrated only on a 2D setting with simplified geometry.

To address these limitations, this chapter proposes a data-driven operator learning framework that approximates the input–output behavior of CFD simulations. Neural operators learn mappings between infinite-dimensional function spaces, approximating the solution operator of a PDE across a family of initial and boundary conditions [20, 21]. Unlike conventional neural networks [103, 104], which operate in finite-dimensional spaces and typically require fixed spatial grids, neural operators allow predictions to be queried at arbitrary locations within the room and capture how ventilation control shapes the entire airflow and air quality distribution. We build on the general neural operator transformer (GNOT) [29], which extends these ideas to irregular spatial domains and multiple input functions, making it well suited for modeling building airflow with complex geometries and diverse inputs.

4.1.1 Contributions

This chapter develops a data-driven operator learning framework for energy-efficient building ventilation control, validated on a real-world classroom using CFD simulations (Fig-

ure 4.1). While prior studies [105, 106] apply neural operators for airflow *prediction*, we advance the line by integrating operator learning into closed-loop *control* – a substantially harder problem since performance depends on both predictive accuracy and reliable closed-loop behavior under energy and air-quality constraints. The key contributions are:

- *Ensemble neural operator transformer:* an ensemble model that predicts building airflow velocity and CO_2 fields under varying HVAC control and occupancy. It improves prediction accuracy over single-model baselines and achieves a 250,000$\times$ speed-up over CFD.
- *Building control with operator learning:* we embed the learned operator in an optimization framework for energy-efficient ventilation under air-quality constraints. Compared to maximum airflow, rule-based control, and data-driven control using averaged prediction, the proposed approach achieves lower energy use and significantly fewer CO_2 violations.
- *Open-source CFD dataset:* we release a high-fidelity dataset derived from CFD simulations of a real-world classroom, together with the 3D room model and reproduction files, providing a reproducible benchmark for ML-based building ventilation research.

The remainder of this chapter is organized as follows. Section 4.2 surveys related work. Section 4.3 introduces the governing dynamics and formulates the operator learning and control problems. Section 4.4 describes the CFD simulation setup and the BEAR-CFD dataset. Section 4.5 presents the ensemble neural operator transformer and the gradient-based control algorithm. Section 4.6 reports the learning and control experiments. Section 4.7 concludes the chapter.

4.2 Related Work

CFD-based building ventilation research.

Computational fluid dynamics simulations are widely adopted in building ventilation research to accurately capture spatiotemporal airflow and pollutant dynamics [10]. Bianco et

al. [107] employed CFD to assist in the design of ventilation units for buildings, while Gao et al. [105] used CFD to analyze airflow fields in an isothermal chamber under fixed ventilation rates. Bulinska et al. [108] modeled CO_2 distribution in a bedroom to inform optimal sensor placement, while Mou et al. [100] simulated airflow and CO_2 concentration in a seminar room, capturing complex 3D ventilation patterns. Ning et al. [101] evaluated the influence of supply outlet height on indoor air distribution in a bedroom, and Mahmoud [102] applied CFD to analyze the dispersion of human-generated aerosols and CO_2 in classrooms. These studies focus on offline analysis and system design rather than real-time control.

Neural operator methods.

Neural operators learn mappings between infinite-dimensional function spaces, enabling approximation of PDE solution operators across families of initial and boundary conditions. Deep Operator Networks (DeepONets) [21] and Fourier Neural Operators (FNOs) [20] have demonstrated strong performance on turbulent flow prediction [109], vehicle aerodynamics [110], and indoor airflow modeling [105]. The General Neural Operator Transformer (GNOT) [29] extended these capabilities to irregular spatial domains and multiple input functions, making it particularly well suited for building airflow with complex geometries and diverse control inputs. Prior studies [105, 106] have applied neural operators for airflow *prediction*; this chapter advances the line by integrating operator learning into closed-loop *control* for building energy management.

4.3 Problem Formulation

In this section, we introduce the governing dynamics of indoor airflow and CO_2 transport, and formulate the learning and control problems. The CFD simulation setup and dataset description are provided in Section 4.4.

4.3.1 Governing Equations for CO_2 Dynamics

Let $\Omega \subset \mathbb{R}^3$ be the spatial domain of interest, and let $t \in \mathbb{R}^+$ denote time. We denote by $C(x,t)$ the CO_2 concentration at spatial location $x \in \Omega$ and time t. Let $m(t) = \begin{bmatrix} m^r(t) & m^a(t) \end{bmatrix} \in \mathbb{R}^{12}$ collect the control actions (airflow rates $m^r \in \mathbb{R}^6$ and airflow angles $m^a \in \mathbb{R}^6$) for the six groups of supply vents, and let $n_p(t) \in \mathbb{R}$ denote the occupancy number.

The distribution of CO_2 in an indoor environment follows the advection-diffusion equation [28, 111],

$$\frac{\partial C(x,t)}{\partial t} + u(x,t) \cdot \nabla C(x,t) = D_{\text{eff}} \nabla^2 C(x,t) + S(x,t), \tag{4.1}$$

where $C(x,t)$ is the CO_2 concentration (ppm) at location x and time t; $u(x,t)$ is the airflow velocity field obtained by solving the incompressible Navier-Stokes equations, whose boundary conditions at the supply vents depend on the ventilation control $m(t)$ and thus allow the control action to influence $u(x,t)$ throughout the domain; D_{eff} is the effective diffusion coefficient for CO_2 in air; and $S(x,t)$ represents the CO_2 source term, which models occupant-generated CO_2 on the designated occupancy surface. The source depends on the occupancy level $n_p(t)$, and we assume an exhaled air rate of 6 L/min per person following [77].

The governing equations for the airflow velocity field $u(x,t)$ follow the incompressible Navier–Stokes equations introduced in Chapter 3, augmented in the present 3D setting with species transport for O_2, CO_2, H_2O, and N_2, an energy equation for temperature, and a k–ω SST turbulence closure. The full set of equations and boundary conditions used in the CFD solver is detailed in Section 4.4.

4.3.2 Data-Driven Modeling with a Neural Operator

Solving PDEs numerically is computationally expensive, making traditional CFD models impractical for real-time building control applications. As discussed above, the adjoint-based PDE optimization of Chapter 3 inherits this cost: each control iteration requires a full forward and backward PDE solve on the simulation mesh. To overcome this limitation, we aim to learn

a neural operator $\mathscr{G}_\theta$ that efficiently maps historical CO_2 concentrations, control actions, and occupancy levels to future CO_2 concentration distributions. Formally, the neural operator learns the mapping

$$\mathscr{G}_\theta : \big(C(x,\tau)_{\tau\in[t-H,t]}, m, n_p\big) \;\mapsto\; C(x,\tau)_{\tau\in(t,t+T]}, \tag{4.2}$$

where $C(x,\tau)_{\tau\in[t-H,t]}$ are the historical CO_2 fields over the window $[t-H,t]$ and $C(x,\tau)_{\tau\in(t,t+T]}$ are the predicted future CO_2 fields over $(t,t+T]$. Incorporating historical CO_2 concentrations accounts for temporal dependencies inherent to the system's physics, such as diffusion and advection dynamics. We assume that the control m and occupancy n_p remain *fixed* over $(t,t+T]$. In building control applications, the transient dynamics of airflow and CO_2 transport evolve rapidly over a short horizon, typically within a few minutes, while control settings and occupancy levels generally remain constant during these intervals (for example, a fixed HVAC setpoint over a 5-minute or 15-minute interval [67]). This assumption thus maintains consistency between the physical model and practical control timescales.

In practice, we rarely have continuous access to the CO_2 distribution over space and time, and instead rely on a discretized approximation of the underlying functions. For simplicity we reuse the symbols t,H,T as discrete time indices. Let $\{x_i\}_{1\le i\le N_x}$ denote a spatial grid and let $C_i^t \in \mathbb{R}$ denote the CO_2 concentration at spatial grid point x_i and time step t. Collecting the recent history of CO_2 fields over the last H time steps together with the fixed parameters m and n_p, we define the discrete input set

$$\mathscr{A} = \{(x_i, C_i^{t-H:t})\}_{1\le i\le N_x} \cup \{m\} \cup \{n_p\}. \tag{4.3}$$

Although $\mathscr{A}$ is finite-dimensional, it represents the sampled version of the underlying function $C(x,\tau)$ over the history window $[t-H,t]$. From this discrete representation, the neural operator $\mathscr{G}_\theta$ produces predicted future concentrations $\widehat{C}^{t+1:t+T}_{1\le i\le N_x}$ (abbreviated $\widehat{C}$), which approximate the

continuous output function $\widehat{C}(x,\tau)$ for $\tau \in (t, t+T]$:

$$\widehat{C}^{t+1:t+T}_{1\le i\le N_x} = \mathscr{G}_\theta(\mathscr{A}). \tag{4.4}$$

4.3.3 Ventilation Control Optimization

The learned neural operator $\mathscr{G}_\theta$ can be integrated into the building ventilation control optimization problem. The resulting formulation is given in (4.5).

$$\min_{m=[m^r,m^a]} \quad w_1 \left\|\widehat{C} - C_{\text{target}}\right\|_2^2 + w_2 \left\|m - m^{(0)}\right\|_2^2 + w_3 \left\|m^r\right\|_1, \tag{4.5a}$$

$$\text{s.t.} \quad \widehat{C} \leftarrow \mathscr{G}_\theta(\{C^{t-H:t}_{1\le i\le N_x}\} \cup \{m\} \cup \{n_p\}), \tag{4.5b}$$

$$\underline{m^r} \le m^r \le \overline{m^r}, \quad \underline{m^a} \le m^a \le \overline{m^a}. \tag{4.5c}$$

In the objective (4.5a), the first term quantifies the deviation between the predicted CO_2 concentrations $\widehat{C}$ and the desired CO_2 level C_{target} over the prediction horizon T, a formulation commonly used in indoor air quality ventilation control studies [26, 112]. The second term penalizes deviations of the optimized control actions from the previous control action $m^{(0)}$. This is important for real-world building management, where large deviations in control actions are typically avoided to maintain system stability and operational safety [28, 65]. Encouraging control actions to remain close to $m^{(0)}$ also reduces the risk of extrapolating beyond the model's training domain, thereby increasing the reliability of the predictions. The third term represents energy consumption, measured through the ℓ_1 norm of the ventilation rate [28]. The coefficients w_1, w_2, w_3 balance the relative importance of these objectives. Constraint (4.5b) states that the predicted CO_2 concentrations are obtained via the trained neural operator model in (4.4), while constraint (4.5c) ensures that both mechanical ventilation rates m^r and vent angles m^a remain within their physical limits.

Remark 4.1. *In this chapter, we do not explicitly model or control indoor temperature in the problem formulation. This choice is motivated by the fact that CO_2 concentration responds on a much faster time scale than temperature, which evolves more slowly because of the building's thermal inertia. By focusing on the faster dynamics of airflow and CO_2 transport, we are able to evaluate the proposed control strategy in a more responsive setting. Thermal effects are still captured, as the CFD simulations described in Section 4.4 solve the full energy equation for the transport of thermal energy within the airflow and include occupant heat sources and boundary temperature conditions. Extending the formulation to include indoor temperature modeling and thermal control is a valuable direction for future work, and the framework is compatible with such multi-objective settings.*

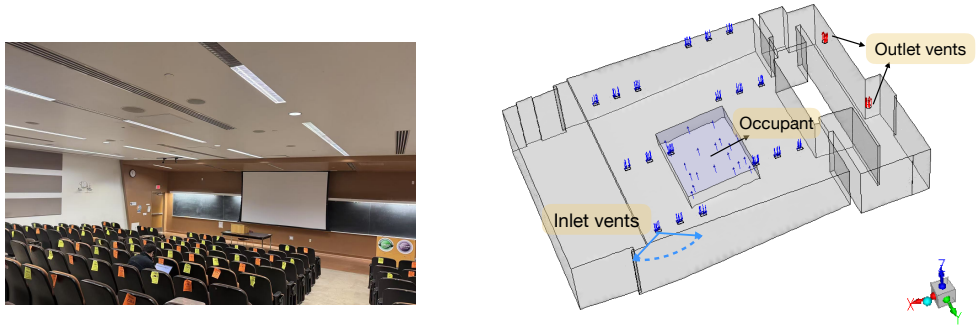
4.4 The BEAR-CFD Data

The CFD simulation is developed based on a real-world classroom located at the University of California, San Diego. The classroom is equipped with a ceiling-mounted ventilation system that includes 2 outlet vents and 18 inlet vents grouped into six zones, each allowing independent control of airflow rates and supply airflow angles to optimize energy efficiency while maintaining high indoor air quality. Figure 4.2(a) illustrates the physical classroom and its corresponding 3D computer-aided design (CAD) representation, and Figure 4.2(b) presents an example of the CO_2 concentration and velocity fields. This section provides a detailed description of the simulation setup and the open-source dataset.

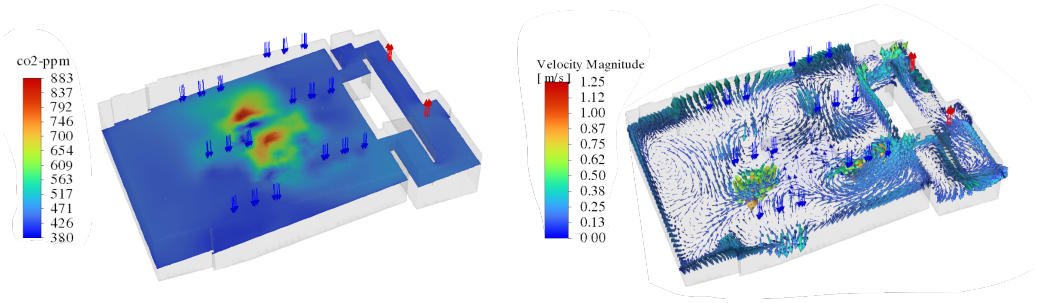
4.4.1 CFD Simulation Setup

Geometry.

The simulated domain represents a mechanically ventilated classroom with dimensions of $19\,\mathrm{m} \times 13\,\mathrm{m} \times 3.5\,\mathrm{m}$; see Figure 4.2(a). The ventilation system consists of 18 rectangular inlet vents, each measuring $0.1349\,\mathrm{m}$ in width and $0.3048\,\mathrm{m}$ in height, and 2 ceiling-mounted outlet



(a) Picture of the studied room and geometrical model of the room.



(b) Visualization of CO_2 field and airflow velocity field.

Figure 4.2. (a) A picture of the studied classroom and the geometry of the CFD model: a room with a ventilation system including 18 inlet vents and 2 outlet vents on the ceiling. Occupants are modeled as a single rectangular cuboid with a prescribed CO_2 mass flux to represent the CO_2 effects of varying occupancy levels. (b) Visualization of the CFD simulation results of CO_2 concentration and airflow velocity fields, one example from the developed CFD dataset.

vents. Fresh air is supplied through the inlets at prescribed velocity and angle conditions, and exhausted through the outlets, forming a displacement ventilation flow pattern.

Occupant modeling.

Occupants are collectively represented by a single, centrally positioned rectangular cuboid, visualized in Figure 4.2(a, right). This abstraction aggregates the influence of multiple seated individuals into a compact volume that approximates their combined occupied zone [113, 114]. CO_2 emissions from occupants are represented as a surface mass flux boundary condition and are modeled using the species transport equation [108, 102]. To simulate varying occupancy levels (ranging from 10 to 80 people), we prescribe a total CO_2 mass flux of $n_p \times 0.00012$ kg/s on the cuboid surface, where n_p is the number of occupants. This corresponds

to a constant breathing rate of 6 L/min per person, following the simplified exhalation model in [108]. A constant surface temperature is applied to the cuboid.

Mesh.

The fluid domain is discretized into approximately 0.245 million tetrahedral elements to resolve airflow and CO_2 transport dynamics. Local mesh refinement is applied near critical regions, including the occupant cuboid, inlet and outlet vents, and wall boundaries, to capture steep gradients in velocity and scalar fields. Mesh quality is assessed using standard metrics: the minimum orthogonal quality is 0.172, and the maximum aspect ratio is 43.33, indicating an acceptable mesh for transient indoor airflow simulations.

Numerical models and boundary conditions.

Airflow and CO_2 dynamics within the classroom are simulated using the commercial CFD software ANSYS Fluent [115]. The incompressible Navier-Stokes equations are solved to capture airflow behavior, coupled with species transport equations for CO_2, O_2, H_2O, and N_2. Turbulence is modeled using the k-ω SST (shear stress transport) model, which is well suited for predicting indoor near-wall flow behavior [116]. The governing equations are discretized using the finite volume method, with second-order schemes for both momentum and species transport. The energy equation is also activated to solve for temperature.

The boundary surfaces include walls, inlet and outlet vents, and the occupant surface. A summary of the boundary conditions is provided in Table 4.1. Inlets are modeled as velocity boundaries with prescribed temperature (T_{inlet}), CO_2 concentration (C_{inlet}), velocity magnitude (m^r), and velocity angle (m^a) [108, 102]. To introduce airflow variability, the inlet velocities and angles for vent groups are randomly sampled from uniform distributions. A maximum velocity of 3.24 m/s corresponds to 10 air changes per hour (ACH), based on operational data. Outlets are modeled as pressure boundaries, and all walls are treated as no-slip, adiabatic surfaces with fixed temperature.

Table 4.1. Boundary conditions for the CFD simulation.

Boundary surface	Boundary conditions
Inlet vents	velocity inlet, $C_{\text{inlet}} = 400$ ppm, $T_{\text{inlet}} = 20°\text{C}$, $m^r \sim U[0.324, 3.24]$ m/s, $m^a \sim U[45, 135]^\circ$
Outlet vents	pressure outlet
Walls	no-slip and adiabatic walls, $T_{\text{wall}} = 21°\text{C}$
Occupant surface	mass flow inlet, $C_{\text{occupant}} = 40{,}000$ ppm, $T_{\text{occupant}} = 25°\text{C}$ [108]

Remark 4.2. *The CFD simulations follow standard modeling practices for indoor airflow [114, 108, 117] and are designed to provide consistent and reproducible training data for future research. The simulation setup assumes constant air density (incompressible flow), omitting buoyancy effects. This simplification is reasonable in mechanically ventilated classrooms where forced ventilation dominates airflow. Nevertheless, we acknowledge that in densely occupied spaces, buoyancy forces may contribute to airflow patterns. Extending the CFD setup to incorporate buoyancy remains an important direction for future work, and the framework is readily adaptable to such enhanced simulations as well as to other CFD configurations and building control systems.*

4.4.2 BEAR-CFD Dataset

The BEAR-CFD dataset is generated through a structured CFD simulation workflow designed to capture the spatiotemporal dynamics of indoor CO_2 levels under varying ventilation and occupancy conditions. The dataset is publicly available at https://ucsdsmartbuilding.github.io/CFD-DATA.html.

The simulations include both steady-state and transient flow cases. Specifically, we run 10 steady-state simulations under different boundary conditions. These steady-state results are then used as *initial conditions* for 300 transient simulations, each lasting 30 minutes of physical time and resolved at 10-second time steps. This initialization strategy ensures that each transient simulation begins from a realistic state shaped by prior control conditions. Without this step, starting from uniform or arbitrary states (for example, the same CO_2 everywhere) could

introduce artificial transients unrelated to the actual control inputs. Simulation outputs are saved every 30 seconds, resulting in 60 time steps per transient case. Transient simulations capture the temporal evolution of airflow dynamics, reflecting unsteady effects influenced by varying boundary conditions. This transient data is used to train our neural operators.

For each simulation, key control parameters, including supply airflow rates (m_i^r) and airflow angles (m_i^a) for the i-th group of inlet vents ($i = 1, \ldots, 6$), as well as the occupant count (n_p), are independently sampled from uniform distributions:

$$\begin{aligned} &m_i^r \sim U[\underline{m^r}, \overline{m^r}], \quad m_i^a \sim U[45^\circ, 135^\circ], \quad i \in \{1, \ldots, 6\}, \\ &n_p \sim U[10, 80], \end{aligned} \tag{4.6}$$

where m_i^r is the airflow rate for the i-th group of vents, bounded between $\underline{m^r} = 0.324$ m/s (10% of maximum) and $\overline{m^r} = 3.24$ m/s; m_i^a is the airflow angle of the i-th group of vents, spanning 45° to 135°; and n_p is the number of occupants in the classroom. Each transient simulation follows this two-phase procedure: (1) *steady-state initialization*, in which a random initial condition is selected from the 10 steady-state simulations; and (2) *transient simulation*, in which control parameters (m_i^r, m_i^a) and occupancy n_p are sampled, and the simulation runs for 30 minutes with CO_2 and airflow fields recorded at 30-second intervals over $\overline{T} = 60$ time steps.

CO_2 concentrations are monitored at two critical planes:

- *HVAC surface:* a horizontal plane at 2.9-meter height near the ventilation inlets, capturing the supply and returning air quality.
- *People surface:* a horizontal plane at 1.6-meter height (the average breathing height for standing adults), representing air quality at occupant exposure levels [118].

In our CFD simulation, the sitting surface is near the occupancy boundary, leading to potential inaccuracies from numerical artifacts and boundary effects. We therefore focus on heights where airflow and CO_2 dispersion are more reliably captured.

Table 4.2. Data fields in the BEAR-CFD dataset (pickle format).

Field	Description
HVAC surface	ndarray $(N_{\text{hvac}}, 3)$, spatial coordinates of grid points on HVAC surfaces
CO_2-HVAC	ndarray $(N_{\text{hvac}}, \overline{T})$, CO_2 concentration time series at HVAC surface
People surface	ndarray $(N_{\text{people}}, 3)$, spatial coordinates of grid points on people surfaces
CO_2-People	ndarray $(N_{\text{people}}, \overline{T})$, CO_2 concentration time series at people surface
steady case	int, identifier for the initial steady-state condition used in the simulation
n_p	int, number of occupants
m_i^r	float, airflow rate (m/s) for i-th group of vents
m_i^a	float, angle (°) for i-th group of vents

The dataset is distributed in both ANSYS Fluent's native format (`.cas` and `.dat` files) and Python pickle (`.pkl`) format. The Fluent files preserve the complete simulation environment and solution data, while the pickle format enables efficient programmatic access to the numerical results through standard Python libraries. The dataset includes spatiotemporal fields of CO_2 concentrations, airflow rates, vent angles, and occupancy, as detailed in Table 4.2.

4.5 Methodology

This chapter proposes a data-driven operator learning framework to model indoor air quality and optimize ventilation control for energy efficiency. The core component is an ensemble neural operator transformer $\mathscr{G}_\theta$ that processes query points, supply airflow rates, airflow angles, and historical CO_2 concentrations to predict future CO_2 fields, which are then used to determine optimal ventilation parameters by solving (4.5).

4.5.1 Ensemble Neural Operator Transformer

To address the limited-data regime, we enhance the general neural operator transformer (GNOT) [29] with ensemble learning [119]. We describe the input encoding and the ensemble of GNOTs.

Input encoding.

The model takes as input the spatial mesh, historical CO_2 concentrations, and control parameters (ventilation actions and occupancy). Three MLPs f_{w1}, f_{w2}, f_{w3} map these heterogeneous inputs into a shared embedding space of dimension n_e: $Y_{\text{mesh}} = (f_{w1}(x_i))_{i=1}^{N_x}$ encodes mesh points, $Y_C = (f_{w2}(x_i, C_i^{t-H:t}))_{i=1}^{N_x}$ encodes past CO_2 at each location, and $Y_{\text{param}} = f_{w3}([m, n_p])$ encodes the 13-dimensional control parameters.

Ensemble of GNOTs.

GNOT updates these features using a heterogeneous normalized cross-attention layer followed by a normalized self-attention layer, with a geometric gating mechanism that combines expert feed-forward networks based on query coordinates [29]; N such blocks are stacked.

While GNOT is originally designed to output only the mean of the prediction, we enhance its robustness by introducing an ensemble-based extension. Instead of training GNOT to minimize mean squared error or relative error, we modify it to predict a probability distribution for future CO_2 concentrations, characterized by the mean (μ_C) and variance (σ_C^2). Specifically, the model predicts

$$\mu_C, \sigma_C^2 = \mathscr{G}_\theta(\mathscr{A}), \tag{4.7a}$$

$$\mu_C = \mathbb{E}\big[\widehat{C}_{1\le i\le N_x}^{t+1:t+T}\big], \quad \sigma_C^2 = \mathrm{Var}\big[\widehat{C}_{1\le i\le N_x}^{t+1:t+T}\big], \tag{4.7b}$$

where $[\mu_C]_i^t$ and $[\sigma_C^2]_i^t$ denote the mean and variance at location i and time step t. The model is trained using the negative log-likelihood (NLL) loss,

$$\mathscr{L} = \frac{1}{N_x T} \sum_{i=1}^{N_x} \sum_{k=1}^{T} \left(\frac{\log\left(2\pi[\sigma_C^2]_i^{t+k}\right)}{2} + \frac{\left(C_i^{t+k} - [\mu_C]_i^{t+k}\right)^2}{2[\sigma_C^2]_i^{t+k}} \right), \tag{4.8}$$

where $C_i^{t+k} \in \mathbb{R}$ is the true value at x_i and time $t+k$. By minimizing the NLL loss, the model learns to jointly optimize the mean and variance predictions, effectively mitigating overfitting [119].

We then train an ensemble of neural operator transformers, with the final prediction obtained by averaging outputs from multiple independently trained models. Let $\mu_C^{(n)}$ denote the mean prediction of the n-th model. The ensemble prediction is

$$\mu_C^{\text{ensemble}} = \frac{1}{N_{\text{ensemble}}} \sum_{n=1}^{N_{\text{ensemble}}} \mu_C^{(n)}, \tag{4.9a}$$

$$\mu_C^{(n)}, (\sigma^2)_C^{(n)} = \mathscr{G}_{\theta_n}(\mathscr{A}), \tag{4.9b}$$

where N_{ensemble} is the number of models in the ensemble and θ_n are the parameters of the n-th trained neural operator. This ensemble approach not only improves prediction accuracy but also enhances the reliability of ventilation control, as demonstrated in the experiments.

4.5.2 Control Algorithm

With the learned ensemble neural operator transformer in hand, we are ready to solve the building control problem in (4.5). Recall that the control objective is

$$\mathscr{L}(m) = w_1 \|\widehat{C} - C_{\text{target}}\|_2^2 + w_2 \|m - m^{(0)}\|_2^2 + w_3 T \|m^r\|_1, \tag{4.10}$$

Algorithm 5. Gradient-based algorithm for solving (4.5)

Require: Trained neural operator transformers $\mathscr{G}_\theta$; control inputs $C_{1\le i\le N_x}^{t-H:t}, n_p$

Ensure: Initial control actions $m^{(0)}$

1: **for** $ite = 0, 1, \ldots,$ MaxIte **do**
2: **obtain** future CO_2 predictions $\widehat{C}$ via (4.9)–(4.11)
3: **evaluate** the loss $\mathscr{L}(m)$ via (4.10)
4: **compute** the gradient $\nabla_m \mathscr{L}(m)$
5: **update** the control vector with step size η:

$$m^{(ite+1)} \leftarrow m^{(ite)} - \eta \nabla_m \mathscr{L}(m)$$

6: **project** $m^{(ite+1)}$ to satisfy the box constraints (4.5c)
7: **end for**
8: **Return** m

where the first term is

$$\|\widehat{C} - C_{\text{target}}\|_2^2 = \frac{1}{N_x T} \sum_{k=1}^{T} \sum_{i=1}^{N_x} \left(\widehat{C}_i^{t+k} - [C_{\text{target}}]_i^{t+k}\right)^2,$$

and w_1, w_2, w_3 are weighting coefficients. The predicted CO_2 concentrations $\widehat{C}$ are generated by the ensemble neural operator (4.9):

$$\widehat{C} = \mu_C^{\text{ensemble}}. \tag{4.11}$$

To solve (4.5), we leverage the differentiability of the neural operator $\mathscr{G}_\theta$ and propose a gradient-based method to update the control actions. This mirrors the gradient-descent philosophy of Chapter 3, but the expensive adjoint solve is now replaced by standard backpropagation through the trained neural operator, which runs in milliseconds on a GPU. To ensure that the control vector m remains within feasible bounds (4.5c), we use a projected gradient descent method: after computing the gradient of the objective with respect to m, the control vector is updated and then clipped to satisfy the predefined bounds. The optimization procedure is summarized in Algorithm 5.

Table 4.3. The ℓ_2 error for five independently trained neural operator transformer models (Model 1 to Model 5) and their ensemble.

	Model 1	Model 2	Model 3	Model 4	Model 5	Ensemble
ℓ_2 error (Train)	6.35%	6.33%	6.33%	6.35%	6.33%	**5.9%**
ℓ_2 error (Test)	12.09%	11.83%	11.82%	12.74%	13.01%	**10.90%**

4.6 Numerical Experiments

In this section, we evaluate the performance of the proposed framework through two sets of experiments: (1) learning experiments to assess the accuracy of the ensemble neural operator, and (2) control experiments to evaluate the effectiveness of the data-driven ventilation control framework. The source code, input data, and trained models are available on GitHub.[1] All experiments are conducted on NVIDIA GeForce RTX 2080 Ti GPUs.

4.6.1 Learning Results

We train the ensemble neural operator transformer to predict CO_2 levels on the *people surface*. The number of query points is $N_x = 7462$, and we select $H = 12$ and $T = 6$, meaning that the model uses data from the past 12 time steps (equivalent to 6 minutes) to forecast CO_2 levels over the next 6 time steps (3 minutes). The dataset is divided into an 80% training set and a 20% testing set to evaluate model performance. We train $N_{\text{ensemble}} = 5$ independent models and use the mean of their predictions as the final output. We use the AdamW optimizer with a cyclical learning rate schedule, and each model is trained for 200 epochs.

To evaluate model performance, we use the average ℓ_2 relative error as the primary metric. Let $C^{(d)}, \widehat{C^{(d)}} \in \mathbb{R}^{N_x \times T}$ denote the ground truth and the predicted mean future CO_2 concentrations for the d-th sample, respectively, and let D denote the total dataset size. The error is defined as

$$\ell_2 = \frac{1}{D} \sum_{d=1}^{D} \frac{\left\| \widehat{C^{(d)}} - C^{(d)} \right\|_2}{\left\| C^{(d)} \right\|_2}. \tag{4.12}$$

[1] https://github.com/alwaysbyx/BuildingControlCFD

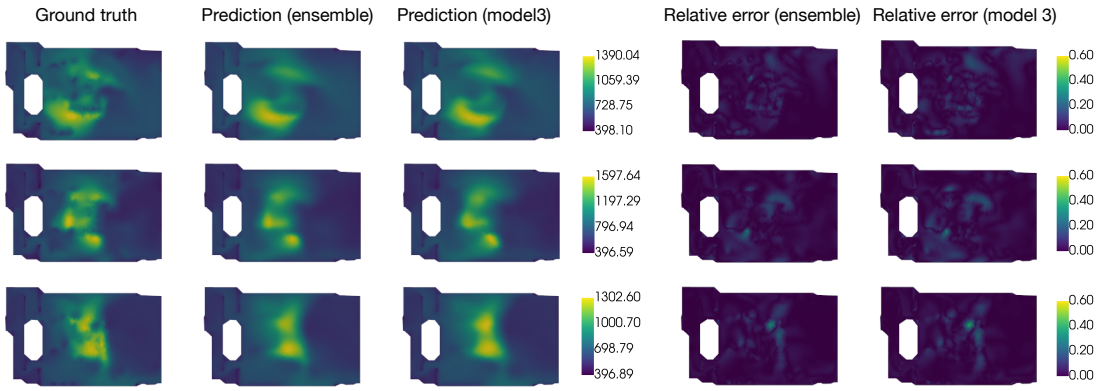


Figure 4.3. Operator learning: visualization of the ground truth, predictions from the ensemble model and Model 3, and the relative errors between the ground truth and predictions at the final time step.

Table 4.3 lists the ℓ_2 error metrics for the five trained models (Model 1 to Model 5) and the ensemble model on both the training and testing datasets. Notably, the ensemble model achieves the lowest errors, with a training ℓ_2 error of 5.9% and a testing ℓ_2 error of 10.90%. The ensemble consistently outperforms the individual models, leading to more robust and accurate predictions. The performance improvement can be attributed to the ensemble's ability to mitigate potential overfitting of individual models by averaging out their prediction errors.

Figure 4.3 shows the ground truth CO_2 concentration fields along with predictions from the ensemble model and Model 3 (the best-performing individual model), together with their relative errors for three test cases. Overall, the neural operator framework shows strong capability in predicting complex spatial CO_2 distributions under varying control parameters. The ensemble model achieves lower relative errors and more accurately captures spatial patterns compared to Model 3. In Section 4.6.4, we further demonstrate its effectiveness in activating control actions and optimizing air quality.

4.6.2 Ventilation Control Results

We evaluate control performance under three occupancy scenarios: high ($n_p = 75$), medium ($n_p = 45$), and low ($n_p = 15$). Following ASHRAE Standard 62.1 [120] (4–6 air changes per hour for classrooms), all scenarios initialize at 5 ACH with inlet angles fixed at

90° downward. We compare six control strategies: **Max** (maximum airflow at 90°); **Baseline** (ASHRAE 5 ACH at 90°); **Rule-based** (airflow proportional to occupancy, 90°); **DL-Avg**, a neural network predicting average CO_2 used inside (4.5); **DL-ROM**, the deep learning-based reduced-order surrogate of [121] integrated into (4.5); and **Ours**, where the ensemble neural operator transformer is embedded in (4.5) to optimize control actions.

For the optimization solver, we choose $C_{\text{target}} = 400$ ppm to ensure that the CO_2 concentration loss is more broadly activated across the spatial domain, enabling more effective optimization of the control actions. We set $w_1 = 1$, $w_2 = 0.4$, $w_3 = 0.12$ through empirical tuning to balance air quality, energy consumption, and control smoothness. These weights are used consistently across all control experiments.

We validate the control strategies using CFD simulations. Performance is measured using three CO_2 metrics: mean CO_2 (ppm), peak CO_2 (ppm), and CO_2 violation (%), defined as $(\text{Peak } CO_2 - 1200)/1200$. As suggested by [91], 1200 ppm is the maximum acceptable CO_2 level for human health. For each metric, we record both the temporal average over the simulation period and the value at the final time step. Energy consumption is modeled as the percentage of the maximum power required to operate the ventilation system at its maximum airflow rate, providing a *normalized* measure relative to the system's peak capacity:

$$E(m) = \frac{1}{6}\sum_{i=1}^{6} \frac{m_i^r}{\overline{m^r}} \times 100\%, \tag{4.13}$$

where m_i^r is the ventilation rate for the i-th group of vents and $\overline{m^r} = 3.24$ m/s is the maximum ventilation rate.

The results are summarized in Table 4.4. The experimental results demonstrate the effectiveness of the proposed control strategy in balancing air quality and energy consumption for the first two cases. In Case 3, under low occupancy, the optimization favors maintaining higher air quality, leading to slightly higher energy consumption than the baseline control. Compared to the Max control strategy, our method achieves comparable air quality performance

Table 4.4. Comparison of CO_2 metrics (mean, peak, and violation percentages) and energy consumption across different control strategies. "Average" refers to the temporal mean over the simulation period, while "Final Step" refers to the CO_2 level at the end of the simulation. The proposed approach achieves significant ventilation energy savings compared to the other control strategies while maintaining acceptable CO_2 violation levels.

Case	Control Strategy	Mean CO_2 (ppm)		Peak CO_2 (ppm)		$CO_2 > 1200$ ppm (%)		Energy
		Average	Final	Average	Final	Average	Final	(%)
1	Max Control	565.6	534.0	895.7	854.2	0.00	0.00	100.0
	Baseline Control	616.1	612.3	1203.0	1143.6	1.32	0.00	50.0
	Rule-based Control	567.4	533.5	898.1	807.9	0.00	0.00	93.8
	Data-driven (DL-Avg)	604.4	595.2	1021.6	971.6	0.00	0.00	83.3
	Data-driven (DL-ROM)	626.2	635.8	1300.1	1326.7	8.34	10.56	44.6
	Ours	600.7	587.9	1060.2	1004.0	**0.00**	**0.00**	**65.8**
2	Max Control	546.7	512.2	800.0	741.3	0.00	0.00	100.0
	Baseline Control	599.9	595.9	1097.3	1108.4	0.00	0.00	50.0
	Rule-based Control	593.7	584.6	1057.6	1059.3	0.00	0.00	56.2
	Data-driven (DL-Avg)	584.7	562.1	912.7	914.6	0.00	0.00	83.0
	Data-driven (DL-ROM)	602.7	600.1	1062.2	1066.2	0.00	0.00	50.8
	Ours	605.0	600.0	1044.1	881.4	**0.00**	**0.00**	**43.8**
3	Max Control	532.3	497.6	754.0	692.0	0.00	0.00	100.0
	Baseline Control	585.2	578.0	1034.2	1013.9	**0.00**	**0.00**	**50.0**
	Rule-based Control	650.7	712.7	1361.4	1623.1	15.67	35.26	18.8
	Data-driven (DL-Avg)	576.9	566.0	946.1	978.3	0.00	0.00	66.1
	Data-driven (DL-ROM)	590.3	590.2	1121.6	1208.4	0.13	0.70	49.4
	Ours	589.4	585.4	984.4	949.8	0.00	0.00	55.2

while reducing energy usage by 34–56%. Compared to the baseline control, it effectively lowers CO_2 levels and adheres to air quality constraints with a slight increase in energy use. Rule-based control suffers from CO_2 violations (for example, a 35.2% final-step violation in Case 3) due to its reliance on well-chosen static operation schedules. Unlike rule-based systems, which require manual fine-tuning of thresholds for different room layouts and occupancy levels, the proposed method autonomously adapts to operating conditions and reduces energy consumption by 12–28% in Cases 1–2 while maintaining safe CO_2 levels.

In addition, the proposed method outperforms the data-driven baselines (DL-Avg and DL-ROM) by maintaining good energy efficiency without incurring any air-quality violations. DL-Avg is limited by predicting only average CO_2 concentration, which fails to capture spatial variations or map control actions to zone-level distributions. DL-ROM captures spatiotemporal

airflow but is less effective at modeling the relationship between control inputs and the resulting CO_2 distributions, leading to occasional violations.

4.6.3 Multi-Step Ventilation Control Results

To assess the proposed operator learning model in a more realistic control scenario, we extend the single-step optimization of Section 4.6.2 to a multi-step planning horizon. Specifically, a 30-minute ventilation horizon is considered with 10 planning steps, each optimizing control over the next 6 time steps (3 minutes), yielding 60 control steps in total. The initial step ($t = 0$) is initialized using historical CO_2 data from Scenario 1 in Section 4.6.2, and subsequent steps use CO_2 levels predicted by recursively rolling out the learned operator with the previously optimized controls.

Figure 4.4 illustrates the performance of the proposed control strategy compared to the other control methods. The occupancy profile (top) shows significant variations over time, and the proposed strategy dynamically adjusts ventilation in response: it increases airflow during high-occupancy periods to mitigate CO_2 accumulation, maintains moderate ventilation afterwards to clear residual CO_2, and reduces airflow during low-occupancy intervals for greater energy savings.

Over the control period, the total energy consumption for Max, Baseline, Rule-based, DL-Avg, DL-ROM, and our method is 100%, 50%, 63.1%, 55.3%, 57.6%, and 50.6%, respectively. CO_2 violations are 0, 17, 10, 37, 7, and 1 for Max, Baseline, Rule-based, DL-Avg, DL-ROM, and ours, respectively. The relatively poor performance of DL-Avg stems from its reliance on predicting only the average CO_2 concentration, without capturing the dynamic spatial shifts of peak concentrations over time, leading to both higher energy consumption and more CO_2 violations. The controller also exhibits sensible spatial behavior, consistently routing the highest airflow through the most effective vent group while throttling others.

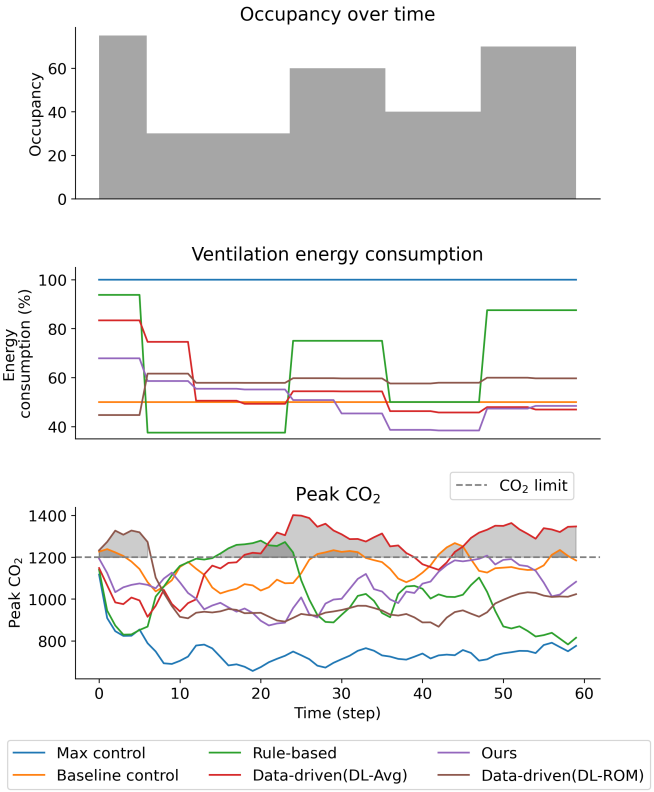


Figure 4.4. Comparison of occupancy profile, energy consumption, and peak CO_2 concentration over a 60-step period. The top panel illustrates dynamic occupancy changes, the middle panel shows corresponding energy consumption as a percentage of the maximum, and the bottom panel depicts CO_2 concentration levels. The dashed line in the bottom panel indicates the 1200 ppm threshold.

4.6.4 Ensemble Effectiveness and Runtime

Setting $w_3 = 0$ in (4.5) (controller prioritizes air quality), the ensemble consistently saturates the airflow rate at every inlet vent – the optimal action in this regime – whereas individual models occasionally fail to saturate all vents, confirming that aggregating predictions across independently trained operators reduces both predictive and optimization noise.

In terms of runtime, the neural operator completes a 3-minute (six-step) transient prediction in under 0.005 seconds, versus 1253.7 seconds for the equivalent ANSYS Fluent CFD solve on 6 CPU cores – a $\sim 250{,}000\times$ speed-up. Training the five-model ensemble takes 16

GPU hours on 2 RTX 2080 Ti GPUs, and CFD data generation totals $\sim$1100 CPU hours; this one-time cost amortizes across all subsequent real-time closed-loop control.

4.7 Chapter Summary

This chapter developed an operator learning framework for energy-efficient ventilation control that preserves CFD fidelity while enabling real-time optimization. An ensemble of neural operator transformers maps past CO_2 fields, occupancy, and control actions to future CO_2 fields. The learned operator is embedded in an optimization-based control framework that parallels Chapter 3's PDE-constrained optimization, but replaces the expensive adjoint solve with millisecond-scale backpropagation. Using CFD of a real-world classroom as ground truth, the approach achieves substantial energy savings over maximum-airflow, rule-based, and data-driven baselines while reducing CO_2 violations. The CFD dataset is released publicly.

Together, Chapters 3 and 4 illustrate a common design pattern: encode the governing physics, differentiate through the dynamics, and embed the resulting differentiable model in a gradient-based control loop. Chapter 3 showed that this pattern enforces constraint satisfaction when the PDE solver is used verbatim, but is limited to low-dimensional geometries. Chapter 4 showed that replacing the solver with a neural surrogate preserves the spatial structure while scaling to realistic 3D classrooms.

A remaining limitation is that the control design in this chapter is still based on an open-loop prediction-and-optimization pipeline. The neural operator captures detailed spatio-temporal indoor dynamics, but it is learned by minimizing prediction error on CFD-generated trajectories rather than by directly optimizing the closed-loop control objective. As a result, when occupancy patterns, boundary conditions, actuator settings, room layouts, or CO_2 distributions move outside the training domain, prediction errors can accumulate during rollout and may lead to suboptimal ventilation decisions or missed localized air-quality violations. This motivates the following work, which shifts the emphasis from learning a structured prediction model for control to

learning a structured control policy itself. Future directions also include extending to multi-zone systems with thermal comfort alongside air quality, and real-world experiments under dynamic occupancy.

Acknowledgments

The authors gratefully acknowledge Ling Zhong and Jialin Ye for the framework visualization design. This work was supported by a Schmidt Sciences AI2050 Early Career Fellowship, NSF grant ECCS-2442689, and DOE grant DE-SC0025495.

Chapter 4, in full, contains material from Y. Bian, O. Schmidt, and Y. Shi, "Operator Learning for Energy-Efficient Building Ventilation Control with Computational Fluid Dynamics Simulation of a Real-World Classroom," *Applied Energy*, 2026. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Optimization-Guided Reinforcement Learning for Energy System Control

Chapter 2 differentiated through an agent's private optimization problem to jointly identify its cost parameters and physical constraints from observed price-responsive behavior. Chapters 3 and 4 differentiated through the governing PDE dynamics, directly via the adjoint method and indirectly via a learned neural operator, to optimize building HVAC control end-to-end. In each case, the structure of an optimization-based or physics-based model was preserved as a *differentiable layer* inside a larger learning pipeline, buying sample efficiency and interpretability that black-box learners do not provide. These frameworks, however, all operate in open-loop or supervised regimes: parameters are fit to historical trajectories, and the resulting policy is then deployed as a fixed optimization problem. What remains missing is a closed-loop counterpart in which the optimization-based policy is refined directly from the cost signal it induces in the real environment, without relying on value-function approximation or expert demonstrations.

In this chapter, we study how to train an optimization-based control policy end-to-end using reinforcement learning (RL), where the policy is implicitly defined as the solution to a parametric optimal control problem with learnable cost and dynamics models, and its parameters are updated from actual closed-loop cost feedback rather than imitation data. The resulting framework, which we call DiffOP, derives analytical policy gradients by applying implicit differentiation to the underlying optimization problem and integrating the result with the standard

policy-gradient estimator. Under mild regularity conditions, we prove that DiffOP converges to an ε-stationary point within $\mathcal{O}(\varepsilon^{-1})$ iterations, and we validate its performance on nonlinear benchmark control tasks and a constrained voltage regulation problem on the IEEE 13-bus distribution system.

5.1 Introduction

Operating complex physical systems in an effective manner is of critical importance to society. Power grids [122], commercial and industrial infrastructures [123], transportation networks [124], and robotic systems [125] all require control policies that are not only high-performing but also interpretable, with reliability and safety guarantees. To this end, optimization-based policies such as model predictive control (MPC) have been studied with known [8, 9] or learned system dynamics [126], offering a principled framework for performance optimization, constraint satisfaction, and transparent decision-making.

Optimization-based policies formulate control as a mathematical optimization problem whose objective minimizes a system cost subject to dynamic equations and state-action constraints. Prior work in this area often learns system dynamics and cost functions separately from historical data to improve predictive accuracy [126, 127, 128]. However, this decoupled learning procedure leads to the *objective mismatch* problem [129, 11]: the learned models perform well in isolation (for example, they produce accurate state predictions) but fail to guide optimal control decisions, because the training objectives do not reflect closed-loop control performance.

To enable end-to-end learning of optimization-based policies, recent works [130, 19, 2, 32] have made the optimization layer differentiable, allowing the dynamics and cost functions to be jointly trained by gradient descent. These approaches primarily target supervised imitation-learning settings, where the optimization policy is trained to mimic expert demonstrations. More recently, research has moved beyond imitation learning to explore *reinforcement learning*-based training of optimization-based policies, aiming to

improve closed-loop performance through direct interaction with the environment. In particular, existing works have integrated MPC with RL [131, 132, 133, 134], leveraging the fact that MPC naturally defines a state-action value function Q, which can be updated via Q-learning to enhance closed-loop performance.

Building on this line of work, the approach developed in this chapter departs from value-function approximation and instead treats the optimization-based policy as a fully differentiable control module. We introduce DiffOP, a novel optimal control framework based on a **Diff**erentiable **O**ptimization-based **P**olicy. DiffOP computes analytical policy gradients by integrating trajectory derivatives derived from Pontryagin's Maximum Principle (PMP) [2, 32] with the standard policy-gradient algorithm [13], enabling end-to-end learning in model-free RL settings.

5.1.1 Contributions

The main contributions of this chapter are as follows:

- We propose DiffOP, which learns optimization-based control policies via policy gradients. A key technical innovation is the joint learning of both the cost function and system dynamics using environmental rewards, combining implicit differentiation with the standard REINFORCE update.
- We provide, to the best of our knowledge, the first non-asymptotic convergence guarantee for learning optimization-based policies via policy gradients. Specifically, we show that DiffOP converges to an ε-stationary point within $\mathcal{O}(\varepsilon^{-1})$ iterations.
- We validate DiffOP on a set of nonlinear control tasks and a real-world voltage control problem with safety constraints, demonstrating its ability to learn cost and dynamics models with superior control performance compared to existing RL-based MPC methods and differentiable optimization approaches in supervised settings.

The remainder of the chapter is organized as follows. Section 5.2 reviews related work. Section 5.3 formalizes the optimization-based policy and the policy-learning problem as a bi-level optimization. Section 5.4 develops the policy-gradient algorithm, including the analytical gradient of the optimization-based policy and its use inside REINFORCE. Section 5.5 establishes the non-asymptotic convergence guarantee. Section 5.6 reports experimental results. Section 5.7 concludes the chapter.

5.2 Related Work

Differentiable optimization for control.

Recent advances have enabled end-to-end learning of model-based control policies (for example, MPC) by making the underlying optimization differentiable, so that gradients with respect to optimization parameters can be propagated to downstream losses. One approach is to unroll the optimization steps within the computational graph and apply automatic differentiation [130, 135]. This is memory-intensive and may yield inaccurate gradients due to truncation. To address these limitations, Amos et al. [19] proposed differentiable MPC by approximating the original problem with iterative LQR and differentiating through its KKT conditions, though this does not support state constraints. Subsequently, Jin et al. [2] derived exact analytical gradients by differentiating through Pontryagin's Maximum Principle (PMP), enabling support for state-action constraints and improving computational efficiency. More recently, Xu et al. [32] introduced IDOC, which applies variable elimination to the KKT system to directly compute trajectory derivatives. While these methods have primarily been applied to imitation learning, the work in this chapter builds on [32] to enable gradient computation for direct policy optimization in an RL setting.

Synthesis of MPC and RL.

Supervised learning of MPC policies often relies on minimizing proxy losses (for example, prediction error), which may not align with the true control objectives. To overcome this,

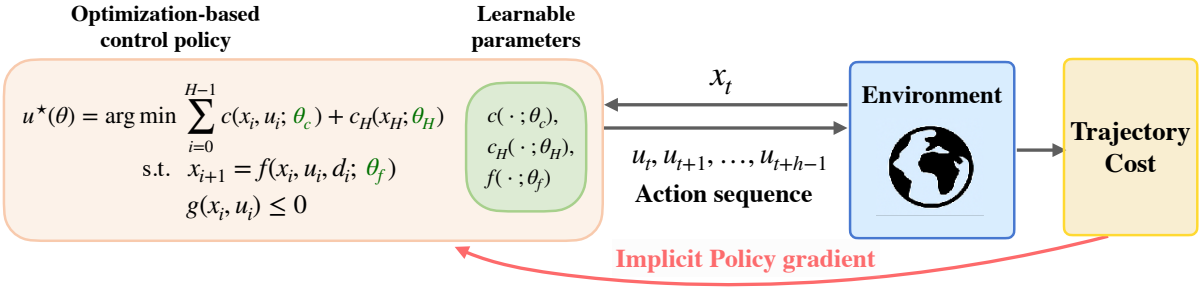


Figure 5.1. Overview of the DiffOP framework. An optimization-based control policy generates an action sequence by solving a parameterized optimal control problem with learnable dynamics and cost models. The environment executes the action sequence and returns cost feedback, which is used to compute policy gradients for updating the parameters.

there is increasing interest in integrating RL with MPC to enhance closed-loop performance. For example, Chen et al. [84] leverage Proximal Policy Optimization (PPO) to adapt linear dynamics models within the differentiable-MPC framework of [19], relying on LQR approximations. Because MPC naturally defines a state-action value function, several works [136, 137, 134] employ Q-learning with temporal-difference updates to learn improved MPC policies. Seel et al. [138] combine Q-learning with deterministic policy gradients to improve convergence. Our work builds on this direction but differs in that, instead of relying on value-function approximation or LQR-based structure, we formulate the control policy as the solution to a parameterized nonlinear optimization problem and derive exact analytical gradients via implicit differentiation. Beyond algorithmic design, we contribute theoretical convergence results, building on policy-gradient theory [139] and recent work in bi-level optimization [140, 141].

5.3 DiffOP: A Differentiable Optimization Policy

This section introduces DiffOP, the optimization-based control policy, and formalizes its learning process as a bi-level optimization problem. Throughout the chapter, we use i to denote the planning step within the optimization horizon and t to denote the actual time step in the control system.

5.3.1 Optimization-Based Control Policy

The optimization-based control policy is defined as

$$
\begin{aligned}
u^{\star}_{0:H-1}(x_{\text{init}};\theta) = \arg\min_{u} \; & \sum_{i=0}^{H-1} c(x_i,u_i;\theta_c) + c_H(x_H;\theta_H) \\
\text{subject to } & x_0 = x_{\text{init}}, \\
& x_{i+1} = f(x_i,u_i;\theta_f), \\
& g(x_i,u_i) \le 0, \quad i = 0,\dots,H-1,
\end{aligned}
\tag{5.1}
$$

where H is the planning horizon, $x_{\text{init}} \in \mathbb{R}^n$ is the initial state, and $x_i \in \mathbb{R}^n$, $u_i \in \mathbb{R}^m$ are the system state and control action at the i-th planning step. The terms $c(x_i,u_i;\theta_c)$ and $c_H(x_H;\theta_H)$ model the instantaneous and terminal costs with learnable parameters θ_c and θ_H, $f(x_i,u_i;\theta_f)$ models the system dynamics with learnable parameters θ_f, and $g(\cdot)$ represents state and control constraints, which are known to the control agent. In summary, the policy (5.1) is a parametric representation of the *unknown* cost and dynamics functions with parameters $\theta = (\theta_c,\theta_H,\theta_f) \in \mathbb{R}^d$.

Given a state x, DiffOP generates a sequence of control actions by solving (5.1). The policy output is

$$
\pi_\theta(x) = \{u^{\star}_0(x;\theta),\dots,u^{\star}_{h-1}(x;\theta)\}, \tag{5.2}
$$

where $u^{\star}_i$ denotes the i-th action from the optimized sequence and $1 \le h \le H$ specifies how many actions are executed before the next observation is received.

Remark 5.1. *Although our policy optimizes over a finite horizon, we do not label it "MPC" because it is not restricted to receding-horizon execution. DiffOP allows the execution of multiple actions per optimization step ($h \ge 1$). As shown in both the algorithm and experiments, DiffOP naturally accommodates arbitrary h through its policy-gradient formulation, supporting both step-wise (when $h = 1$, as in standard MPC) and trajectory-level control (when $h > 1$). Moreover, any standard optimization solver [142, 143] can be used to solve the policy.*

5.3.2 Policy Learning as a Bi-Level Optimization

We are interested in learning the parameters $\theta = (\theta_c, \theta_H, \theta_f) \in \mathbb{R}^d$ to minimize the overall control cost in the real environment. The policy optimization problem is formulated as

$$\min_{\theta} \quad C(\theta) := \mathbb{E}\left[\sum_{t=0}^{T} c(x_t, u_t; \phi_c)\right] \tag{5.3a}$$

$$\text{subject to} \quad x_{t+1} = f(x_t, u_t; \phi_f), \quad \forall t, \tag{5.3b}$$

$$\{u_t, \ldots, u_{t+h-1}\} \sim \pi_\theta(x_t). \tag{5.3c}$$

Upper level: unknown system (5.3a)**–**(5.3b)**.**

Equation (5.3a) describes the ground-truth system cost model with parameters ϕ_c, and (5.3b) describes the ground-truth system dynamics parameterized by ϕ_f. Both are *unknown* to the agent. The decision maker seeks an optimization-based policy of the form (5.1) with parameters $\theta = (\theta_c, \theta_H, \theta_f)$ that minimize the expected accumulated cost in the real system.

Lower level: stochastic optimization-based policy (5.3c)**.**

The lower-level optimization solves the policy (5.1) to determine the control action at each time step. To enable exploration during policy learning, we introduce stochasticity by reparameterizing $\{u_t, \ldots, u_{t+h-1}\} \sim \pi_\theta(x_t)$ as a truncated Gaussian distribution. At each decision point t, the policy generates a sequence of h actions by solving the optimization problem, and each action is sampled according to

$$u_i \sim \mathcal{N}(u_i^\star, \sigma^2 I), \quad \text{clipped to } [u_i^\star \pm \beta\sigma^2 I], \tag{5.4}$$

for $i = t, \ldots, t+h-1$. Here $u_i^\star$ is the deterministic action obtained by solving the optimization-based policy (5.2), $\sigma^2 I$ is the covariance matrix with $\sigma^2 > 0$ controlling the exploration noise, and $\beta > 0$ is a hyperparameter regulating the truncation range. This independent Gaussian sampling is used for simplicity and analytical tractability in the convergence analysis. In practice, DiffOP

Algorithm 6. DiffOP

Input: $\theta^{(0)} = (\theta_c^{(0)}, \theta_H^{(0)}, \theta_f^{(0)})$, learning rate η
for $k = 0, 1, \ldots, K-1$ **do**
  for $n = 1, \ldots, N$ **do**
    Sample the action as in (5.2)–(5.4), $\forall t$
    Collect trajectory data $(x_t^{(n)}, u_t^{(n)}, u_t^{\star(n)}), \forall t$
    Compute the gradient $\nabla_\theta u_t^{\star(n)}$ as in Proposition 5.2
  end for
  Estimate policy gradient $\widehat{\nabla}_\theta C(\theta^{(k)})$ as in Proposition 5.3
  Update policy $\theta^{(k+1)} \leftarrow \theta^{(k)} - \eta \widehat{\nabla}_\theta C(\theta^{(k)})$
end for

can be extended to sample the action sequence jointly from a multivariate Gaussian, allowing for correlated noise across the planning horizon. The expectation $\mathbb{E}$ in (5.3a) is taken over the initial state distribution $x_0 \sim \mathscr{D}$ and the trajectory $(x_0, u_0, \ldots, x_T, u_T)$ induced by the stochastic policy (5.3c). By choosing a truncated Gaussian rather than a standard Gaussian distribution, we explicitly bound the deviation of sampled actions from their optimal counterparts. This bounded exploration controls the variance of the policy gradients and will be used to establish gradient *boundedness* and *convergence*.

Crucially, we view problem (5.3) as a *policy optimization* task in which the policy is implicitly defined through an optimization problem. This formulation enables the adoption of policy-gradient methods for end-to-end training.

5.4 Proposed Algorithm

The proposed algorithm is summarized in Algorithm 6. It builds on the classic policy-gradient framework [13] and incorporates trajectory derivatives derived from Pontryagin's Maximum Principle [2, 32] to compute gradients of the actual cost with respect to the policy parameters $\theta := \{\theta_c, \theta_H, \theta_f\}$.

At each training iteration k, the algorithm collects a batch of N trajectories by sampling actions from the current optimization-based policy. For each trajectory, it computes the gradients

$\nabla_\theta u^\star$ of the optimal actions with respect to the policy parameters. These gradients are then combined into an estimate of the overall policy gradient $\widehat{\nabla}_\theta C(\theta)$, which drives the parameter update via standard gradient descent. To derive these gradients, we first present Proposition 5.2, which characterizes the differentiability of the optimization-based policy with respect to its parameters using first-order optimality conditions and the implicit function theorem.

5.4.1 Gradient of the Optimization-Based Policy

Let $x_i^\star, u_i^\star$ be the optimal state and action obtained by solving the optimization policy (5.1), where $i = 0, \ldots, H-1$ indexes the planning step, and let $\zeta^\star = (x_0^\star, u_0^\star, x_1^\star, u_1^\star, \ldots, u_{H-1}^\star, x_H^\star)$ denote the optimal trajectory over the horizon H. Stacking all constraints into a single vector-valued constraint $\kappa := (\kappa_{-1}, \kappa_0, \ldots, \kappa_H) \in \mathbb{R}^{n_\kappa}$ with $n_\kappa = (H+1)n + q$, we set $\kappa_{-1} = x_0^\star$ (the input to the policy) and

$$\kappa_i = \begin{cases} (\tilde{g}(x_i^\star, u_i^\star),\ x_{i+1}^\star - f(x_i^\star, u_i^\star; \theta_f)), & i = 0, \ldots, H-1, \\ \tilde{g}(x_H^\star), & i = H, \end{cases} \tag{5.5}$$

where $\tilde{g}(x_i^\star, u_i^\star)$ is the subset of active inequality constraints at the i-th step and q is the total number of active inequality constraints. Problem (5.1) can be solved with general-purpose solvers [144, 145] to determine both the optimal solution $\zeta^\star$ and its active constraint set κ.

Proposition 5.2 (Gradient of the optimization-based policy)**.** *Suppose $u^\star$ is the solution of the optimization-based policy* (5.1) *and $\zeta^\star$ is the resulting trajectory. Assume $c(\cdot), c_H(\cdot), f(\cdot), g(\cdot)$*

are twice differentiable in a neighborhood of $(\theta, \zeta^\star)$. *Let*

$$
\begin{aligned}
A &= \nabla_\zeta \kappa(\zeta^\star;\theta),\\
B &= \nabla^2_{\theta\zeta} J(\zeta^\star;\theta) - \sum_{i=-1}^{H}\sum_{j=1}^{|\kappa_i|} \lambda_{i,j}\nabla^2_{\theta\zeta}[\kappa_i(\zeta^\star;\theta)]_j,\\
C_m &= \nabla_\theta \kappa(\zeta^\star;\theta),\\
D &= \nabla^2_{\zeta\zeta} J(\zeta^\star;\theta) - \sum_{i=-1}^{H}\sum_{j=1}^{|\kappa_i|} \lambda_{i,j}\nabla^2_{\zeta\zeta}[\kappa_i(\zeta^\star;\theta)]_j.
\end{aligned}
$$

If $\operatorname{rank}(A) = n_\kappa$ *and* D *is non-singular, the gradient* $\nabla_\theta u_i^\star$ *takes the form*

$$
\nabla_\theta u_i^\star = [\nabla_\theta \zeta^\star]_{(n+m)i+n:(n+m)(i+1)}, \tag{5.6}
$$

with

$$
\nabla_\theta \zeta^\star = D^{-1}A^\mathsf{T}(AD^{-1}A^\mathsf{T})^{-1}(AD^{-1}B - C_m) - D^{-1}B,
$$

where n, m *are the state and action dimensions, and the Lagrange multiplier* $\lambda \in \mathbb{R}^{n_\kappa}$ *satisfies* $\lambda^\mathsf{T} A = \nabla_\zeta J(\zeta^\star;\theta)$.

Here $\lambda_{i,j}$ is the dual variable corresponding to the j-th element of $\kappa_i(\zeta^\star;\theta)$. Proposition 5.2 is a direct application of constrained optimization differentiation [32]; a detailed derivation is reported in [31].

5.4.2 Policy Gradient Estimation

We now turn to the estimation of the policy gradient. Let $\tau = (x_0, u_0, x_1, u_1, \ldots, u_{T-1}, x_T)$ be the trajectory induced by the policy. The analytical policy-gradient update rule takes the following form.

Proposition 5.3 (Policy gradient update)**.** *Consider the policy learning problem* (5.3)*. The policy gradient admits the closed-form expression*

$$\nabla_\theta C(\theta) = \mathbb{E}\left[L(\tau) \left(\sum_{t=0}^{T} \frac{1}{\sigma^2} [\nabla_\theta u_t^\star]^\mathsf{T} (u_t - u_t^\star) \right) \right], \tag{5.7}$$

where u_t and $u_t^\star$ are the actual control action and the corresponding optimal solution produced by the policy at time t, and $L(\tau) = \sum_{t=0}^{T} c(x_t, u_t; \phi_c)$ is the total trajectory cost.

Proof. The result follows from the REINFORCE gradient estimator [13] applied to a stochastic policy modeled as a Gaussian centered at the optimization-based solution $u_t^\star$. A detailed derivation is provided in [31]. □

By establishing differentiability of $u_t^\star$ with respect to θ through the implicit function theorem (Proposition 5.2), we estimate the policy gradient in practice by Monte Carlo sampling over N trajectories:

$$\widehat{\nabla}_\theta C(\theta^{(k)}) = \frac{1}{N} \sum_{n=1}^{N} \left[L(\tau^{(n)}) \sum_{t=0}^{T} \frac{1}{\sigma^2} [\nabla_\theta u_t^{\star(n)}]^\mathsf{T} (u_t^{(n)} - u_t^{\star(n)}) \right]. \tag{5.8}$$

Although Algorithm 6 uses the plain REINFORCE estimator [13], any of the standard variance-reduction techniques [146] can be incorporated without changing the core analysis.

5.5 Non-Asymptotic Convergence Analysis

We present the main theoretical result of this chapter, which establishes the convergence properties of DiffOP when trained with Algorithm 6. Because the policy is implicitly defined through an optimization problem, the sensitivity of its solution with respect to the policy parameters plays a central role in the analysis.

5.5.1 Convergence Under Bounded Sensitivity

We begin by stating a sufficient condition for convergence: bounded first- and second-order sensitivity of the optimization-defined policy with respect to its parameters.

Assumption 5.4 (Bounded first- and second-order sensitivity)**.** *There exist constants $M_{u1} > 0$ and $M_{u2} > 0$ such that for all planning steps i, the solution $u_i^\star(x;\theta)$ to the optimization-defined policy satisfies*

$$\|\nabla_\theta u_i^\star(x;\theta)\| \le M_{u1}, \qquad \left\|\nabla_\theta^2 u_i^\star(x;\theta)\right\| \le M_{u2}.$$

This assumption ensures that the policy responds smoothly to changes in the parameters θ. Next, we assume standard regularity conditions on the initial state distribution and the trajectory cost [139].

Assumption 5.5. *The initial state distribution $\mathscr{D}$ is supported in a region with a finite radius D_0. For any initial state $x_0 \sim \mathscr{D}$, the trajectory cost $L(\tau)$ is bounded, i.e., there exists a positive constant $M_L > 0$ such that $L(\tau) \le M_L$.*

We can now characterize the convergence of the proposed algorithm.

Theorem 5.6 (Convergence of DiffOP with policy gradient)**.** *Suppose Assumptions 5.4 and 5.5 hold. For any $\varepsilon > 0$ and $\nu \in (0,1)$, define the smoothness constant*

$$L_C = M_L T\left(\sqrt{m}\beta M_{u2} + \tfrac{1}{\sigma^2} M_{u1}^2 + Tm\beta^2 M_{u1}^2\right),$$

choose step size $\eta = \frac{1}{4L_C}$, the number of policy iterations K, and the number of sampled trajectories per policy-gradient step $N = \frac{2m\beta^2 M_L^2 T^2 M_{u1}^2}{\varepsilon^2} \log \frac{2Kd}{\nu}$. Then, with probability at least $1-\nu$,

$$\min_{k=0,\dots,K-1} \|\nabla_\theta C(\theta^{(k)})\|^2 \le \frac{16 L_C (C(\theta^{(0)}) - C(\bar{\theta}))}{K} + 3\varepsilon, \tag{5.9}$$

where $\bar{\theta}$ is the global optimum of (5.3)*.*

Proof. The detailed proof builds on standard stochastic approximation techniques [139], adapted to the implicit policy structure induced by optimization-based control. The argument proceeds in three steps: (i) bounded sensitivity (Assumption 5.4) combined with the truncated Gaussian sampling (5.4) implies that the score function $\nabla_\theta \log \pi_\theta(u_t|x_t)$ and its Hessian are bounded; (ii) these bounds imply L_C-smoothness of $C(\theta)$; and (iii) a Hoeffding concentration bound on the REINFORCE estimator yields the stated sample complexity. The full derivation is provided in [31]. □

5.5.2 Sufficient Conditions for Bounded Sensitivity

We now describe a common class of optimization-based policies that satisfy the sensitivity bounds in Assumption 5.4. Specifically, we consider the unconstrained, strongly convex version of (5.1), for which concrete sufficient conditions can be established.

Let $x_0 = x_{\text{init}}$ denote the initial state, and let $u = [u_0, \dots, u_{H-1}] \in \mathbb{R}^{mH}$ be the control sequence generated by the optimization-based policy with parameters θ. The unrolled cost function can be written as

$$J(x_{\text{init}}, u, \theta) = \sum_{i=0}^{H-1} c(x_i, u_i; \theta_c) + c_H(x_H; \theta_H), \tag{5.10}$$

subject to the dynamics $x_0 = x_{\text{init}}$, $x_{i+1} = f(x_i, u_i; \theta_f)$ for $i = 0, \dots, H-1$.

Assumption 5.7. *The function $J(x_{init}, u, \theta)$ is μ-strongly convex with respect to u.*

Strong convexity ensures the uniqueness of the optimal solution and the stability of the mapping from θ to the control sequence $u^\star(x; \theta)$. This injectivity yields well-defined gradients and underpins the convergence guarantees.

Assumption 5.8. *Let $z = (x_{init}, u, \theta)$ and let $\mathscr{Z} \subset \mathbb{R}^{n_z}$ be a compact set. The unrolled cost function $J(z)$ satisfies:*

- *$\nabla_z J(z)$ is L_1-Lipschitz: $\|\nabla_z J(z) - \nabla_z J(z')\| \le L_1 \|z - z'\|$ for all $z, z' \in \mathscr{Z}$.*

- $\nabla_\theta \nabla_u J(z)$ *is* L_2*-Lipschitz:* $\|\nabla_\theta \nabla_u J(z) - \nabla_\theta \nabla_u J(z')\| \le L_2 \|z - z'\|$ *for all* $z, z' \in \mathscr{Z}$.
- $\nabla_u^2 J(z)$ *is* L_3*-Lipschitz:* $\|\nabla_u^2 J(z) - \nabla_u^2 J(z')\| \le L_3 \|z - z'\|$ *for all* $z, z' \in \mathscr{Z}$.

Assumption 5.8 captures the smoothness of the unrolled cost function and its partial derivatives with respect to the optimization variables and policy parameters. These conditions ensure that the gradients and Hessians involved in the policy-sensitivity analysis are well behaved. Under Assumptions 5.7 and 5.8, bounded sensitivity follows.

Proposition 5.9. *Under Assumptions 5.7 and 5.8, the optimization-defined policy* π_θ *satisfies the bounded-sensitivity condition in Assumption 5.4. In particular, there exist constants*

$$M_{u1} = \frac{L_1}{\mu}, \qquad M_{u2} = \frac{L_2}{\mu} + \frac{L_1 L_2 + L_1 L_3}{\mu^2} + \frac{L_1^2 L_3}{\mu^3},$$

such that for all planning steps i,

$$\|\nabla_\theta u_i^\star(x;\theta)\| \le M_{u1}, \qquad \left\|\nabla_\theta^2 u_i^\star(x;\theta)\right\| \le M_{u2}.$$

Proof sketch. Applying the implicit function theorem [147] to the first-order optimality condition $\nabla_u J(x_{\text{init}}, u^\star(\theta), \theta) = 0$ gives $\nabla_\theta u^\star(\theta) = -[\nabla_u^2 J(z^\star)]^{-1} \nabla_\theta \nabla_u J(z^\star)$, and the first bound follows by strong convexity ($\|[\nabla_u^2 J]^{-1}\| \le 1/\mu$) together with L_1-Lipschitzness of $\nabla_u J$. The second bound follows by expanding $\|\nabla_\theta u^\star(\theta) - \nabla_\theta u^\star(\theta')\|$ via the chain rule and bounding each factor using Assumption 5.8. The full derivation is provided in [31]. □

Discussion.

The analysis above establishes the first non-asymptotic convergence guarantee for learning optimization-based policies via policy gradients: DiffOP converges to an ε-stationary point within $\mathscr{O}(1/\varepsilon)$ iterations under mild smoothness and strong convexity assumptions. Notably, this matches the standard convergence rate of first-order policy-gradient methods, despite the added complexity of differentiating through an implicit optimization-based policy. The

Table 5.1. Nonlinear experimental settings, adapted from [2].

Systems	Objective parameters θ_c, θ_H	Dynamic parameters θ_f
Cartpole	$c(x_i,u_i) = \|\theta_c^{\mathsf{T}}(x_i - x_{\text{nom}})\|^2 + \|u_i\|^2$	length and mass for each link
Two-link robot arm		length and mass for each link
6-DoF quadrotor maneuvering	$c_H(x_H) = \|\theta_H^{\mathsf{T}}(x_H - x_{\text{nom}})\|^2$	mass, wing length, inertia matrix

central technical challenge lies in bounding the sensitivity of the optimization solution with respect to the policy parameters, which we address by introducing structural assumptions on the inner optimization problem and leveraging analytical tools from bi-level optimization theory [140, 141]. In more general settings (non-convex or constrained optimization), the solution mapping can become non-smooth or even discontinuous, especially near constraint boundaries or saddle points, potentially leading to unbounded gradients and unstable updates; for instance, small parameter changes can trigger abrupt changes in the active constraint set, breaking the smoothness assumptions required for implicit differentiation. Extending the convergence analysis to such cases is an important direction for future work. In practice, however, many smooth non-convex constrained problems exhibit locally bounded sensitivities [148] and satisfy Assumption 5.4 empirically; the experiments that follow include constrained control tasks to illustrate this.

5.6 Experiments

This section evaluates DiffOP on nonlinear control tasks and a constrained voltage regulation problem on the IEEE 13-bus distribution system.

5.6.1 Nonlinear Control Tasks

We follow the experimental setup of [2] and summarize the environment details in Table 5.1. All dynamical systems are discretized with the Euler scheme $x_{t+1} = x_t + \Delta t \cdot f(x_t, u_t)$ with $\Delta t \in \{0.05, 0.1\}$ s.

We compare DiffOP against the following optimization-based control baselines. In all methods, the forward pass is solved with the CasADi optimization solver [142]:

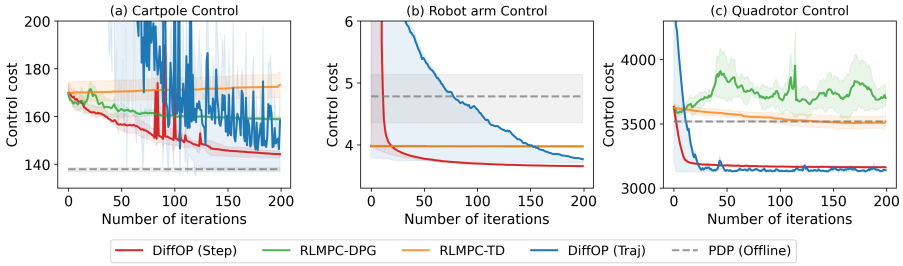


Figure 5.2. Control cost versus training iteration on the nonlinear control tasks. Solid lines indicate the mean cost over 5 runs, and shaded regions denote the 20th–80th percentile range.

- **PDP (offline)** [2]: learns the optimization parameters via Pontryagin Differentiable Programming from expert trajectories. As a purely offline method, it does not adapt online; reported performance is taken directly from the original paper.
- **RLMPC-TD** [136]: a model-free RL approach that updates a short-horizon MPC policy using temporal-difference (TD) Q-learning.
- **RLMPC-DPG** [138]: a variant of RLMPC-TD that also uses a deterministic policy gradient (DPG) to update the MPC parameters.

DiffOP is evaluated under two deployment modes to highlight its flexibility:

- **DiffOP (Step):** at each time step, apply only the first action of the optimized trajectory, mirroring the receding-horizon scheme of RLMPC-TD and RLMPC-DPG.
- **DiffOP (Traj):** generate the entire optimized control sequence once at the start of the episode and execute it open loop. This trajectory-optimization setting boosts temporal consistency at the cost of robustness.

For a fair comparison, the planning horizon is set to 6 for Cartpole and 12 for the Robot Arm and Quadrotor across RLMPC-TD, RLMPC-DPG, and DiffOP (Step). DiffOP (Traj) sets $H = T$ (the full episode length). For all DiffOP experiments we use $\beta = 0.01$, $\eta = 0.1$, and $N = 10$. The RL-based MPC baselines are implemented with the MPC4RL framework [149].

Results are reported in Figure 5.2. Compared to the RL-based MPC baselines, DiffOP (Step) achieves faster convergence and a lower final control cost across all nonlinear tasks; RLMPC-TD and RLMPC-DPG often converge to suboptimal solutions and may fail to consistently reduce the control cost. Compared to PDP (offline), DiffOP (Traj) achieves lower cost on the Robot Arm and Quadrotor tasks, showing the benefit of policy refinement through online interaction: PDP relies solely on supervised expert data, whereas DiffOP (Traj) fine-tunes the policy via RL and adapts better to the true system dynamics. These results highlight the effectiveness of DiffOP under both receding-horizon and open-loop settings, and its robustness across control tasks with different levels of nonlinearity and complexity.

5.6.2 Voltage Control with Power Injection Constraints

We further evaluate DiffOP on a practical voltage control scenario involving reactive power constraints, using the IEEE 13-bus radial distribution benchmark. Following the setup of [150], we use the Pandapower toolbox [151] to simulate the power network under two voltage-disturbance scenarios: (1) high voltage due to excess PV generation under strong sunlight; and (2) low voltage caused by peak loads with insufficient generation. For each scenario, load profiles are varied to induce deviations of 5%–15% from the nominal voltage v^{nom}. The goal of voltage control is fast voltage restoration to within a 5% acceptable range with minimal control effort. The nominal bus voltage is 4.16 kV with an acceptable operating range of $\pm 5\%$, i.e., $[3.952\ \text{kV}, 4.368\ \text{kV}]$. State constraints are explicitly enforced inside the optimization-based policy.

Let $v_i \in \mathbb{R}^3$ denote the squared voltage magnitudes, $q_i \in \mathbb{R}^3$ the reactive power injections, and $u_i \in \mathbb{R}^3$ the control inputs at planning step i. The optimization-based control policy for

voltage regulation is

$$\begin{aligned}
\arg\min_{u} \quad & \sum_{i=0}^{H-1} \left(u_i^\top C_u u_i + (v_i - v^{\text{nom}})^\top C_v (v_i - v^{\text{nom}}) \right) \\
& + (v_H - v^{\text{nom}})^\top C_v (v_H - v^{\text{nom}}) && \text{(5.11a)} \\
\text{s.t.} \quad & v_{i+1} = A q_i + v^{\text{env}}, \quad q_{i+1} = q_i + u_i, && \text{(5.11b)} \\
& \underline{q} \le q_i \le \overline{q}. && \text{(5.11c)}
\end{aligned}$$

The objective (5.11a) penalizes both voltage deviation from v^{nom} and control effort, where C_u is a fixed parameter reflecting actuation cost. Equation (5.11b) models the voltage and reactive-power dynamics, where v^{env} is the initial voltage determined by load profiles and is considered static due to time-scale separation between load change and the voltage control process. Constraint (5.11c) ensures that reactive-power injections remain within operational limits. The policy parameters are $\theta = \{C_v, A\}$, which are learned to optimize both transient and steady-state performance.

DiffOP is evaluated against all prior baselines as well as two state-of-the-art algorithms designed for voltage control:

- **Stable-DDPG** [152]: an RL-trained monotone neural policy for voltage control with theoretical voltage-restoration guarantees.
- **TASRL** [150]: an RL policy that combines Stable-DDPG with a safe gradient flow to achieve fast voltage restoration while guaranteeing steady-state optimality under limited reactive-power capacity.

All optimization-based policies are trained on 10,000 trajectories of length $T = 30$ seconds. The planning horizon is set to $H = 6$ for RLMPC-TD, RLMPC-DPG, and DiffOP (Step); DiffOP (Traj) and PDP (offline) use $H = 30$ with open-loop execution. The overall performance is summarized in Table 5.2. Stable-DDPG and TASRL numbers are reported from [150].

Table 5.2. Voltage control performance on 500 scenarios of the IEEE 13-bus system. Lower cost indicates better performance. DiffOP (Traj) and PDP use horizon $H = 30$ with open-loop execution; DiffOP (Step), RLMPC-DPG, and RLMPC-TD use $H = 6$ with first-action execution.

	Transient cost	Steady-state cost
DiffOP (Step)	**-6.81**	**-0.11**
RLMPC-DPG	-6.11	-0.10
RLMPC-TD	-4.62	-0.07
DiffOP (Traj)	**-6.80**	**-0.11**
PDP (offline)	-5.86	-0.09
Stable-DDPG	-5.61	-0.09
TASRL	-6.76	**-0.11**

(a) DiffOP (Traj) (b) DiffOP (Step)

Figure 5.3. Voltage and reactive-power trajectories under DiffOP. Solid lines show post-training responses; dashed lines show pre-training performance. Gray shaded areas denote the safe operating bounds.

DiffOP consistently achieves the lowest transient and steady-state costs across all baselines. Compared to the MPC-based approaches, it effectively learns both cost and dynamics parameters and achieves the best transient and steady-state performance simultaneously. Notably, DiffOP also outperforms RL-based neural-network policies on transient cost and matches TASRL's steady-state cost. Figure 5.3 visualizes voltage and reactive-power trajectories before and after training: after policy optimization, both variants exhibit significantly improved voltage regulation and faster convergence, with all voltage trajectories stabilized inside the safe operating bounds, in contrast to the untrained policy. Although DiffOP (Step) and DiffOP (Traj) use

different planning configurations, both variants consistently learn effective control policies, highlighting that the framework naturally supports both receding-horizon and open-loop execution.

5.6.3 Implementation Details

Initialization.

For the nonlinear control tasks, all optimization-based policies are initialized following [2]. For the voltage control task, all approaches are initialized with $A = 0.5I$ and $C_v = I$; under this initialization the resulting policy does not drive the voltage into the acceptable 5% deviation range, so all improvement must come from learning.

DiffOP hyperparameters.

For both task families we use $N = 10$ trajectory samples per gradient estimate, exploration parameter $\beta = 0.01$, and learning rate $\eta = 0.1$. DiffOP (Step) uses $H = 6$ for Cartpole and voltage control, and $H = 12$ for the Robot Arm and Quadrotor; DiffOP (Traj) uses $H = T$ equal to the full episode length.

RLMPC-TD and RLMPC-DPG.

Both are implemented using the MPC4RL framework [149]. For each control task, a grid search is performed over learning rates $\{10^{-1}, 10^{-2}, \ldots, 10^{-8}\}$ and (for RLMPC-DPG) exploration rates $\{10^{-1}, \ldots, 10^{-5}\}$ to select the best-performing setting.

PDP (offline).

We follow [2] and use the authors' published code. For the voltage control task, 10,000 trajectories are generated using expert demonstrations from TASRL [150] and used to train a PDP policy with $H = T$, matching DiffOP (Traj).

Runtime.

Table 5.3 reports average runtimes for the voltage control task. All experiments run on a single thread of an Intel Core i7-11700K CPU. DiffOP (Step) uses a shorter horizon than DiffOP (Traj), so each forward/backward pass is cheaper; however, DiffOP (Step) performs both

Table 5.3. Average runtime on the voltage control task.

	DiffOP		PDP (offline)	RLMPC-DPG	RLMPC-TD
	Traj	Step			
Forward (solving the policy) [s]	0.026	0.011	0.026	–	–
Backward (computing gradients) [s]	0.010	0.002	0.010	–	–
Total training time [hours]	0.37	1.36	1.94	0.86	0.72

operations at every time step in a trajectory, while DiffOP (Traj) performs them only once per trajectory, so DiffOP (Traj) has the shorter total training time. Compared to PDP (offline), the policy-gradient computation in DiffOP does not significantly increase the backward-pass time. PDP (offline) is trained entirely on offline data with a training time determined by the number of iterations (10,000 iterations in this setup, with 10 trajectories per iteration). RLMPC-TD and RLMPC-DPG are implemented through the MPC4RL framework [149]; we report their total runtime without separating forward and backward passes.

5.7 Chapter Summary

This chapter developed DiffOP, a framework for learning optimization-based control policies via analytical implicit policy gradients. The policy is defined as the solution to a parameterized optimal control problem with learnable cost and dynamics models, and its parameters are updated directly from closed-loop cost feedback using a REINFORCE-style estimator combined with trajectory derivatives obtained by implicit differentiation of the inner optimization. Under bounded sensitivity of the optimization-defined policy (Assumption 5.4) and standard regularity conditions on the trajectory cost, we established the first non-asymptotic convergence guarantee for policy-gradient learning of optimization-based policies: DiffOP converges to an ε-stationary point within $\mathcal{O}(\varepsilon^{-1})$ iterations. We further showed that strong convexity and Lipschitz smoothness of the unrolled cost function are sufficient conditions for bounded sensitivity. Experiments on nonlinear benchmark control tasks and a constrained voltage regulation problem on the IEEE 13-bus system demonstrated that DiffOP consistently outperforms both RL-based MPC methods and differentiable optimization approaches trained in supervised settings.

Viewed in the context of this dissertation, DiffOP closes a loop that has been forming across the preceding chapters. Chapter 2 used differentiable optimization for *identification* of price-responsive demand response agents from observed behavior. Chapters 3 and 4 used differentiable physical models for *open-loop* design of energy-efficient HVAC control actions. DiffOP uses differentiable optimization as the *policy itself*, trained end-to-end from closed-loop environmental feedback. The same structural idea, preserving the optimization (or physics) inside the learning pipeline, is instantiated across all three settings.

Several directions remain open. First, improving sample complexity, either through variance-reduced gradient estimators or by exploiting structure in the dual variables, would make policy-gradient training of optimization-based policies more practical for high-dimensional energy systems. Second, extending the convergence analysis beyond the strongly convex case, to weakly convex or smooth non-convex inner problems and to constrained settings with active-set changes, remains an important theoretical challenge. Third, it would be of significant interest to further investigate the robustness of DiffOP in comparison to model-free RL approaches, and to extend the experimental results to a wider range of real-world deployments, including real-time grid operation and cyber-physical infrastructure.

The main boundary of DiffOP lies in the structure of the inner optimization problem. The framework is most appropriate for energy-system control problems whose inner policy layer is a continuous optimization problem and whose optimal solution varies smoothly with the learnable parameters over the region visited during training. This includes many convex or locally regular nonlinear programs, such as economic dispatch with smooth generation costs, storage arbitrage with convex degradation models, HVAC control with continuous actuator limits, optimal power-flow approximations with fixed network topology, and voltage or reactive-power control problems with continuously varying injections and box constraints. In these settings, even when constraints are present, DiffOP can be applied when the active set remains locally stable, constraint qualifications hold, second-order sufficient conditions are satisfied, and the solver returns a consistent local optimizer.

By contrast, DiffOP is not a general-purpose differentiable layer for all energy-system optimization problems. It is not designed to directly differentiate through inherently discrete or combinatorial inner problems, such as unit commitment with binary on/off variables, discrete tap-changing transformers, capacitor-bank switching, or market-clearing rules with block bids and nonconvex acceptance constraints. These problems typically produce piecewise-constant or discontinuous decision maps, so small changes in learned parameters may cause abrupt changes in the optimal action rather than smooth policy variations. Similarly, inner problems with multiple equally optimal solutions, degenerate binding constraints, complementarity conditions, nonsmooth penalty functions, or abrupt protection/control logic fall outside the clean differentiability regime unless they are relaxed, smoothed, regularized, or replaced by a differentiable surrogate. Thus, the present form of DiffOP should be viewed as a policy-learning method for smooth continuous optimal-control layers in energy systems, rather than as a differentiable solver for arbitrary mixed-integer, nonsmooth, or discontinuous operational problems.

Acknowledgments

This work is supported by a Schmidt Sciences AI2050 Early Career Fellowship, NSF grant ECCS-2442689, and DOE grant DESC0025495. The work of J. Feng is supported by the UC-National Laboratory In Residence Graduate Fellowship L24GF7923.

Chapter 5, in full, contains material from Y. Bian, J. Feng, and Y. Shi, "DiffOP: Reinforcement Learning of Optimization-Based Control Policies via Implicit Policy Gradients," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. The dissertation author was the primary investigator and author of this paper.

Chapter 6

Structured Policy Representations for On-Policy Reinforcement Learning

Chapter 5 showed that when the policy is defined implicitly as the solution of a parametric optimal control problem, it can be trained end-to-end through reinforcement learning with provable convergence guarantees. In this setting, the embedded optimization structure serves as a powerful source of inductive bias by incorporating domain knowledge directly into the policy class.

In this chapter, rather than further enriching the internal structure of the policy, we change only the actor representation and evaluate it within modern state-of-the-art on-policy methods, including PPO [14], TRPO [153], and SPO [154]. This shift still fits the broader thesis agenda: for energy systems and other high-stakes control problems alike, the choice of policy representation is itself a structural design decision that can materially affect sample efficiency, stability, and deployability. From the perspective of *policy gradient variance*, we study which policy representations and actor architectures are most effective for stable and efficient on-policy reinforcement learning.

We show that discrete categorical policies, which cast the policy update as a cross-entropy-like objective over discretized action bins, can substantially reduce gradient variance and improve optimization stability relative to continuous Gaussian policies (Figure 6.1). Building on this insight, we further propose a regularized actor architecture, RN-D, that improves training

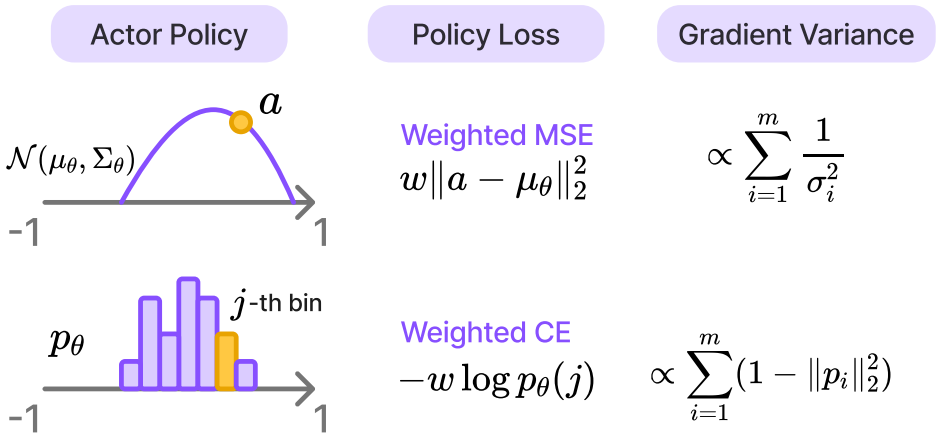


Figure 6.1. Two perspectives on continuous Gaussian and discrete categorical actor policies: gradient variance (left) and optimization loss structure (right).

stability and scalability for discretized actors. Across standard locomotion benchmarks and ManiSkill, spanning both state-based and vision-based tasks, our approach consistently improves control performance and accelerates convergence. Additional experiments under PPO, TRPO, and SPO show that these gains are not tied to a single on-policy surrogate objective.

6.1 Introduction

On-policy reinforcement learning (RL) is a central paradigm for sequential decision-making, underpinning widely used algorithms such as policy-gradient methods and their modern variants [155, 156, 157, 14]. Despite their favorable theoretical properties and widespread adoption, these methods remain notoriously difficult to optimize in practice. For example, Proximal Policy Optimization (PPO) is often reported to exhibit unstable training dynamics, pronounced sensitivity to hyperparameters, and strong dependence on policy parameterization choices [16]. Even on standard continuous-control benchmarks, reliable performance typically requires careful "code-level" optimization [158]. These observations suggest that the primary challenges of on-policy RL are not solely due to insufficient data or model capacity, but are closely tied to the optimization behavior induced by architectural and parameterization choices.

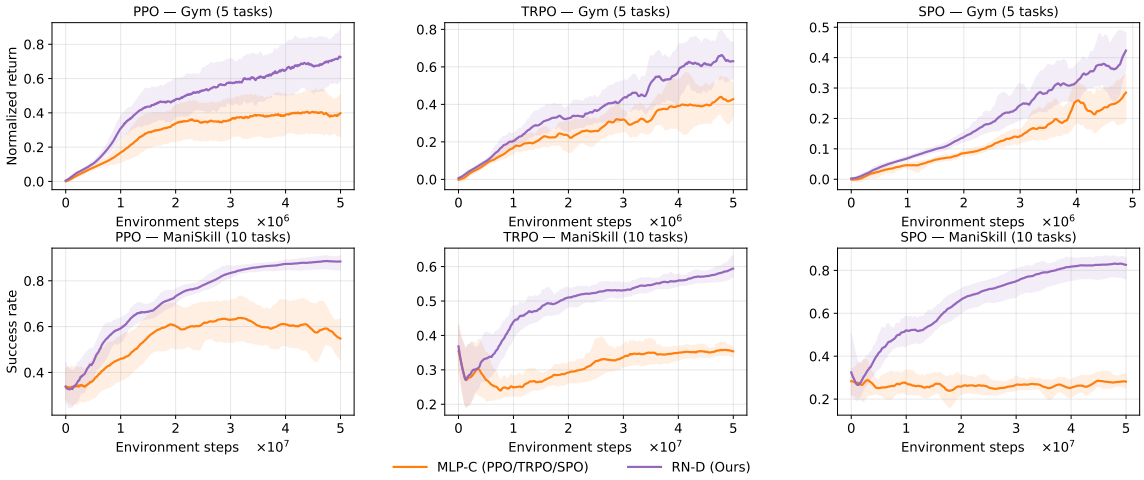


Figure 6.2. Comparison of RN-D with the standard continuous MLP actor under PPO, TRPO, and SPO on Gym locomotion and ManiSkill manipulation benchmarks.

Considerable effort has gone into understanding and improving the training performance of on-policy RL, including improved advantage estimators [159], analyses highlighting the role of value estimation [160], and the design of constrained policy update mechanisms [156, 14, 154]. In contrast, the parameterization of the policy itself is often treated as a secondary design choice; continuous control tasks typically adopt Gaussian policies with diagonal covariance as a default following [161], largely due to their analytical and computational convenience.

Recent work challenges this convention from two complementary directions. First, earlier on-policy studies show that discretizing continuous actions and optimizing categorical policies can yield substantial gains over Gaussian actors [37, 162]. Second, value-function learning has increasingly embraced classification-style objectives: distributional RL trains categorical return distributions with a cross-entropy loss [34], and recent work shows that replacing mean squared error with cross-entropy can improve robustness to noisy bootstrapped targets and support scaling to higher-capacity networks [35, 163]. These advances are particularly pronounced in off-policy RL, where the value function is the primary object being optimized and directly shapes the policy update. A symmetric perspective applies to on-policy RL: the actor is the object being optimized, and discretized categorical actors turn the policy update into a cross-

entropy-like objective over action bins, making architectural and optimization choices that benefit classification especially relevant. A closely related phenomenon appears in on-policy RL for language models, where the policy is categorical over tokens and PPO-style optimization is widely used for RLHF [164, 36, 165, 166]. On the theoretical side, Agarwal et al. [167] show that policy-gradient methods admit global convergence guarantees with the softmax policy parameterization under exact gradients and adequate exploration.

Motivated by this connection, we revisit actor design in on-policy RL for continuous control tasks through the lens of classification-style objectives. We make a simple drop-in change to the policy parameterization: instead of a Gaussian policy with continuous action outputs, we use a factorized categorical policy, which turns the on-policy policy loss into a cross-entropy-like objective over selected bins. We then pair this parameterization with a regularized actor network that promotes more stable training.

6.1.1 Contributions

The contributions of this chapter can be summarized as follows:

- We revisit actor parameterization and architecture as underexplored levers for improving on-policy continuous control, and study discretized categorical policies whose optimization resembles cross-entropy loss.
- We propose a regularized actor network design (RN-D: **R**egularized **N**etwork for **D**iscrete action policies) for discretized on-policy learning that improves optimization stability and reduces gradient variance during training.
- We demonstrate that our approach consistently improves control performance and accelerates convergence across standard locomotion benchmarks and ManiSkill, covering both state-based and vision-based tasks, and that the same trend holds across PPO, TRPO, and SPO.

The remainder of the chapter is organized as follows. Section 6.2 reviews related work. Section 6.3 introduces preliminaries on on-policy actor optimization and the two actor families considered in this chapter. Section 6.4 revisits discretized categorical policies through a policy-gradient variance analysis and an optimization-view comparison between cross-entropy and squared-error objectives. Section 6.5 presents the RN-D actor network architecture. Section 6.6 reports the main experimental results, and Section 6.7 presents extended ablations. Section 6.8 concludes the chapter.

6.2 Related Work

On-policy reinforcement learning.

On-policy policy-gradient methods optimize the expected return by differentiating through a stochastic policy, dating back to REINFORCE [161] and the policy-gradient theorem with function approximation [33]. Modern deep actor-critic systems improved stability and throughput via parallel data collection and actor-critic updates [168]. A central challenge in on-policy learning is balancing stability and sample reuse. Natural-gradient and trust-region perspectives [169, 170] motivated algorithms that constrain policy updates, most notably TRPO [153]. PPO [14] popularized a simpler clipped surrogate objective that supports multiple epochs of minibatch updates, and generalized advantage estimation (GAE) further improved the variance–bias tradeoff in advantage computation [159]. In parallel, several large-scale empirical studies have shown that the performance of on-policy methods depends strongly on "implementation ingredients" beyond the high-level objective, including normalization, clipping, network design, and optimization details [16, 171]. Motivated by these findings, the work in this chapter targets a complementary axis: we study how the *policy representation itself*, specifically categorical parameterizations combined with the actor network architecture, affects optimization behavior under standard on-policy training.

Actor policy parameterization for continuous control.

Most continuous-control actor-critic implementations parameterize the policy with a diagonal Gaussian distribution [16] due to its simplicity and reparameterization-friendly sampling; however, this choice can interact poorly with exploration and gradient variance. To address the limitations of Gaussian policies with bounded action spaces, prior work replaces them with finite-support Beta distributions [172], yielding a bias-free, lower-variance policy that performs well empirically. By considering the extremes along each action dimension, Bernoulli distributions have also been used to replace the Gaussian parameterization of continuous control and show improved performance on several continuous-control benchmarks [38]. Within PPO-style training, PPO-CMA adapts the exploration covariance inspired by CMA-ES to mitigate premature variance collapse in Gaussian policies [173].

An orthogonal line of work replaces continuous distributions with *discretized* categorical policies over per-dimension action bins. Tang and Agrawal [37] showed that factorized categorical policies can scale to high-dimensional control and often improve on-policy optimization, and further proposed ordinal parameterizations to explicitly encode the natural ordering of action bins. More recently, unimodal discrete parameterizations (for example, Poisson-based) were introduced to enforce ordering structure and reduce undesirable multimodality and gradient variance in discretized actors [162].

Network architectures.

Recent advances in computer vision and natural language processing have been largely driven by scaling model capacity [174]. In supervised learning, architectural choices, including skip connections, normalization, and depth/width scaling, are known to strongly influence optimization. Residual networks [175] improve the trainability of deep models via identity pathways, while batch normalization [176] and layer normalization [177] help stabilize activations and gradients. In contrast, network design and systematic scaling have been comparatively less

explored in deep RL, where many standard benchmarks and implementations [178, 16] still rely on relatively shallow MLP backbones (e.g., 2–3 layers) for both the actor and the critic.

Several recent works revisit careful network designs for deep RL, often highlighting how network capacity can substantially influence performance [174, 179, 180]. SimBa [174] and its follow-up SimBa-v2 [181] propose scalable MLP backbones for deep RL, showing that residual pathways and improved normalization can unlock consistent performance gains as model capacity increases. BroNet provides an extensive study of critic scaling and introduces a regularized residual MLP architecture to unlock performance gains when increasing critic capacity [182]. While these works primarily improve performance by scaling backbones and strengthening value learning (often in off-policy settings), a systematic understanding of how *actor* network architecture influences *on-policy* optimization remains less mature. We complement this line of research by studying actor network design principles under on-policy training, and by considering discretized categorical actors whose policy objective resembles a cross-entropy loss over action bins.

6.3 Preliminaries

We consider standard episodic RL in continuous-control Markov decision processes (MDPs) with state space $\mathscr{S}$, action space $\mathscr{A} \subset \mathbb{R}^m$, transition dynamics $P(\cdot \mid s,a)$, reward function $r(s,a)$, and discount factor $\gamma \in (0,1)$. A stochastic actor policy $\pi_\theta(a \mid s)$ is parameterized by θ and induces the discounted return

$$J(\theta) \triangleq \mathbb{E}_{\tau\sim\pi_\theta}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t)\Big], \tag{6.1}$$

where $\tau = (s_0,a_0,s_1,a_1,\ldots)$ is a trajectory generated by interacting with the environment under π_θ.

6.3.1 On-Policy Actor Optimization

On-policy methods such as Proximal Policy Optimization (PPO) [14] update the actor using data collected from the current policy. Let $\pi_{\theta_{\text{old}}}$ denote the behavior policy used to sample a batch of trajectories. PPO maximizes a clipped surrogate objective that stabilizes policy updates by constraining the change in action probabilities:

$$\mathscr{L}^{\text{PPO}}(\theta) = \mathbb{E}\Big[\min\big(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t\big)\Big], \tag{6.2}$$

where $r_t(\theta) \triangleq \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the importance ratio, $\varepsilon > 0$ is the clipping threshold, and $\hat{A}_t$ is an advantage estimate computed from a learned value function $V_\phi(s)$ (for example, via generalized advantage estimation). Unless stated otherwise, we consider the standard actor-critic setup where ϕ is updated by minimizing a squared-error value loss and the actor is updated by maximizing (6.2).

6.3.2 Policy Parameterizations for Continuous Control

We consider two common stochastic actor families for continuous control: a continuous Gaussian policy and a discrete categorical policy constructed via action discretization. Without loss of generality, we assume the action space $\mathscr{A} = [-1,1]^m$.

Continuous Gaussian actor.

The continuous actor models $\pi_\theta(a \mid s)$ as a multivariate Gaussian distribution

$$\pi^c(a \mid s) = \mathscr{N}\big(\mu_\theta(s), \Sigma_\theta(s)\big), \tag{6.3}$$

where $a \in \mathbb{R}^m$. The mean $\mu_\theta(s) \in \mathbb{R}^m$ is produced by a neural network, and the covariance is typically diagonal, $\Sigma_\theta(s) = \text{Diag}(\sigma_\theta^2(s)) \in \mathbb{R}^{m\times m}$, with $\sigma_\theta(s) \in \mathbb{R}^m_+$ either state-dependent or state-independent.

Discrete categorical actor via uniform discretization.

In contrast, a discrete actor represents the policy as a categorical distribution over a finite set of discretized actions. For each action dimension $i \in \{1,\dots,m\}$, we define a uniform discretization of $[-1,1]$ into K bins:

$$\mathscr{A}_i = \left\{ \tfrac{2j}{K-1} - 1 \;\middle|\; j = 0,1,\dots,K-1 \right\}. \tag{6.4}$$

The policy over these K discrete choices is parameterized by logits $z_\theta(s)_{i,j}$ and a softmax:

$$\pi^d(a^i_j \mid s) = \frac{\exp\left(z_\theta(s)_{i,j}\right)}{\sum_{k=1}^{K} \exp\left(z_\theta(s)_{i,k}\right)}, \quad j = 1,\dots,K. \tag{6.5}$$

We assume conditional independence across action dimensions, yielding the factorized joint policy

$$\pi^d(a \mid s) = \prod_{i=1}^{m} \pi^d(a^i_{j_i} \mid s), \tag{6.6}$$

where $a = (a^1_{j_1},\dots,a^m_{j_m})$ corresponds to selecting one bin index j_i per dimension. The resulting action lies in the original range $[-1,1]^m$ by construction. Beyond the plain softmax over per-bin logits, some works [37, 162] explore structured discrete actor parameterizations that exploit the ordinal structure of discretized action bins to improve optimization stability. In this chapter we focus on the simplest baseline, uniform discretization with an independent softmax categorical policy per action dimension, to isolate the effect of discretizing the actor without additional architectural constraints.

Training with PPO.

Both the continuous Gaussian actor and the discretized categorical actor can be trained under the same on-policy optimization framework. The only policy-specific ingredient is the log-likelihood $\log \pi_\theta(a_t \mid s_t)$ used to form the importance ratio in the PPO surrogate objective: it is evaluated under a Gaussian density for π^c and under a categorical probability mass function

for π^d. All other components of the training pipeline, including on-policy rollout collection, advantage estimation, and clipped surrogate maximization, are identical.

6.4 Revisiting Discrete Categorical Policies

This section revisits discretized categorical policies for continuous control from two complementary viewpoints (illustrated in Figure 6.1). First, we analyze how the policy-gradient estimator behaves under a factorized categorical policy and how its variance depends on the discretization. Second, we provide an optimization perspective that connects PPO actor updates to standard supervised losses: cross-entropy for categorical actors and squared-error (under Gaussian likelihood) for continuous actors. These perspectives motivate the empirical and algorithmic choices studied later in the chapter.

6.4.1 Policy-Gradient Estimator and Variance

On-policy policy gradient.

To analyze the stochastic gradient structure underlying PPO, we consider the policy-gradient term obtained by differentiating the loss in (6.2) with respect to θ:

$$g(\theta) \triangleq \mathbb{E}_{(s_t,a_t)\sim\pi_{\theta_{\text{old}}}}\Big[w_t(\theta)\hat{A}_t\nabla_\theta \log \pi_\theta(a_t \mid s_t)\Big], \tag{6.7}$$

where $w_t(\theta)$ denotes the effective PPO importance weight induced by clipping,

$$w_t(\theta) = \begin{cases} r_t(\theta), & \text{if unclipped,} \\ 0, & \text{if clipped,} \end{cases} \tag{6.8}$$

$r_t(\theta)$ is the importance ratio, and $\hat{A}_t$ is an advantage estimator. In practice, (6.7) is estimated from finite on-policy rollouts, so its optimization behavior is strongly influenced by the variance of the random vector $w_t(\theta)\hat{A}_t\nabla_\theta \log \pi_\theta(a_t \mid s_t)$.

A vanilla policy-gradient toy setting.

To isolate the intrinsic stochasticity of the score function, consider a simplified one-step on-policy setting with a fixed state s and the vanilla REINFORCE estimator (i.e., $w_t(\theta) \equiv 1$). Assume further that the return is a constant $R_t \equiv R$, independent of the sampled action [162]. The stochastic gradient estimator becomes

$$\hat{g}(\theta) = R \nabla_\theta \log \pi_\theta(a \mid s), \quad a \sim \pi_\theta(\cdot \mid s). \tag{6.9}$$

Although the true gradient is zero in this setting (since R does not depend on a), the estimator (6.9) generally has nonzero variance; this is precisely the variance that baselines and advantages are designed to reduce in practice.

Proposition 6.1 (Gradient variance for Gaussian vs. categorical policies)**.** *Fix a state s and consider the one-step REINFORCE estimator $\hat{g}(\theta) = R \nabla_\theta \log \pi_\theta(a \mid s)$ with $a \sim \pi_\theta(\cdot \mid s)$ and constant return R. The conditional covariances for the two policy families are as follows.*

[1] ***Gaussian (gradient w.r.t. mean).*** *Let $\pi^c(a \mid s) = \mathcal{N}(\mu, \Sigma)$ with diagonal $\Sigma = \mathrm{Diag}(\sigma^2)$ and treat $\mu \in \mathbb{R}^m$ as the parameter (with Σ fixed). Then*

$$\mathbb{E}\big[\|\hat{g}_\mu\|_2^2 \mid s\big] = R^2 \,\mathrm{Tr}(\Sigma^{-1}) = \sum_{i=1}^{m} \frac{R^2}{\sigma_i^2}. \tag{6.10}$$

[2] ***Categorical (gradient w.r.t. logits).*** *Let the factorized categorical policy be $\pi^d(a \mid s) = \prod_{i=1}^m \pi^d(a^i_{j_i} \mid s)$, where $p_i(s) \in \Delta^{K-1}$ is the per-dimension softmax probability vector over K bins and $j_i \sim \mathrm{Cat}(p_i(s))$. Treating the per-dimension logits $z_i(s) \in \mathbb{R}^K$ as parameters and stacking $z(s) = [z_1(s); \ldots; z_m(s)] \in \mathbb{R}^{mK}$,*

$$\mathbb{E}\big[\|\hat{g}_z\|_2^2 \mid s\big] = R^2 \sum_{i=1}^{m} \Big(1 - \|p_i(s)\|_2^2\Big) \;\le\; mR^2\Big(1 - \frac{1}{K}\Big). \tag{6.11}$$

Moreover, the inequality is tight if and only if $p_i(s)$ is uniform over the K bins for all i.

Proof. Fix a state s and write $\hat{g}(\theta) = R\nabla_\theta \log \pi_\theta(a \mid s)$ with $a \sim \pi_\theta(\cdot \mid s)$ and constant R. We use the standard identity

$$\mathbb{E}\big[\nabla_\theta \log \pi_\theta(a \mid s) \mid s\big] = \nabla_\theta \int \pi_\theta(a \mid s)\, da = \nabla_\theta 1 = 0, \tag{6.12}$$

so $\mathbb{E}[\hat{g} \mid s] = 0$ and therefore the conditional covariance satisfies $\mathrm{Cov}(\hat{g} \mid s) = \mathbb{E}[\hat{g}\hat{g}^\top \mid s]$ and, in particular, $\mathbb{E}[\|\hat{g}\|_2^2 \mid s] = \mathrm{Tr}(\mathrm{Cov}(\hat{g} \mid s))$.

[1] Gaussian policy (gradient w.r.t. mean). Let $\pi^c(a \mid s) = \mathcal{N}(\mu, \Sigma)$ with fixed diagonal $\Sigma = \mathrm{Diag}(\sigma^2)$ and parameter $\mu \in \mathbb{R}^m$. The log-density is

$$\log \pi^c(a \mid s) = -\tfrac{1}{2}(a-\mu)^\top \Sigma^{-1}(a-\mu) + C,$$

so $\nabla_\mu \log \pi^c(a \mid s) = \Sigma^{-1}(a-\mu)$. Therefore $\hat{g}_\mu = R\Sigma^{-1}(a-\mu)$ and

$$\begin{aligned}
\mathbb{E}\big[\|\hat{g}_\mu\|_2^2 \mid s\big] &= R^2\, \mathbb{E}\big[(a-\mu)^\top \Sigma^{-2}(a-\mu) \mid s\big] \\
&= R^2\, \mathrm{Tr}\Big(\Sigma^{-2}\, \mathbb{E}[(a-\mu)(a-\mu)^\top \mid s]\Big) \\
&= R^2\, \mathrm{Tr}(\Sigma^{-2}\Sigma) = R^2\, \mathrm{Tr}(\Sigma^{-1}).
\end{aligned}$$

For diagonal $\Sigma = \mathrm{Diag}(\sigma^2)$, $\mathrm{Tr}(\Sigma^{-1}) = \sum_{i=1}^m \sigma_i^{-2}$, yielding (6.10).

[2] Categorical policy (gradient w.r.t. logits). Consider one action dimension i with logits $z_i \in \mathbb{R}^K$ and softmax probabilities $p_i = \mathrm{softmax}(z_i) \in \Delta^{K-1}$. Let $j_i \sim \mathrm{Cat}(p_i)$ and denote the sampled one-hot vector by $e_{j_i} \in \mathbb{R}^K$. For the log-probability $\log \pi^d(a^i_{j_i} \mid s) = \log p_{i,j_i}$, the softmax score w.r.t. logits is

$$\nabla_{z_i} \log p_{i,j_i} = e_{j_i} - p_i.$$

Hence the per-dimension REINFORCE gradient is $\hat{g}_{z_i} = R\,(e_{j_i} - p_i)$, and its conditional second moment is

$$\begin{aligned}
\mathbb{E}\big[\|\hat{g}_{z_i}\|_2^2 \mid s\big] &= R^2\,\mathbb{E}\big[\|e_{j_i} - p_i\|_2^2 \mid s\big] \\
&= R^2\,\mathbb{E}\big[\|e_{j_i}\|_2^2 + \|p_i\|_2^2 - 2\,e_{j_i}^\top p_i \mid s\big] \\
&= R^2\left(1 + \|p_i\|_2^2 - 2\,\mathbb{E}[p_{i,j_i} \mid s]\right) \\
&= R^2\left(1 + \|p_i\|_2^2 - 2\sum_{k=1}^{K} p_{i,k}^2\right) = R^2\,(1 - \|p_i\|_2^2).
\end{aligned}$$

Assuming the m action dimensions factorize, $\pi^d(a \mid s) = \prod_{i=1}^m \pi^d(a^i_{j_i} \mid s)$, so the full log-probability is the sum across i and the full gradient w.r.t. $z = [z_1; \ldots; z_m]$ is the concatenation of per-dimension gradients: $\hat{g}_z = [\hat{g}_{z_1}; \ldots; \hat{g}_{z_m}]$. Therefore $\|\hat{g}_z\|_2^2 = \sum_{i=1}^m \|\hat{g}_{z_i}\|_2^2$, and

$$\mathbb{E}\big[\|\hat{g}_z\|_2^2 \mid s\big] = \sum_{i=1}^{m} \mathbb{E}\big[\|\hat{g}_{z_i}\|_2^2 \mid s\big] = R^2 \sum_{i=1}^{m} \big(1 - \|p_i(s)\|_2^2\big), \tag{6.13}$$

which proves the first equality in (6.11).

For the upper bound, for any $p \in \Delta^{K-1}$, Cauchy-Schwarz gives $\|p\|_2^2 \geq \frac{1}{K}(\|p\|_1)^2 = \frac{1}{K}$, hence $1 - \|p\|_2^2 \leq 1 - \frac{1}{K}$. Applying this to each $p_i(s)$ yields

$$\mathbb{E}\big[\|\hat{g}_z\|_2^2 \mid s\big] \leq R^2 \sum_{i=1}^{m} \left(1 - \frac{1}{K}\right) = mR^2\left(1 - \frac{1}{K}\right).$$

Moreover, $\|p\|_2^2 = 1/K$ holds if and only if p is uniform over the K bins, so the inequality is tight if and only if $p_i(s)$ is uniform for all i. □

Insight.

For continuous Gaussian actors, gradient variance can grow sharply as the policy standard deviation decreases, a behavior that commonly occurs in practice as exploration is reduced over the course of training. Even at early training stages where the standard deviation remains relatively large (e.g., $\sigma = 1$), the per-dimension gradient-variance gap between Gaussian and

categorical policies is lower-bounded by $1/K$, so the total gap scales linearly with the action dimension m.

6.4.2 Optimization View: Cross-Entropy vs. Squared Error

Gaussian actor: (weighted) squared error.

For a diagonal Gaussian with (state-dependent or state-independent) variance, the negative log-likelihood is

$$-\log \pi_\theta^c(a \mid s) = \tfrac{1}{2}(a-\mu_\theta(s))^\top \Sigma_\theta(s)^{-1}(a-\mu_\theta(s)) + \tfrac{1}{2}\log|\Sigma_\theta(s)| + \text{const.} \tag{6.14}$$

If $\Sigma_\theta(s) = \sigma^2 I$ is fixed, minimizing (6.14) with respect to $\mu_\theta(s)$ is equivalent to minimizing a mean squared error:

$$-\log \pi_\theta^c(a \mid s) \;\propto\; \|a-\mu_\theta(s)\|_2^2. \tag{6.15}$$

Combined with (6.7), the Gaussian actor update admits an interpretation as a weighted mean squared error regression, with weights determined by the advantage and the importance ratio.

Categorical actor: (weighted) cross-entropy.

For a discretized categorical policy, a sampled action corresponds to a bin index j_i per dimension. The negative log-likelihood for each dimension is

$$-\log \pi_\theta^d(a^i \mid s) = -\log p_i(j_i \mid s), \tag{6.16}$$

which is exactly the cross-entropy between a one-hot target e_{j_i} and the predicted distribution $p_i(s)$. Under (6.7), the objective reduces to a weighted cross-entropy loss, where the weights are given by the advantage-modulated importance ratio (with clipping). Accordingly, the categorical actor update corresponds to minimizing this weighted cross-entropy over the sampled action bins.

Insight.

On-policy RL training differs across policy families primarily through the likelihood model that defines $\log \pi_\theta(a \mid s)$: categorical policies induce (weighted) cross-entropy objectives over bins, while Gaussian policies induce (weighted) quadratic objectives over continuous actions. Although prior work provides strong empirical evidence that cross-entropy can replace the mean squared error regression loss for value-function learning in deep RL [35, 181], the consequences of analogous loss choices for actor-network learning remain largely unexplored. Motivated by viewing the policy objective through this loss-based lens (weighted MSE for Gaussian actors vs. weighted cross-entropy for categorical actors), we hypothesize that the *classification*-style structure of the categorical update can better exploit neural-network capacity. This perspective motivates the use of discrete categorical actors as a scalable design choice for training large RL policies.

6.5 Actor Network Architecture

We adopt a unified actor network architecture for discrete categorical policies that emphasizes optimization stability and scalability across high-dimensional action spaces. The architecture is composed of a feature-extraction stage, a stack of residual feedforward blocks, and a final output projection to categorical action logits.

Feature extraction.

The actor first maps raw observations to a latent feature representation using a task-dependent encoder. For low-dimensional state inputs, we employ a multi-layer perceptron (MLP) encoder composed of linear layers and nonlinear activations. For high-dimensional observations (e.g., images), a convolutional neural network (CNN) encoder is used instead. The encoder produces a shared latent representation that is consumed by subsequent feedforward blocks.

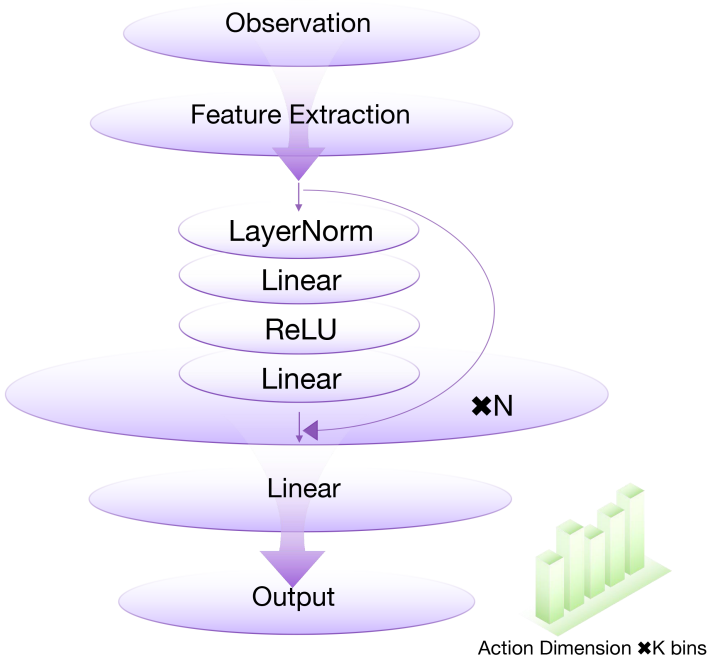


Figure 6.3. Proposed **R**egularized **N**etwork for **D**iscrete action policies (RN-D). The actor consists of a feature extractor (MLP or CNN) followed by pre-LayerNorm residual MLP blocks, enabling stable optimization and improved scalability. The output layer produces categorical logits over K bins for each action dimension.

Residual feedforward blocks.

Following feature extraction, the actor applies a stack of N residual feedforward blocks. Each block is equipped with a residual connection,

$$\mathbf{h}_{\ell+1} = \mathbf{h}_\ell + \mathrm{FFN}(\mathrm{LN}(\mathbf{h}_\ell)),$$

where $\mathrm{LN}(\cdot)$ denotes layer normalization. We employ pre-layer normalization by applying LayerNorm before each feedforward block, a design that improves optimization stability in deep architectures, including Transformer models [183]. Following [184], the feedforward network adopts an inverted-bottleneck structure: the hidden dimension is first expanded from d_h to $4d_h$ and passed through a ReLU activation, and a second linear layer projects the representation back to d_h.

Output layer.

The final hidden representation is passed through a linear projection that outputs logits for a discrete categorical distribution. Specifically, for an action space with d dimensions and K bins per dimension, the output layer produces $d \times K$ logits, which parameterize independent categorical distributions over each action dimension.

Relation to prior architectures.

The proposed actor architecture shares structural similarities with several existing designs. Residual MLPs with normalization have been explored in prior RL works such as SimBa [174, 181] and BroNet [185], as well as in the feedforward sublayers of Transformer models [184]. The contribution of this chapter does not lie in introducing a novel network module, but rather in demonstrating that this architectural choice is particularly well-suited for discrete categorical actor policies. As shown in the experiments, combining this network with categorical action parameterization leads to improved optimization behavior compared to conventional shallow MLP actors commonly used in on-policy RL.

6.6 Experiments

We evaluate the proposed discrete policy with regularized network (RN-D) across a diverse set of locomotion and manipulation benchmarks. The main experiments use PPO, and we further test the same actor design under TRPO and SPO to assess whether the improvement is algorithm-specific.

Experimental setup.

We evaluate the method on three environment families: Gym locomotion [186], ManiSkill [187] with state-based observations, and ManiSkill with RGB-based observations. These tasks cover a range of control challenges, including locomotion and dexterous manipulation, varying embodiments, and different observation modalities. Specifically, we consider 5 standard Gym MuJoCo locomotion tasks (`HalfCheetah-v4`, `Ant-v4`, `Hopper-v4`, `Humanoid-v4`, and

`Walker2d-v4`), 10 ManiSkill manipulation tasks with low-dimensional state observations, and 5 ManiSkill tasks with image-based observations, where policies operate on raw RGB inputs via a convolutional encoder. The implementation is based on the CleanRL PPO codebase [178] and the ManiSkill benchmark suite [187], and unless otherwise specified we adopt the default hyperparameters from these codebases without task-specific retuning. Gymnasium/MuJoCo experiments are run on an NVIDIA H800, while ManiSkill experiments are run on 2× NVIDIA GeForce RTX 5090 GPUs; all results are averaged over 5 random seeds.

Baselines.

The main experiments are conducted with PPO [14], using implementations adapted from the CleanRL [178] and ManiSkill codebases [187]. We compare **RN-D**, our regularized-network discrete (categorical) actor, against three baselines:

- **RN-C**: a continuous policy parameterized by a factorized Gaussian with the same regularized network.
- **MLP-C**: a continuous factorized-Gaussian policy with a standard MLP actor (the widely adopted PPO actor in standard implementations).
- **MLP-D**: a discrete categorical policy with a standard MLP actor.

Across all four settings, the critic (value network) architecture is kept identical to ensure a fair comparison. For discrete actors, we use $K = 41$ bins throughout. To evaluate algorithmic robustness, we also compare RN-D with the standard continuous MLP actor (MLP-C) under PPO, TRPO [153], and SPO [154] on Gym locomotion and ManiSkill manipulation benchmarks.

Metrics.

To aggregate performance across benchmarks, we use task-standard metrics and normalize returns when needed. For Gym MuJoCo locomotion, we report TD3-normalized return [188], defined as

$$\text{TD3-Normalized}(x) := \frac{x - x_{\text{random}}}{x_{\text{TD3}} - x_{\text{random}}}, \tag{6.17}$$

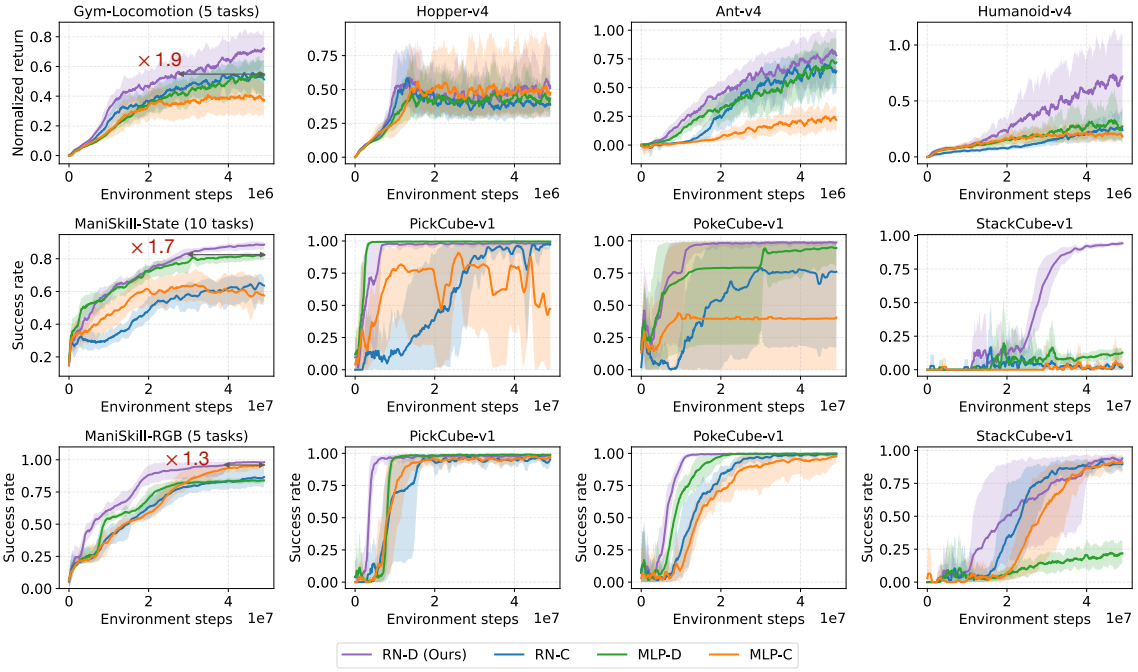


Figure 6.4. Aggregate learning curves across benchmarks. Each subplot reports the mean performance over tasks within a benchmark (MuJoCo locomotion: normalized return; ManiSkill: success rate) as a function of environment steps. Curves are averaged over 5 random seeds; shaded regions denote 95% stratified bootstrap confidence intervals. The red annotations indicate a sample-efficiency speedup.

where x is the raw episodic return and $(x_{\text{random}}, x_{\text{TD3}})$ are the random-policy and TD3 reference scores of each task. For ManiSkill benchmarks, we report success rate, i.e., the fraction of evaluation episodes that achieve the task goal.

Overall performance.

Figure 6.4 shows that **RN-D (ours)** consistently achieves the best performance across both locomotion and ManiSkill. On Gym locomotion, RN-D attains higher normalized returns than the Gaussian baselines, with particularly clear improvements on harder tasks such as Ant and Humanoid. On ManiSkill, RN-D yields the highest average success on the 10 state-based tasks and is the only variant that reliably solves the challenging StackCube task, where other methods remain near zero success for most of training. The same trend holds for vision-based

ManiSkill tasks: RN-D maintains its advantage when paired with CNN encoders and reaches near-saturated success more reliably than the baselines.

Beyond final performance, RN-D reaches strong performance with fewer environment steps. The red annotations in Figure 6.4 indicate a sample-efficiency speedup: RN-D matches the best baseline's performance using roughly $1.3\times$–$1.9\times$ fewer interaction steps. Comparing ablations reveals complementary effects. For state-based settings (locomotion and ManiSkill-State), RN-C versus MLP-C (blue vs. orange) shows that the regularized backbone improves performance even with Gaussian actors, while MLP-D versus MLP-C (green vs. orange) indicates that discretization alone can also be beneficial. For vision-based tasks, the CNN encoder provides a strong representation and narrows the gap among actor variants; in some cases MLP-C is already competitive. Nevertheless, RN-D still achieves the best final performance and consistently improves sample efficiency.

The multi-algorithm results in Figure 6.2 exhibit the same qualitative pattern. Under PPO, TRPO, and SPO, RN-D achieves faster convergence and higher final performance than the standard continuous MLP actor, indicating that the benefit of discretized categorical actors with regularized networks is not specific to PPO's clipped surrogate.

Gradient variance.

We plot the policy-gradient variance (log scale) on five MuJoCo tasks as a representative illustration in Figure 6.5. Across training, the Gaussian policy (RN-C/MLP-C) exhibits consistently larger variance, often by orders of magnitude, than its discrete categorical counterpart (RN-D/MLP-D). This disparity is most pronounced for the standard MLP actor: MLP-C (orange) stays far above MLP-D (green) and increases steadily with environment steps, consistent with the Gaussian actor's standard deviation shrinking over training and amplifying gradient variability. Replacing the MLP with a regularized network reduces variance for both policy classes. Overall, the discrete categorical policy with a regularized network (RN-D, purple) achieves the lowest policy-gradient variance throughout training.

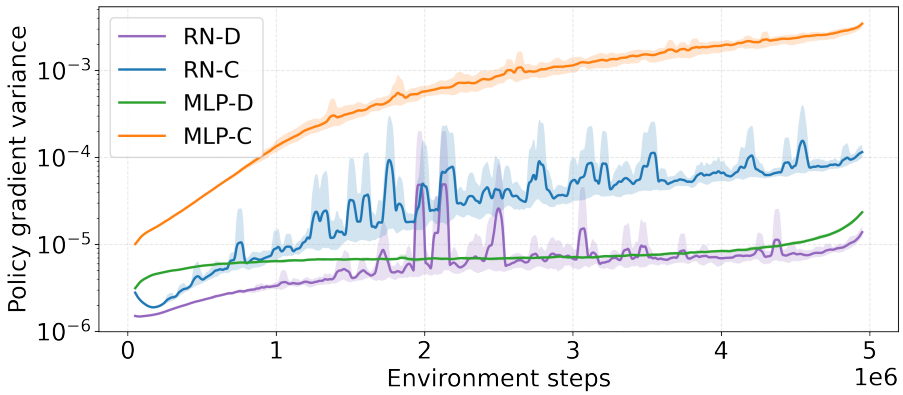


Figure 6.5. Evolution of the policy-gradient variance over training (log scale).

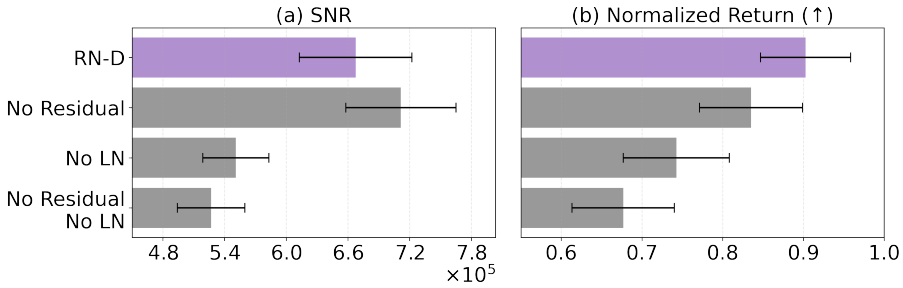


Figure 6.6. Component analysis. (a) Gradient signal-to-noise ratio (SNR) on Gym locomotion tasks. Bars show the mean and error bars denote one standard deviation across tasks. (b) Average normalized return. Higher SNR correlates with higher returns, with RN-D achieving the best performance.

Component analysis.

We visualize the gradient signal-to-noise ratio (SNR) and normalized return on Gym locomotion tasks, where SNR is computed as the squared norm of the mean policy gradient divided by its variance across mini-batches during training. Following [16], we report the 95th-percentile performance over training and summarize gradient signal quality using the mean SNR. Figure 6.6(a) shows that residual connections and layer normalization each improve SNR relative to a plain MLP actor. Figure 6.6(b) shows a consistent trend in performance, suggesting a strong correlation between gradient signal quality (SNR) and normalized return across architectures.

6.7 Extended Studies

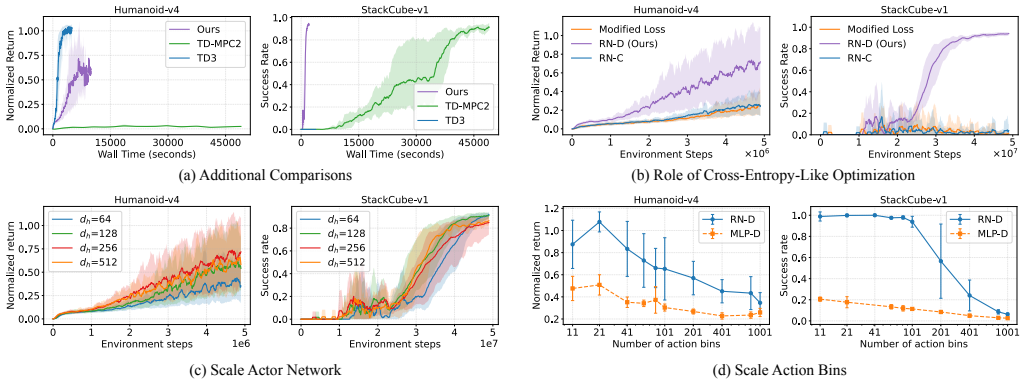


Figure 6.7. Extended studies on sample efficiency, optimization objective, and scaling.

6.7.1 Additional Comparisons with State-of-the-Art Methods

Figure 6.7(a) compares RN-D with TD3 [188] and TD-MPC2 [179] on Humanoid-v4 and StackCube-v1 using wall-clock time. RN-D remains competitive on Humanoid-v4 and is substantially more effective on StackCube-v1, where it reaches high success while TD3 fails to learn and TD-MPC2 improves only later in training. These results suggest that the proposed on-policy discretized actor is especially useful in manipulation settings where exploration and policy optimization are difficult.

6.7.2 Role of Cross-Entropy-Like Optimization

To isolate the role of the optimization objective, we introduce a *loss-swap* ablation that keeps the discretized action representation but replaces the categorical likelihood with a Gaussian likelihood. The actor still outputs logits over K bins, but the action mean is computed as

$$\mu_{\theta,i}(s) = \sum_{k=1}^{K} p_{\theta,i}(k \mid s)\, c_k, \tag{6.18}$$

where $p_{\theta,i}(k \mid s) = \mathrm{softmax}(z_{\theta,i}(s))_k$ and c_k is the kth bin center. We then use

$$\pi_\theta^{\mathrm{swap}}(a \mid s) = \mathcal{N}\big(a; \mu_\theta(s), \Sigma\big). \tag{6.19}$$

Figure 6.7(b) shows that replacing the categorical likelihood with this Gaussian likelihood causes a substantial performance drop, with the loss-swapped variant closely matching the continuous Gaussian baseline. Thus, discretization alone is not sufficient; the cross-entropy-like categorical objective is central to the observed gains.

6.7.3 Scaling the Actor Network

We next scale the actor width $d_h \in \{64, 128, 256, 512\}$ while keeping the PPO objective, critic, and other hyperparameters fixed. Figure 6.7(c) shows that wider actors generally learn faster on Humanoid-v4 and StackCube-v1, although the final return can be non-monotonic. Actor capacity is therefore an effective lever for accelerating on-policy learning, but larger networks do not always improve asymptotic performance.

6.7.4 Scaling the Discrete Bins

Finally, we vary the number of action bins from 21 to 1001, with 41 bins used in the main experiments. Figure 6.7(d) shows that discrete policies perform best with a small to moderate number of bins, while very fine discretization can degrade performance. This suggests a tradeoff: coarse discretization preserves the optimization advantages of categorical policies, whereas too many bins make the actor predict overly fine-grained action distributions. More efficient action tokenization is a promising way to retain precision without losing the optimization benefits of discreteness.

6.8 Chapter Summary

This chapter revisited actor design for on-policy reinforcement learning in continuous control and showed that policy parameterization and architecture provide a strong, underexplored lever for improving on-policy training. The proposed RN-D approach replaces the standard diagonal-Gaussian actor with a discretized categorical policy over action bins and pairs it with a regularized residual actor network, while keeping the critic and the underlying on-policy

objective fixed. Across locomotion benchmarks and ManiSkill (state- and vision-based) tasks, this simple actor-side change consistently improves final performance, accelerates convergence, and yields more stable learning. Results under PPO, TRPO, and SPO further show that the improvement is not tied to one particular on-policy algorithm. The loss-swap ablation shows that the gains are not merely due to discretizing the action space but arise from the cross-entropy-like optimization objective induced by the categorical likelihood.

Placed alongside the preceding chapters, this chapter completes the thesis's broader program of using *structured models for sample-efficient learning and control*. Chapters 2–4 installed structure *inside* the model itself—embedding optimization, physics-based PDEs, and neural operators directly into the function class—to improve data efficiency and generalization for energy-system applications. Chapter 5 pushed that structure into the policy representation and trained it end-to-end via reinforcement learning. This chapter adopts a complementary posture: rather than enriching the internal structure of the policy or the model, it holds the RL algorithm fixed and asks what choice of *policy representation and network architecture* best matches the optimization geometry of on-policy RL. The answer, consistently across benchmarks and modalities, is that a discretized categorical actor combined with a regularized residual network delivers lower gradient variance, higher gradient signal-to-noise ratio, and stronger downstream performance than the Gaussian/MLP defaults that dominate the literature. From the thesis perspective, this result reinforces a unifying theme: imposing the right structural prior—whether in the model, the policy, or the actor parameterization—is a principled and broadly effective strategy for improving sample efficiency and learning stability.

The findings carry particular relevance for energy systems and other high-stakes control settings, where data collection is expensive, training must be reliable, and policies may need to be deployed under stringent operational constraints. In such domains, reducing gradient variance and accelerating convergence through better policy structure directly translates into lower data requirements and more predictable training behavior, complementing the model-level structural choices studied in the earlier chapters.

Several directions appear promising. While this chapter isolates actor-side effects by holding the critic fixed, understanding actor-critic interactions remains an important direction: jointly scaling or regularizing the critic and actor may reveal more symmetric design principles for on-policy optimization. Extending RN-D to energy control tasks—where action spaces are often bounded, piecewise, or low-dimensional but optimization is challenging due to long horizons and sparse reward signals—is a natural next step that directly connects this chapter's contributions to the energy-systems setting motivating the thesis. In addition, we envision applying the approach to more complex embodied settings, including long-horizon manipulation, contact-rich control, and multi-object tasks, to better characterize its benefits and limitations in challenging real-world regimes.

Acknowledgments

The authors thank Haiwen Yu for contributions to the framework visualization design.

Chapter 6, in full, contains material from Y. Bian*, J. Feng*, T. Wang, Y. Li, S. Gao, and Y. Shi, "RN-D: Discretized Categorical Actors with Regularized Networks for On-Policy Reinforcement Learning," *International Conference on Machine Learning*, 2026 (*equal contribution). The dissertation author was a co-primary investigator and co-author of this paper.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This dissertation studied how structured models can improve sample-efficient learning and control in energy systems by embedding optimization, physics, and policy structure directly into trainable decision-making pipelines. Across the thesis, the central argument has been that energy applications are neither well served by purely black-box learning nor by purely hand-crafted models alone. Instead, the most effective approaches are those that preserve the right problem structure inside the learning architecture itself, so that data-driven adaptation and model-based rigor reinforce rather than oppose one another.

The main contributions of the thesis can be summarized in three parts:

- **Optimization-guided models for strategic behavior learning and control.** Chapters 2 and 5 show how optimization structure can be embedded directly into learning: first for forecasting strategic demand response behavior from observed price-responsive actions, and then for reinforcement learning with policies defined implicitly by parametric optimal control problems.

- **Dynamics-guided models for high-fidelity and scalable building control.** Chapters 3 and 4 develop structured models for healthy and energy-efficient building operation, first through PDE-constrained learning and control based on airflow, temperature, and

contaminant dynamics, and then through neural operator surrogates that retain the relevant spatial structure while making real-time optimization practical.

- **Structured policy representations for stable reinforcement learning.** Chapter 6 shows that even when the outer learning algorithm is fixed, the structure of the policy class itself remains a powerful design lever, and that carefully chosen actor parameterizations can improve gradient variance, training stability, and downstream control performance.

Taken together, these contributions support a unifying thesis-level perspective: structured learning is most valuable when the chosen structure matches the semantics of the decision problem. In the settings studied here, this meant preserving latent optimization models for strategic agents, preserving governing dynamics for cyber-physical control, or preserving favorable optimization geometry through policy representation. The resulting methods improve not only prediction and control performance, but also sample efficiency, interpretability, and operational reliability.

7.2 Lessons and Reflections

One central lesson from the experimental results is that structure helps most when it captures the dominant mechanism that actually generates the data or governs the control problem. In Chapter 2, optimization-guided forecasting performs well because price-responsive behavior is not arbitrary: it is induced by hidden objectives and operational constraints. In Chapters 3 and 4, dynamics-guided control improves performance because indoor air quality and thermal behavior are fundamentally spatiotemporal phenomena that cannot be fully understood through low-dimensional averages alone. In Chapter 5, optimization-defined policies provide a useful inductive bias because many control problems are naturally expressed through constrained objectives. In Chapter 6, the same principle appears at a different level: even when the environment dynamics are unchanged, the structure of the policy representation can strongly shape optimization behavior.

A second lesson is that the benefits of structure depend critically on the quality and scope of the modeling assumptions. When the assumed structure is well aligned with the problem, it can reduce sample complexity dramatically and improve generalization beyond the training distribution. At the same time, stronger modeling assumptions also create clearer failure modes. Exact PDE-based control offers strong physical fidelity, but it incurs a large computational burden. Neural operator surrogates improve tractability, but they inherit limitations from the data used to train them and may degrade under distribution shift. Optimization-based forecasting can recover meaningful latent parameters under favorable assumptions, yet partial observability may make full identification impossible even when accurate forecasting remains achievable. These results suggest that structure should not be viewed as universally beneficial in the abstract; its value depends on how well it matches the governing geometry of the task.

A third lesson concerns the trade-offs among accuracy, efficiency, robustness, and deployability. Richer structure often improves data efficiency and interpretability, but it can increase modeling effort and computational cost. More approximate models or surrogates improve speed and scalability, but they may weaken theoretical guarantees or introduce new sources of mismatch. The practical question is therefore not whether one should always use the most detailed possible model, but what level of structure is sufficient to preserve the aspects of the problem that matter for decision-making. Across the thesis, the most successful methods were those that retained the essential semantics of the system while remaining simple enough to train, optimize, and deploy.

A more personal reflection emerging from this dissertation is that research progress is fundamentally iterative. At the completion of a project, there is often a brief sense of arrival: a method works, the theoretical picture becomes clearer, or the empirical results move beyond the current baselines. Yet that moment is never the end of the story. With time and distance, the same work begins to look different. Its enabling assumptions become more visible, alternative formulations begin to seem plausible, and limitations that once appeared secondary start to define the next set of questions.

This pattern recurred repeatedly in my own work. In DiffOP, I was initially excited by the idea that reinforcement learning could be used to tune an MPC policy whose structure was itself defined by an optimization problem. It seemed to offer a compelling way to combine model-based control with data-driven adaptation. As the work progressed toward higher-dimensional control tasks, however, the limitations became more apparent: obtaining a sufficiently good initial policy was already difficult, and reliable training became harder still. A similar experience arose in the work on residual regularized categorical actors for on-policy reinforcement learning. The method showed surprisingly strong improvements even without hyperparameter tuning, which made the early results especially encouraging. But after that initial excitement, a different question became more important: does the current formulation truly capture the structure of continuous action? Human reasoning may often proceed through discrete abstractions, but many physical actions, such as reaching or grasping, are inherently continuous. This suggests that an important next step is not merely to choose between discrete and continuous structure, but to understand how the two can be combined more faithfully.

In this sense, the value of research lies not only in building a method that works, but also in learning how to question it, reinterpret it, and move beyond it. The cycle of proposing, validating, revisiting, and rethinking has been one of the most important lessons of this thesis, and it is also part of what makes research intellectually durable.

7.3 Future Work

Several research directions follow naturally from this dissertation, as summarized in Figure 7.1. For optimization-guided models, an important next step is to broaden the class of latent decision models that can be learned from data. This includes richer strategic interactions among multiple agents, stochastic and dynamic market settings, online adaptation under nonstationary behavior, and partial-observation settings in which only aggregate actions or delayed outcomes are available. Another promising direction is to strengthen the theory for optimization-guided

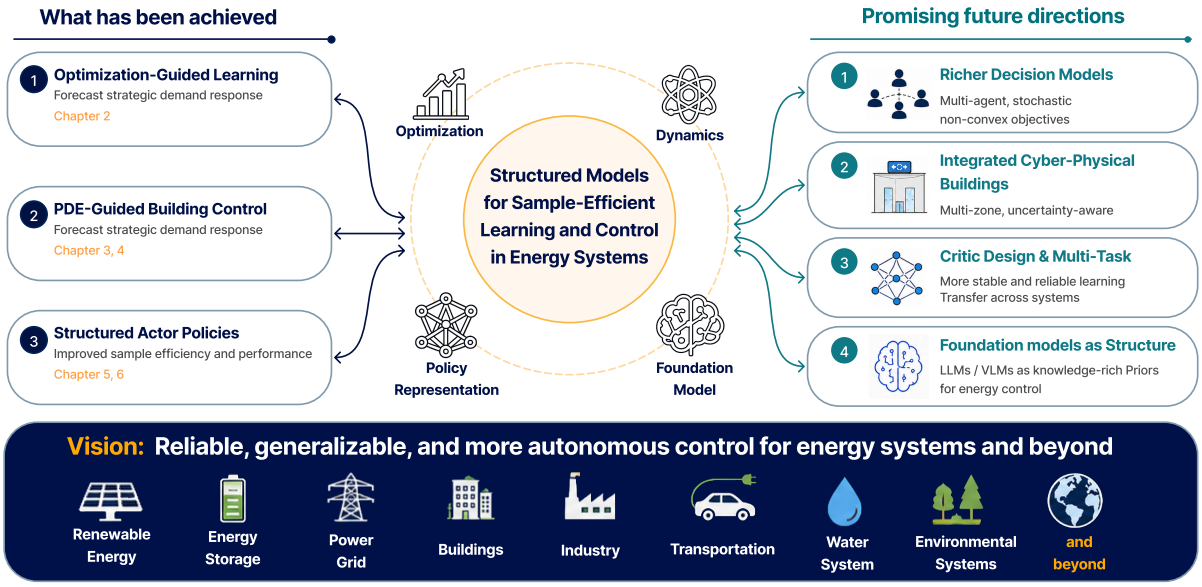


Figure 7.1. Prospective research directions extending the contributions of this dissertation, spanning richer agent modeling, integrated cyber-physical building control, structured reinforcement learning, and foundation-model-grounded decision-making.

learning beyond strongly convex or smooth settings, especially when active sets change, objectives are weakly convex, or the learned optimization layer is embedded inside larger closed-loop systems.

For dynamics-guided models, future work should move toward more realistic and integrated cyber-physical building control. This includes three-dimensional multi-zone environments, tighter coupling between airflow, temperature, humidity, and occupancy uncertainty, and the use of adaptive sensing and estimation together with control. For operator learning, a major opportunity is to develop surrogates that are not only fast and accurate, but also uncertainty-aware, robust to distribution shift, and reliable when embedded in receding-horizon optimization. More broadly, the interaction between physical simulators, operator surrogates, and control design remains a rich area for both methodological and application-driven advances.

For reinforcement learning and optimization-based policy learning, an important future direction is to deepen the study of structured policy and value-function representations. This dissertation has shown that incorporating optimization structure and improved policy represen-

tations can substantially improve sample efficiency and control performance. However, many open questions remain in how to design the critic, how to regularize value learning, and how to improve the interaction between actor and critic during training. In actor-critic reinforcement learning, the critic not only evaluates the current policy but also shapes the policy improvement direction. Poorly designed critics can introduce bias, instability, or overfitting to bootstrapped targets, while better critic architectures and regularization mechanisms may provide more reliable learning signals. Future work could therefore investigate structure-aware critic design, low-rank or modular value-function parameterizations, uncertainty-aware critics, and principled actor-critic coupling mechanisms that improve stability, robustness, and performance in continuous control. Another important direction is to move beyond single-task control toward multi-task and generalizable control. Most methods studied in this dissertation focus on learning one controller for one task or one system configuration. In real energy systems, however, controllers must often operate across different buildings, operating conditions, occupant behaviors, weather patterns, market environments, and device constraints. This motivates the development of structured reinforcement learning methods that can transfer across tasks while preserving domain-specific inductive biases. Future work could explore multi-task policy training, task-conditioned value functions, modular control architectures, and representation learning methods that separate general control knowledge from system-specific parameters. Such methods may help bridge the gap between high-performing controllers in simulation and broadly deployable controllers for real-world energy systems.

Finally, a particularly exciting direction is to leverage foundation models as another form of structure for sample-efficient learning and control in energy systems. The structured models developed in this dissertation incorporate explicit prior knowledge through optimization layers, physical dynamics, and control-theoretic representations. Foundation models, including large language models and vision-language models, offer a complementary source of structure: they are pretrained on massive amounts of text, code, visual data, and domain knowledge, and can therefore distill useful information about energy systems, physical constraints, operational

objectives, and control workflows. When appropriately grounded, these models may reduce the amount of task-specific data and expert effort required to design learning-based controllers. For example, language and vision-language models could help interpret building layouts, equipment descriptions, sensor streams, weather forecasts, or market signals; translate high-level operational goals into optimization or reinforcement learning objectives; suggest model structures and constraints; diagnose failure modes; and coordinate simulation, learning, and deployment pipelines. In this sense, foundation models can serve not only as automation tools, but also as knowledge-rich priors that complement optimization, physics, and reinforcement learning structures. Carefully integrating them with verifiable optimization, physical simulators, and closed-loop control algorithms could open a path toward more autonomous, generalizable, and sample-efficient control for energy systems and beyond.

Overall, these directions point toward a broader vision of **structured learning and control for energy systems and beyond**. The central message of this dissertation is that carefully designed structure—whether from optimization, physics, policy and value-function representations, or pretrained foundation models—can reduce the effective complexity of learning and make data-driven control more sample-efficient, interpretable, and reliable. Future progress will require moving from individual structured components toward integrated learning-and-control systems that can reason about strategic behavior, model complex physical dynamics, optimize policies in closed loop, generalize across tasks, and adapt to new environments with limited data. By combining domain knowledge with scalable learning architectures, structured models provide a promising foundation for building intelligent decision-making systems that are not only effective in simulation, but also robust, transferable, and deployable in real-world energy systems.

Bibliography

[1] M. Jin, S. Liu, S. Schiavon, and C. Spanos. Sensing by proxy: occupancy detection based on indoor CO2 concentration. *arXiv preprint arXiv:1511.05274*, 2015.

[2] W. Jin, Z. Wang, Z. Yang, and S. Mou. Pontryagin differentiable programming: An end-to-end learning and control framework. *Advances in Neural Information Processing Systems*, 33:7979–7992, 2020.

[3] K.J. Chua, S.K. Chou, W.M. Yang, and J. Yan. Achieving better energy-efficient air conditioning–a review of technologies and strategies. *Applied Energy*, 104:87–104, 2013.

[4] W.W. Che, C.Y. Tso, L. Sun, D.Y.K. Ip, H. Lee, C.Y.H. Chao, and A.K.H. Lau. Energy consumption, indoor thermal comfort and air quality in a commercial office with retrofitted heat, ventilation and air conditioning (hvac) system. *Energy and Buildings*, 201:202–215, 2019.

[5] U.S. Energy Information Administration. Form eia-860 detailed data with previous form data (eia-860a/860b). Technical report, U.S. Energy Information Administration, 2021.

[6] Federal Energy Regulatory Commission. Electric storage participation in markets operated by regional transmission organizations and independent system operators. *Order No. 841*, 2018.

[7] E. Cartwright. FERC order no. 2222: Participation of distributed energy resource aggregations in markets operated by regional transmission organizations and independent system operators. *Federal Energy Regulatory Commission*, 2020.

[8] M. Morari and J.H. Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(4-5):667–682, 1999.

[9] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control: Theory and Algorithms*. Springer, 2017.

[10] W. Tian, X. Han, W. Zuo, and M.D. Sohn. Building energy simulation coupled with CFD for indoor environment: A critical review and recent applications. *Energy and Buildings*, 165:184–199, 2018.

[11] N. Lambert, B. Amos, O. Yadan, and R. Calandra. Objective mismatch in model-based reinforcement learning. *arXiv preprint arXiv:2002.04523*, 2020.

[12] C. Michoski, M. Milosavljević, T. Oliver, and D.R. Hatch. Solving differential equations using deep neural networks. *Neurocomputing*, 399:193–212, 2020.

[13] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

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

[15] Z. Zhang, A. Chong, Y. Pan, C. Zhang, and K.P. Lam. A deep reinforcement learning approach to using whole building energy model for hvac optimal control. *Building Simulation*, 12:1–14, 2018.

[16] M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, et al. What matters in on-policy reinforcement learning? a large-scale empirical study. *arXiv preprint arXiv:2006.05990*, 2020.

[17] A. Kovacs. Inverse optimization approach for identifying demand response models. *Energy*, 216:119257, 2021.

[18] E.T. Maddalena, Y. Lian, and C.N. Jones. Data-driven methods for building control—a review and promising future directions. *Control Engineering Practice*, 95:104211, 2020.

[19] B. Amos, I. Jimenez, J. Sacks, B. Boots, and J.Z. Kolter. Differentiable MPC for end-to-end planning and control. In *Advances in Neural Information Processing Systems*, 2018.

[20] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

[21] L. Lu, P. Jin, G. Pang, Z. Zhang, and G.E. Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

[22] California Independent System Operator. Energy storage enhancements. CAISO Proposal, 2022.

[23] Z. Liang et al. Data-driven marginal offer price recovery in wholesale electricity markets. *IEEE Transactions on Power Systems*, 2023.

[24] Y. Bian, N. Zheng, Y. Zheng, B. Xu, and Y. Shi. Demand response model identification and behavior forecast with OptNet: a gradient-based approach. In *Proceedings of the Thirteenth ACM International Conference on Future Energy Systems (e-Energy '22)*, 2022.

[25] Y. Bian, N. Zheng, Y. Zheng, B. Xu, and Y. Shi. Predicting strategic energy storage behaviors via a gradient-based approach. *arXiv preprint*, 2024.

[26] G. Li et al. Tube-based robust MPC for building HVAC systems with indoor air quality constraints. *Energy and Buildings*, 265:112089, 2022.

[27] F. Shang et al. Developing smart air purifier control strategies for better IAQ and energy efficiency using reinforcement learning. *Building and Environment*, 242:110556, 2023.

[28] Y. Bian and Y. Shi. Ventilation and indoor air quality control through pde-constrained optimization. *arXiv preprint*, 2024.

[29] Z. Hao, Z. Wang, H. Su, C. Ying, Y. Dong, S. Liu, Z. Cheng, J. Song, and J. Zhu. GNOT: A general neural operator transformer for operator learning. In *International Conference on Machine Learning*, 2023.

[30] Y. Bian et al. Operator learning for energy-efficient building ventilation control with computational fluid dynamics. *arXiv preprint*, 2025.

[31] Y. Bian, J. Feng, and Y. Shi. DiffOP: Reinforcement learning of optimization-based control policies via implicit policy gradients. *arXiv preprint arXiv:2411.07484*, 2025.

[32] M. Xu, T.L. Molloy, and S. Gould. Revisiting implicit differentiation for learning problems in optimal control. In *Advances in Neural Information Processing Systems*, 2024.

[33] R.S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, 1999.

[34] M.G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, 2017.

[35] J. Farebrother, J. Orbay, Q. Vuong, A.A. Taıga, Y. Chebotar, T. Xiao, A. Irpan, S. Levine, P.S. Castro, A. Faust, et al. Stop regressing: Training value functions via classification for scalable deep RL. *arXiv preprint arXiv:2403.03950*, 2024.

[36] D. Guo et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint*, 2025.

[37] Y. Tang and S. Agrawal. Discretizing continuous action space for on-policy optimization. In *AAAI Conference on Artificial Intelligence*, 2020.

[38] T. Seyde, I. Gilitschenski, W. Schwarting, B. Stellato, M. Riedmiller, M. Wulfmeier, and D. Rus. Is bang-bang control all you need? solving continuous control with Bernoulli policies. In *Advances in Neural Information Processing Systems*, 2021.

[39] Hamed Mohsenian-Rad. Optimal bidding, scheduling, and deployment of battery systems in california day-ahead energy market. *IEEE Transactions on Power Systems*, 31(1):442–453, 2015.

[40] Shubhrajit Bhattacharjee, Ramteen Sioshansi, and Hamidreza Zareipour. Energy storage participation in wholesale markets: The impact of state-of-energy management. *IEEE Open Access Journal of Power and Energy*, 2022.

[41] Mostafa Rezaeimozafar, Rory FD Monaghan, Enda Barrett, and Maeve Duffy. A review of behind-the-meter energy storage systems in smart grids. *Renewable and Sustainable Energy Reviews*, 164:112573, 2022.

[42] International Renewable Energy Agency. Behind-the-meter batteries, 2019.

[43] Neil McIlwaine, Aoife M Foley, D John Morrow, Dlzar Al Kez, Chongyu Zhang, Xi Lu, and Robert J Best. A state-of-the-art techno-economic review of distributed and embedded energy storage for energy systems. *Energy*, 229:120461, 2021.

[44] CAISO. Energy storage enhancements market power mitigation, February 2022.

[45] Christoph Graf, Emilio La Pera, Federico Quaglia, and Frank A Wolak. Market power mitigation mechanisms for wholesale electricity markets: Status quo and challenges. *Work. Pap. Stanf. Univ*, 2021.

[46] Hamed Mohsenian-Rad. Coordinated price-maker operation of large energy storage units in nodal energy markets. *IEEE Transactions on Power Systems*, 31(1):786–797, 2015.

[47] Jesus E Contereras-Ocana, Yishen Wang, Miguel A Ortega-Vazquez, and Baosen Zhang. Energy storage: Market power and social welfare. In *2017 IEEE Power & Energy Society General Meeting*, pages 1–5. IEEE, 2017.

[48] Truong X Nghiem and Colin N Jones. Data-driven demand response modeling and control of buildings with Gaussian processes. In *2017 American Control Conference (ACC)*, pages 2919–2924. IEEE, 2017.

[49] José Vuelvas and Fredy Ruiz. A novel incentive-based demand response model for Cournot competition in electricity markets. *Energy Systems*, 10(1):95–112, 2019.

[50] Ricardo Fernández-Blanco, Juan Miguel Morales, and Salvador Pineda. Forecasting the price-response of a pool of buildings via homothetic inverse optimization. *Applied Energy*, 290:116791, 2021.

[51] Tianguang Lu, Zhaoyu Wang, Jianhui Wang, Qian Ai, and Chong Wang. A data-driven Stackelberg market strategy for demand response-enabled distribution systems. *IEEE Transactions on Smart Grid*, 10(3):2345–2357, 2018.

[52] John Duchi, Stephen Boyd, and Jacob Mattingley. Sequential convex programming. *Notes for EE364b, Stanford University*, 2018.

[53] New york independent system operator – energy market and operation data.

[54] Andrew Wilson, Danielle Esterhuysen, and Dominic Hains. 2020 Performance Review — UQ's 1.1 MW Battery Project. University of Queensland. Accessed: Dec. 3, 2022.

[55] Timuçin Harputlugil and Pieter de Wilde. The interaction between humans and buildings for energy efficiency: A critical review. *Energy Research & Social Science*, 71:101828, 2021.

[56] Mehzabeen Mannan and Sami G Al-Ghamdi. Indoor air quality in buildings: a comprehensive review on the factors influencing air pollution in residential and commercial structure. *International Journal of Environmental Research and Public Health*, 18(6):3276, 2021.

[57] Abhinandana Boodi, Karim Beddiar, Malek Benamour, Yassine Amirat, and Mohamed Benbouzid. Intelligent systems for building energy and occupant comfort optimization: A state of the art review and recommendations. *Energies*, 11(10):2604, 2018.

[58] Stephen Xia, Peter Wei, Yanchen Liu, Andrew Sonta, and Xiaofan Jiang. RECA: A multi-task deep reinforcement learning-based recommender system for co-optimizing energy, comfort and air quality in commercial buildings. In *Proceedings of the 10th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 99–109, 2023.

[59] Ashkan Haji Hosseinloo, Saleh Nabi, Anette Hosoi, and Munther A Dahleh. Data-driven control of COVID-19 in buildings: a reinforcement-learning approach. *IEEE Transactions on Automation Science and Engineering*, 2023.

[60] Yixin Dong, Li Zhu, Sui Li, and Martin Wollensak. Optimal design of building openings to reduce the risk of indoor respiratory epidemic infections. In *Building Simulation*, pages 1–14. Springer, 2022.

[61] UCSD. Facilities management response to the COVID-19 pandemic. https://blink.ucsd.edu/facilities/management/covid-19.html, 2023.

[62] REHVA. COVID-19 guidance. https://www.rehva.eu/activities/covid-19-guidance, 2020.

[63] Luigi Schibuola and Chiara Tambani. High energy efficiency ventilation to limit COVID-19 contagion in school environments. *Energy and Buildings*, 240:110882, 2021.

[64] Naohide Shinohara, Koichi Tatsu, Naoki Kagi, Hoon Kim, Jun Sakaguchi, Isamu Ogura, Yoshiko Murashima, Hiromu Sakurai, and Wataru Naito. Air exchange rates and advection-diffusion of CO_2 and aerosols in a route bus for evaluation of infection risk. *Indoor Air*, 32(3):e13019, 2022.

[65] Sheng Zhang, Zhengtao Ai, and Zhang Lin. Novel demand-controlled optimization of constant-air-volume mechanical ventilation for indoor air quality, durability and energy saving. *Applied Energy*, 293:116954, 2021.

[66] Markus Gwerder, Dimitrios Gyalistras, Frauke Oldewurtel, Beat Lehmann, Katharina Wirth, Vanessa Stauch, and Jürg Tödtli. Potential assessment of rule-based control for integrated room automation. In *10th REHVA World Congress Clima*, pages 9–12, 2010.

[67] Yuexin Bian, Xiaohan Fu, Bo Liu, Rohith Rachala, Rajesh K Gupta, and Yuanyuan Shi. BEAR-Data: Analysis and applications of an open multizone building dataset. In *Proceedings of the 10th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 240–243, 2023.

[68] Elence Xinzhu Chen, Xu Han, Ali Malkawi, Runyu Zhang, and Na Li. Adaptive model predictive control with ensembled multi-time scale deep-learning models for smart control of natural ventilation. *Building and Environment*, page 110519, 2023.

[69] Yize Chen, Yuanyuan Shi, and Baosen Zhang. Optimal control via neural networks: A convex approach. In *International Conference on Learning Representations*, 2018.

[70] Ján Drgoňa, Javier Arroyo, Iago Cupeiro Figueroa, David Blum, Krzysztof Arendt, Donghun Kim, Enric Perarnau Ollé, Juraj Oravec, Michael Wetter, Draguna L Vrabie, et al. All you need to know about model predictive control for buildings. *Annual Reviews in Control*, 50:190–232, 2020.

[71] Chi Zhang, Yuanyuan Shi, and Yize Chen. BEAR: Physics-principled building environment for control and reinforcement learning. In *Proceedings of the 14th ACM International Conference on Future Energy Systems*, pages 66–71, 2023.

[72] Tianshu Wei, Yanzhi Wang, and Qi Zhu. Deep reinforcement learning for building HVAC control. In *Proceedings of the 54th Annual Design Automation Conference 2017*, pages 1–6, 2017.

[73] Liang Yu, Yi Sun, Zhanbo Xu, Chao Shen, Dong Yue, Tao Jiang, and Xiaohong Guan. Multi-agent deep reinforcement learning for HVAC control in commercial buildings. *IEEE Transactions on Smart Grid*, 12(1):407–419, 2020.

[74] SungKu Heo, KiJeon Nam, Jorge Loy-Benitez, Qian Li, SeungChul Lee, and ChangKyoo Yoo. A deep reinforcement learning-based autonomous ventilation control system for smart indoor air quality management in a subway station. *Energy and Buildings*, 202:109440, 2019.

[75] Zechariah Lau, Ian M Griffiths, Aaron English, and Katerina Kaouri. Predicting the spatio-temporal infection risk in indoor spaces using an efficient airborne transmission model. *Proceedings of the Royal Society A*, 478(2259):20210383, 2022.

[76] Sai Ranjeet Narayanan and Suo Yang. Airborne transmission of virus-laden aerosols inside a music classroom: Effects of portable purifiers and aerosol injection rates. *Physics of Fluids*, 33(3), 2021.

[77] Yi He, Yingnan Chu, Haiyan Zang, Jinyan Zhao, and Yehao Song. Experimental and CFD study of ventilation performance enhanced by roof window and mechanical ventilation system with different design strategies. *Building and Environment*, 224:109566, 2022.

[78] Renyi Zhang, Yixin Li, Annie L Zhang, Yuan Wang, and Mario J Molina. Identifying airborne transmission as the dominant route for the spread of COVID-19. *Proceedings of the National Academy of Sciences*, 117(26):14857–14863, 2020.

[79] Runxin He and Humberto Gonzalez. Zoned HVAC control via PDE-constrained optimization. In *2016 American Control Conference (ACC)*, pages 587–592. IEEE, 2016.

[80] Shumon Koga, Mouhacine Benosman, and Jeff Borggaard. Learning-based robust observer design for coupled thermal and fluid systems. In *2019 American Control Conference (ACC)*, pages 941–946. IEEE, 2019.

[81] Tianqi Xiao and Fengqi You. Building thermal modeling and model predictive control with physically consistent deep learning for decarbonization and energy optimization. *Applied Energy*, 342:121165, 2023.

[82] A.M. Farahmand, S. Nabi, and D.N. Nikovski. Deep learning for estimation and control of partial differential equations. *arXiv preprint*, 2017.

[83] Bharathan Balaji, Hidetoshi Teraoka, Rajesh Gupta, and Yuvraj Agarwal. ZonePAC: Zonal power estimation and control via HVAC metering and occupant feedback. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, pages 1–8, 2013.

[84] Bingqing Chen, Zicheng Cai, and Mario Bergés. GNU-RL: A precocial reinforcement learning solution for building HVAC control using a differentiable MPC policy. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 316–325, 2019.

[85] A. McNamara, A. Treuille, Z. Popovic, and J. Stam. Fluid control using the adjoint method. *ACM Transactions on Graphics*, 23(3):449–456, 2004.

[86] Jacques Louis Lions. *Optimal Control of Systems Governed by Partial Differential Equations*, volume 170. Springer, 1971.

[87] P. Holl, V. Koltun, K. Um, and N. Thuerey. PhiFlow: A differentiable PDE solving framework for deep learning via physical simulations. https://github.com/tum-pbs/PhiFlow, 2020.

[88] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. 2017.

[89] F. Kreith. *The CRC Handbook of Thermal Engineering*. CRC Press, 1999.

[90] P.F. Linden. The fluid mechanics of natural ventilation. *Annual Review of Fluid Mechanics*, 31:201–238, 1999.

[91] Wei Zhang, Wentao Wu, Leslie Norford, Na Li, and Ali Malkawi. Model predictive control of short-term winter natural ventilation in a smart building using machine learning algorithms. *Journal of Building Engineering*, 73:106602, 2023.

[92] Behrang Chenari, João Dias Carrilho, and Manuel Gameiro Da Silva. Towards sustainable, energy-efficient and healthy ventilation strategies in buildings: A review. *Renewable and Sustainable Energy Reviews*, 59:1426–1447, 2016.

[93] Shanrui Shi, Shohei Miyata, and Yasunori Akashi. Event-driven model-based optimal demand-controlled ventilation for multizone vav systems: Enhancing energy efficiency and indoor environmental quality. *Applied Energy*, 377:124683, 2025.

[94] Yuan Gao, Shohei Miyata, and Yasunori Akashi. Energy saving and indoor temperature control for an office building using tube-based robust model predictive control. *Applied Energy*, 341:121106, 2023.

[95] Wei Su, Zhengtao Ai, Jing Liu, Bin Yang, and Faming Wang. Maintaining an acceptable indoor air quality of spaces by intentional natural ventilation or intermittent mechanical ventilation with minimum energy use. *Applied Energy*, 348:121504, 2023.

[96] Yan Du, Helia Zandi, Olivera Kotevska, Kuldeep Kurte, Jeffery Munk, Kadir Amasyali, Evan Mckee, and Fangxing Li. Intelligent multi-zone residential hvac control strategy based on deep reinforcement learning. *Applied Energy*, 281:116117, 2021.

[97] Ting Yang, Liyuan Zhao, Wei Li, Jianzhong Wu, and Albert Y Zomaya. Towards healthy and cost-effective indoor environment management in smart homes: A deep reinforcement learning approach. *Applied Energy*, 300:117335, 2021.

[98] Hao Wang, Xiwen Chen, Natan Vital, Edward Duffy, and Abolfazl Razi. Energy optimization for hvac systems in multi-vav open offices: A deep reinforcement learning approach. *Applied Energy*, 356:122354, 2024.

[99] Chunxiao Li, Can Cui, and Ming Li. A proactive 2-stage indoor co2-based demand-controlled ventilation method considering control performance and energy efficiency. *Applied Energy*, 329:120288, 2023.

[100] Jianqiang Mou, Shan Cui, and David Wee Yang Khoo. Computational fluid dynamics modelling of airflow and carbon dioxide distribution inside a seminar room for sensor placement. *Measurement: Sensors*, 23:100402, 2022.

[101] Mao Ning, Song Mengjie, Chan Mingyin, Pan Dongmei, and Deng Shiming. Computational fluid dynamics (cfd) modelling of air flow field, mean age of air and co2 distributions inside a bedroom with different heights of conditioned air supply outlet. *Applied Energy*, 164:906–915, 2016.

[102] Mohamed Mahmoud Abdelkareem Mahmoud, P Bahl, AFV de A Aquino, CR Maclntyre, S Bhattacharjee, D Green, N Cooper, C Doolan, and C de Silva. A numerical framework

for the analysis of indoor air quality in a classroom. *Journal of Building Engineering*, 92:109659, 2024.

[103] Q Zhou and R Ooka. Neural network for indoor airflow prediction with cfd database. In *Journal of Physics: Conference Series*, volume 2069, page 012154. IOP Publishing, 2021.

[104] Alok Warey, Shailendra Kaushik, Bahram Khalighi, Michael Cruse, and Ganesh Venkatesan. Data-driven prediction of vehicle cabin thermal comfort: using machine learning and high-fidelity simulation results. *International Journal of Heat and Mass Transfer*, 148:119083, 2020.

[105] Hu Gao, Weixin Qian, Jiankai Dong, and Jing Liu. Rapid prediction of indoor airflow field using operator neural network with small dataset. *Building and Environment*, 251:111175, 2024.

[106] Xiaoxiao Ding, Haotian Zhang, Weirong Zhang, Weijia Zhang, and Yingli Xuan. A fourier neural operator-based method for rapid prediction of 3d indoor airflow dynamics. In *Building Simulation*, pages 1–17. Springer, 2025.

[107] Nicola Bianco, Andrea Fragnito, Marcello Iasiello, and Gerardo Maria Mauro. A cfd multi-objective optimization framework to design a wall-type heat recovery and ventilation unit with phase change material. *Applied Energy*, 347:121368, 2023.

[108] Anna Bulińska, Zbigniew Popiołek, and Zbigniew Buliński. Experimentally validated cfd analysis on sampling region determination of average indoor carbon dioxide concentration in occupied space. *Building and Environment*, 72:319–331, 2014.

[109] Yunpeng Wang, Zhijie Li, Zelong Yuan, Wenhui Peng, Tianyuan Liu, and Jianchun Wang. Prediction of turbulent channel flow using fourier neural operator-based machine-learning strategy. *Physical Review Fluids*, 9(8):084604, 2024.

[110] Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, et al. Geometry-informed neural operator for large-scale 3d pdes. *Advances in Neural Information Processing Systems*, 36, 2024.

[111] Anna Bulińska and Zbigniew Buliński. A cfd analysis of different human breathing models and its influence on spatial distribution of indoor air parameters. *Computer Assisted Methods in Engineering and Science*, 22(3):213–227, 2017.

[112] Xinyi Sha, Zhenjun Ma, Subbu Sethuvenkatraman, and Wanqing Li. Online learning-enhanced data-driven model predictive control for optimizing hvac energy consumption, indoor air quality and thermal comfort. *Applied Energy*, 383:125341, 2025.

[113] Yize Chen, Yuanyuan Shi, and Baosen Zhang. Modeling and optimization of complex building energy systems with deep neural networks. In *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pages 1368–1373. IEEE, 2017.

[114] Andrea Carlo D'Alicandro and Alessandro Mauro. Experimental and numerical analysis of co2 transport inside a university classroom: effects of turbulent models. *Journal of Building Performance Simulation*, 16(4):434–459, 2023.

[115] User Manual. Ansys fluent 12.0. *Theory Guide*, 67, 2009.

[116] Mohamed Abuhegazy, Khaled Talaat, Osman Anderoglu, and Svetlana V Poroseva. Numerical investigation of aerosol transport in a classroom with relevance to covid-19. *Physics of Fluids*, 32(10), 2020.

[117] Kevin Weekly, Nikolaos Bekiaris-Liberis, Ming Jin, and Alexandre M Bayen. Modeling and estimation of the humans' effect on the co 2 dynamics inside a conference room. *IEEE Transactions on Control Systems Technology*, 23(5):1770–1781, 2015.

[118] ASHRAE Standard 55–thermal environmental conditions for human occupancy. Technical report, ASHRAE Inc., Atlanta, GA, 1992.

[119] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[120] ASHRAE. Ventilation for acceptable indoor air quality: ASHRAE standard 62.1. American Society of Heating, Refrigerating and Air-Conditioning Engineers, 2010.

[121] P. Pant, R. Doshi, P. Bahl, and A.B. Farimani. Deep learning for reduced order modelling and efficient temporal evolution of fluid simulations. *Physics of Fluids*, 33(10), 2021.

[122] Jan Machowski, Zbigniew Lubosny, Janusz W Bialek, and James R Bumby. *Power system dynamics: stability and control*. John Wiley & Sons, 2020.

[123] Doseok Jang, Larry Yan, Lucas Spangher, and Costas J Spanos. Active reinforcement learning for robust building control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 22150–22158, 2024.

[124] Rudy R Negenborn, Bart De Schutter, and Johannes Hellendoorn. Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. *Engineering Applications of Artificial Intelligence*, 21(3):353–366, 2008.

[125] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. John Wiley & Sons, 2020.

[126] Yize Chen, Yuanyuan Shi, and Baosen Zhang. Optimal control via neural networks: A convex approach. In *International Conference on Learning Representations (ICLR)*, 2019.

[127] Tim Salzmann, Elia Kaufmann, Jon Arrizabalaga, Marco Pavone, Davide Scaramuzza, and Markus Ryll. Real-time neural mpc: Deep learning model predictive control for quadrotors and agile robotic platforms. *IEEE Robotics and Automation Letters*, 8(4):2397–2404, 2023.

[128] Kunwu Zhang, Qi Sun, and Yang Shi. Trajectory tracking control of autonomous ground vehicles using adaptive learning mpc. *IEEE Transactions on Neural Networks and Learning Systems*, 32(12):5554–5564, 2021.

[129] Priya Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning in stochastic optimization. *Advances in neural information processing systems*, 30, 2017.

[130] Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *International Conference on Machine Learning (ICML)*, pages 4732–4741. PMLR, 2018.

[131] Hossein Nejatbakhsh Esfahani, Arash Bahari Kordabad, and Sébastien Gros. Approximate robust nmpc using reinforcement learning. In *2021 European Control Conference*, pages 132–137. IEEE, 2021.

[132] Min Lin, Zhongqi Sun, Yuanqing Xia, and Jinhui Zhang. Reinforcement learning-based model predictive control for discrete-time systems. *IEEE Transactions on Neural Networks and Learning Systems*, 35(3):3312–3324, 2023.

[133] Rudolf Reiter, Jasper Hoffmann, Dirk Reinhardt, Florian Messerer, Katrin BaumgÃĪrtner, Shamburaj Sawant, Joschka Boedecker, Moritz Diehl, and Sebastien Gros. Synthesis of model predictive control and reinforcement learning: Survey and classification. *arXiv preprint arXiv:2502.02133*, 2025.

[134] Nathan P Lawrence, Philip D Loewen, Michael G Forbes, R Bhushan Gopaluni, and Ali Mesbah. A view on learning robust goal-conditioned value functions: Interplay between rl and mpc. *arXiv preprint arXiv:2502.06996*, 2025.

[135] Risheng Liu, Xin Fan, Shichao Cheng, Xiangyu Wang, and Zhongxuan Luo. Proximal alternating direction network: A globally converged deep unrolling framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[136] Sébastien Gros and Mario Zanon. Data-driven economic nmpc using reinforcement learning. *IEEE Transactions on Automatic Control*, 65(2):636–648, 2019.

[137] Sébastien Gros and Mario Zanon. Reinforcement learning based on mpc and the stochastic policy gradient method. In *2021 American Control Conference*, pages 1947–1952. IEEE, 2021.

[138] Katrine Seel, Sébastien Gros, and Jan Tommy Gravdahl. Combining q-learning and deterministic policy gradient for learning-based mpc. In *2023 62nd IEEE Conference on Decision and Control*, pages 610–617. IEEE, 2023.

[139] Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. Stochastic variance-reduced policy gradient. In *International Conference on Machine Learning (ICML)*, pages 4026–4035. PMLR, 2018.

[140] Kaiyi Ji, Junjie Yang, and Yingbin Liang. Bilevel optimization: Convergence analysis and enhanced design. In *International Conference on Machine Learning (ICML)*, pages 4882–4892. PMLR, 2021.

[141] Jeongyeol Kwon, Dohyun Kwon, Stephen Wright, and Robert D Nowak. A fully first-order method for stochastic bilevel optimization. In *International Conference on Machine Learning*, pages 18083–18113. PMLR, 2023.

[142] Joel AE Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[143] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106:25–57, 2006.

[144] Philip E Gill, Walter Murray, and Michael A Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM review*, 47(1):99–131, 2005.

[145] Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[146] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoffrey Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. *International Conference on Learning Representations (ICLR)*, 2017.

[147] Walter Rudin. *Principles of Mathematical Analysis*. McGraw-Hill, 3rd edition, 1964.

[148] Damien Scieur, Gauthier Gidel, Quentin Bertrand, and Fabian Pedregosa. The curse of unrolling: Rate of differentiating through optimization. *Advances in Neural Information Processing Systems*, 35:17133–17145, 2022.

[149] Dirk Reinhardt, Katrin Baumgärtner, Jonathan Frey, Moritz Diehl, and Sebastien Gros. Mpc4rl-a software package for reinforcement learning based on model predictive control. In *2024 IEEE 63rd Conference on Decision and Control*, pages 1787–1794. IEEE, 2024.

[150] Jie Feng, Wenqi Cui, Jorge Cortés, and Yuanyuan Shi. Bridging transient and steady-state performance in voltage control: A reinforcement learning approach with safe gradient flow. *IEEE Control Systems Letters*, 7:2845–2850, 2023.

[151] Konstantin Turitsyn, Petr Sulc, Scott Backhaus, and Michael Chertkov. Options for control of reactive power by distributed photovoltaic generators. *Proceedings of the IEEE*, 99(6):1063–1073, 2011.

[152] Yuanyuan Shi, Guannan Qu, Steven Low, Anima Anandkumar, and Adam Wierman. Stability constrained reinforcement learning for real-time voltage control. In *2022 American Control Conference*, pages 2715–2721. IEEE, 2022.

[153] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[154] Zhengpeng Xie, Qiang Zhang, Fan Yang, Marco Hutter, and Renjing Xu. Simple policy optimization. In *Forty-second International Conference on Machine Learning*, 2025.

[155] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[156] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France, 07–09 Jul 2015. PMLR.

[157] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1928–1937, New York, New York, USA, 20–22 Jun 2016. PMLR.

[158] Shengyi Huang, Rousslan Fernand Julien Dossa, Antonin Raffin, Anssi Kanervisto, and Weixun Wang. The 37 implementation details of proximal policy optimization. *The ICLR Blog Track 2023*, 2022.

[159] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[160] Tao Wang, Ruipeng Zhang, and Sicun Gao. Improving value estimation critically enhances vanilla policy gradient. In *Forty-second International Conference on Machine Learning*, 2025.

[161] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[162] Yuanyang Zhu, Zhi Wang, Yuanheng Zhu, Chunlin Chen, and Dongbin Zhao. Discretizing continuous action space with unimodal probability distributions for on-policy reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[163] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, pages 1–7, 2025.

[164] Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs

beyond the base model? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[165] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025.

[166] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, page 1279–1297, New York, NY, USA, 2025. Association for Computing Machinery.

[167] Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22(98):1–76, 2021.

[168] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PmLR, 2016.

[169] Karl W Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *International Conference on Machine Learning*, pages 2020–2027. PMLR, 2021.

[170] Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *Advances in neural information processing systems*, 30, 2017.

[171] Sven Gronauer, Martin Gottwald, and Klaus Diepold. The successful ingredients of policy gradient algorithms. In *IJCAI*, pages 2455–2461, 2021.

[172] Po-Wei Chou, Daniel Maturana, and Sebastian Scherer. Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution. In *International conference on machine learning*, pages 834–843. PMLR, 2017.

[173] Perttu Hämäläinen, Amin Babadi, Xiaoxiao Ma, and Jaakko Lehtinen. Ppo-cma: Proximal policy optimization with covariance matrix adaptation. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2020.

[174] Hojoon Lee, Dongyoon Hwang, Donghu Kim, Hyunseung Kim, Jun Jet Tai, Kaushik Subramanian, Peter R Wurman, Jaegul Choo, Peter Stone, and Takuma Seno. Simba: Simplicity bias for scaling up parameters in deep reinforcement learning. *arXiv preprint arXiv:2410.09754*, 2024.

[175] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[176] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[177] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[178] Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and JoÃĢo GM AraÃšjo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022.

[179] Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control, 2024.

[180] Nicklas Hansen, Hao Su, and Xiaolong Wang. Learning massively multitask world models for continuous control, 2025.

[181] Hojoon Lee, Youngdo Lee, Takuma Seno, Donghu Kim, Peter Stone, and Jaegul Choo. Hyperspherical normalization for scalable deep reinforcement learning. *arXiv preprint arXiv:2502.15280*, 2025.

[182] Michal Nauman, Marek Cygan, Carmelo Sferrazza, Aviral Kumar, and Pieter Abbeel. Bigger, regularized, categorical: High-capacity value functions are efficient multi-task learners. *arXiv preprint arXiv:2505.23150*, 2025.

[183] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International conference on machine learning*, pages 10524–10533. PMLR, 2020.

[184] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[185] Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample efficient continuous control. *Advances in neural information processing systems*, 37:113038–113071, 2024.

[186] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[187] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *Robotics: Science and Systems*, 2025.

[188] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.